ASSEMBLER

KASSETTE
MO5 E TO7-70

INFOGRAMES

ASSEMBLER

Autor : William Hennebois.

I - ALLGEMEINES

Sie sind nun im Besitz eines ASSEMBLER-Programmes. Dieses Programm erlaubt Ihnen alle Möglichkeiten ihres Computers THOMSON auszuschöpfen. Dieser ASSEMBLER teilt sich in drei Untereinheiten auf:

Der Editor beinhaltet alle Befehle und Funktionen, die es Ihnen gestatten die Assemblertexte einzugeben (Quell-oder Ursprungsprogramm genannt).

Der ASSEMBLER ist ein Programm, welches das Ursprungsprogramm In binären Code übersetzt, der vom Mikroprozessor 68Ø9 verstanden wird.

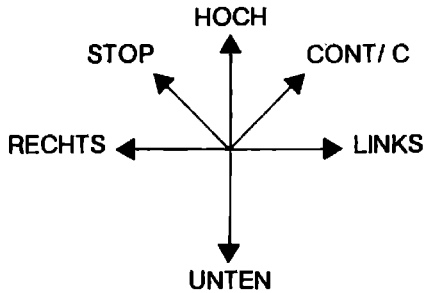
Der Debugger hilft Ihnen die Programme zu testen, indem er von zahlreichen Möglichkeiten wie Haltepunkte, Disassembler, Prüfung und Änderung der Register und Speicher Gebrauch macht.

Der Umgang mit diesem Programm verlangt selbstverständlich einige Kenntnisse der Assemblersprache und des Mikroprozessors 68Ø9, mit dem ihr Computer THOMSON ausgestattet ist. Falls Sie nicht über diese Kenntnisse verfügen sollten, können Sie sie in der entsprechenden Fachliteratur nachlesen. Sie finden anschliessend eine Liste der benützten Ausdrücke vor.

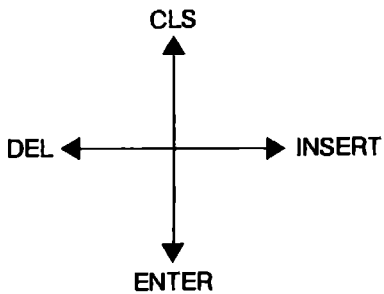
Kennzeichnung der Dateien.
(<Dateiname>.<Zusatz>)
Der Dateiname kann bis zu 8 Zeichen enthalten.
Der Zusatz kann bis zu 3 Zeichen enthalten.
Wenn kein Zusatz angegeben wird, bildet das System die Zusätze:
- ASS für die Textdatei (das Quellprogramm).
- BIN für die Binärdatei (den erzeugten Objektcode).

Symbole.
$N1 Der Wert N1 ist in Hexadezimal geschrieben. Beispiel: NAME EQU$ E7CC
Adresse der ersten Zeile, Beispiel: LIST
* Adresse der letzten Zeile, Beispiel: DEL #, *
Mit dem Joystick des ersten Spiels kann man den Cursor auf dem Bildschirm bewegen. Er ersetzt auch einige Tasten des Editors.

Bedienungsanleitung des Joysticks.



Bedienungsanleitung des Joysticks beim Knopfdruck.



Das Zeilenformat.
Das Zeilenformat ist wie folgt:
(Zeilennummer) (Adressename) (Anweisung) (Arbeitsbereich)
Beispiel:

```
ØØØ1Ø    LOOP    LDA#1
ODER
ØØØ1Ø             LDA#%1111
```

Mit der Hilfe der ⭾ - Taste können Sie in Achterschritten tabulieren. Beim Übertreten des Bildschirmrandes wird die momentane Zeile angezeigt.

Editortasten.
Die folgende Funktionsbeschreibung der Tasten bezieht sich nur auf das Assemblerprogramm.

ENTER — Zeigt dem Computer an, daß die geschriebene Zeile beendet ist und ins Programm übernommen werden soll. Der Cursor geht anschließend wieder an den Anfang der nächsten Zeile zurück. Dies entspricht dem Carriage Return (Wagenrücklauf).

STOP	beendet die Ausführung eines Befehls. Um wieder zu starten, drücken Sie auf ENTER.
CNT	Die CONTROL-Taste muß gleichzeitig mit einer anderen Taste gedrückt werden, die die Richtung ändert.
CNT. C	Unterbricht den laufenden Befehl oder die automatische Numerierung, wenn Sie angeordnet war. Eine gerade eingegebene Zeile wird nicht berücksichtigt. Der Computer ist wieder im Befehls-Modus.
CNT. X	Löscht den Rest der Linie von der Cursor-Stelle ab aus.
CLS	Löscht den gesamten Bildschirm aus.
DEL	Taste "AUSLÖSCHEN". Sie löscht das Zeichen aus, das sich über dem Cursor befindet; der Rest der Zeile rechts vom Cursor rückt dann eine Stelle nach links.
INS	Taste "INSERTION" (Einfügung). Sie schafft ein Leerzeichen über dem Cursor und verschiebt den Rest der Zeile nach rechts.
⇆	Verschiebt den Cursor in Achterschritten.

Editor Befehle.

LIST	Listet den Quelltext auf dem Bildschirm.
LIST N1, N2	Listet die Zeilen zwischen N1 und N2 auf.
LLIST	Druckt den Quelltext aus. Kann mit STOP oder CONT/ C beendet werden.
LLIST N1, N2	Druckt die Zeilen zwischen N1 und N2 aus.
DEL N1, N2	Löscht die Zeilen zwischen N1 und N2 aus.
FXXXXX	Sucht das Wort XXXXX im Text nach.
F	Sucht die nächste Zeile, die das gesuchte Wort enthält.
AUTO	Numeriert die Zeilen automatisch.
AUTO N1, N2	Numeriert in N1-Schritten von N2 aus.
RENUM	Numeriert in Zehner Schritten einen schon eingegebenen Text.
RENUM N1, N2	Numeriert in N1 Schritten von N2 aus.
ZONE N1	Hält die Adresse des Speichers fest.
EXEC	Ausführung des Programms.
EXEC N1	Ausführung des Programms ab Adresse N1.
EXEC Datei	Laden und Ausführung einer Datei.
LOAD Datei	Laden einer Assemblerquelle.
SAVE Datei	Speichert das Programm auf Kassette.
LOADM Datei	Lädt ein Programm in binärem Format. Die Ausführungsadresse befindet sich in EXEC.

SAVEM N1,N2,N3: Datei.
Speichert den Speicher zwischen N1 und N2 auf Kassette. N3 ist die Ausführungsadresse.

TBUG Ruft den Debugger-Mode auf.

MERGE Datei Verbindet die Quellprogramm Datei mit dem im Speicher befindlichen Quellprogramm.

SCREEN N1, N2, N3,N4
Ändert die Bildschirmfarbe. N1 = Farbe der ASSEMBLER, N2 = TBUG-Farbe, N3 = Hinter grundfarbe, N4 = Bildrahmenfarbe.

Der ASSEMBLER.

ASS Startet den ASSEMBLER mit folgenden Wahlmöglichkeiten.

CO Zeigt den Text und den Code an.

IMP Wählt den Drucker.

ECR Zeigt den Text an.

TL Zeigt die Adressentabelle an.

RM Assembliert den Text direkt in einen reservierten Speicherbereich. (Vergessen Sie nicht, den Bereich vorher mit dem Befehl ZONE zu reservieren).

:Datei Der Objektcode wird (als Binärdatei) auf dem Kassettenrekorder gespeichert.

Beispiel:
Die Befehlszeile ASS CO, TL: CODE assembliert den Quelltext zu einer Binärdatei mit Kennzeichnung CODE.BIN auf der Kassette, und zeigt das Assemblerlisting und die Symboltabelle auf dem Bildschirm an.

Pseudo-Anweisung des ASSEMBLERS.
Eine Pseudo-Anweisung entspricht keinem Mikroprozessorbefehl, sondern ist eine Anweisung an den Assembler. Die folgenden Pseudo-Anweisungen sind also spezifisch für dieses Programm. Sie dienen zur Verwaltung des Speicherbereichs, der Definition von symbolischen Adressen oder Konstanten, und auch zur Steuerung des Druckers.

– **EQU** Zuweisung eines Werts an ein Symbol.
Syntax: <Symbol> EQU <Ausdruck>
Effekt: EQU weist dem <Symbol> den Wert (8 oder 16 bit) des <Ausdrucks> zu, der im Operandenfeld steht.

– **RMB** Reservierung von Speicherplatz.
Syntax: <Symbol> RMB <Ausdruck>
Effekt: RMB hält ab der durch <Symbol> gekennzeichneten Adresse soviele Bytes an Speicherplatz frei, wie durch den <Ausdruck> angegeben werden.

– **ORG** Festlegung der Anfangsadresse eines Programms im Speicher.
Syntax: ORG <Ausdruck>
Effekt: Der Assembler nimmt <Ausdruck> als Anfangsadresse für den erzeugten Objektcode an.

– **FCB** Definition von 8-bit Konstanten.
Syntax: <Symbol> FCB < Ausdruck >, < Ausdruck >, < Ausdruck >, ...
Effekt: Die durch <Ausdruck> ... gebildeten 8-bit Konstanten werden ab der momentanen PC-Adresse (die durch <Symbol> gekennzeichnet wird) in den erzeugten Code eingefügt.

– **FDB** Definition von 16-bit Konstanten.
Syntax: <Symbol> FDB <Ausdruck>, <Ausdruck>, <Ausdruck>, ...
Effekt: Wie bei FCB; jedoch werden 16-bit Konstanten eingefügt.

– **FCC** Definition einer Zeichenkette.
Syntax: <Symbol> FCC </ ><Zeichenkette></ >.
Effekt: Die <Zeichenkette> wird als Folge von ASCII-Zeichen ab der momentanen PC-Adresse in den Code eingefügt.

– **END** Ende des Quellprogramms.
Syntax: END <Ausdruck>
Effekt: END markiert das Ende des Quelltextes. Die Programmzeilen nach END werden vom Assembler nicht berücksichtigt. Der <Ausdruck> hält die Anfangsadresse des Programms fest.

– **UTIL** ändert die Druckparameter während des Assemblerlaufs.
Wenn %1ØØØ Wahl des Druckers.
Wenn %Ø1ØØ Ausdruck der Adressennamentabelle.
Wenn %ØØ11 Ausdruck des Quellprogramms und des Codes.
Beispiel:

```
ØØØ1Ø    LDA    #1
ØØØ2Ø    UTIL   %ØØ11
ØØØ3Ø    STA    $ØØØØ
ØØØ4Ø*
```

Tippen Sie: ASS.
Das Quellprogramm und der Code werden ab der Zeile ØØØ2Ø angezeigt.

Konstantentypen.

$----	Hexadezimal
%----	Binär
!-	ASCII
----	Dezimal
Adressenname	Maximal 8 Zeichen außer (X, Y, A, B, PCR, D, CCR)
*	Adressenname der den momentanen Wert des PCR darstellt.

Operation mit Adresssennamen.
Das Programm ASSEMBLER ermöglicht Berechnungen mit den Adressennamen.

Die Addition und Subtraktion sind ohne Einschränkung möglich.

```
Beispiel: ØØØ1Ø    LOOP    EQU   $4Ø
          ØØØ2Ø            ORG   $96ØØ
          ØØØ3Ø            LDB #LOOP+1*        setzt den Wert $41 in B.
```

Diese Operationen können auch mit negativen Werten erfolgen.

```
Beispiel: ØØØ1Ø    LOOP    EQU   $4Ø
          ØØØ2Ø            ORG   $96ØØ
          ØØØ3Ø            LDB #-LOOP+1*       setzt den Wert $C1 in B.
```

oder

```
Beispiel: ØØØ1Ø    LOO1    EQU   5Ø
          ØØØ2Ø    LOO2    EQU   1Ø
          ØØØ3Ø            ORG   $96ØØ
          ØØØ4Ø            LDB #-LOO1+LOO2* setzt den Wert $D8 in B.
```

TBUG Befehle.

TBUG ist ein leistungsstarker Debugger.
Man erhält ihn durch den Befehl TBUG vom Editor aus.

D N1, N2	Anzeige der Hexadezimalwerte und ASCII zwischen den Adressen N1 und N2.
I N1	Ermöglicht Änderungen des Speicherinhaltes von N1 ab.
V	Macht alle Register sichtbar.
R reg. N1	Ermöglicht den Register "Reg" mit dem Wert N1 zu laden.
IMP, com	Wählt den Drucker für den folgenden Befehl 1.
GN1, N2	Startet Programmausführung an Adresse N1 und setzt Haltepunkt (Break point) auf der Adresse N2.
P	Führt ein Programm Schritt für Schritt aus.
PN1	Führt ein Programm Schritt für Schritt aus ab Adresse N1. /Leertaste/führt Anweisung aus; mit/P/wird das Programm fortgesetzt CONT/C Ende.
Q	Verlasse TBUG.
SN1,N2	Disassembliert zwischen den Adressen N1 und N2.

II - BENUTZUNG EINES TO7-7Ø

A - Wie das Programm geladen wird.
Dieses Programm benötigt folgende Mindestaustattung:
- Zentraleinheit TO7-7Ø,
- Basic - Programm - Modul,
- Programm - Rekorder,
- Joysticks und Fernsehgerät.

Achten Sie darauf, daß alle Geräte richtig angeschlossen und eingeschaltet sind.
Legen Sie das Basic - Programm - Modul in das vorgesehene Modulfach bei ausgeschalteter Zentraleinheit.
Verbinden Sie den Programm-Rekorder mit der Seite der Zentraleinheit.
Nach dem Einschalten wird das Menü angezeigt.
Legen Sie die Kassette in den Programm-Rekorder mit der TO7-7Ø Seite ein, drükken Sie auf lesen. Es ist völlig normal, daß der Rekorder nicht startet.
Wählen Sie den Punkt 1, um in Basic zu gelangen.
Geben Sie **LOADM** ein.
Der Programm-Rekorder startet automatisch, das Programm wird geladen und der Programm-Rekorder schaltet automatisch wieder aus.

B - Besonderheiten der TO7-7Ø Version.
Der ASSEMBLER besetzt 11 K Byte. Das Ursprungsprogramm ist von der Adresse $8FBØ ab gelegt. Wenn Sie während eines Ladevorgangs mit EXEC oder LOADM, den Fehler:
"Chargement sur le système" (Ladung auf System) auslösen, bedenken Sie daß sich nach dem Befehl DEL #,* kein Ursprungsprogramm mehr im Speicher befindet.
Der Reset darf nur im dringendsten Fall angewendet werden. Er führt zum Verlust des Ursprungsprogramms, das sich im Speicher befindet, und nur ein EXEC & H6B7D kann es wieder starten.

C - Einsprungadressen TO7-7Ø.
Ihr TO7-7Ø verfügt im ROM über eine Serie von Routinen, die Sie unter der Bedingung, daß Sie die nötigen Parameter laden, benützen können.
Sie können die Routinen mit JSR oder JMP aufrufen. Die Register des Mikroprozessors 68Ø9, die Speicher Parameter der Base Page des Monitor, (von $6ØØØ bis $6ØFF) sind im Anhang beschrieben.

PUTC	(E8Ø3)	Gebe ein Zeichen aus.
GETC	(E8Ø6)	Lese Tastatureingabe.
KTST	(E8Ø9)	Schnelle Tastaturtabfrage.
DRAW	(E8ØC)	Zeichne eine Linie.
PLOT	($E8ØF)	Bilde einen Punkt im Grafikmodus ab.
RSCO	($E812)	Steuerung des Schnittstellenadapters.

K7CO	($E815)	Lese/Schreibe Kassette.
GETL	($E818)	Lese Lightpen Position.
LPIN	($E81B)	Lese Lightpen Schalter.
NOTE	($E81E)	Tonerzeugung.
GETP	($E821)	Stelle die Farbe eines Punktes dar.
GETS	($E824)	Lese den Bildschirm.
JOYS	($E827)	Lese Joystick Position.
DKCO	($E82A)	Disk-Kontrolle.
MENU	($E82B)	Kehre zum Monitor zurück.
KBIN	($E83Ø)	Ausgabe Unterbrechungsprogramm.
CHPL	($E833)	Bilde einen Punkt Im Zeichenmodus ab.

Genaue Beschreibung einer Routine.

PUTC ($EØ3)

Diese Routine macht die alphanumerische Bildschirmsteuerung, das Scrolling, die Stelle des Cursors und die Größe und Farbattribute. Alle Argumente sind durch des Register B gezogen.
Definition eines Bildschirmfensters zwischen den Zeilen 1 und 21:
3 Aufrufe sind nötig:

UNTERER ENDPUNKT.

```
ØØØ1Ø              LDB   # $1F       *US-CODE
ØØØ2Ø              BSR   PUTC1
ØØØ3Ø              LDB   # $1Ø+1     *B=OBERER TEIL
ØØØ4Ø              BSR PUTC1
ØØØ5Ø              LDB   # $1Ø+1     *B=UNTERER TEIL
ØØØ6Ø    PUTC1     JMB   PUTC
```

OBERER ENDPUNKT.

```
ØØØ1Ø              LDB   # $1F       *B=US-CODE
ØØØ2Ø              BSR   PUTC1
ØØØ3Ø              LDB   # $2Ø+Ø     *B=OBERER TEIL
ØØØ4Ø              BSR   PUTC1
ØØØ5Ø              LDB   # $2Ø+1     *B=UNTERER TEIL
ØØØ6Ø    PUTC1     JMP PUTC
```

Um den Cursor zu positionieren sind drei Aufrufe nötig:

```
ØØØ1Ø              LDB   # $1F
ØØØ2Ø              BSR   PUTC1
ØØØ3Ø              LDB   # LIGNE     *B=ZEILENNUMMER
ØØØ4Ø              BSR   PUTC2
ØØØ6Ø              LDB   # COLONNE   *B=SPALTENNUMMER
ØØØ7Ø    PUTC2     ORB   # $4Ø       *B=B+$4Ø
ØØØ8Ø    PUTC1     JMP   PUTC
```

Definition von Attributen: 2 Aufrufe sind nötig.
Die Farbe.
Der Befehl ESC ($1B), dann das Farbattribut.

```
ØØØ1Ø                  LDB   #$1B     *B=ESC
ØØØ2Ø                  JSR   PUTC
ØØØ3Ø                  JMP   ----     *(BILDRAHMEN, HINTERGRUND,
                                       VORDERGRUND)

ØØØ1Ø  BILDRAHMEN      LDB   #$6Ø+FARBE
ØØØ2Ø                  JSR   PUTC

ØØØ1Ø  HINTERGRUND     LDB   #$5Ø+FARBE
ØØØ2Ø                  JSR   PUTC

ØØØ3Ø  VORDERGRUND     LDB   #$4Ø+FARBE
ØØØ4Ø                  JSR   PUTC

doppelte Höhe                #>$4D
doppelte Breite              #>$4D
doppelte Höhe und Breite     #>$4F
normale Grösse               #>$4C
Video Invers                 #>$5C
```

Darstellung eines Zeichen.
Jedes Zeichen mit ASCII-Code kann an der Stelle des Cursors abgebildet werden.
Das Zeichen wird ins Register B geladen.
Beispiel:

```
ØØØ1Ø          LDX   #TEXT    *X=TEXTADRESSE
ØØØ2Ø  LOOP     LDB   ,X+      *B=ASCII ZEICHEN
ØØØ3Ø           JSR   PUTC     *DARSTELLUNG
ØØØ4Ø           CMPB  #4       *Wenn B=4 DANN STOP
ØØØ5Ø           BNE   LOOP
ØØØ6Ø           RTS
ØØØ7Ø  TEXT     FCC   /ASSEMBLER/
ØØØ6Ø           FCB   4
```

PLOT ($E8ØF)
Diese Routine bildet einen Punkt im Grafikmodus ab, der durch die X/Y - Koordinaten definiert wird.
VORDERGRUND = FARBE CHDRAW = Ø oder ASCII - Code.
X = C (Ø < C < 32Ø)
Y = L(< Ø < 1 < 2ØØ)
Rufen sie PLOT auf.

DRAW ($E8ØC)
Diese Routine zeichnet eine Linie, die zwischen dem letzten definierten Punkt PLOTXPLOTY und dem Punkt X, Y gezogen wird. Die Farbattribute sind gesetzt Vordergrund=Farbe. CHDRAW=Ø oder ASCII - Code.

X=Kolonne
Y = Linie
Rufen Sie DRAW auf.

KTST ($E8Ø9)
Diese Routine prüft, ob eine Taste gedrückt wurde
C=Carryflag
C=Ø es wurde keine Taste gedrückt
C=1 es wurde eine Taste gedrückt.

GETC ($E8Ø6)
Diese Routine sendet den ASCII-Code eines Zeichens im B-Register zurück. Wenn am definierten Ort kein darstellbares Zeichen vorhanden ist, sendet B eine Ø zurück.

NOTE ($E81E7)
Diese Routine spielt einen Ton, der sich im Register-B befindet.

OKTAVE: OKTAVE	WERT
1	16
2	Ø8
3	Ø4
4	Ø2
5	Ø1

TONDAUER: NOTE	WERT
Ganze Note	96
Punktierte halbe Note	72
Halbe Note	48
Punktierte viertel Note	36
Viertel Note	24
Punktierte achtel Note	18
Achtel Note	12
Punktierte sechzehntel Note	Ø9
Sechzehntel Note	Ø6
Punktierte zweiunddreissigstel Note	Ø5
Zweiunddreissigstel Note	Ø3

TEMPO von 1 bis 255.
KLANGFARBE von Ø bis 5.

JOYS ($E827)
Diese Routine zeigt die Position des Joysticks. Register A muss mit der Nummer des Joysticks geladen sein, dessen Position Sie empfangen wollen (Ø oder 1). Register B sendet eine Zahl zwischen Ø und 8, die die Position des Joysticks gemäss folgender Tabelle bezeichnet.
Ø => Neutral
1 => Nord

2 = > Nordost
3 = > Osten
4 = > Südosten
5 = > Süden
6 = > Südwesten
7 = > Westen
8 = > Nordwesten

Das Carryflag ist gesetzt, wenn der Auslöser (Aktion-Knopf) gedrückt worden ist, andernfalls null.

Diese Liste ist nicht vollständig : wenn Sie alle Möglichkeiten des Monitors wissen wollen, lesen Sie bitte in den entsprechenden Büchern nach.

D-TO7/7Ø Beispielprogramm.

Das folgende Programm ist ein Anwendungsbeispiel der Monitorroutinen. Es zeigt Ihnen auch, wie ein Programm aufgebaut ist. Dieses Programm ermöglicht eine Figur mit Hilfe der 4 Pfeile auf der Tastatur auf dem Bildschirm zu bewegen.

```
ØØ18Ø                ORG    $BØØØ
ØØ19Ø  ECRAN         EQU    $4ØØØ
ØØ2ØØ  CONTROL       EQU    $E7C3
ØØ21Ø  PUTC          EQU    $E8Ø3
ØØ22Ø  GETC          EQU    $E8Ø6
ØØ23Ø  TAIL          RMB    2
ØØ24Ø  TAB1          FCB    1, 2, 3, 4, 5
ØØ25Ø  TABDES        FDB    MOTIF, MOTIF
ØØ26Ø  *
ØØ27Ø  *             DIE DATEN DER GRAFIK
ØØ28Ø  *
ØØ29Ø  MOTIF         FCB    $16, $4
ØØ3ØØ                FCB    $Ø,$CØ,$3F,$E8,$FC,$E8,$Ø,$CØ
ØØ31Ø                FCB    $7,$E8,$CØ,$EF,$3,$EF,$EØ,$E8
ØØ32Ø                FCB    $F8,$EC,$Ø,$C7,$Ø,$C7,$1F,$E9
ØØ33Ø                FCB    $81,$EC,$8Ø,$EF,$Ø,$C7,$1,$E9
ØØ34Ø                FCB    $82,$EC,$4Ø,$EF,$Ø,$C7,$1,$E9
ØØ35Ø                FCB    $84,$EC,$4Ø,$EF,$Ø,$C7,$1,$E9
ØØ36Ø                FCB    $84,$EC,$46,$Ef,$ø,$C7,$1,$E9
ØØ37Ø                FCB    $83,$EC,$89,$EF,$Ø,$C7,$1,$E9
ØØ38Ø                FCB    $8Ø,$EC,$4,$EF,$8E,$EF,$1,$E9
ØØ39Ø                FCB    $87,$EC,$F4,$EF,$92,$EF,$1,$E9
ØØ4ØØ                FCB    $84,$EC,$17,$EF,$22,$EF,$1,$E9
ØØ41Ø                FCB    $84,$EC,$1Ø,$EF,$24,$EF,$1,$E9
ØØ42Ø                FCB    $84,$EC,$7F,$EF,$9C,$EF,$1,$E9
ØØ43Ø                FCB    $84,$EC,$5Ø,$EF,$8Ø,$EF,$1,$E9
ØØ44Ø                FCB    $84,$EC,$53,$EF,$FE,$EF,$1,$E9
ØØ45Ø                FCB    $84,$EC,$52,$EF,$82,$EF,$1,$E9
ØØ46Ø                FCB    $84,$EC,$52,$EF,$82,$EF,$1,$E9
```

```
00470                   FCB     $84,$EC,$52,$EF,$82,$EF,$1,$E9
00480                   FCB     $84,$EC,$52,$EF,$82,$EF,$1,$E9
00490                   FCB     $FC,$EC,$52,$EF,$82,$EF,$1F,$E9
00500                   FCB     $3,$E0,$52,$EF,$83,$EF,$E0,$E8
00510                   FCB     $0,$C0,$73,$EF,$83,$EF,$0,$C0
00520                   *
00530   *               VERSCHIEBUNG
00540   *
00550   MOV     PSHS    B               *WENN KEINE BEWEGUNG
00560           TSTA
00570           BEQ     AFD1
00580           LDB     1,Y
00590           CMPA    =11
00600           BNE     AF3             *WENN UNTEN
00610           TSTB
00620           BEQ     AF3
00630           DECB
00640   AF3     CMPA    #10
00650           BNE     AF4             *WENN OBEN
00660           CMPB    #44
00670           BEQ     AF4
00680           INCB
00690   AF4     STB     1,Y
00700           LDB     2,Y
00710           CMPA    #8
00720           BNE     AF5             *WENN LINKS
00730           DECB
00740           BPL     AF5
00750           LDB     #39
00760   AF5     CMPA    #9
00770           BNE     AF2             *WENN RECHTS
00780           INCB
00790           CMPB    #40
00800           BNE     AF2
00810           CLRB
00820   AF2     STB     2,Y
00830   AFD1    LDU     1,Y
00840           PULS    B, PCR
00850   *
00860   *               MOTIVDARSTELLUNG
00870   *
00880   SET     PSHS    X
00890           LDD     1,Y
00900           PSHS    D
00910           CLRB
00920   LOP1    PSHS    B,U
```

```
00930           CLRB
00940   SET1    LDA     CONTROL     *EINSTELLUNG DES
00950           ORA     #1          *ZEICHENSPEICHERS
00960           STA     CONTROL
00970           LDA     ,X+         *DARSTELLUNG DES
00980           ORA     ,U          *ZEICHENSPEICHERS
00990           STA     ,U
01000           DEC     CONTROL     *FARBSPEICHERS
01010           LDA     ,X+         *DARSTELLUNG
                                    *DES FARBSPEICHERS
01020           STA     ,U+
01030           INCB
01040           CMPB    TAIL+1
01050           BNE     SET1        *VERTIKALE SCHLEIFE
01060           PULS    B,U
01070           LEAU    40,U
01080           INCB
01090           CMPB    TAIL
01100           BNE     LOP1        *HORIZONTALE SCHLEIFE
01110           PULS    X, D, PCR
01120   *
01130   *       LÖSCHEN DES BILDES
01140   *
01150   RSET    PSHS    X
01160           LDB     TAIL
01170   LOP 11  PSHS    B,U
01180           LDB     TAIL+1
01190   LOOOP   LDA     CONTROL
01200           ANDA    #$FE        *EINSTELLUNG
                                    *DES ZEICHENSPEICHERS
01210           STA     CONTROL     *FARBE
01220           LDA     #$C0        *AUSLÖSCHEN
01230           STA     ,U
01240           INC     CONTROL     *ZEICHENSPEICHER
01250           CLR     ,U+         *AUSLÖSCHEN
01260           DECB
01270           BNE     LOOOP       *VERTIKALE SCHLEIFE
01280           PULS    B,U
01290           LEAU    40,U
01300           DECB
01310           BNE     LOP11       *HORIZONTALE SCHLEIFE
01320           PULS    X,PCR
01340   *       DARSTELLUNG UND VERSCHIEBUNG
01350   *
01360   AFOB    LDY     #TAB1       *ZEIGT DIE KOORDINATEN
01370           JSR     DEFIG       *X ZEIGT AUFDIE GRAFIK
```

```
Ø138Ø            JSR    POTU      *U ZEIGT AUF DEM BILDSCHIRM
Ø139Ø            LDD    ,X++
Ø14ØØ            STD    TAIL      *D = GRÖSSE DES BILDES
Ø141Ø            BSR    RSET      *AUSLÖSCHEN
Ø142Ø  AF1Ø      LDA    3,Y       *A = VERSCHIEBUNG
Ø143Ø            BEQ    AF9
Ø144Ø            JSR    MOV       *VERSCHIEBUNG
Ø145Ø  AF9       JSR    POTU      *U ZEIGT AUF DEM BILDSCHIRM
Ø1455                              *+ VERSCHIEBUNG
Ø146Ø            JSR    SET       *DARSTELLUNG
Ø147Ø  AF1       RTS
Ø148Ø  *
Ø149Ø  *         B = NUMMER DES BILDES
Ø15ØØ  *
Ø151Ø  DEFIG     LDB    ,Y
Ø152Ø            DECB
Ø153Ø            LDX    #TABDES   *X ZEIGT AUF DIE TABELLE
Ø1535                              *DES BILDES
Ø154Ø            ASLB
Ø155Ø            LDX    B,X       *X ZEIGT AUF DEM BILDSCHIRM
Ø156Ø            RTS
Ø157Ø  *         BERECHNUNG DER BILDSCHIRMADRESSE
Ø158Ø  POTU      PSHS   A,B
Ø159Ø            LDA    1,Y
Ø16ØØ            LDB    #16Ø      *16Ø=4*4Ø
Ø161Ø            MUL
Ø162Ø            ADDD   #BILDSCHIRM
Ø163Ø            TFR    D,U       *VERTIKALE ADDITION
Ø164Ø  AFFT      LDA    2,Y
Ø165Ø            LEAU   A,U       *HORIZONTALE ADDITION
Ø166Ø            PULS   B,A,PCR
Ø167Ø  *
Ø168Ø  *         ANFANG DES PROGRAMMS
Ø169Ø  *
Ø17ØØ  START     LDX    #RAZ
Ø171Ø  STA2      LDB    ,X+       *LÖSCHEN DES BILDSCHIRM
Ø172Ø            JSR    PUTC
Ø173Ø            CMPB   #1
Ø174Ø            BNE    STA2
Ø175Ø  DEBUT     JSR    GETC      *TASTATUR ANGABE
Ø176Ø            STB    TAB1+3    *SPEICHERN
Ø1765                              *IN DEN KOORDINATEN
Ø177Ø            JSR    AFOB      *DARSTELLUNG
Ø178Ø            LDX    #$13ØØ
Ø179Ø  LOP 1Ø    LEAX   -1,X      *ABWARTEN
Ø18ØØ            BNE    LOP1Ø
```

```
Ø181Ø                BRA     DEBUT       *SCHLEIFE
Ø182Ø   *
Ø183Ø   RAZ          FCB     $1F,$2Ø,$2Ø,$1F,$12,$14,$1B,$6Ø,$1B,$5Ø,$1B
Ø1835                FCB     $4Ø,12,1
Ø184Ø                END     START
Ø185Ø   *
Ø186Ø   *
Ø187Ø   *            ASSEMBLER BEFEHLE
Ø188Ø   *
Ø189Ø   *
Ø19ØØ    *           TIPPEN ZONE $BØØØ/ENTER/
Ø191Ø   *                       ASS RM      /ENTER/
Ø192Ø   *                       EXEC        /ENTER/
Ø193Ø   *            ODER
Ø194Ø   *            TIPPEN ASS : ESSAI/ENTER/
Ø195Ø   *            Der Programm-Rekorder ist zum Speichern bereit.
Ø196Ø   *            In diesem Fall wird das Programm durch
Ø197Ø   *            "LOADM" geladen.
```

E – Die wichtigen Adressen.

DIE BILDSCHIRMADRESSEN

$4ØØØ	STAD	Erstes Byte
$5F4Ø	ENDAD	Letztes Byte
$4118	CURSOR	Herkunft der 1. Kolonne
$4258	TELCUR	Herkunft der 2. Kolonne
$414Ø	ORGROL	Herkunft des ROLLUP
$428Ø	ORGBL	Herkunft des ROLLUP mit doppelter Höhe
$5F18	DERCUR	Kursor auf der letzten Zeile

EINGANG - AUSGANGADRESSEN

6846

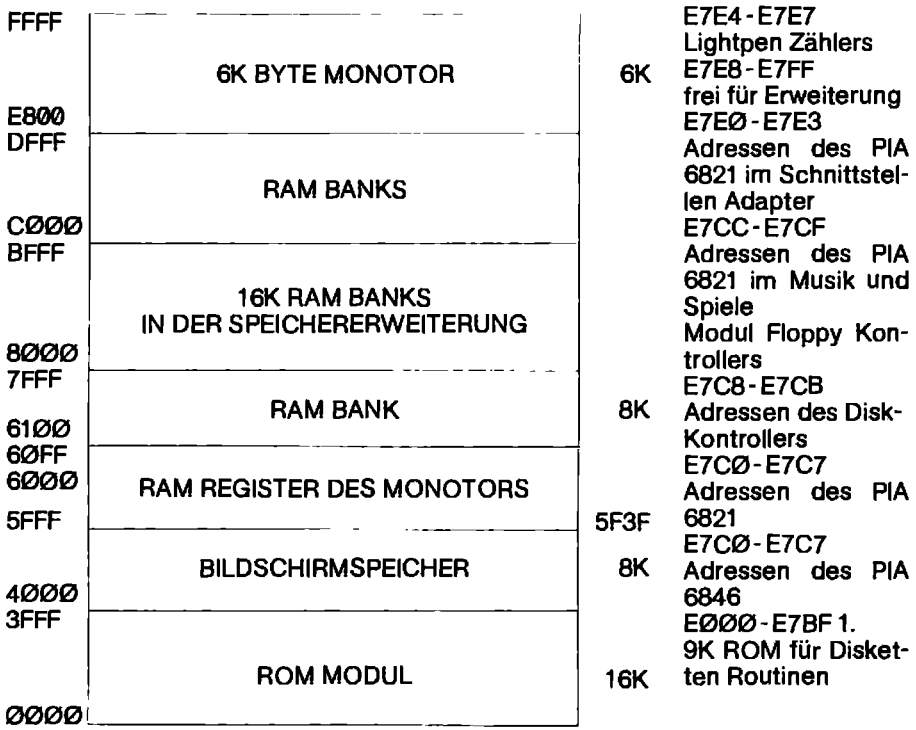
$E7CØ	CSR	Composite Status Register
$E7C1	CRC	Kontrollregister Port P
$E7C2	DDRC	Richtungsregister Port P
$E7C3	PRC	Datenregister Port P
$E7C5	TCR	TIMER CONTROL REGISTER
$E7C6	TMSB	TIMER M.S.B.
$E7C7	TLSB	TIMER L.S.B.

PIA 6821 System

$E7C8	PRA	Datenregister Port A
$E7C9	PRA	Datenregister Port B BO-B2 Multiplexing der Tastatur B3-B7 Auswahl der RAM-Banks
$E7CA	CRA	Kontrollregister Port A

$E7CB	CRB	Kontrollregister Port B CB1 Eingang Lightpen- Impulse CB2 Mischbild schalten
		SPIELE/Musik PIA 6821
$E7CC	PRA1	Datenregister Port A1
$E7CD	ORB1	Datenregister Port B1
$E7CE	CRA1	Kontrollregister Port A1
$E7CF	CRB1	Kontrollregister Port B1
		PIA 6821 im Schnittstellenadapter
$E7EØ	PRA2	Datenregister Port A2
$E7E1	PRB2	Datenregister Port B2
$E7E2	CRA2	Kontrollregister Port A2
$E7E3	CRB2	Kontrollregister Port B2

F – Die Speicheraufteilung.



G - Vom Monitor belegte RAM-Register (Zeropage).
Die nun folgenden Adressen sind im Hexadezimalsystem angeben:

$6ØØØ		
$6Ø18	TERMIN	Zeilenende-Tabelle.
$6Ø19	STATUS	diverse Flags: B7=Semigrafik, B6=Fast-Scroll-Modus, B5=User-Interruptprogramm, B4=Punktgragik ohne Farbe, B3=Flag der Tastaturabfrage, B2=Cursor ein/aus, B1=reserviert, BØ=Flag für Autorepeat der Tastatur.
$6Ø1A		
$6Ø1B	TABPT	Zeiger der Zeilenende-Tabelle.
$6Ø1B	RANG	momentane logische Zeilennummer.
$6Ø1C		
$6Ø1D	TOPTAB	Zeiger auf den Anfang der Zeilenende-Tabelle.
$6Ø1D	TOPRAN	Erste logische Zeile des Bildschirmfensters.
$6Ø1E		
$6Ø1F	BOTTAB	Zeiger auf das Ende der Zeilenende-Tabelle.
$6Ø1F	BOTRAN	Letzte logische Zeile des Bildschirmfensters.
$6Ø2Ø	COLN	momentane logische Spalte.
$6Ø21		
$6Ø22	IRQPT	Adresse der IRQ-Interruptroutine.
$6Ø23		
$6Ø24	FIR QPT	Adresse der FIRQ-Interruptroutine.
$6Ø25		
$6Ø26	NMIPT	Adresse der NMI-Interruptroutine.
$6Ø27		
$6Ø28	TIMEPT	Adresse der Timer-Interruptroutine.
$6Ø29	K7.OPC	OP-Code für Rekorderfunktionen.
$6Ø2A	K7.STA	Statusbyte des Rekorders.
$6Ø2B	RS OPC	OP-Code für Kommunikations-Interface.
$6Ø2C	RS STA	Statusbyte des Kommunikations-Interface.
$6Ø2D		
$6Ø2E	USERAF	Anfang der User-Zeichentabelle.
$6Ø2F		
$6Ø3Ø	SWI1	Adresse der SWI-Interruptroutine.
$6Ø31		
$6Ø32	TEMPO	Abspielgeschwindigkeit einer Melodie.
$6Ø33		
$6Ø34	DUREE	Relative Länge einer Note (1-96).
$6Ø35	TIMBRE	der Note (Ø=DAUERTON).
$6Ø36		
$6Ø37	OKTAVE	Oktave (1, 2, 4, 8, oder 16).
$6Ø38	FORME	Farbcode für Grafikmodus (-8 bis + 15).
$6Ø39	ATRANG	Flags zur Bildschirmsteuerung: B7= Scroll-Flag, B6=reserviert, B5= reserviert, B4=reserviert,

		B3=reserviert, B2=reserviert, B1=einfache oder doppelte Zeichenbreite, BØ=einfache oder doppelte Zeichenhöhe.
$6Ø3A	ATRSCR	Flags für den Fullscreen-Modus. B7=Fullscreen-Hintergrundflag, B6=Fullscreen Vordergrundflag, B5=reserviert, B4=reserviert, B3=reserviert, B2=reserviert, B1=einfache oder doppelte Zeichenbreite, BØ=einfache oder doppelte Zeichenhöhe.
$6Ø3B	COLOUR	Aktuelle Vorder-/Hintergrundfarbe. (Zeichen).
$6Ø3C	TELETL	FFH=Page-Modus (kein Scrollen).
$6Ø3D		
$6Ø3E	PLOTX	Horiz. Position des letzten Grafikpunktes.
$6Ø3F		
$6Ø4Ø	PLOTY	Vert. Position des letzten Grafikpunktes.
$6Ø41	CHDRAW	Parameter für DRAW-Funktion: ASCII-Code=Zeichenmodus, Ø=Grafikmodus.
$6Ø42	CURSFL	Flag, verhindert unzulässige Zeilenverbindung.
$6Ø43	COPCHR	FFH=BS und HT kopieren des momentanen Zeichens.
$6Ø44		
$6Ø45	BAUDS	Übertragungsrate der seriellen Schnittstelle.
$6Ø46	NOMBRE	Wortlänge der seriellen Schnittstelle.
$6Ø47	GRCODE	Druckercode für Umschaltung in Grafik-Modus.
$6Ø48	DK. OPC	OP-Code für die Disketten-Funktionen.
$6Ø49	DK. DRV	Nummer des angesprochenen Laufwerkes.
$6Ø4A		
$6Ø4B	DK. TRK	Nummer der angesprochenen Spur.
$6Ø4C	DK. SEC	Nummer des angesprochenen Sektors.
$6Ø4D	DK. NUM	Faktor der Sektorenverflechtung.
$6Ø4E	DK. STA	Statusbyte des Disk-Controllers.
$6Ø4F		
$6Ø5Ø	DK. BUF	Anfangsadresse des Disk-Buffers.
$6Ø51		
$6Ø52	TRACKØ	Position des Schreib-/Lesekopfes in Drive Ø.
$6Ø53		
$6Ø54	TRACK1	Position des Schreib-/Lesekopfes in Drive 1.
$6Ø55		
$6Ø56	TRACK2	Position des Schreib-/Lesekopfes in Drive 2.
$6Ø57		
$6Ø58	TRACK 3	Position des Schreib-/Lesekopfes in Drive 3.
$6Ø59	SEQUCE	Flag zur Kennung der Bildschirm-Steuersequenz.
$6Ø5A		
$6Ø5B	SCRPT	Aktuelle Cursorposition.
$6Ø5C	SAVCOL	Sicherung des aktuellen Farbcodes.
$6Ø5D	ASCII	Code des letzten dargestellten Zeichens.
$6Ø5E	KEY	Code der letzten betätigten Taste.

$6Ø5F	CMPTKB	Zähler des Tastatur-Autorepeats.
$6Ø6Ø		
$6Ø61	STADR	Startadresse des aktuellen Bildschirm-Speichers.
$6Ø62		
$6Ø63	ENDDR	Endadresse des aktuellen Bildschirm-Speichers.
$6Ø64	TCRSAV	Sicherung aktueller Timer-Zustand.
$6Ø65		
$6Ø66	TCTSAV	Sicherung aktueller Zählerstand des Timers.
$6Ø67	LATCLV	Verzögerungsfaktor des Autorepeat.
$6Ø68		
$6Ø69	SAVATR	Sicherung der aktuellen Bildschimattribute.
$6Ø6A	US1	Flag für US-Sequenz.
$6Ø6B	COMPT	Zähler für Zeichenwiederholung.
$6Ø6C		
$6Ø6D	TEMP	Temporäres Register für Datenübertragung.
$6Ø6E		
$6Ø6F	SAVEST	Sicherung des Stack-Pointers.
$6Ø7Ø	ACCENT	Flag für Akzent-Sequenz.
$6Ø71	SS2GET	Flag zur Darstellung akzentuierter Zeichen.
$6Ø72	SS3GET	Flag zur Darstellung akzentuierter Zeichen.
$6Ø73	BUZZ	Flag für den Signalton-Summer reserviert.
$6Ø74		
$6Ø75	EFCMPT	Zähler für gelöschte Zeichen (DEL-Taste).
$6Ø76		
$6Ø77	BLOCZ	Ständig Ø für Initialisierungen.
$6Ø78	SCROLS	Flag für Soft-Scroll.
$6Ø79		
$6Ø7E	TABCHX	Pointer der Menü-Auswahltabelle.
$6Ø7F	RUNFLG	Flag für Menüposition 2 (Auto-Start).
$6Ø8Ø	DKFLG	Flag für Disk-Controller.
$6Ø81		Stack-System.
$6ØCD		
$6ØCE	PTCLAV	Pointer der Tastaturtabelle.
$6ØCF		
$6ØDØ	PTGENE	Pointer der Standardzeichentabelle.
$6ØD1	APPLIC	Prüfsumme des laufenden Anwendungsprogrammes.
$6ØD2	DECALG	Einstellung des Lightpen.
$6ØD3		
$6ØEA	LPBUFF	Puffer für Lightpen-Lesedaten.
$6ØEB		
$6ØFD		Reserviert.
$6ØFE		
$6ØFF	TSTRST	Flag für Kalt-oder Warmstart.

Einige übersetzungen.

Instruction illégale bedeutet Illegal instruction

Mauvaise ponctuation	bedeutet	Syntax error
Adressage illégal	bedeutet	Illegal adressing
Dépassement de capacité	bedeutet	Operand too large
Label trop loin	bedeutet	Branch out of range
Label incorrect	bedeutet	Bad label
Multi label	bedeutet	Multiple defined label
Table des labels saturée	bedeutet	Symbols table full
Label introuvable	bedeutet	Undefined label
Assemblage en mémoire réservée	bedeutet	Bad memory assembling
Mémoire pleine	bedeutet	Memory full
Ligne occupée	bedeutet	Line already exist
Comment ?	bedeutet	What ?
Disquette protégée	bedeutet	Disk protected
Lecteur pas prêt	bedeutet	Disk not ready
Disquette illisible	bedeutet	Reading disk impossible
Erreur de parité	bedeutet	Parity error
Disquette pleine	bedeutet	Disk full
Discordance de type	bedeutet	Type mismatch
Fichier incomplet	bedeutet	File format error
Disquette déjà initialisée	bedeutet	Disk already formatted
Fichier déjà ouvert	bedeutet	File already opened
Mémoire en ROM	bedeutet	Memory in ROM
Disque	bedeutet	Disk
Reste	bedeutet	Remain
Bloc(s)	bedeutet	Blok(s)
Libre	bedeutet	Free
Nom du fichier incorrect	bedeutet	Bad file name
Fichier introuvable	bedeutet	File not found
Plus de données	bedeutet	No more data
Fichier fermé	bedeutet	File closed

III - BENUTZUNG EINES MO5 E

A - Wie das Programm geladen wird.

Dieses Programm benötigt folgende Mindestausstattung:
Zentraleinheit MO5 E,
Programm-Rekorder,
Joysticks und Fernsehgerät.

Achten Sie darauf, daß alle Geräte richtig angeschlossen und eingeschaltet sind.

1. Verbinden Sie den Programm-Rekorder mit der Seite der Zentraleinheit.
2. Nach dem Einschalten wird das Menü angezeigt.
3. Legen Sie die Kassette in den Programm-Rekorder mit der MO5 E Seite ein, drücken Sie auf lesen >. Es ist völlig normal, daß der Rekorder nicht startet.
4. Wählen Sie den Punkt 1, um in Basic zu gelangen. Geben Sie **LOADM** ein.

Der Programm-Rekorder startet automatisch, das Programm wird geladen und der Programm-Rekorder schaltet sich automatisch wieder aus.

B - Besonderheiten der MO5 E Version.

Der ASSEMBLER besetzt 11K Byte. Das Ursprungsprogramm ist von der Adresse $5Ø2Ø ab gelegt. Wenn Sie während eines Ladevorgangs mit EXEC oder LOADM, den Fehler: "Chargement sur le sytème" (Ladung auf System) auslösen, bedenken Sie daß sich nach dem Befehl DEL=,* kein Ursprungsprogramm mehr im Speicher befindet. Der Reset darf nur im dringendsten Fall angewendet werden. Er führt zum Verlust des Ursprungsprogramms, das sich im Speicher befindet und nur ein EXEC &H2C84 kann es wieder starten.

C - Einsprungsadressen MO5 E.

Ihr MO5 E verfügt im ROM über eine Serie von Routinen, die Sie unter der Bedingung, daß Sie die nötigen Parameter laden, benützen können. Den Zugang zu einer Monitor MO5 E Routine erreichen Sie durch die Anweisung "SWI" gefolgt von einem Code auf einem Byte. Zu einem gleichen S/P gehören zwei unterschiedliche Codes mit 7 Bits, diese 7 schwachen Bits bilden den Routine-Code. Das Bit 7 hat folgende bedeutung:
Wenn er Ø ist, ein simuliertes "JSR",
Wenn er 1 ist, ein simuliertes "JMP".

		MONITOR CODE
PUTC	($Ø2)	Gebe ein Zeichen aus.
GETC	($ØA)	Lese Tastatureingabe.
KTST	($ØC)	Schnelle Tastaturabfrage.
DRAW	($ØE)	Zeichne eine Linie.
PLOT	($Ø1Ø)	Bilde einen Punkt im Grafikmodus ab.
RSCO	($24)	Steurung des Schnittstellenadapters.

K7CO	($2Ø)	Lese/Schreibe Kassette.
GETL	($18)	Lese Lightpen Position.
LPIN	($16)	Lese Lightpen Schalter.
NOTE	($1E)	Tonerzeugung.
GETP	($14)	Stelle die Farbe eines Punktes dar.
GETS	($1A)	Lese den Bildschirm.
JOYS	($1C)	Lese Joystick Position.
MENU	($ØØ)	Kehre zum Monitor zurück.
CHPL	($12)	Bilde einen Punkt im Zeichenmodus ab.

Genaue Beschreibung einer Routine.

PUTC ($Ø2)

Diese Routine macht die alphanumerische Bildschirmsteuerung, das Scrolling, die Stelle des Cursors und die Größe und Farbattribute. Alle Argumente sind durch des Register B gezogen.
Definition eines Bildschirmfensters zwischen den Zeilen 1 und 21 : 3 Aufrufe sind nötig :

UNTERER ENDPUNKT.

```
ØØØ1Ø          LDB   #$1F        *US-CODE
ØØØ2Ø          BSR   PUTC1
ØØØ3Ø          LDB   #$1Ø+2      *B=OBERER TEIL
ØØØ4Ø          BSR   PUTC1
ØØØ5Ø          LDB   #$1Ø+1      *B=UNTERER TEIL
ØØØ6Ø PUTC1    SWI
ØØØ7Ø          FCB   $2+$8Ø
```

OBERER ENDPUNKT.

```
ØØØ1Ø          LDB   #$1F        *B=US-CODE
ØØØ2Ø          BSR   PUTC1
ØØØ3Ø          LDB   #$2Ø+Ø      *B=OBERER TEIL
ØØØ4Ø          BSR   PUTC1
ØØØ5Ø          LDB   #$2Ø+1      *B=UNTERER TEIL
ØØØ6Ø PUTC1    SWI
ØØØ7Ø          FCB $2+8Ø
```

Um den Cursor zu positionieren sind drei Aufrufe nötig.

```
ØØØ1Ø          LDB   #$1F        B=US
ØØØ2Ø          BSR   PUTC1
ØØØ3Ø          LDB   # LIGNE     *B=ZEILENNUMMER
ØØØ4Ø          BSR   PUTC2
ØØØ6Ø          LDB   # COLONNE   *B=SPALTENNUMMER
ØØØ7Ø PUTC2    ORB   #$4Ø        *B=B+$4Ø
ØØØ7Ø PUTC2    ORB   #4Ø         *B=B+$4Ø
ØØØ8Ø PUTC1    SWI
ØØØ9Ø          FCB   $2+$8Ø
```

Definition von Attributen: 2 Aufrufe sind nötig.
Die Farbe.
Der Befehl ESC ($13), dann das Farbattribut.

```
00010                   LDB   #$1B  *B=ESC
00020                   SWI
00025                   FCB   $2
00030                   JMP   ----  *(BILDRAHMEN, HINTERGRUND,
                                     VORDERGRUND)
00010  BILDRAHMEN       LDB   #$60+FARBE
00020                   SWI
00030                   FCB   $2+$80
00010  HINTERGRUND      LDB   #$50+FARBE
00020                   SWI
00030                   FCB   $2+$80
00030  VORDERGRUND      LDB   #$40+FARBE
00040                   SWI
00050                   FCB   $2+$80
```

doppelte Höhe => $4D
doppelte Breite => $4E
doppelte Höhe und Breite => $4F
normale Grösse => $4C
Video Invers => $5C

Darstellung eines Zeichens.
Jedes Zeichen mit ASCII-Code kann an der Stelle des Cursors abgebildet werden.
Das Zeichen wird ins Register B geladen.
Beispiel:

```
00010                   LDX   #TEXT *X=TEXTADRESSE
00020  LOOP             LDB   ,X+   *B=ASCII ZEICHEN
00030                   SWI
00035                   FCB   $2
00040                   CMPB  #4    *WENN B=4 DANN STOP
00050                   BNE   LOOP
00060                   RTS
00070  TEXT             FCC   /ASSEMBLER/
00080                   FCB   4
```

PLOT ($10)
Diese Routine bildet einen Punkt im Grafikmodus ab, der durch di X/Y-Koordinaten definiert wird.
VORDERGRUND=Farbe CHDRAW=0 oder ASCII-Code
X=C(0 < C < 320)
Y=L(< 0 < 1 < 200)
Rufen Sie PLOT auf.

DRAW ($ØE)
Diese Routine zeichnet eine Linie, die zwischen dem letzten definierten Punkt PLOTXPLOTY und dem Punkt X, Y gezogen wird. Die Farbattribute sind gesetzt.
VORDERGRUND=Farbe CHDRAW=Ø oder ASCII-Code
X=Kolonne
Y=Linie
Rufen Sie DRAW auf.

KTST ($ØC)
Diese Routine prüft, ob eine Taste gedrückt wurde.
C=Carryflag.
C=Ø es wurde keine Taste gedrückt
C=1 es wurde eine Taste gedrückt.

GETC ($ØA)
Diese Routine sendet den ASCII-Code eines Zeichens im B-Register zurück. Wenn am definierten Ort kein darstellbares Zeichen vorhanden ist, sendet B eine Ø zurück.

NOTE ($1E)
Diese Routine spielt einen Ton, der sich im Register -B befindet.

OKTAVE: OKTAVE	WERT
1	16
2	Ø8
3	Ø4
4	Ø2
5	Ø1

TONDAUER: NOTE	WERT
GANZE NOTE	96
PUNKTIERTE HALBE NOTE	72
HALBE NOTE	48
PUNKTIERTE VIERTEL NOTE	36
VIERTEL NOTE	24
PUNKTIERTE ACHTEL NOTE	18
ACHTEL NOTE	12
PUNKTIERTE SECHZEHNTEL NOTE	Ø9
SECHZEHNTEL NOTE	Ø6
PUNKTIERTE ZWEIUNDDREISSIGSTEL NOTE	Ø5
ZWEIUNDDREISSIGSTEL NOTE	Ø3

TEMPO von 1 bis 255.
KLANGFARBE von Ø bis 5.

JOYS ($1C)
Diese Routine zeigt die Position des Joysticks. Register A muß mit der Nummer des Joysticks geladen sein, dessen Position Sie empfangen wollen (Ø oder 1). Register B sendet eine Zahl zwischen Ø und 8, die die Position des Joysticks gemäss folgender Tabelle bezeichnet.
Ø=> Neutral
1=> Nord
2=> Nordost
3=> Osten
4=> Südosten
5=> Süden
6=> Südwesten
7=> Westen
8=> Nordwesten
Das Carryflag ist gesetzt, wenn der Auslöser (Aktion-Knopf) gedrückt worden ist, andernfalls null.

Diese Liste ist nicht vollständig : wenn Sie alle Möglichkeiten des Monitors wissen wollen lesen Sie bitte in den entsprechenden Büchern nach.

D - MO5 E Beispielprogramm.
Das folgende Programm ist ein Anwendungsbeispiel der Monitorroutinen. Es zeigt Ihnen auch, wie ein Programm aufgebaut ist. Dieses Programm ermöglicht eine Figur mit Hilfe der 4 Pfeile auf der Tastatur auf dem Bildschirm zu bewegen.

```
ØØ2ØØ *
ØØ21Ø                  ORG     $9ØØØ
ØØ22Ø  ECRAN           EQU     $ØØØØ
ØØ23Ø  CONTROL         EQU     $A7CØ
ØØ24Ø  GETC            EQU     $A
ØØ25Ø  PUTC            EQU     $2
ØØ26Ø  TAIL            RMB     2
ØØ27Ø  TAB1            FCB     1, 2, 3, 4, 5
ØØ28Ø  TABDES          FDB     MOTIF, MOTIF
ØØ29Ø  *
ØØ3ØØ  *                       DIE DATEN DER GRAFIK
ØØ31Ø  *
ØØ32Ø  MOTIF           FCB     $16,$4
ØØ33Ø                  FCB     $Ø,$Ø,$3F,$5Ø,$FC,$5Ø,$Ø,$Ø
ØØ34Ø                  FCB     $7,$5Ø,$CØ,$57,$3,$57,$EØ,$5Ø
ØØ35Ø                  FCB     $F8,$54,$Ø,$7,$Ø,$7,$1F,$51,
ØØ36Ø                  FCB     $81,$54,$8Ø,$57,$Ø,$7,$1,$51
ØØ37Ø                  FCB     $82,$54,$4Ø,$57,$Ø,$7,$1,$51
ØØ38Ø                  FCB     $84,$54,$4Ø,$57,$Ø,$7,$1,$51
ØØ39Ø                  FCB     $84,$54,$46,$57,$Ø,$7,$1,$51
ØØ4ØØ                  FCB     $83,$54,$89,$57,$Ø,$7,$1,$51
```

```
00410          FCB    $80,$54,$4,$57,$8E,$57,$1,$51
00420          FCB    $87,$54,$F4,$57,$92,$57,$1,$51
00430          FCB    $84,$54,$17,$57,$22,$57,$1,$51
00440          FCB    $84,$54,$10,$57,$24,$57,$1,$51
00450          FCB    $84,$54,$7F,$57,$9C,$57,$1,$51
00460          FCB    $84,$54,$50,$57,$80,$57,$1,$51
00470          FCB    $84,$54,$53,$57,$FE,$57,$1,$51
00480          FCB    $84,$54,$52,$57,$82,$57,$1,$51
00490          FCB    $84,$54,$52,$57,$82,$57,$1,$51
00500          FCB    $84,$54,$52,$57,$82,$57,$1,$51
00510          FCB    $84,$54,$52,$57,$82,$57,$1,$51
00520          FCB    $FC,$54,$52,$57,$82,$57,$1F,$51
00530          FCB    $3,$40,$52,$57,$83,$57,$E0,$50
00540          FCB    $0,$0,$73,$57,$83,$57,$0,$0
00550  *
00560  *       VERSCHIEBUNG
00570  *
00580  MOV     PSHS   B          *WENN KEINE BEWEGUNG
00590          TSTA
00600          BEQ    AFD1
00610          LDB    1,Y
00620          CMPA   #11
00630          BNE    AF3        *WENN UNTEN
00640          TSTB
00650          BEQ    AF3
00660          DECB
00670  AF3     CMPA   #10
00680          BNE    AF4        *WENN OBEN
00690          CMPB   #44
00700          BEQ    AF4
00710          INCB
00720  AF4     STB    1,Y
00730          LDB    2,Y
00740          CMPA   #8
00750          BNE    AF5        *WENN LINKS
00760          DECB
00770          BPL    AF5
00780          LDB    #39
00790  AF5     CMPA   #9
00800          BNE    AF2        *WENN RECHTS
00810          INCB
00820          CMPB   #40
00830          BNE    AF2
00840          CLRB
00850  AF2     STB    2,Y
00860  AFD1    LDU    1,Y
```

```
00870           PULS    B,PCR
00880 *
00890 *         MOTIVDARSTELLUNG
00900 *
00010 SET       PSHS    X
00920           LDD     1,Y
00930           PSHS    D
00940           CLRB
00950 LOP1      PSHS    B,U
00960           CLRB
00970 SET1      LDA     CONTROL   *EINSTELLUNG
00980           ORA     #1        *DES ZEICHENSPEICHERS
00990           STA     CONTROL
01000           LDA     ,X+       *DARSTELLUNG DES
01010           ORA     ,U        *ZEICHENSPEICHERS
01020           STA     ,U
01030           DEC     CONTROL   *FARBSPEICHER
01040           LDA     ,X+       *DARSTELLUNG
                                  *DES FARBSPEICHERS
01050           STA     ,U+
01060           INCB
01070           CMPB    TAIL+1
01060           BNE     SET1      *VERTIKALE SCHLEIFE
01090           PULS    B,U
01100           LEAU    40,U
01110           INCB
01120           CMPB    TAIL
01130           BNE     LOP1      *HORIZONTALE SCHLEIFE
01140           PULS    X,D,PCR
01150 *
01160 *         LÖSCHEN DES BILDES
01170 *
01180 RSET      PSHS    X
01190           LDB     TAIL
01200 LOP11     PSHS    B,U
01210           LDB     TAIL+1
01220 LOOOP     LDA     CONTROL
01230           ANDA    #$FE      *EINSTELLUNG
                                  *DES ZEICHENSPEICHERS.
01240           STA     CONTROL   *FARBE
01250           CLR     ,U
01260           INC     CONTROL   *ZEICHENSPEICHER
01270           CLR     ,U+       *AUSLÖSCHEN
01280           DECB
01290           BNE     LOOOP     *VERTIKALE SCHLEIFE
01300           PULS    B,U
```

```
Ø131Ø          LEAU   4Ø,U
Ø132Ø          DECB
Ø133Ø          BNE    LOP11       *HORIZONTALE SCHLEIFE
Ø134Ø          PULS   X,PCR
Ø136Ø  *       DARSTELLUNG UND VERSCHIEBUNG
Ø137Ø  *
Ø138Ø  AFOB    LDY    #TAB1       *ZEIGT DIE KORRDINATEN
Ø139Ø          JSR    DEFIG       *X ZEIGT DEN GRAFIK
Ø14ØØ          JSR    POTU        *U ZEIGT AUF DEN BILDSCHIRM
Ø141Ø          LDD    ,X++
Ø142Ø          STD    TAIL        *D=GRÖSSE DES BILDES
Ø143Ø          BSR    RSET        *AUSLÖSCHEN
Ø144Ø  AF1Ø    LDA    3,Y         *A=VERSCHIEBUNG
Ø145Ø          BEQ    AF9
Ø146Ø          JSR    MOV         *VERSCHIEBUNG
Ø147Ø  AF9     JSR    POTU        *U ZEIGT AUF DEN BILDSCHIRM
Ø1475                             *VERSCHIEBUNG
Ø148Ø          JSR    SET         *DARSTELLUNG
Ø149Ø  AF1     RTS
Ø15ØØ  *
Ø151Ø  *       B=NUMMER DES BILDES
Ø152Ø  *
Ø153Ø  DEFIG   LDB    ,Y
Ø154Ø          DECB
Ø155Ø          LDX    #TABDES     *X ZEIGT AUF DIE TABELLE
Ø1555                             *DES BILDES
Ø156Ø          ASLB
Ø157Ø          LDX    B,X         *X ZEIGT AUF DEN BILDSCHIRM
Ø158Ø          RTS
Ø159Ø  *       BERECHNUNG DER BILDSCHIRMADRESSE
Ø16ØØ  POTU    PSHS   A,B
Ø161Ø          LDA    1,Y
Ø162Ø          LDB    #16Ø        *16Ø=4*4Ø
Ø163Ø          MUL
Ø164Ø          ADDD   #BILDSCHIRM
Ø165Ø          TFR    D,U         *VERTIKALE ADDITION
Ø166Ø  AFFT    LDA    2,Y
Ø167Ø          LEAU   A,U         *HORIZONTALE ADDITION
Ø168Ø          PULS   B,A,PCR
Ø169Ø  *
Ø17ØØ  *       ANFANG DES PROGRAMMS
Ø171Ø  *
Ø172Ø  START   LDX    #RAZ
Ø173Ø  STA2    LDB    ,X+         *LÖSCHEN DES BILDSCHIRM
Ø174Ø          SWI
Ø175Ø          FCB    PUTC
```

```
01760           CMPB    #4
01770           BNE     STA2
01780  DEBUT    SWI             *TASTATUR ANGABE
01790           FCB     GETC
01800           STB     TAB1+3  *SPEICHERN
01805                           *IN DEN KOORDINATEN
01810           JSR     AFOB    *DARSTELLUNG
01820           LDX     #$1300
01830  LOP10    LEAX    -1,X    *ABWARTEN
01840           BNE     LOP10
01850           BRA     DEBUT   *SCHLEIFE
01860  *
01870  RAZ      FCB     $1F,$20,$20,$1F,$12,$14,$1B,$60,$1B,$50,$1B
01875           FCB     $40,12,4
01880           END     START
01890  *
01900  *
01910  *        ASSEMBLIER BEFEHLE
01920  *
01930  *
01940  *        TIPPEN ZONE $9000/ ENTER/
01950  *               ASS RM      / ENTER/
01960  *               EXECT       / ENTER/
01970  *        ODER
01980  *        TIPPEN ASS : ESSAI/ ENTER/
01990  *        In diesem Fall wird das programm durch
02000  *        "LOADM" geladen
```

ZUSATZ

E - Die wichtigen Adressen

		Die Bildschirmadressen
$0000	STAD	ErstesByte
$1F40	ENDAD	Letztes Byte
$A7C0	PRA	Daten Register PORT A
$A7C1	PRB	Daten Register PORT B
$A7C2	CRA	Kontrol Register PORT A
$A7C3	CRB	Kontrol Register PORT B
		Spiele/Musik PIA 6821
$A7CC	PRA1	Daten Register PORT A
		PA07 (Eingabe) Joystickspositon

$A7CD	PRB1	Daten Register PORT B Bits Ø-5 (Eingabe) Digital-Analog-Wandler Bit 6 Eingabe Aktion-Knopf Joystick Ø Bit 7 Eingabe Aktion-Knopf Joystick 1
$A7CE	CRA1	Kontrol Register PORT A1
$A7CF	CRB1	Kontrol Register PORT B1

F-Vom Monitor belegte RAM-Register (Zeropage)

$2Ø19	STATUS	b4=nur Punktgrafik b3=Summer ausgeschaltet
$2Ø1B	RANG	Enthält die momentane logische Zeilennummer
$2Ø1C	COLN	Momentane logische Spalte
$2Ø1 E	TOPRAN	Enthält die erste logische Zeile des momentanen Bildschirmfensters
$2Ø2Ø	BOTRAN	Enthält die letzte logische Zeile des momentanen Bildschirmfensters
$2Ø29	FORME	Farbe für Punkt oder Strichzeichnungen (-16, +15)
$2Ø2B	COLOUR	Momentane Vordergrung und Hintergrundfarben
$2Ø36	CHDRAW	ASCII-Code eines Grafikzeichens im Zeichenmodus
$2Ø39	TEMPO,2	Allgemeines Tempo einer Melodie
$2Ø3B	DUREE,2	Relative Tondauer
$2Ø3D	TIMBRE	Bedämpfung
$2Ø3E	OKTAVE,2	Oktave einer Note
$2Ø42	PROPC	OP-Code Kommunikations-Controller
$2Ø43	PRSTA	Statusanzeige Kommunikations-Controller
$2Ø48	DKOPC	OP-Code Disketten-Controller
$2Ø49	DKDRV	Laufwerksnummer
$2Ø4A	DKTRK,2	Spurnummer
$2Ø4C	DKSEC	Sektorennummer
$2Ø4E	DKSTA	Status Anzeige Disketten-Controller
$2Ø4F	DKBUF,2	E/H-Pufferadresse Diskette
$2Ø5E	SWI1PT,2	SWI-Zeiger
$2Ø61	TIMEPT,3	Adressebenutzer IRQ Interrupt Routine und Flag zur IRQ Steuerung
$2Ø64	IRQPT,3	Adresse standard IRQ Interrupt Routine und standard Flag zur IRQ Steuerung
$2Ø67	FIRQPT,3	Adresse FIRQ Interrupt Routine und Flag zur FIRQ Steuerung
$2Ø6A	SIMUL,3	Adresse Monitor-Einsprungtabelle und Flag
$2Ø6D	CHRPTR,3	Adresse Tastatur Dekodierttabelle und Flag
$2Ø7Ø	USERAF,3	Adresse Benutzer Zeichen und Flag
$2Ø73	GENTR,3	Adresse standard Zeichen
$2Ø76	LATCLV	Tastaturverzögerung
$2Ø77	GRCODE	Codewort zum Kopieren eines Grafikbildschirms

G - Speicheraufteilung

ØØØØ	-	1FFF	2X8K Bildschirmspeicher
2ØØØ	-	2ØFF	Monitor Register
21ØØ	-	21FF	Basic oder Anwender Register
22ØØ	-	5FFF	Benutzerspeicher
6ØØØ	-	9FFF	Benutzerspeicher
AØØØ	-	A7BF	1.9K zur Diskettensteuerung
A7CØ	-	A7C3	6821 System PIA
A7C4	-	A7CB	Frei
A7CC	-	A7CF	6821 Spiele - PIA
A7DØ	-	A7DF	Disketten-Controller
A7EØ	-	A7E3	6821 Kommunikations - PIA
A7E4	-	A7E7	Lichtgriffelzähler
A7E8	-	A7FF	Frei für PIA- und ACIA-Erweiterungen
A8ØØ	-	AFFF	2K Frei
BØØØ	-	EFFF	ROM Benutzercassette
CØØØ	-	EFFF	BASIC (frei von BØØØH bis BFFFH)
FØØØ	-	FFFF	4K Monitor

Verzweigungsbefehle.

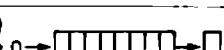
Instruction	Forms	Addressing Mode Relative			Description	5	3	2	1	0
		Op	~	#		H	N	Z	V	C
BCC	BCC	24	3	2	Branch C = 0	•	•	•	•	•
	LBCC	10 24	5(6)	4	Long Branch C = 0	•	•	•	•	•
BCS	BCS	25	3	2	Branch C = 1	•	•	•	•	•
	LBCS	10 25	5(6)	4	Long Branch C = 1	•	•	•	•	•
BEQ	BEQ	27	3	2	Branch Z = 1	•	•	•	•	•
	LBEQ	10 27	5(6)	4	Long Branch Z = 1	•	•	•	•	•
BGE	BGE	2C	3	2	Branch ≥ Zero	•	•	•	•	•
	LBGE	10 2C	5(6)	4	Long Branch ≥ Zero	•	•	•	•	•
BGT	BGT	2E	3	2	Branch > Zero	•	•	•	•	•
	LBGT	10 2E	5(6)	4	Long Branch > Zero	•	•	•	•	•
BHI	BHI	22	3	2	Branch Higher	•	•	•	•	•
	LBHI	10 22	5(6)	4	Long Branch Higher	•	•	•	•	•
BHS	BHS	24	3	2	Branch Higher or Same	•	•	•	•	•
	LBHS	10 24	5(6)	4	Long Branch Higher or Same	•	•	•	•	•

Instruction	Forms	Addressing Mode			Description	5	3	2	1	0
		Relative				H	N	Z	V	C
		Op	~	#						
BLE	BLE	2F	3	2	Branch ≤ Zero	•	•	•	•	•
	LBLE	10 2F	5(6)	4	Long Branch ≤ Zero	•	•	•	•	•
BLO	BLO	25	3	2	Branch Lower	•	•	•	•	•
	LBLO	10 25	5(6)	4	Long Branch Lower	•	•	•	•	•
BLS	BLS	23	3	2	Branch Lower or Same	•	•	•	•	•
	LBLS	10 23	5(6)	4	Long Branch Lower or Same	•	•	•	•	•
BLT	BLT	2D	3	2	Branch < Zero	•	•	•	•	•
	LBLT	10 2D	5(6)	4	Long Branch < Zero	•	•	•	•	•
BMI	BMI	2B	3	2	Branch Minus	•	•	•	•	•
	LBMI	10 2B	5(6)	4	Long Branch Minus	•	•	•	•	•
BNE	BNE	26	3	2	Branch Z = 0	•	•	•	•	•
	LBNE	10 26	5(6)	4	Long Branch Z = 0	•	•	•	•	•
BPL	BPL	2A	3	3	Branch Plus	•	•	•	•	•
	LBPL	10 2A	5(6)	4	Long Branch Plus	•	•	•	•	•
BRA	BRA	20	3	2	Branch Always	•	•	•	•	•
	LBRA	16	5	3	Long Branch Always	•	•	•	•	•
BRN	BRN	21	3	2	Branch Never	•	•	•	•	•
	LBRN	10 21	5	4	Long Branch Never	•	•	•	•	•
BSR	BSR	8D	7	2	Branch to Subroutine	•	•	•	•	•
	LBSR	17	9	3	Long Branch to Subroutine	•	•	•	•	•
BVC	BVC	28	3	2	Branch V = 0	•	•	•	•	•
	LBVC	10 28	5(6)	4	Long Branch V = 0	•	•	•	•	•
BVS	BVS	29	3	2	Branch V = 1	•	•	•	•	•
	LBVS	10 29	5(6)	4	Long Branch V = 1	•	•	•	•	•

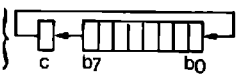
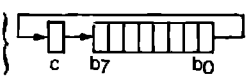
Instruction	Forms	Addressing modes											
		Immediate			Direct			Indexed			Extended		
		Op	~	#	Op	~	#	Op	~	#	Op	~	#
ABX		3A											
ADC	ADCA	89	2	2	99	4	2	A9	4+	2+	B9	5	3
	ADCB	C9	2	2	D9	4	2	E9	4+	2+	F9	5	3
ADD	ADDA	8B	2	2	9B	4	2	AB	4+	2+	BB	5	3
	ADDB	CB	2	2	DB	4	2	EB	4+	2+	FB	5	3
	ADDD	C3	4	3	D3	6	2	E3	6+	2+	F3	7	3
AND	ANDA	84	2	2	94	4	2	A4	4+	2+	B4	5	3
	ANDB	C4	2	2	D4	4	2	E4	4+	2+	F4	5	3
	ANDCC	1C	3	2									
ASL	ASLA												
	ASLB												
	ASL				08	6	2	68	6+	2+	78	7	3
ASR	ASRA												
	ASRB												
	ASR				07	6	2	67	6+	2+	77	7	3
BIT	BITA	85	2	2	95	4	2	A5	4+	2+	B5	5	3
	BITB	C5	2	2	D5	4	2	E5	4+	2+	F5	5	3
CLR	CLRA												
	CLRB												
	CLR				0F	6	2	6F	6+	2+	7F	7	3
CMP	CMPA	81	2	2	91	4	2	A1	4+	2+	B1	5	3
	CMPB	C1	2	2	D1	4	2	E1	4+	2+	F1	5	3
	CMPD	10	5	4	10	7	3	10	7+	3+	10	8	4
		83			93			A3			B3		
	CMPS	11	5	4	11	7	3	11	7+	3+	11	8	4
		8C			9C			AC			BC		
	CMPU	11	5	4	11	7	3	11	7+	3+	11	8	4
		83			93			A3			B3		
	CMPX	8C	4	3	9C	6	2	AC	6+	2+	BC	7	3
	CMPY	10	5	4	10	7	3	10	7+	3+	10	8	4
		8C			9C			AC			BC		
COM	COMA												
	COMB												
	COM				03	6	2	63	6+	2+	73	7	3
CWAI		3C	≥20	2									
DAA													

Inherent			Description	5	3	2	1	0
Op	~	#		H	N	Z	V	C
3A	3	1	B + X → X (Unsigned)	•	•	•	•	•
			A + M + C → A	↑	↑	↑	↑	↑
			B + M + C → B	↑	↑	↑	↑	↑
			A + M → A	↑	↑	↑	↑	↑
			B + M → B	↑	↑	↑	↑	↑
			D + M:M + 1 → D	•	↑	↑	↑	↑
			A . M → A	•	↑	↑	0	•
			B . M → B	•	↑	↑	0	•
			CC . IMM → CC					7
48	2	1	A	8	↑	↑	↑	↑
58	2	1	B	8	↑	↑	↑	↑
			M	8	↑	↑	↑	↑
47	2	1	A	8	↑	↑	•	↑
57	2	1	B	8	↑	↑	•	↑
			M	8	↑	↑	•	↑
			Bit Test A (M — A)	•	↑	↑	0	•
			Bit Test B (M — B)	•	↑	↑	0	•
4F	2	1	0 → A	•	0	1	0	0
5F	2	1	0 → B	•	0	1	0	0
4F	2	1	0 → M	•	0	1	0	0
			Compare M from A	8	↑	↑	↑	↑
			Compare M from B	8	↑	↑	↑	↑
			Compare M:M + 1 from D	•	↑	↑	↑	↑
			Compare M:M + 1 from S	•	↑	↑	↑	↑
			Compare M:M + 1 from U	•	↑	↑	↑	↑
			Compare M:M + 1 from X	•	↑	↑	↑	↑
			Compare M:M + 1 from Y	•	↑	↑	↑	↑
43	2	1	$\overline{A}$ → A	•	↑	↑	0	1
53	2	1	$\overline{B}$ → B	•	↑	↑	0	1
			$\overline{M}$ → M	•	↑	↑	0	1
			CC / IMM → CC Wait for Interrupt					7
19	2	1	Decimal Adjust A	•	↑	↑	0	1

Instruction	Forms	Addressing modes											
		Immediate			Direct			Indexed			Extended		
		Op	~	#	Op	~	#	Op	~	#	Op	~	#
DEC	DECA												
	DECB												
	DEC				0A	6	2	6A	6+	2+	7A	7	3
EOR	EORA	88	2	2	98	4	2	A8	4+	2+	B8	5	3
	EORB	C8	2	2	D8	4	2	E8	4+	2+	F8	5	3
EXG	R1, R2	1E	8	2									
INC	INCA												
	INCB												
	INC				0C	6	2	6C	6+	2+	7C	7	3
JMP					0E	3	2	6E	3+	2+	7E	4	3
JSR					9D	7	2	AD	7+	2+	BD	8	3
LD	LDA	86	2	2	96	4	2	A6	4+	2+	B6	5	3
	LDB	C6	2	2	D6	4	2	E6	4+	2+	F6	5	3
	LDD	CC	3	3	DC	5	2	EC	5+	2+	FC	6	3
	LDS	10	4	4	10	6	3	10	6+	3+	10	7	4
		CE			DE			EE			FE		
	LDU	CE	3	3	DE	5	2	EE	5+	2+	FE	6	3
	LDX	8E	3	3	9E	5	2	AE	5+	2+	BE	6	3
	LDY	10	4	4	10	6	3	10	6+	3+	10	7	4
		8E			9E			AE			BE		
LEA	LEAS							32	4+	2+			
	LEAU							33	4+	2+			
	LEAX							30	4+	2+			
	LEAY							31	4+	2+			
LSL	LSLA												
	LSLB												
	LSL				08	6	2	68	6+	2+	78	7	3
LSR	LSRA												
	LSRB												
	LSR				04	6	2	64	6+	2+	74		3
MUL													
NEG	NEGA												
	NEGB												
	NEG				00	6	2	60	6+	2+	70	7	3

Inherent			Description	5	3	2	1	0
Op	~	#		H	N	Z	V	C
4A	2	1	A − 1 → A	•	↑	↑	↑	•
5A	2	1	B − 1 → B	•	↑	↑	↑	•
			M − 1 → M	•	↑	↑	↑	•
			A ∀ M → A	•	↑	↑	0	•
			B ∀ M → A	•	↑	↑	0	•
			R1 → R2(2)	•	•	•	•	•
4C	2	1	A + 1 → A	•	↑	↑	↑	•
5C	2	1	B + 1 → B	•	↑	↑	↑	•
			M + 1 → M	•	↑	↑	↑	•
			EA(3) → PC	•	•	•	•	•
			Jump to Subroutine	•	•	•	•	•
			M → A	•	↑	↑	0	•
			M → B	•	↑	↑	0	•
			M:M + 1 → D	•	↑	↑	0	•
			M:M + 1 → S	•	↑	↑	0	•
			M:M + 1 → U	•	↑	↑	0	•
			M:M + 1 → X	•	↑	↑	0	•
			M:M + 1 → Y	•	↑	↑	0	•
			EA(3) → S	•	•	•	•	•
			EA(3) → U	•	•	•	•	•
			EA(3) → X	•	•	↑	•	•
			EA(3) → Y	•	•	↑	•	•
48	2	1	A	•	↑	↑	↑	↑
58	2	1	B 	•	↑	↑	↑	↑
			M	•	↑	↑	↑	↑
44	2	1	A	•	0	↑	•	↑
54	2	1	B 	•	0	↑	•	↑
			M	•	0	↑	•	↑
3D	11	1	A x B → D (Unsigned)	•	•	↑	•	9
40	2	1	$\overline{A}$ + 1 → A	8	↑	↑	↑	↑
50	2	1	$\overline{B}$ + 1 → B	8	↑	↑	↑	↑
			$\overline{M}$ + 1 → M	8	↑	↑	↑	↑

Instruction	Forms	Addressing modes											
		Immediate			Direct			Indexed			Extended		
		Op	~	#	Op	~	#	Op	~	#	Op	~	#
NOP													
OR	ORA	8A	2	2	9A	4	2	AA	4+	2+	BA	5	3
	ORB	CA	2	2	DA	4	2	EA	+	2+	FA	5	3
	ORCC	1A	3	2									
PSH	PSHS	34	5+4	2									
	PSHU	36	5+4	2									
PUL	PULS	35	5+4	2									
	PULU	37	5+4	2									
ROL	ROLA												
	ROLB												
	ROL				09	6	2	69	6+	2+	79	7	3
ROR	RORA												
	RORB												
	ROR				06	6	2	66	6+	2+	76	7	3
RTI													
RTS													
SBC	SBCA	82	2	2	92	4	2	A2	4+	2+	B2	5	3
	SBCB	C2	2	2	D2	4	2	E2	4+	2+	F2	5	3
SEX													
ST	STA				97	4	2	A7	4+	2+	B7	5	3
	STB				D7	4	2	E7	4+	2+	F7	5	3
	STD				DD	5	2	ED	5+	2+	FD	6	3
	STS				10	6	3	10	6+	3+	10	7	4
					DF			EF			FF		
	STU				DF	5	2	EF	5+	2+	FF	6	3
	STX				9F	5	2	AF	5+	2+	BF	6	3
	STY				10	6	3	10			10	7	4
					9F			AF	6+	3+	BF		
SUB	SUBA	80	2	2	90	4	2	A0	4+	2+	B0	5	3
	SUBB	C0	2	2	D0	4	2	E0	4+	2+	F0	5	3
	SUBD	83	4	3	93	6	2	A3	6+	2+	B3	7	3

Inherent			Description	5	3	2	1	0
Op	~	#		H	N	Z	V	C
12	2	1	N° Opération	•	•	•	•	•
			A V M → A	•	↑	↑	0	•
			B V M → B	•	↑	↑	0	•
			CC V IMM → CC				7	
			Push Registers on S Stack	•	•	•	•	•
			Push Registers on U Stack	•	•	•	•	•
			Pull Registers on S Stack	•	•	•	•	•
			Pull Registers on U Stack	•	•	•	•	•
49	2	1	A	•	↑	↑	↑	↑
59	2	1	B	•	↑	↑	↑	↑
			M 	•	↑	↑	↑	↑
46	2	1	A	•	↑	↑	•	↑
56	2	1	B	•	↑	↑	•	↑
			M 	•	↑	↑	•	↑
3B	6/15	1	Return from Interrupt					7
39	5	1	Return from Subroutine	•	•	•	•	•
			A−M−C → A	8	↑	↑	↑	↑
			B−M−C → B	8	↑	↑	↑	↑
1D	2	1	Sign Extend B into A	•	↑	↑	0	•
			A → M	•	↑	↑	0	•
			B → M	•	↑	↑	0	•
			D → M:M + 1	•	↑	↑	0	•
			S → M:M + 1	•	↑	↑	0	•
			U → M:M + 1	•	↑	↑	0	•
			X → M:M + 1	•	↑	↑	0	•
			Y → M:M + 1	•	↑	↑	0	•
			A−M → A	8	↑	↑	↑	↑
			B−M → B	8	↑	↑	↑	↑
			D−M:M + 1 → D	•	↑	↑	↑	↑

Instruction	Forms	Addressing modes											
		Immediate			Direct			Indexed			Extended		
		Op	~	#	Op	~	#	Op	~	#	Op	~	#
SWI	SWI(6) SWI2(6) SWI3(6)												
SYNC													
TFR	R1, R2	1F	6	2									
TST	TSTA TSTB TST				0D	6	2	6D	6+	2+	7D	7	3

Légendee :

OP Opération Code (Hexa)
~ Number of MPU Cycles
= Number of Program Bytes
+ Arithmetic Plus
− Arithmetic MInus
. Multiply
M Complement of M
→ Transfer Into
H Half-carry (from bit 3)
N Negative (sign bit)
Z Zero (Reset)
V Overflow, 2's complement
C Carry from ALU
↑ Test and set if true, cleared otherwise
• Not Affected
CC Condition Code Register
: Concatenation
V Logical or
. Logical and
V Logical Exclusive or

Inherent			Description	5	3	2	1	0
Op	~	#		H	N	Z	V	C
3F	19	1	Software Interrupt 1	•	•	•	•	•
10 3F	20	2	Software Interrupt 2	•	•	•	•	•
11 3F	20	1	Software Interrupt 3	•	•	•	•	•
13	≥ 4	1	Synchronize to Interrupt	•	•	•	•	•
			R1 → R2 (2)	•	•	•	•	•
4D	2	1	Test A	•	↑	↑	0	•
5D	2	1	Test B	•	↑	↑	0	•
			Test M	•	↑	↑	0	•

Notes :

1. This column gives a base cycle and byte count. To obtain total count, add the values obtained from the INDEXED ADDRESSING MODE table, in Appendix F.
2. R1 and R2 may be any pair of S bit or any pair of 16 bit registers.
   The S bit registers are : A, B, CC, DP.
   The 16 bit registers are : X, Y, U, S, D, PC.
3. EA is the effective address.
4. The PSH and PUL instructions require 5 cycles plus 1 cycle for each byte pushed or pulled.
5. 5(6) means : 5 cycles if branch not taken, 6 cycles if taken (Branch Instructions).
6. SWI sets I and F bits. SWI2 and SWI3 do not affect I and F.
7. Conditions Codes set as a direct result of the instruction.
8. Value of half-carry flag is undefined.
9. Special Case – Carry set if b7 is SET.