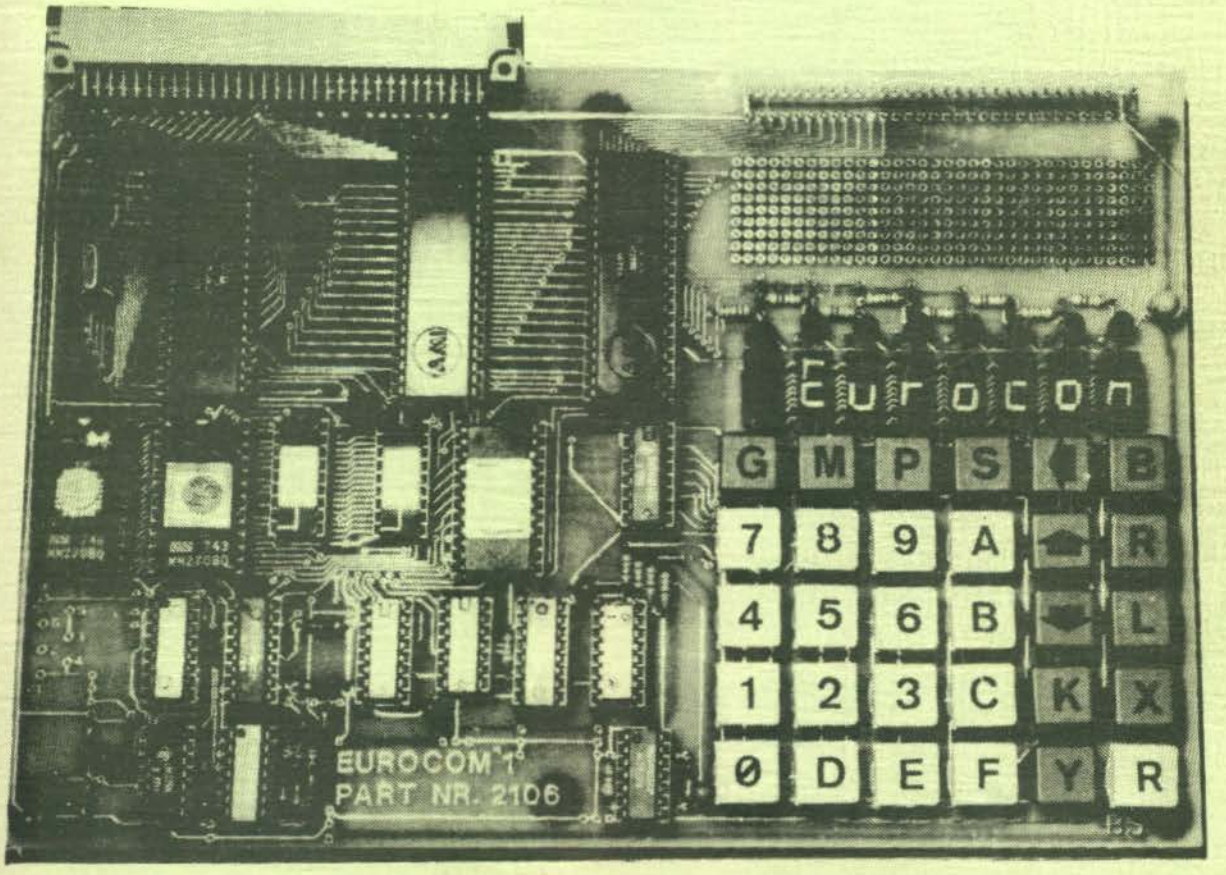
ELTEC

EUROCOM-1

MIKROCOMPUTER



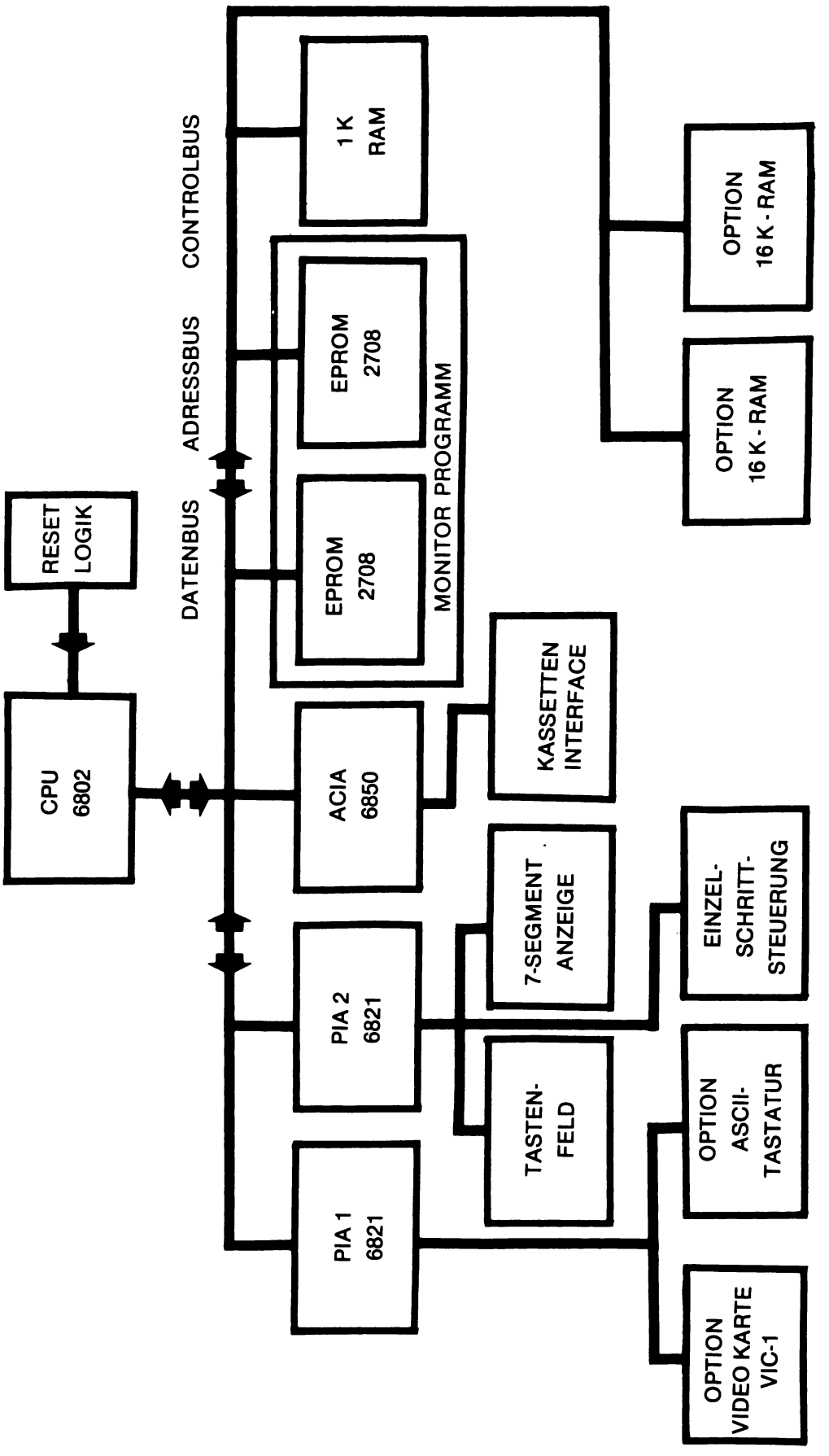
Begleitmaterial mit Beispielprogrammen

ELTEC Elektronik GmbH Neubrunnenstraße 10 6500 Mainz Postfach 1847 Tel. 06131/26411

I N H A L T S V E R Z E I C H N I S

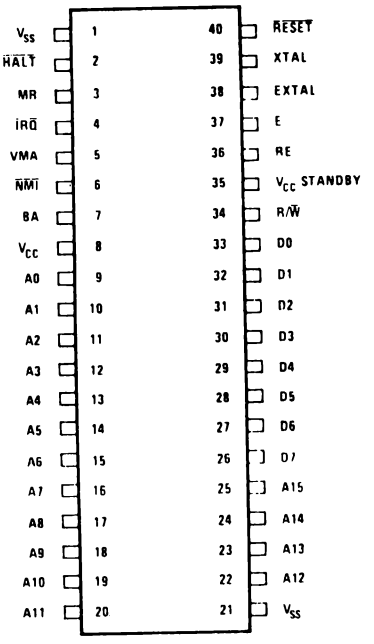
Blockschaltbild..... Seite 3
Pinbelegung der integrierten Schaltkreise Seite 4
Anschlußbelegungen der VG-Leisten..... Seite 6
Anschlußbelegung des EUROCOM-Netzteils... Seite 7
Schaltplan des EUROCOM-Netzteils..... Seite 8
Anschluß-Beispiel für Cassettenrecorder.. Seite 9
Monitorfunktionen des EUROCOM 1 ..... Seite 1o
Anhänge ..... Seite 25
Fehlermeldungen ..... Seite 25
Cassetten- und Fileformat ..... Seite 26
Adressbelegung ..... Seite 28
Wichtige Adressen im Monitor-Arbeitsbereich ..... Seite 29
Anleitung zur Initialisierung des Parallen-Interface-Adapters (PIA)..... Seite 33
Programmierung der Interrupt-Eingänge und Steuerleitungen des PIA ..... Seite 35
Listing des Monitors des EUROCOM 1 ..... Seite 4o
Beispielprogramme ..... Seite 62
Datenblätter ..... Seite 91
der CPU S68o2 ..... Seite 91
der PIA S682o1 ..... Seite 111
der ACIA S685o ..... Seite 123

BLOCKSCHALTBILD EUROCOM 1



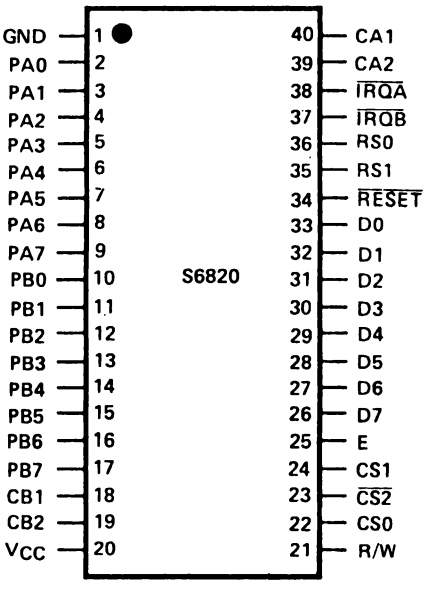
S6802

MICROPROCESSOR



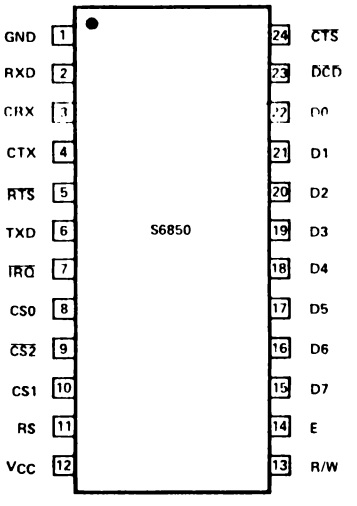
S6820I

PERIPHERAL INTERFACE ADAPTER (PIA)



S6850

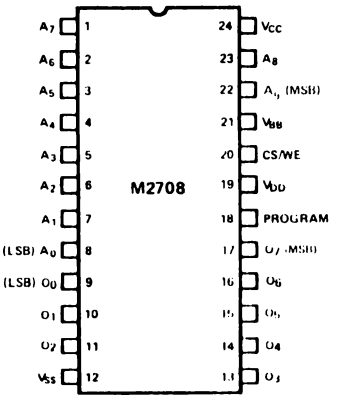
ASYNCHRONOUS COMMUNICATION INTERFACE ADAPTER (ACIA)



M2708

1K x 8

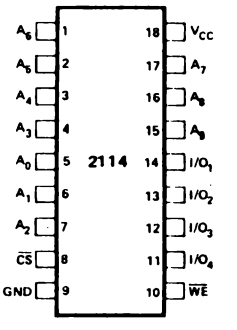
E PROM



2114

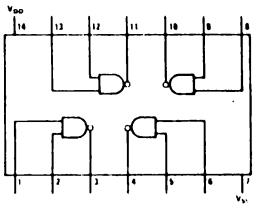
1024 X 4 BIT

STATIC RAM

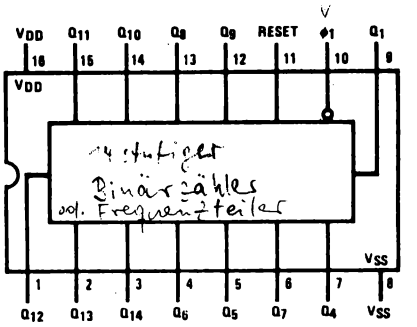


Takt

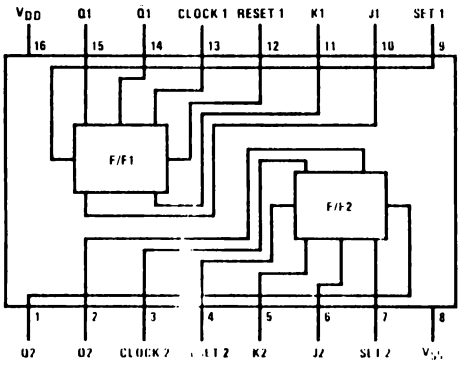
CD4011



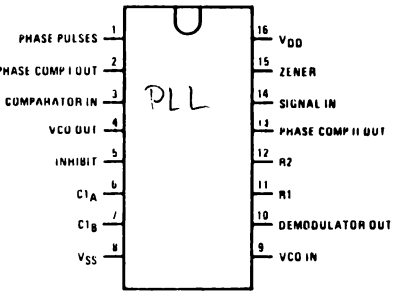
CD4020



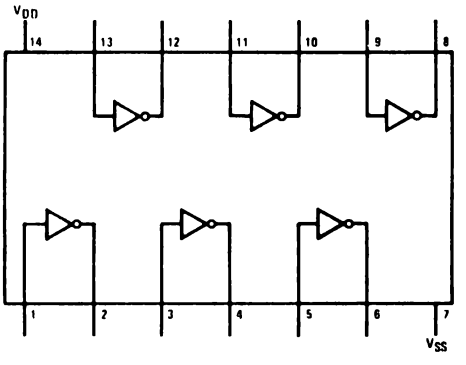
CD4027



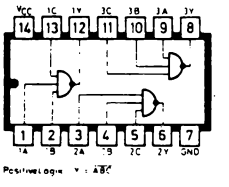
CD4046



CD4069

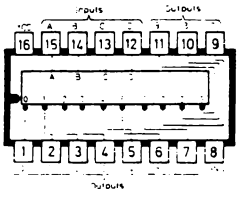


SN7410 SN74LS10



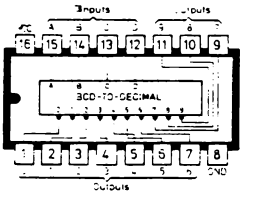
SN7442A SN74LS42

BCD -> Dez.



SN74145 SN74LS145

BCD -> Dez



ANSCHLUSSBELEGUNGEN DER VG-LEISTEN

	c VG1	a	Nr	c VG2	a	
	GND	GND	1	GND	GND	
PIA1	CB2	A15	2	CB2		PIA2
	CB1	A14	3	CB1		
	PB7	A13	4	PB7		
	PB6	A12	5	PB6		
	PB5	A11	6	PB5		
	PB4	A1Ø	7	PB4		
	PB3	A 9	8	PB3		
	PB2	A 8	9	PB2		
	PB1	A 7	1Ø	PB1		
	PBØ	A 6	11	PBØ		
	PA7	A 5	12	PA7		
	PA6	A 4	13	PA6		
	PA5	A 3	14	PA5		
	PA4	A 2	15	PA4		
	PA3	A 1	16	PA3		
	PA2	A Ø	17	PA2		
	PA1	R/W	18	PA1		
	PAØ	DØ	19	PAØ		
	$\overline{IRQ}$	D1	2Ø			FREI
	CA2	D2	21	CA2		
	CA1	D3	22	CA1		
	$\overline{NMI}$	D4	23			
	$\overline{HALT}$	D5	24			
	BA	D6	25			
[ACIA]	RXD	D7	26			FREI
	TXD	$\overline{RES}$	27			
	VMA	E	28			
	MR	*300B	29			
	+12	- 5	3Ø	+12	- 5	
	+ 5	+ 5	31	+ 5	+ 5	
	GND	GND	32	GND	GND	

* Anschluß 3 u. 4 vom S 6850 Taktfrequenz 300 Baud

ANSCHLUSS-BELEGUNG DES EUROCOM-NETZTEILES

V G- Leiste

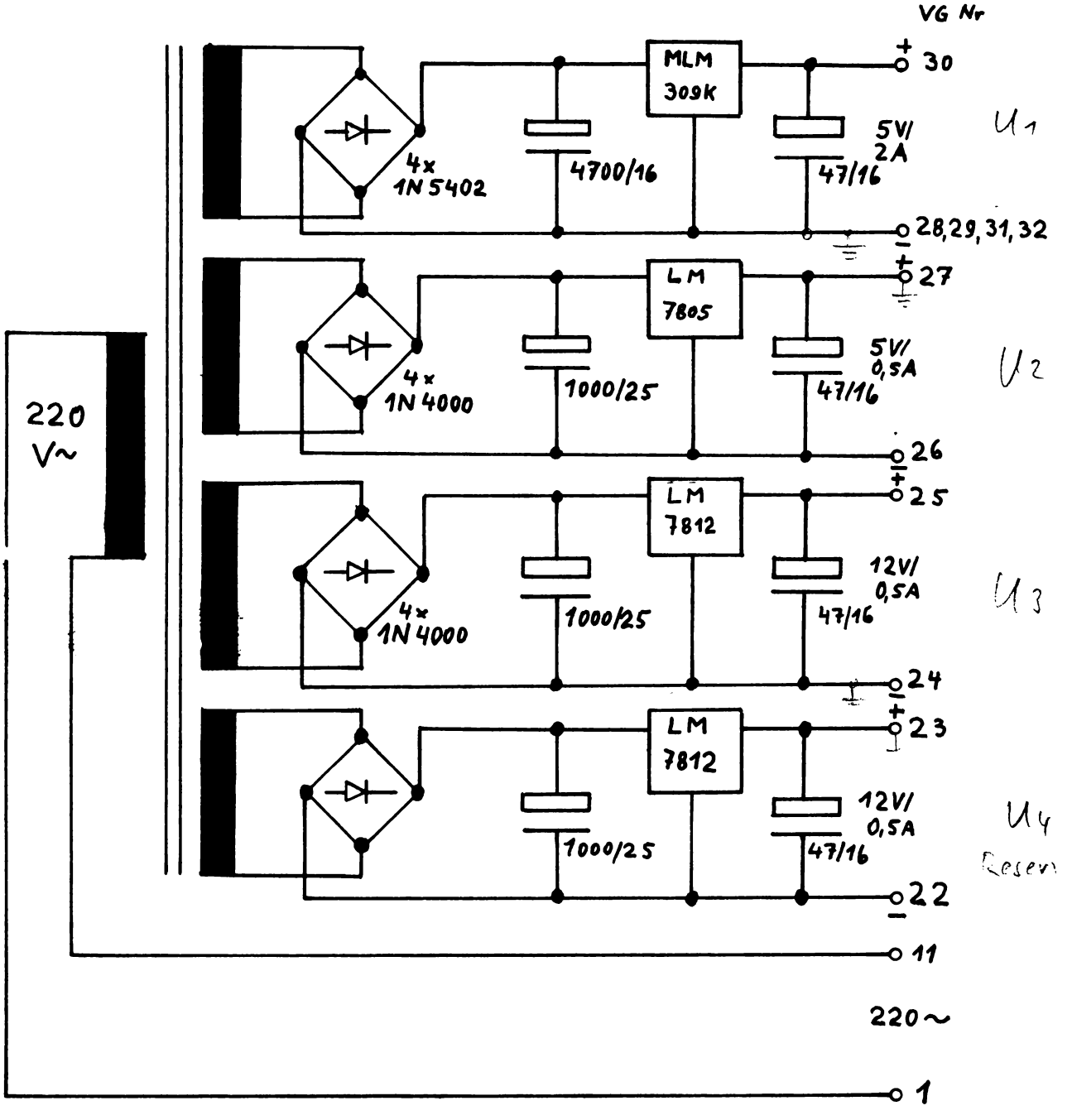
a+c	Nr	
22Ø V	1	Netzleitung
	2	
	3	
	4	
	5	
	6	
	7	
	8	
	9	
	1Ø	
22Ø V	11	Netzleitung
	12	
	13	
	14	
	15	
	16	
	17	
	18	
	19	
	2Ø	
	21	
-	22	(verbinden mit -12V/Ø,5A) Reserve !
+ 12	23	(verbinden mit GND) "
-	24	verbinden mit GND
+ 12	25	verbinden mit +12V/Ø,5A
-	26	verbinden mit - 5V/Ø,5A
+ 5	27	verbinden mit GND
-	28	verbinden mit GND
- 5	29	verbinden mit GND
+ 5	3Ø	verbinden mit + 5V/2A
-	31	verbinden mit GND
- 5	32	verbinden mit GND

ACHTUNG:

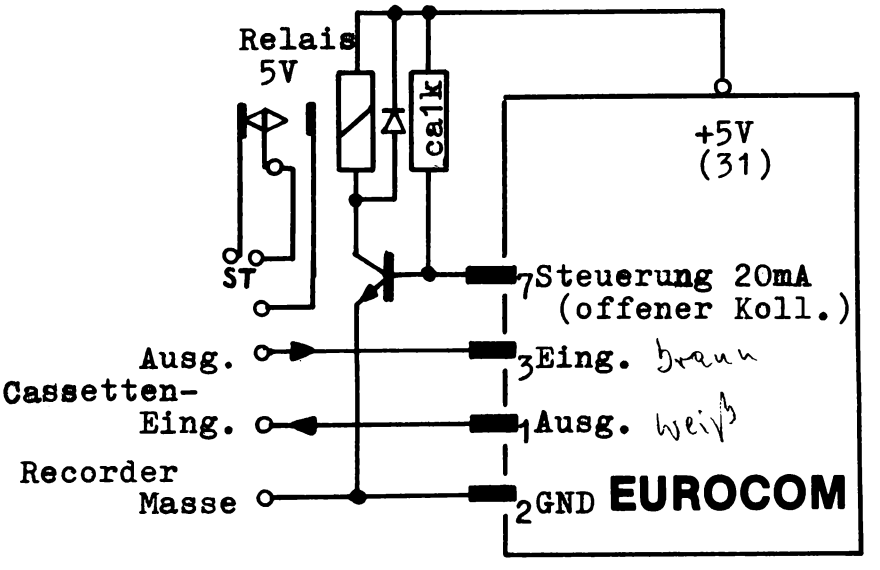
ANSCHLUSS-BELEGUNG VON EUROCOM-1 UND NETZTEIL STIMMEN NICHT ÜBEREIN !

- 12V/Ø,5A wird für spätere Erweiterungen benötigt.

SCHALTPLAN DES EUROCOM-NETZGERÄTES



ANSCHLUSS-BEISPIEL FÜR CASSETTENRECORDER



Steuerung kann natürlich auch von Hand erfolgen.

MONITOR

```
EEEEE    L        TTTTTTT    EEEEE     CCC
E        L           T       E        C   C
E        L           T       E        C
EEE      L           T       EEE      C
E        L           T       E        C
E        L           T       E        C   C
EEEEE    LLLLLLL     T       EEEEE     CCC
```

E U R O C O M 1

M O N I T O R F U N K T I O N E N

BESCHREIBUNG DER MONITORFUNKTIONEN DES EUROCOM 1

A C H T U N G
BEIM BETRIEB MIT ZWEI GETRENNTEN STROMVERSORGUNGEN
MUESSEN DIE -5 VOLT Z U E R S T EIN- UND
Z U L E T Z T AUSGESCHALTET WERDEN.

NACH DEM EINSCHALTEN BEFINDET SICH DAS MONITORPROGRAMM
DES EUROCOM 1 AUTOMATISCH IM GRUNDZUSTAND, DER
H A U P T K O N T R O L L S C H L E I F E (HKS).
DIE ANZEIGE BLINKT JETZT IM SEKUNDENRYTHMUS:
"EUROCOM" - "CONTROL".

DIE HKS KANN AUS JEDEM ZUSTAND DER MASCHINE DURCH DRUCK
AUF DIE "RESET"-TASTE ERREICHT WERDEN (TASTE "R", WEISS!!).
WENN SICH DER PROZESSOR IN EINER EWIGEN PROGRAMMSCHLEIFE
BEFINDET, AUS DER ER VON SELBST NICHT MEHR ZUM MONITOR
ZURUECKKEHREN KANN, IST DIE "RESET"-TASTE ZU BETAETIGEN.

DIE MONITORFUNKTIONEN KOENNEN VON DER HKS AUS DURCH DIE
BETAETIGUNG FOLGENDER (GRUENER) TASTEN AUFGERUFEN WERDEN:

TASTE	BEZEICHNUNG	BEDEUTUNG
M	MEM	MEMORY CHANGE AND CHECK
G	GO	GO TO START OF PROGRAM
P	POINT	SET A BREAKPOINT
S	STEP	SINGLE STEP
B	BRANCH	COMPUTE RELATIVE BRANCH ADDRESS
R	RECORD	RECORD A FILE VIA ACIA
L	LOAD	LOAD A FILE FROM ACIA
K	KILL	KILL BREAKPOINTS
X	---	ABHAENGIG VOM BENUTZERPROGRAMM
Y	---	WIE X

DIE BEDEUTUNG DIESER TASTEN KANN SICH AENDERN, SOBALD
DER PROZESSOR SICH NICHT MEHR IN DER HKS BEFINDET.
DIES WIRD NOCH IM DETAIL DARGESTELLT.

DIE HEXADEZIMALTASTATUR (0..9,A..F, WEISS!!) UND
DIE TASTEN ← ("BACKARROW"="BA"), ↑ ("UPARROW"="UA") UND
↓ ("DOWNARROW"="DA") HABEN I N N E R H A L B
DER EINZELNEN MONITORFUNKTIONEN SPEZIELLE BEDEUTUNGEN,
IN DER HKS JEDOCH KEINE.

```
*******
* MEM *
*******
```

DIESE TASTE STARTET AUS DER HKS HERAUS EIN HILFSPROGRAMM, MIT DEM MAN
- DEN INHALT JEDER SPEICHERZELLE DES EUROCOM 1 ZUR ANZEIGE BRINGEN KANN
- JEDE SPEICHERZELLE DES SCHREIBLESESPEICHERS AENDERN KANN
- BEQUEM FORTLAUFEND PROGRAMME EINTRAGEN UND UEBERPRUEFEN KANN.

NACH BETAETIGUNG DER TASTE "MEM" ERSCHEINT RECHTS AUF DER ANZEIGE EIN ZEICHEN, DAS DEN BUCHSTABEN "M" DARSTELLT. DAMIT WIRD EINE "ADRESSEINGABE" ANGEFORDERT.

A D R E S S E I N G A B E

UEBER DIE HEXADEZIMALTASTATUR MUESSEN GENAU VIER ZEICHEN EINGEGEBEN WERDEN, Z.B. "A","4","0","0". DIE ZEICHEN WERDEN VON RECHTS IN DIE ANZEIGE GESCHOBEN. BIS ZUR EINGABE DER VIERTEN STELLE BEHALTEN ALLE FUNKTIONSTASTEN DIE GLEICHE WIRKUNG WIE IN DER HKS. DIE ADRESSEINGABE KANN ALSO NOCH DURCH BETAETIGUNG EINER GEEIGNETEN FUNKTIONS-TASTE ABGEBROCHEN BZW. KORRIGIERT WERDEN.
NACH EINGABE DER VIERTEN STELLE BLEIBT DIE ANZEIGE CA. 1 SEC. STEHEN , DANN FAEHRT DER MONITOR MIT DER AUSFUEHRUNG DER VORHER ANGEWAEHLTEN MONITORFUNKTION FORT.

NACH DER ADRESSEINGABE SPRINGT DIE ADRESSE AUF DIE LINKE SEITE DER ANZEIGE UND GANZ RECHTS ERSCHEINT IN HEXADEZI-MALER DARSTELLUNG DER INHALT DERJENIGEN SPEICHERZELLE, DEREN ADRESSE EINGEGEBEN WURDE.

INNERHALB DER "MEM"-FUNKTION WERDEN DIE LINKEN VIER STELLEN DER ANZEIGE ALS "ADRESSFELD", DIE BEIDEN STELLEN GANZ RECHTS ALS "DATENFELD" BEZEICHNET.

DAS DATENFELD KANN JETZT DURCH EINGABE UEBER DIE HEXA-DEZIMALTASTATUR VERAENDERT WERDEN: JEDES NEUE ZEICHEN WIRD VON RECHTS IN DAS DATENFELD GESCHOBEN. AUF DIESE WEISE LASSEN SICH AUCH ETWAIGE EINGABEFEHLER LEICHT KORRIGIEREN.

DIESE VERAENDERUNG DES DATENFELDES HAT VORERST NOCH KEINEN EINFLUSS AUF DEN INHALT DES SPEICHERS. DAS DATENFELD WIRD ERST DANN IN DIE ANGEWAEHLTE SPEICHERZELLE UEBERTRAGEN, WENN EINE DER DREI TASTEN "BA","UA" ODER "DA" BETAETIGT WIRD.

"BA": DER INHALT DES DATENFELDES WIRD IN DIE SPEICHERZELLE EINGESCHRIEBEN. DANN PRUEFT DER MONITOR, OB DER SCHREIBVORGANG ERFOLGREICH WAR. WAR ER DAS NICHT, SO FOLGT DIE MELDUNG "ERROR 1" (SH. ANHANG 1). WAR DER SCHREIBVORGANG ERFOLGREICH, SO KEHRT DER MONITOR IN DIE HKS ZURUECK.

"UA": BEWIRKT FAST DAS GLEICHE WIE "BA", ABER ES ERFOLGT
K E I N E RUECKKEHR IN DIE HKS, SONDERN DAS
ADRESSFELD WIRD UM 1 ERNIEDRIGT UND IM DATENFELD
WIRD DER INHALT DIESER NEUEN ADRESSE DARGESTELLT.

"DA": WIRKT WIE DIE TASTE "UA", MIT DEM UNTERSCHIED,
DASS DIE ADRESSE UM 1 ERHOEHT STATT ERNIEDRIGT WIRD.

DIE UEBRIGEN FUNKTIONSTASTEN HABEN IN DER "MEM"-FUNKTION
DIE GLEICHE WIRKUNG WIE IN DER HKS.
DURCH BETAETIGEN EINER FUNKTIONSTASTE WIRD ABER DAS
DATENFELD N I C H T IN DEN SPEICHER UEBERTRAGEN,
SONDERN N U R DURCH DIE "ARROW"-TASTEN!!!

B E I S P I E L

AB ADRESSE A400 SOLLEN IN DEN SPEICHER DIE ZAHLEN
03,06,09,0C,0F,12,15,18,1B,1E EINGETRAGEN WERDEN.

TASTE	ANZEIGE	KOMMENTAR
RESET	EUROCOM CONTROL	ZWECKMAESSIG, UM ALTE BREAKPOINTS ZU LOESCHEN, DIE SONST BEI SPAETERER BENUTZUNG VON "KILL" ODER "RESET" DEN SPEICHER UNERWUENSCHT VER-AENDERN KOENNTEN.
M	M	"MEM-FUNKTION AUFRUFEN
A	M A	BEGINN DER ADRESSEINGABE
4	M A4	
0	M A40	
0	M A400	1 SEC WARTEN
	A400 **	** = ALTER INHALT VON A400
0	A400 *0	BEGINN DER DATENEINGABE
3	A400 03	
DA	A401 **	NAECHSTE ADRESSE
0	A401 *0	
7	A401 07	HIER WURDE EIN FEHLER GEMACHT
0	A401 70	KORREKTUR!
6	A401 06	
DA	A402 **	NAECHSTE ADRESSE
.		
.		USW.
.		
E	A409 1E	LETZTE EINGABE
UA	A408 1B	BEGINN DER "RUECKWAERTSKONTROLLE"

DURCH FORTGESETZTE BETAETIGUNG VON "UA" KANN MAN
DIE EINGEGEBENEN ZAHLEN IN UMGEKEHRTER REIHENFOLGE
DURCHGEHEN, UEBERPRUEFEN UND GEGEBENENFALLS KORRIGIEREN.

A C H T U N G: VERSUCHT MAN AUF ADRESSEN AUSSERHALB DES
REGULAEREN RAM- ODER ROM-SPEICHERS ZUZUGREIFEN, SO KANN
DAS ZUM "AUSSTEIGEN" DES COMPUTERS FUEHREN. DER RECHNER
KANN DANN NUR MIT DER "RESET"-TASTE WIEDER GEWECKT WERDEN.

EBENSO SOLLTE MAN ES AUF JEDEN FALL VERMEIDEN, MIT DER
"MEM"-FUNKTION IN DEN RAM-ARBEITSBEREICH DES MONITORS
(S.H. ANHANG 4) ZU SCHREIBEN.

```
******
* GO *
******
```

AUS DER HKS HERAUS STARTET DIESE TASTE DAS BENUTZER-
PROGRAMM AN EINER ANZUGEBENDEN ADRESSE.

NACH BETAETIGUNG DER TASTE "GO" ERSCHEINT RECHTS AUF DER
ANZEIGE DER BUCHSTABE "G". DER MONITOR ERWARTET JETZT EINE
V I E R S T E L L I G E A D R E S S E I N G A B E,
NAEMLICH DIE STARTADRESSE DES BENUTZERPROGRAMMS.

DIE ADRESSE WIRD UEBER DIE HEXADEZIMALTASTATUR EINGEGEBEN;
DIE EINGEGEBENEN ZEICHEN SCHIEBEN SICH VON RECHTS IN DIE
ANZEIGE HINEIN. NACH DEM VIERTEN ZEICHEN BLEIBT DIE ANZEIGE
NOCH CA. 1 SEC. STEHEN, DANN STARTET DER PROZESSOR AN DER
EINGEGEBENEN ADRESSE (VERGL. BESCHREIBUNG DER "MEM"-FUNKTION,
ABSCHNITT "ADRESSEINGABE").

B E I S P I E L

TASTE	ANZEIGE	KOMMENTAR
G	G	"GO" AUFRUFEN
A	G A	BEGINN DER ADRESSEINGABE
4	G A4	
0	G A40	
0	G A400	LETZTE EINGABE, 1 SEC WARTEN. DER PROZESSOR STARTET DIE PRO-GRAMMAUSFUEHRUNG BEI ADRESSE A400.

DORT SOLLTE NATUERLICH DANN SCHON EIN BENUTZERPROGRAMM
STEHEN, ANDERNFALLS WIRD MAN DIE MASCHINE HOECHSTWAHR-
SCHEINLICH NUR NOCH MIT DER "RESET"-TASTE IN DIE HKS
ZURUECKHOLEN KOENNEN.

```
*********
* POINT *
*********
```

VON DER HKS AUS BEWIRKT POINT DAS SETZEN EINES BREAK-
POINTS.
LAEUFT DER PROZESSOR BEI DER ABARBEITUNG DES BENUTZER-
PROGRAMMS AUF EINEN BREAKPOINT, SO WIRD DAS PROGRAMM
UNTERBROCHEN UND DIE KONTROLLE AN DEN BREAKPOINTSERVICE
DES MONITORS UEBERGEBEN.
DER BENUTZER KANN DANN DIE CPU-REGISTER KONTROLLIEREN
UND AENDERN,DAS PROGRAMM FORTSEZEN LASSEN ODER DEN
SINGLE-STEP-MODE (SH. TASTE "STEP") EINSCHALTEN.

NACH DRUCK AUF "POINT" ERSCHEINT RECHTS AUF DER ANZEIGE
EIN "P". DAMIT WIRD EINE ADRESSEINGABE ANGEFORDERT,
NAEMLICH DIE ADRESSE, AUF DIE DER BREAKPOINT GESETZT
WERDEN SOLL.
DER MONITOR LEGT NUN DIE EINGEGEBENE ADRESSE SOWIE DEN
BEFEHL, DER UNTER DIESER ADRESSE IM PROGRAMM STEHT, IN
SEINEM BREAKPOINTSPEICHER AB.
DANACH SCHREIBT ER IN DIE ANGEGEBENE ADRESSE DEN BEFEHL
"SWI" (SOFTWARE-INTERRUPT, CODE 3F) UND GEHT ZURUECK
ZUR HKS.

ES KOENNEN MAXIMAL 15 BREAKPOINTS GESETZT WERDEN, UEBER-
SCHREITUNG FUEHRT ZU ERROR 3 (SH. ANHANG 1); ES IST NICHT
ZULAESSIG, EINEN BREAKPOINT AUF EINE ADRESSE ZU SETZEN,
DIE SCHON DEN BEFEHL "SWI" ENTHAELT (ERROR 4).

BEHANDLUNG VON BREAKPOINTS DURCH DEN EUROCOM - MONITOR

WENN DER PROZESSOR BEI DER ABARBEITUNG EINES PROGRAMMES
AUF DEN BEFEHL SWI (=3F) TRIFFT, RETTET ER ZUNAECHST DIE
CPU-REGISTER (PC,INDEX,ACCU A, ACCU B, CONDITION CODE)
IN DEN STACK. DANN SETZT ER DEN PC NEU AUF DIE ADRESSE,
DIE VOM INTERRUPTVEKTOR "SWIV" ANGEGEBEN WIRD (SH.
ANHANG 4). "SWIV" WIRD BEI JEDEM RESET VOM MONITOR MIT
DER STARTADRESSE DER BREAKPOINTSERVICEROUTINE VORBESETZT.

SOLANGE MAN DEN INHALT VON "SWIV" UNVERAENDERT LAESST,
FUEHRT DER BEFEHL SWI STETS IN DEN BREAKPOINTSERVICE,
DIE AUSFUEHRUNG DES PROGRAMMS IST ALSO ZUNAECHST UNTER-
BROCHEN. DER BREAKPOINTSERVICE STARTET DANN DIE
REGISTER-ANZEIGE&MODIFIKATIONS-ROUTINE (=RR).

DIESE ROUTINE WIRD SPAETER UNTER "STEP" NOCH
AUSFUEHRLICHER BESCHRIEBEN. HIER NUR SOVIEL:
RR MELDET SICH MIT DER ANZEIGE "PC=XXXX" ('XXXX'=ADRESSE
DES SWI-BEFEHLS, DER DEN "BREAK" VERURSACHTE). JETZT
BESTEHT DIE MOEGLICHKEIT, DIE CPU-REGISTER DES BENUTZER-
PROGRAMMS ZU KONTROLLIEREN UND EVTL. ZU MODIFIZIEREN.

DER BREAKPOINTSERVICE KANN MIT DEN TASTEN "GO", "STEP", "KILL" UND "BA" VERLASSEN WERDEN.

WURDE DER BREAKPOINTSERVICE DURCH "KILL" VERLASSEN, KEHRT DER MONITOR IN DIE HKS ZURUECK. MIT "STEP", "GO" UND "BA" WIRD DAS BENUTZERPROGRAMM FORTGESETZT, INDEM DER PROZESSOR DIE CPU-REGISTER RESTAURIERT, WIE SIE JETZT IM STACK STEHEN. D.H. DIE VOM BENUTZER IN DER RR AUSGEFUEHRTEN REGISTERMODIFIKATIONEN WIRKEN SICH JETZT AUS.

MAN BEACHTE: DER JEWEILIGE BREAKPOINT WIRD BEIM VERLASSEN DES BREAKPOINTSERVICE MIT DEN OBENGENANNTEN VIER TASTEN GELOESCHT UND DURCH DEN URSPRUENGLICHEN BEFEHL ERSETZT - DIES GESCHIEHT KORREKT ALLERDINGS NUR, WENN DIE ADRESSE DES BREAKPOINTS IM BREAKPOINTSPEICHER ENTHALTEN WAR UND DER PC DES BENUTZERPROGRAMMS NICHT VERAENDERT WURDE (SH. ANHANG 4).

IM UEBRIGEN KANN DER BREAKPOINT AUCH MIT DEN TASTEN "MEM","POINT","BRANCH","LOAD" UND "RECORD" VERLASSEN WERDEN. SIE FUEHREN SOFORT ZU DEN ENTSPRECHENDEN MONITORFUNKTIONEN, OHNE DEN BREAKPOINT ZU LOESCHEN.

ABWEICHUNGEN VON DEM OBEN BESCHRIEBENEN REGELVERHALTEN ERHAELT MAN, WENN MAN

- EINEN BREAKPOINT AUF EINE ADRESSE SETZT, DIE 00 ENTHAELT. EIN SOLCHER BREAKPOINT KANN ZWAR NICHT MEHR DURCH "RESET", WOHL ABER NOCH DURCH DEN BREAKPOINTSERVICE GELOESCHT WERDEN

- DEN BEFEHL SWI IM PROGRAMM VERWENDET, IHN ALSO NICHT MIT DER "POINT"-FUNKTION DORT ABSETZT

- DEN PC AN EINEM BREAKPOINT VERAENDERT

- DEN INTERRUPTVERKTOR "SWIV" AENDERT

- DEN BREAKPOINTSPEICHER VERAENDERT.

DIE FOLGEN SIND SCHWER VORAUSZUSAGEN UND HAENGEN Z.T. VON VERALTETEN ODER NOCH UNDEFINIERTEN INHALTEN DES BREAKPOINTSPEICHERS AB.

IN DER PRAXIS JEDOCH STELLT SICH DIE BENUTZUNG VON BREAKPOINTS ALS AUSGESPROCHEN SIMPLE SACHE HERAUS, WENN MAN SICH AN DIE OBIGEN WARNUNGEN HAELT.DIE FOLGENDEN VIER PUNKTE ERMOEGLICHEN DIE REIBUNGSLOSE BENUTZUNG DER BREAKPOINTEIGENSCHAFTEN DES MONITORS:

1. "POINT" XXXX	SETZE EINEN BREAKPOINT AUF ADRESSE XXXX
2. "RESET"	ENTFERNE ALLE BREAKPOINTS, DIE IM BREAKPOINTSPEICHER ALS NOCH AKTIV GEFUEHRT SIND
3. "GO" XXXX "KILL"	ENTFERNE DEN BEI XXXX GESETZTEN BREAKPOINT
4."KILL"	IN DER HKS ENTFERNT "KILL" GENAUSO WIE "RESET" ALLE BREAKPOINTS.

LAEUFT DAS BENUTZERPROGRAMM AUF EINEN BREAKPOINT, SO ERFOLGT DAS VERLASSEN DES BREAKPOINTSERVICE MIT:

5. "KILL"	LOESCHEN DES BREAKPOINT UND RUECKKEHR ZUR HKS
6. "GO"	LOESCHEN DES BREAKPOINT UND FORTSETZUNG DES PROGRAMMS
7. "STEP"	LOESCHEN DES BREAKPOINT UND EINSCHALTEN DES SINGLE-STEP-MODE, AUSFUEHRUNG DES NAECHSTEN PROGRAMMBEFEHLS
8. "BA"	LOESCHEN DES BREAKPOINT UND FOERTSETZUNG DES PROGRAMMS, OHNE DEN SINGLE-STEP-MODE EIN- ODER AUSZUSCHALTEN, D.H. ER BLEIBT AUS, WENN ER AUS WAR UND AN, WENN ER AN WAR.

ES IST OFFENSICHTLICH, DASS DER BREAKPOINT FUER DEN PROGRAMMIERER EIN IDEALES MITTEL ZUR FEHLERSUCHE IST.
ER KANN DAMIT SEIN PROGRAMM AN BELIEBIGEN STELLEN ANHALTEN LASSEN UND ES MIT HILFE DER RR KONTROLLIEREN.
DARUEBERHINAUS BIETET DIE REGISTERMODIFIKATION DIE MOEGLICHKEIT, DEN PROGRAMMLAUF ZU BEEINFLUSSEN.

```
********
* STEP *
********
```

AUS DER HKS HERAUS BEWIRKT DIESE TASTE DEN START EINES PROGRAMMES IM SINGLE-STEP-MODE (= SSM).

NACH BETAETIGUNG DER TASTE "STEP" ERSCHEINT RECHTS AUF DER ANZEIGE EIN "S". WIE BEI "GO" MUSS WIEDER EINE VIER-STELLIGE ADRESSEINGABE ERFOLGEN, NAEMLICH DIE LOGISCHE STARTADRESSE DES BENUTZERPROGRAMMS.

DANACH ERSCHEINT DIE ANZEIGE "PC=XXXX", WOBEI 'XXXX' DIE EBEN EINGEGEBENE ADRESSE IST. DAMIT BEFINDET SICH DER RECHNER IN DER RR, UND ZWAR MIT EINGESCHALTETEM SSM.

DER SINGLE-STEP-MODE (=SSM)

IM SSM HAELT DER RECHNER V O R DER AUSFUEHRUNG JEDES EINZELNEN PROGRAMMBEFEHLS AN UND GEHT IN DIE RR. DORT WIRD DURCH "PC=XXXX" DIE ADRESSE DES BEFEHLS, VOR DESSEN AUS-FUEHRUNG GESTOPPT WURDE, ANGEZEIGT.

SOLANGE DER SSM EINGESCHALTET IST, WIRD DER RECHNER DURCH EINFACHEN DRUCK AUF DIE TASTE "BA" VERANLASST, DEN NAECHSTEN BEFEHL DES BENUTZERPROGRAMMS AUSZUFUEHREN UND DANN SOFORT IN DIE RR ZURUECKZUKEHREN. AUF DIESE WEISE KANN EIN PROGRAMM IN EINZELSCHRITTEN "DURCHGESTEPPT" WERDEN.

IM PRINZIP WIRKT JETZT JEDER EINZELNE BEFEHL DES BENUTZER-PROGRAMMS WIE EIN BREAKPOINT, AUCH WENN KEINER GESETZT WURDE. TATSAECHLICH BENUTZT DER RECHNER FUER DEN SSM NICHT DEN BEFEHL SWI, SONDERN VOM MONITOR SELBST AUSGELOESTE HARDWARE-INTERRUPTS.

ES GIBT ZWEI MOEGLICHKEITEN, UM IN DEN SSM ZU KOMMEN:

1. AUS DER HKS HERAUS MIT "STEP" UND ADRESSEINGABE (SH. OBEN)

2. MIT "STEP" O H N E ADRESSEINGABE, WENN DER MONITOR SICH SCHON IN DER RR BEFINDET (Z.B DURCH EINEN BREAKPOINT).
   IN DIESEM FALL LAESST DER MONITOR DEN PROZESSOR GENAU EINEN PROGRAMMSCHRITT AUSFUEHREN, BEVOR ER IHN DURCH EINEN INTERRUPT ZURUECKRUFT.

DIE REGISTER-ANZEIGE&MODIFIKATIONS-ROUTINE (=RR)

DIE RR WIRD SOWOHL VOM BREAKPOINTSERVICE ALS AUCH VOM SSM GESTARTET, UM DEM BENUTZER ZUGRIFF AUF DIE CPU-REGISTER WAEHREND DER AUSFUEHRUNG SEINES PROGRAMMS ZU BIETEN. AUSSERDEM LAESST SICH IN DER RR DER SSM EIN- UND AUSSCHALTEN.

A. DER ZUGRIFF AUF DIE CPU-REGISTER

DIE RR MELDET SICH GRUNDSAETZLICH MIT DER ANZEIGE: "PC=XXXX" (XXXX=DERZEITIGER STAND DES PC IM BENUTZER-PROGRAMM, ALSO DIE ADRESSE, BEI DER DAS PROGRAMM GEGEBENENFALLS FORTZUSETZEN IST).

DURCH DIE TASTEN "UA" UND "DA" LASSEN SICH NUN SAEMTLICHE CPU-REGISTER KONTROLLIEREN:

TASTE	ANZEIGE	KOMMENTAR
	PC=XXXX	PROGRAMMCOUNTER
DA	SP=XXXX	STACKPOINTER
DA	CC=00XX	CONDITION-CODE-REGISTER
DA	bA=XXXX	ACCU B UND ACCU A
DA	Id=XXXX	INDEXREGISTER
DA	PC=XXXX	USW.

MIT "UA" ERSCHEINEN DIE REGISTER IN UMGEKEHRTER REIHEN-FOLGE.

MIT DER HEXADEZIMALTASTATUR KANN DAS JEWEILS ANGE-ZEIGTE REGISTER MODIFIZIERT WERDEN. MAN BEACHTE ABER, DASS BEI FORTSETZUNG DES BENUTZERPROGRAMMS DIE REGISTER IN DIESER VERAENDERTEN FORM UEBERNOMMEN WERDEN. Z.B. KANN DURCH VERAENDERUNG DES PC EIN SPRUNG ER-ZWUNGEN WERDEN - DIES VERMEIDE MAN ALLERDINGS, WENN MAN SICH AN EINEM BREAKPOINT BEFINDET (WAS IM SSM NICHT IMMER LEICHT ZU ERKENNEN IST).

B. AUSGAENGE AUS DER RR

"GO": SCHALTET DEN SSM AB UND VERANLASST DIE FORT-FUEHRUNG DES BENUTZERPROGRAMMS

"STEP": SCHALTET DEN SSM EIN UND VERANLASST DIE FORT-FUEHRUNG DES BENUTZERPROGRAMMS UM GENAU EINEN BEFEHL UND FUEHRT DANN WIEDER IN DIE RR

"BA": WENN DER SSM AN IST, WIE "STEP"
WENN NICHT, WIE "GO"

"KILL": FALLS DER RECHNER SICH AN EINEM BREAKPOINT BEFINDET (AUCH WENN DER BREAKPOINT IM SSM ERREICHT WURDE), LOESCHT KILL DEN BREAKPOINT UND

FUEHRT ZURUECK ZUR HKS, SONST WIE "BA"

DIE FOLGENDEN TASTEN
"MEM" }
"POINT" }
"BRANCH" } SCHALTEN DEN SSM AB, VERLASSEN DIE RR, OHNE BREAKPOINTS ZU LOESCHEN, LOESEN SOFORT DIE ENTSPRECHEN MONITOR-
"LOAD" } FUNKTIONEN AUS, HABEN
"RECORD" } HIER ALSO DIE GLEICHE WIRKUNG WIE IN DER HKS

C. WARNUNGEN

BEI MODIFIKATION DER CPU-REGISTER IST JEDE VERAENDERUNG DES INTERRUPT-BITS IM CC (2↑4) ZU VERMEIDEN!

STACK-POINTER EBENFALLS NICHT VERAENDERN!

NIEMALS AN EINEM BREAKPOINT DEN PC VERAENDERN!

MONITORROUTINEN MOEGLICHST NICHT IM SSM "DURCH-STEPPEN"! MAN ERKENNT DIE MONITORROUTINEN DARAN, DASS DER PC WERTE GROESSER ALS F7FF ANNIMMT.

KOMMT MAN IM SSM AUF EINEN BREAKPOINT, SO KANN DIE SELBE ADRESSE BIS ZU DREIMAL HINTEREINANDER AUF DEM DISPLAY ERSCHEINEN, DAS PROGRAMM SCHEINT EINEN MOMENT LANG NICHT MEHR WEITER ZU KOMMEN. ES IST SCHWER VORHERZU-SAGEN, IN WELCHEM MOMENT DER BREAKPOINT DANN LOESCHFAEHIG IST. (IN DIESEM FALL EVENTUELL MIT "KILL" STEPPEN.)

BEI DER BENUTZUNG VON BREAKPOINTS DEN INTERRUPTVEKTOR "SWIV" NICHT VERAENDERN!

BEI DER BENUTZUNG DES SSM DIE INTERRUPTVEKTOREN "NMIV" UND "IRQV" NICHT VERAENDERN!

*** *** *** *** *** ***

EINE GUENSTIGE EIGENSCHAFT DES IM EUROCOM 1 VERWENDETEN BREAKPOINT/SINGLE-STEP-KONZEPTES IST ES, DASS MAN AN JEDEM BREAKPOINT IN DEN SSM UND AUS DEM SSM JEDERZEIT ZURUECK IN DEN NORMALBETRIEB SCHALTEN KANN.
AUF DIESE WEISE KANN MAN DEN ZEITAUFWENDIGEN SSM AUF DIE ERFORDERLICHEN BEREICHE BESCHRAENKEN UND DIE UNINTERES-SANTEN BEFEHLSFOLGEN UEBERSPRINGEN.
ES IST DANN NUR NOETIG, AM ANFANG JEDEN BEREICHES, DEN MAN "DURCHSTEPPEN" WILL, EINEN BREAKPOINT ZU SETZEN.

```
********
* KILL *
********
```

"KILL", IN DER HKS BETAETIGT, VERANLASST DIE LOESCHUNG ALLER BREAKPOINTS, DEREN ADRESSEN IM BREAKPOINTSPEICHER STEHEN, WENN SIE DORT ALS N I C H T G E L O E S C H T (BEFEHLSWORT UNGLEICH 0) MARKIERT SIND.

AUS DIESEM GRUNDE IST ES NICHT EMPFEHLENSWERT, EINEN BREAKPOINT AUF EINE ADRESSE ZU SETZEN, DEREN INHALT GLEICH NULL IST.

****
* BRANCH *
****

DIESE FUNKTION BIETET DEM PROGRAMMIERER EINE HILFE
BEI DER BERECHNUNG RELATIVER ADRESSEN, WIE SIE FUER
DIE BRANCH-BEFEHLE (PROGRAMMVERZWEIGUNGEN)
BEIM MC 6802 BENOETIGT WERDEN.

AUS DER HKS, DER MEM-ROUTINE UND EINIGEN ANDEREN MONITOR-
FUNKTIONEN BEWIRKT DER DRUCK AUF DIE TASTE "B", DASS
AM RECHTEN RAND DER ANZEIGE "b" ERSCHEINT.
HIER IST DIE ADRESSE DES BRANCH-BEFEHLS, DESSEN OPERANDEN
MAN BERECHNEN WILL, EINZUGEBEN (ADRESSEINGABE).
DIE ADRESSE SPRINGT NACH LINKS, RECHTS ERSCHEINT DAS
WORT "to".
JETZT FOLGT WIEDER EINE ADRESSEINGABE, UND ZWAR DIE
ADRESSE DES SPRUNGZIELES.

DER COMPUTER RECHNET IN 2-COMPLEMENT-ARITHMETIK DEN
WERT

BANCHADRESSE+2 - ZIELADRESSE AUS,

ALSO GERADE DEN OPERANDEN DES BRANCHBEFEHLS.

UEBERSCHREITET DAS RESULTAT DEN BEREICH -128 BIS +127,
SO WIRD ERROR 8 ANGEZEIGT, WEIL OHNEHIN KEIN BRANCH
DIESEN BEREICH UEBERSCHREITEN KANN.

DAS RESULTAT WIRD DURCH " = XX" ANGEZEIGT,
WOBEI XX DER GESUCHTE OPERAND IST.

****
* RECORD *
****

DIESE TASTE DIENT DER AUFZEICHNUNG EINES SPEICHER-
BEREICHES AUF CASSETTENRECORDER.

NACH BETAETIGEN DER TASTE "RECORD" ERSCHEINT AUF DER
ANZEIGE "bEG". HIER WIRD DIE ANFANGSADRESSE DES AUFZU-
ZEICHNENDEN SPEICHERBEREICHS EINGEGEBEN (ADRESSEINGABE).

NACH DER VIERTEN ZIFFER ERSCHEINT AUF DER ANZEIGE "End",
ES FOLGT DIE EINGABE DER ENDADRESSE DES BEREICHS.
RECHTS AUF DER ANZEIGE ERSCHEINT NUN "r".
JETZT WIRD EINE SECHSSTELLIGE HEXADEZIMALE KENNUNG FÜR
DIE AUFZEICHNUNG EINGEGEBEN. SOLL OHNE KENNUNG (=HEADER)
AUFGEZEICHNET WERDEN, WIRD EINFACH DIE TASTE "BA" BETAE-
TIGT. DAS "r" SPRINGT DANN NACH LINKS, ALS SEI EINE KENN-
UNG GEGEBEN WORDEN.

JETZT KANN MAN DIE TASTE "UA" BENUTZEN, UM DIE MOTOR-
STROMSTEUERUNG (OPTIONAL) FUER DEN CASSETENRECORDER EIN-
ZUSCHALTEN, FALLS DAS ZU POSITIONIERUNGSZWECKEN NOETIG
SEIN SOLLTE. DER ZUSTAND "MOTORSTROM EIN" WIRD
DURCH BLINKEN DES "r" ANGEZEIGT.
MIT DER TASTE "DA" WIRD DANN DER MOTORSTROM WIEDER
ABGESCHALTET.
DAS EIGENTLICHE AUFZEICHNUNGSPROGRAMM SCHALTET SPAETER
DEN STROM SELBSTAENDIG.

NACHDEM DIE CASSETTE POSITIONIERT UND DER RECORDER AUF
AUFNAHME GESTELLT IST, BRAUCHT NUR NOCH EINMAL
DIE TASTE "BA" BETAETIGT ZU WERDEN. DIE AUFNAHME
BEGINNT, WAS DURCH "r-" ANGEZEIGT WIRD.
FUER EINE FEHLERFREIE AUFZEICHNUNG DARF KEINE TASTE MEHR
BETAETIGT WERDEN, BIS DER EUROCOM IN DIE HKS ZURUECK-
KEHRT. DABEI WIRD DER MOTORSTROM AUTOMATISCH WIEDER AB-
GESTELLT.

DA ZU BEGINN JEDER AUFZEICHNUNG 10 SEC DAUERTON ALS
INFORMATIONSFREIER VORSPANN UND AM ENDE 2 SEC DAUERTON ALS
NACHSPANN GESENDET WERDEN, DUERFTE ES NICHT SCHWERFALLEN,
DIE ENTSPRECHENDEN RASTSCHALTER AM RECORDER ZEITIG GENUG
VON HAND ZU BEDIENEN, FALLS KEIN MOTORSTROMRELAIS
VORHANDEN IST,

```
********
* LOAD *
********
```

DIESE FUNKTION DIENT DEM EINLESEN VON CASSETTENFILES, DIE MIT DER "RECORD"-FUNCTION AUFGEZEICHNET WURDEN.

NACH DEM DRUCK AUF DIE TASTE "LOAD" ERSCHEINT RECHTS AUF DER ANZEIGE "L."; JETZT KANN WIEDER EINE GENAU SECHS-STELLIGE KENNUNG ANGEGEBEN WERDEN, DIE DEN ZU LADENDEN FILE BEZEICHNET.
DIE KENNUNG KANN AUCH DURCH "BA" UEBERSPRUNGEN WERDEN, D.H., DASS DER NAECHSTE FILE (MIT ODER OHNE KENNUNG), DER AUF DER CASSETTE ZU FINDEN IST, GELADEN WERDEN SOLL.

NACH EINGABE DER KENNUNG IST DIE MOTORSTEUERUNG GENAUSO ZU HANDHABEN WIE IN DER "RECORD"-FUNCTION, DIE AUSFUEHRUNG WIRD EBENFALLS MIT "BA" GESTARTET, UND DURCH "L-" ANGEZEIGT. JETZT DUERFEN WEDER AM COMPUTER NOCH AM RECORDER MANIPULATIONEN VORGENOMMEN WERDEN, UM LESEFEHLER (ERROR 5 UND 7) ZU VERMEIDEN.

WURDE KEINE KENNUNG GEGEBEN, ERSCHEINT DIE KENNUNG DES FILES, DER GELADEN WIRD, WAEHREND DES LADEVORGANGS AUF DER ANZEIGE.

NACH ABSCLUSS DES LADEVORGANGS SCHALTET DIE LADE-FUNKTION DEN MOTORSTROM WIEDER AB UND DER PROZESSOR KEHRT IN DIE HKS ZURUECK.

ANHANG 1: FEHLERMELDUNGEN DES EUROCOM 1 - MONITORS

WAEHREND EINER FEHLERMELDUNG BLINKT DIE NUMMER DES FEHLERS AUF DER ANZEIGE.

WAEHREND EINER FEHLERMELDUNG BEFINDET SICH DER MONITOR LOGISCH SCHON WIEDER IN DER HAUPTKONTROLLSCHLEIFE.

ERROR	ERKLAERUNG
1	VERSUCH, IN DEN SPEICHER ZU SCHREIBEN, IST MISSLUNGEN, ETWA WEIL DIE ADRESSE SICH IN EINEM BEREICH BEFAND, IN DEN NICHT GESCHRIEBEN WERDEN KANN (ROM, BESTIMMTE PORTREGISTER, NICHT BELEGTE ADRESSEN). SCHLIMMSTENFALLS IST DER RAM SCHADHAFT.
2	DASSELBE WIE ERROR 1, ABER BEIM SETZEN EINES BREAKPOINTS.
3	ES WURDE VERSUCHT, MEHR ALS 15 BREAKPOINTS ZU SETZEN.
4	VERSUCH, EINEN BREAKPOINT AUF EINEN SPEICHERPLATZ ZU SETZEN, AUF DEM SCHON DER BEFEHL SWI(3F) STEHT, Z.B. WEIL DORT SCHON EIN BREAKPOINT GESETZT WURDE, DER NOCH NICHT WIEDER GELOESCHT IST.
5	BEIM LESEN VOM CASSETTENRECORDER WURDE EIN ASCII-CODE GEFUNDEN, DER KEINEM HEXADEZIMALZEICHEN ENTSPRICHT. GRUENDE: ENTWEDER HANDELT ES SICH UM EINE GRUNDSAETZLICH FALSCH FORMATIERTE CASSETTE ODER DIE AUFNAHME BZW. DIE WIEDERGABE WURDE GESTOERT DURCH UNZEITGEMAESSE TASTENBETAETIGUNG AM RECORDER BZW. AM EUROCOMP ODER DIE INFORMATION IST GESTOERT DURCH MAENGEL DES BANDMATERIALS BZW. DES RECORDERS.
6	WIE ERROR 1, ABER IN DER "LOAD"-ROUTINE.
7	PRUEFSUMMENFEHLER (CHECKSUM-ERROR) IN DER "LOAD"-FUNKTION, GRUENDE WIE ERROR 5
8	UEBERSCHREITUNG DES BRANCHBEREICHS BEI DER B-FUNKTION.

ANHANG 2: CASSETTEN- UND FILEFORMAT DES EUROCOM 1

DIE AUFZEICHNUNG AUF CASSETTE ERFOLGT NATUERLICH SERIELL, D.H. DIE 8 BIT EINES BYTE WERDEN NACHEINANDER, BEGINNEND MIT DEM HOECHSTWERTIGEN BIT, AUFGEZEICHNET. DA SICH AUF EINER CASSETTE KEINE GLEICHSPANNUNGEN AUFZEICHNEN LASSEN, MUSS DER WERT EINES BIT, 0 ODER 1, DURCH EINE FREQUENZ VERSCHLUESSELT WERDEN.
DIESE METHODE HEISST FREQUENCY-SHIFT-KEYING (FSK). DER BEIM EUROCOM 1 VERWENDETE "KANSAS-CITY-STANDARD" BESAGT, DASS FUER EINE 0 EIN TON VON 1200 HZ, FUER EINE 1 EIN TON VON 2400 HZ VERWENDET WIRD.

DER ASYNCHRONE INTERFACE-ADAPTER (ACIA) DES EUROCOM ARBEITET MIT 300 BAUD (BITS/SEC). WENN, Z.B. DURCH GLEICHLAUFSCHWANKUNGEN, DAS TIMIMG VON ACIA UND RECORDER NICHT HUNDERTPROZENTIG UEBEREINSTIMMT, KOENNTE LEICHT EINE VERSCHIEBUNG UM 1 BIT AUFTRETEN, WAS ZUM VOELLIGEN ZUSAMMENBRUCH EINER VERNUENFTIGEN INFORMATIONSUEBERTRAGUNG FUEHREN WUERDE. DESWEGEN WERDEN ANFANG UND ENDE EINES JEDEN BYTE MARKIERT:
JEDEM BYTE WIRD EIN STARTBIT VORAUSGESCHICKT, DAS IMMER 0 IST UND ZWEI STOPBITS NACHGESCHICKT, DIE IMMER 1 SIND. DIE STOPBITS DIENEN ALS ZEITLICHE PUFFERZONE, DAS "SYNCHRONISATIONSSIGNAL" FUER DEN BYTEANFANG IST DER SPRUNG VOM STOPBIT (=1) ZUM STARTBIT (=0). AUF DIESE WEISE KOENNEN KLEINERE UNTERSCHIEDE IM TIMING, DIE SICH NACH EINIGEN BYTE ZUR DESYNCHRONISATION AKKUMULIEREN WUERDEN, IMMER WIEDER KOMPENSIERT WERDEN.

SOFTWARESEITIG ARBEITET DER EUROCOM 1 MIT DEM MOTOROLA-FORMAT. DABEI WIRD EIN DATENBYTE DURCH ZWEI ASCII-CHARACTERS DARGESTELLT, DIE DEN BEIDEN HEXADEZIMALEN ZEICHEN ENTSPRECHEN, MIT DENEN EIN BYTE UEBLICHERWEISE WIEDERGEGEBEN WIRD; Z.B. WIRD EIN BYTE, DAS IN BINAERSCHREIBWEISE DIE FORM "11001010" HAT (=HEXADEZIMAL "CA"), AUF CASSETTE GESCHRIEBEN DURCH DIE BEIDEN BYTES "01000011" UND "01000001" (HEXADEZIMAL 43 UND 41), WAS GENAU DEN ASCII-CODES FUER DIE BUCHSTABEN "C" UND "A" ENTSPRICHT. DIE SCHEINBARE VERSCHWENDUNG, DIE IN DIESEM FORMAT LIEGT, WIRD MEHR ALS NUR AUSGEGLICHEN ERSTENS DURCH DIE LEICHTE UND SICHERE FESTSTELLBARKEIT VON FEHLERN DER AUFZEICHNUNG UND ZWEITENS DURCH DIE UNIVERSELLE KOMPATIBILITAET. MAN KOENNTE Z.B. EINE DERARTIGE AUFZEICHNUNG VOM CASSETTENRECORDER (NATUERLICH DEMODULIERT) AUF EINEN TELETYPE GEBEN UND HAETTE SOFORT EIN VERSTAENDLICHES LISTING DER AUFZEICHNUNG.

DER EUROCOM 1 ZEICHNET SEINE PROGRAMME AUF DEM CASSETTENRECORDER ALS "FILES" AUF, D.H. SO, DASS ER BEIM WIEDERLESEN DIE ANFAENGE DER PROGRAMME ERKENNEN UND UNTERSCHEIDEN KANN, OHNE DIE GANZE CASSETTE VON VORNE LESEN ZU MUESSEN. ZU DIESEM ZWECK UND ZUR SYNCHRONISATION MIT DEM ERSTEN STARTBIT WIRD JEDER FILE MIT EINEM DAUERTON, DER ALSO KEIN STARTBIT ENTHAELT, BEGONNEN UND BEENDET. DAMIT IST AUCH GENUG ZEIT FUER DIE MANUELLE BEDIENUNG DES RECORDERS.

DER AUFBAU EINES EUROCOM-FILE (MOTOROLA-FORMAT)

IM FOLGENDEN BEDEUTET JEDE ZWEISTELLIGE ZAHL
EBENSO WIE JEDES XX EINE ASCII-CODEZAHL.

AUSGEGEBENE ASCII-CODES	BEDEUTUNG BEZUEGLICH PROGRAMM, TELETYPE UND LOCHSTREIFEN
10"DAUERTON	KEINE WIRKUNG
10*00	VORSPANN AUS TRANSPORTLOECHERN
0D,0A	WAGENRUECKLAUF, ZEILENVORSCHUB
00,00,00,00	ZEIT FUER WAGENRUECKLAUF
53,30	"S0", ZEICHEN FUER HEADERBLOCK-ANFANG
XX,XX,XX	NAME DES FILE IN KLARTEXT,
XX,XX,XX	INSGESAMT 6 ASCII-ZEICHEN
0D,0A	WAGENRUECKLAUF, ZEILENVORSCHUB
00,00,00,00	ZEIT FUER WAGENRUECKLAUF, ENDE DES HEADERBLOCKS

FALLS BEI DER AUFZEICHNUNG DURCH DEN EUROCOM 1 KEINE KENNUNG ANGEGEBEN WURDE, WIRD KEIN HEADERBLOCK AUFGEZEICHNET.

53,31	"S1", ZEICHEN FUER ANFANG PROGRAMMZEILE
XX,XX	"ZEILENLAENGE"=L+3, WOBEI L DIE ANZAHL DER PROGRAMMBYTES IN DIESER ZEILE IST; DIE +3 RUEHREN VON ZEILENLAENGE UND ANFANGSADRESSE HER. DIE TATSAECHLICHE LAENGE DIESER ZEILE IN ASCII-ZEICHEN BETRAEGT ALSO 2*(L+3).
XX,XX,XX,XX	ANFANGSADRESSE DIESER ZEILE IM SPEICHER
XX,XX	1.STES PROGRAMMBYTE DIESER ZEILE
XX,XX	2.TES PROGRAMMBYTE
.	
.	USW.
.	
XX,XX	L-TES PROGRAMMBYTE DIESER ZEILE
0D,0A	WAGENRUECKLAUF, ZEILENVORSCHUB
00,00,00,00	ZEIT FUER WAGENRUECKLAUF

JETZT WERDEN SOVIELE PROGRAMMZEILEN IN DIESEM FORMAT AUSGEGEBEN WIE NOETIG. JEDE ZEILE BEGINNT MIT "S1" UND ENDET MIT WAGENRUECKLAUF UND ZEILENVORSCHUB.

MAN BEACHTE, DASS DER MAXIMALE WERT FUER L = 252 IST. DER EUROCOM 1 VERWENDET IN DER REGEL L=24. DAS KANN MAN ABER AENDERN (SPEICHERPLATZ "LENGTH", SH. ANHANG 4)

53,39	"S9", ZEICHEN FUER ENDE DES PROGRAMMS
2" DAUERTON	KEINE WIRKUNG

ANHANG 3: ADRESSBELEGUNG DES EUROCOM 1

ADRESSE	HARDWARE	BEDEUTUNG
0000 --007F	CPU-RAM	FREI FUER ANWENDER, DIRECT ADRESSING.
0080 --7FFF	NICHT BELEGT	FREI FUER SPEICHERERWEITERUNGEN
8004	PIA 1 2	DATA-REGISTER A DES USER-PORTS
8005	PIA 1 2	DATA-REGISTER B DES USER-PORTS
8006	PIA 1 2	CONTROL-REGISTER A DES USER-PORTS
8007	PIA 1 2	CONTROL-REGISTER B DES USER-PORTS
8008	ACIA	PORT FUER SERIELLEN I/O, IST AN
8009	ACIA	CASSETTENINTERFACE ANGESCHLOSSEN
8010 --8013	PIA 2 1	TASTATUR, ANZEIGE UND EINZELSCHRITTSTEUERUNG
A400 --AXXX	RAM	FREI FUER ANWENDER BIS AXXX. AXXX WIRD DURCH DEN STACKPOINTER ANGEZEIGT, DER BEI INTERRUPTS, JSR, BSR UND PUSH-BEFEHLEN DEKREMENTIERT WIRD. UEBERLAPPUNGEN ZWISCHEN BENUTZERPROGRAMM UND STACK HABEN KATASTROPHALE FOLGEN FUER DAS PROGRAMM.
AXXX+1		STACKSPITZE (TOP)
--A7A7		STACKANFANG (BOTTOM)
A7A8 --A7FF		MONITOR ARBEITSBEREICH
F800 --FFFF	ROM	MONITOR FIRMWARE

ANHANG 4: WICHTIGE ADRESSEN IM MONITOR-ARBEITSBEREICH

ADRESSE	SYMB. NAME	BEDEUTUNG
A7A7	STACK	HOECHSTE ADRESSE DES STACK
A7A8 -A7AF	DISPL	ANZEIGEPUFFER, ENTHAELT 7-SEGMENTCODE, DER DURCH DIE MONITOR-SUBROUTINE "DISP" AUF DIE ANZEIGE GEBRACHT WIRD. DER INHALT VON #$A7A8 WIRD AUF DEM ANZEIGEELEMENT GANZ LINKS DARGESTELLT, DER INHALT VON #$A7A9 AUF DEM NAECHSTEN ETC.
A7B0 -A7B7		ASCII-PUFFER, ENTHAELT ASCII-CODE, DER DURCH DIE MONITOR-SUBROUTINE "BUILD" IN 7-SEGMENT-CODE UMGEWANDELT UND IM ANZEIGEPUFFER ABGELEGT WIRD (#$A7B0 NACH #$A7A8, #$A7B1 NACH #$A7A9 ETC.).
A7B8	CARRY	WIRD DURCH DIE "SHIFT"-ROUTINEN VON RECHTS IN DEN ASCII-PUFFER GESCHOBEN.
A7B9	KEY	HIER LEGT DIE "DISP"-ROUTINE DIE CODEZAHL FUER EINE EVTL. GEDRUECKTE TASTE AB.
A7BE A7BF	NMIV	INTERRUPTVEKTOR FUER NMI, WIRD BEI "RESET" MIT #$FB4C="SSLOOP" VORBESETZT.
A7C0 A7C1	SWIV	INTERRUPTVEKTOR FUER SWI, WIRD BEI "RESET" MIT #$FAEB="SWIR" VORBESETZT.
A7C2 A7C3	IRQV	INTERRUPTVEKTOR FUER IRQ, WIRD WIE "NMIV" VORBESETZT.
A7C4	LENGTH	ANZAHL PROGRAMMBYTES JE DATENSATZ DER "RECORD"-FUNKTION, WIRD BEI "RESET" MIT #24 VORBESETZT.
A7D3 -A7FF		BREAKPOINTSPEICHER, WIRD FOLGENDERMASSEN BENUTZT:

JE DREI AUFEINANDERFOLGENDE SPEICHERPLAETZE BESCHREIBEN EINEN BREAKPOINT, Z.B.:

A7XX	PAGEBYTE DER BREAKPOINTADRESSE
A7XX+1	LOWBYTE DER ADRESSE
A7XX+2	ORIGINALINHALT DES BREAKPOINTS

DABEI KANN "XX" DIE WERTE D3, D6, D9, DC, DF, ... ETC. ANNEHMEN.

DER BREAKPOINTSPEICHER SPIELT EINE ENTSCHEIDENDE ROLLE BEI DER RESTAURATION VON BREAKPOINTS.

1. DIE BETAETIGUNG DER TASTE "RESET" ZU EINEM BELIEBIGEN ZEITPUNKT LOEST STETS FOLGENDEN ABLAUF AUS:

DER BREAKPOINTSPEICHER WIRD DARAUFHIN DURCHSUCHT, OB DORT IRGENDEIN BREAKPOINT ALS "AKTIV" VERMERKT IST, DH., DASS EIN "ORIGINALINHALT" UNGLEICH 00 IST. FALLS EIN SOLCHER ORIGINALINHALT GEFUNDEN WIRD, WIRD ER IN DIE ZELLE UEBERTRAGEN, DIE DURCH "BREAKPOINTADRESSE" ANGEGEBEN WIRD (RESTAURATION).
DANN WIRD IN DIE STELLE "ORIGINALINHALT" DES BREAKPOINT-SPEICHERS 00 GESCHREIBEN, UM DEN BREAKPOINT ALS GELOESCHT ZU MARKIEREN (LOESCHUNG).

2. DER GLEICHE ABLAUF WIRD AUSGELOEST, WENN "KILL" IN DER HKS BETAETIGT WIRD.

MAN BEACHTE, DASS DIE RESTAURATION NICHT DAVON ABHAENGT, OB AN DER BREAKPOINTADRESSE IM BENUTZERPROGRAMM DER BREAKPOINTCODE 3F STEHT, SONDERN NUR DAVON, OB "ORI-GINALINHALT" IM BREAKPOINTSPEICHER UNGLEICH 00 IST.

3. ANDERS VERLAEUFT DIE SACHE, WENN DIE MASCHINE BEI DER ABARBEITUNG DES PROGRAMMS AUF DEN BEFEHL SWI LAEUFT. DER PROZESSOR SPRINGT DANN BEKANNTLICH AUF DIE ADRESSE, DIE IM INTERRUPTVEKTOR "SWIV" STEHT (SH. OBEN), ALSO IN DEN BREAKPOINTSERVICE (SOFERN "SWIV" VOM BENUTZER NICHT VERAENDERT WURDE).

WENN DER BREAKPOINTSERVICE MIT "GO", "STEP", "BA" ODER "KILL" WIEDER VERLASSEN WIRD, WIRD ZUNAECHST GEPRUEFT: IST DIE ADRESSE DIESES BREAKPOINTS IM BREAKPOINTSPEICHER ENTHALTEN?

FALLS NEIN: DER MONITOR GEHT SOFORT ZURUECK IN DIE HKS.

FALLS JA: DER "ORIGINALINHALT" AUS DEM BREAKPOINTSPEI-CHER WIRD IN DIE SPEICHERZELLE GESCHRIEBEN, AUF DIE DER PC DES BENUTZERPROGRAMMS ZEIGT. WENN DAS PC-REGISTER IN DER RR NICHT VERAEN-DERT WURDE, IST DIES GERADE DIE ADRESSE DES SWI, DER DEN BREAK VERURSACHTE.
ANSCHLIESSEND WIRD "ORIGINALINHALT" IM BREAK-SPEICHER DURCH 00 UEBERSCHRIEBEN, UM DEN BREAKPOINT ALS GELOESCHT ZU MARKIEREN.

ANHANG 5: WICHTIGE ADRESSEN IM MONITOR FUER DEN BETRIEB VON TASTATUR UND ANZEIGE

UM DEM ANWENDER UNNOETIGEN PROGRAMMIERAUFWAND ZU ERSPAREN, SIND EINIGE UNTERPROGRAMME DES MONITOR FUER TASTATUR-INPUT UND ANZEIGE SO GEHALTEN, DASS SIE VOM ANWENDER AUFGERUFEN WERDEN KOENNEN.

ADRESSE	SYMB. NAME	BEDEUTUNG
F831	DISP4	FUEHRT "DISP" VIERMAL AUS
F833	DISP2	FUEHRT "DISP" ZWEIMAL AUS
F835	DISP	STELLT DEN 7-SEGMENTCODE DES ANZEIGEPUFFERS E I N M A L AUF DER ANZEIGE DAR. PRUEFT WAEHRENDDESSEN, OB EINE TASTE GEDRUECKT WURDE UND LEGT GEGEBENENFALLS DEREN CODE IN "KEY" AB. EIN STEHENDES BILD ERHAELT MAN DURCH AUFRUF VON "DISP" IN EINER SCHLEIFE.
F883	BUILD	KONVERTIERT DEN ASCII-PUFFER IN 7-SEGMENT-CODE, DER IM ANZEIGEPUFFER ABGELEGT WIRD.
F902	SHIF88	FUEHRT SHIFT8 ACHTMAL AUS
F904	SHIF48	FUEHRT SHIFT8 VIERMAL AUS
F906	SHIF28	FUEHRT SHIFT8 ZWEIMAL AUS
F908	SHIFT8	SCHIEBT ASCII-PUFFER UND "CARRY" UM EINE STELLE NACH LINKS, D.H. DER AM WEITESTEN LINKS STEHENDE CHARACTER GEHT VERLOREN, DIE UEBRIGEN CHARACTERS WANDERN UM EINE STELLE NACH LINKS UND DER INHALT VON "CARRY" WIRD AUF DIE LETZTE STELLE RECHTS GESETZT.
F876	WAIT	BENUTZT "BUILD", UM DEN INHALT DES ASCII-PUFFERS AUF DIE ANZEIGE ZU BRINGEN; BENUTZT "DISP", UM DIE ANZEIGE SOLANGE STEHEN ZU LASSEN, BIS EINE TASTE GEDRUECKT WIRD, DEREN CODEZAHL IN "KEY" ABGELEGT WIRD.
F8AD	INDIS4	FUEHRT "INDIS1" VIERMAL AUS.
F8AF	INDIS2	FUEHRT "INDIS1" ZWEIMAL AUS.
F8B1	INDIS1	SCHIEBT ASCII-PUFFER UND "CARRY" UM EINE STELLE NACH LINKS, WARTET AUF EINGABE. FALLS EINE HEXADEZIMALTASTE GEDRUECKT WURDE, SETZT "INDIS1" DEN ENTSPRECHENDEN ASCII-CODE IN "CARRY" AB UND KEHRT ZUM RUFENDEN PROGRAMM ZURUECK. NACH DER RUECKKEHR IST DAS INDEXREGISTER UM 1 ERHOEHT. FALLS EINE ANDERE TASTE GEDRUECKT WURDE, ERFOLGT DIE RUECKKEHR JEDOCH ZUR HAUPTKONTROLLSCHLEIFE. DORT WIRD DIE MIT DEM DRUCK AUFGERUFENE FUNKTION SOFORT GESTARTET.

F97C	INBYTE	SPALTET EIN BYTE IN ZWEI HEXADEZIMAL-TEILE, CODIERT DIESE IN ASCII UND SCHIEBT SIE VON RECHTS IN DEN ASCII-PUFFER. DIE ADRESSE DES ZU KONVERTIERENDEN BYTE IST VORHER INS INDEXREGISTER ZU GEBEN. DAS INDEXREGISTER WIRD UM 1 ERHOEHT.
F97A	INBYT2	FUEHRT INBYTE ZWEIMAL AUS, D.H. DIE BEIDEN ZU KONVERTIERENDEN BYTES MUESSEN IN AUFEINANDERFOLGENDEN SPEICHERPLAETZEN LIEGEN.
F929	BYTE	KONVERTIERT ZWEI HEXADEZIMALZEICHEN VOM ASCII-CODE ZU EINEM BYTE, IST ALSO DAS GEGENSTUECK ZU "INBYTE". DAS INDEXREGISTER MUSS MIT DER ADRESSE DES HOEHERWERTIGEN HEXADEZIMALZEICHENS VORBESETZT SEIN, DAS NIEDRIGERWERTIGE MUSS AUF DER NAECHSTHOEHEREN ADRESSE LIEGEN. NACH DER RUECKKEHR IST DAS INDEXREGISTER UM ZWEI ERHOEHT UND DAS RESULTAT LIEGT IM ACCU A VOR.
FA54	CNTRL	RUECKKEHR-ADRESSE ZUR HAUPTKONTROLLSCHLEIFE.
FE82	CTABLE	CODETAFEL FUER KONVERSIONEN ZWISCHEN ASCII, 7-SEGMENTCODE UND TASTATURCODE. AUFBAU DER TABELLE:

ADRESSE	INHALT	BEDEUTUNG
FE82	30	ASCII-CODE FUER "0"
FE83	C0	7-SEGMENT-CODE FUER "0"
FE84	10	CODE FUER TASTE "0"
FE85	31	ASCII-CODE FUER "1"

USW.

DIE TABELLE ENTHAELT KEINESWEGS ALLE ASCII-ZEICHEN, SONDERN NUR DIE ENGE AUSWAHL, DIE VOM EUROCOM 1 TATSAECHLICH GEBRAUCHT WIRD. ALLE UEBRIGEN ASCII-CODES WERDEN VON "BUILD" ZU BLANKS KONVERTIERT.

Anleitung zur Initialisierung des Parallel-Interface-Adapters (PIA) vom EUROCOM-Mikrocomputer

Der PIA ist ein paralleler Ein-Aussabe-Baustein mit 2 mal 8 parallelen Ein- oder Aussabe-Leitungen, zu denen je 2 Steuerleitungen (CA1, CA2 und CB1, CB2) gehoeren. Intern beinhaltet der PIA je zwei (im Folgenden mit A und B unterschieden) Datenregister, Datenrichtungsregister und Kontrollregister. Die Funktionsweise des PIA-Bausteins wird durch die Programmierung der beiden Kontrollregisters bestimmt. Die Kontrollworte setzen sich aus folgenden Teilen zusammen:

Bit 0 und 1: Arbeitsweise des Interrupt-Eingangs 1 (CA1 und CB1)

Bit 2: entscheidet ueber den Zugriff zum Datenregister oder Datenrichtungsregister

Bit 3 bis 5: Arbeitsweise der Steuerleitung 2 (CA2 und CB2)

Bit 6: Kontroll-Bit der Steuerleitung 2 (IRQA2 und IRQB2)

Bit 7: Kontroll-Bit des Interrupt-Eingangs 1 (IRQA1 und IRQB1)

Zusammengefasste Darstellung der Kontrollregister:

Bit:	7	6	5	4	3	2	1	0
A:	IRQA1	IRQA2	CA2 Control			DDRA Access	CA1 Control	
B:	IRQB1	IRQB2	CB2 Control			DDRB Access	CB1 Control	

Nach Betaetigen der RESET-Taste (logischer Pegel 0 am Reset-Eingang des PIA) sind alle Register des PIA geloescht (alle Bit sind auf 0) Dadurch muss der PIA am Anfang eines Programmes initialisiert werden, ehe er die gewuenschte Funktion hat. Aber auch, wenn waehrend des Programms die Arbeitsweise des PIA geaendert werden soll, muss eine neue Initialisierung vorgenommen werden. Da die einzelnen Daten-Anschluesse des PIA wahlweise als Ein- oder Ausgaenge arbeiten koennen, muss die Richtung der Daten zunaechst im Datenrichtungsregister festgelegt werden. Um aber auf das Datenrichtungsregister zugreifen zu koennen, muss das Bit 2 im Kontrollregister Null sein. Die Funktion dieses Bit ist also folgende:

Bit 2 = 0: Zugriff auf das Datenrichtungsregister
1: Zugriff auf das Datenregister

Nach dem Loeschen von Bit 2 im Datenresister (oder einfach des ganzen Kontrollresisters mit dem "CLR"-Befehl) wird festselegt, ob die einzelnen Bit des PIA als Eingaenge oder Ausgaenge arbeiten sollen. Abspeichern einer logischen 1 bewirkt, dass das entsprechende Bit als Ausgang arbeitet, Abspeichern einer logischen 0, dass das Bit als Eingang arbeitet. Dies sei an einem Beispiel noch naeher erlaeutert. Im folgenden bedeuten:

PIAADR = PIA-Teil A Datenresister oder Datenrichtungs-resister, je nach Zustand von Bit 2 im Kontroll-resister A (Datenresister und Datenrichtungs-resister haben also die gleiche Adresse)
PIAACR = PIA-Teil A Kontrollresister
PIABDR = PIA-Teil B Datenresister oder Datenrichtungs-resister, je nach Zustand von Bit 2 im Kontroll-resister B (beide haben auch hier die gleiche Adresse).
PIABCR = PIA-Teil b Kontrollresister

```
Die Befehle   CLR PIAACR     ermoeglicht Zugriff auf Daten-
                             richtungsregister
              LDA A #$FF     alle 8 Bit auf 1 setzen
              STA A PIAADR   PIA-Teil A hat 8 Ausgaenge
```

bewirken also, dass der PIA-Teil A einen 8-Bit Parallel-Ausgang darstellt. Somit kann man durch einen Store-Befehl (STA oder STX) das Bitmuster, das sich gerade in dem entsprechendem Accumulator befindet, statisch am Ausgang des PIA erscheinen lassen, d.h., die Ausgaenge bleiben solange in dem entsprechenden logischen Zustand bis sie durch einen neuen Store-Befehl veraendert werden.

```
Die Befehle   CLR PIABCR     ermoeglicht Zugriff auf Daten-
                             richtungsregister
              CLR PIABDR     alle 8 Bit auf 0 setzen
                             PIA-Teil B hat 8 Eingaenge
```

bewirken also, dass der PIA-Teil B einen 8-Bit Parallel-Eingang darstellt. Somit kann man durch einen Load-Befehl (LDA oder LDX) das Bitmuster, das gerade am Eingang ansteht, in den entsprechenden Accumulator laden.

```
Die Befehle   CLR PIABCR     ermoeglicht Zugriff auf Daten-
                             richtungsregister
              LDA A #$0F     die unteren 4 Bit des Daten-
                             resisters sind Ausgaenge, die
              STA A PIABCR   oberen 4 Bit sind Eingaenge
```

ermoeglichen den gleichzeitigen Anschluss von 4 Ausgangs- und 4 Eingangsleitungen an den Anschluessen eines Daten-resisters.

Als naechstes muss nun das Bit 2 im Kontrollresister gesetzt werden, damit nicht mehr auf das Datenrichtungs-resister, sondern auf das Datenresister zugegriffen werden kann. Gleichzeitig sollte man aber auch die Funktion von Interrupt- und Steuer-Leitung bestimmen, um mit einem Store-Befehl in das Kontrollresister alle Funktionen des PIA zu definieren.

Programmierung des Interrupt-Einganges CA1
(Gleiches gilt im Folgenden auch fuer CB1)

Der flankengetriggerte CA1-Eingang des PIA dient dazu, den Mikrokomputer mit dem Takt einer an den PIA angeschlossenen Peripherie zu synchronisieren, wobei die Reaktion des PIA wahlweise bei der positiven oder negativen Flanke des Taktsignals erfolgen kann. Die Reaktion des PIA besteht darin, dass bei der entsprechenden Flanke im Kontrollregister das Bit 7 gesetzt wird, und bei entsprechender Programmierung des Kontrollregisters auch ein Hardware-Interrupt (die Interrupt-Ausgangsleitung IRQA geht auf low) ausgeloest werden kann. Wird kein Hardware-Interrupt ausgeloest, muss per Programm das Bit 7 abgefragt werden. Die Warteschleife, die auf das Setzen von Bit 7 wartet, kann folgendermassen aussehen:

```
WAIT  LDA A PIACR   Kontrollregister laden. Ist Bit 7 gesetzt?
      BPL WAIT      Bit 7 noch 0, weiter warten
      LDA A PIADR   Ja, Bit 7 ist 1, neue Eingabe von PIA laden
```

Wichtig ist noch, dass das Bit 7 nicht per Programm geloescht werden kann (CLR-Befehl), sondern erst durch das Lesen des Datenregisters (LDA A PIADR wie oben) automatisch geloescht wird. Die Funktion des CA1-Einganges kann wie folgt programmiert werden:

Bit 0 = 0: das Bit 7 wird durch den CA1-Eingang gesetzt, aber der Interrupt ist abgeschaltet (disabled)
1: das Bit 7 wird durch den CA1-Eingang gesetzt und loest gleichzeitig einen Interupt aus (IRQ)

Bit 1 = 0: Bit 7 wird bei der negativen Flanke von CA1 gesetzt
1: Bit 7 wird von der positiven Flanke von CA1 gesetzt

Programmierung der Steuerleitung CA2 als Eingang
(Gleiches gilt im Folgenden auch fuer CB2)

Die CA2-Steuerleitung kann wahlweise als Eingang oder als Ausgang betrieben werden. Wird die CA2-Steuerleitung als Eingang betrieben, ist die Arbeitsweise die gleiche wie beim CA1-Eingang. Bei Erscheinen der im Kontrollregister programmierten Flanke am Eingang wird Bit 6 im Kontrollregister gesetzt, was bei entsprechender Programmierung auch einen Interrupt ausloesen kann. Wird kein Interrupt ausgeloest, muss per Programm das Bit 6 abgefragt werden. Die Warteschleife, die auf das Setzen von Bit 6 wartet, kann folgendermassen aussehen:

```
WAIT  LDA A PIABCR  Kontrollregister laden
      ASL A         schiebt im Accumulator A das Bit 6
                    an die Stelle von Bit 7
      BPL WAIT      Bit 7 (verschobenes Bit 6) ist nicht
                    gesetzt, Kontrollregister erneut pruefen
      LDA A PIABDR  Bit 7 ist im Accumulator A gesetzt, laden
                    des Datenregisters loescht Bit 6 im Kontroll-
                    register wieder
```

Auch hier kann Bit 6 im Kontrollregister erst durch das nachfolgende Lesen des Datenregisters geloescht werden und nicht durch einen "CLR"-Befehl.

Die Funktion der CA2-Steuerleitung als Eingang kann wie folgt programmiert werden:

Bit 5 = 0: CA2 ist wie CA1 ein Interrupt-Eingang

Bit 3 = 0: das Bit 6 wird durch den CA2-Eingang gesetzt, aber der Interrupt ist abgeschaltet (disabled)
1: das Bit 6 wird durch den CA2-Eingang gesetzt und loest gleichzeitig einen Interrupt aus (IRQ)

Bit 4 = 0: Bit 6 wird bei der positiven Flanke von CA2 gesetzt
1: Bit 6 wird bei der negativen Flanke von CA2 gesetzt

Programmierung der Steuerleitung CA2 als Ausgang
(Gleiches gilt im folgenden auch fuer CB2)

Arbeitet die CA2-Steuerleitung als Ausgang, sind zwei verschiedene Betriebsarten moeglich. In der einen Betriebsart kann der logische Pegel des CA2-Ausgangs direkt vom Programm bestimmt werden, was zum Erzeugen von Strobe- und Trigger-Impulsen per Programm verwendet werden kann. In der zweiten Betriebsart arbeitet der PIA im "HANDSHAKE-MODE". Hierbei kann mit dem CA2-Ausgang ein externes Geraet mit dem Mikroprozessor automatisch synchronisiert werden (naeheres siehe unten). Fuer die Bestimmung des CA2-Ausgangspegels per Programm muss das Kontrollregister wie folgt programmiert werden:

Bit 5 = 1: CA2 ist ein Steuerausgang

Bit 4 = 1: der Pegel des CA2-Ausganges wird durch den logischen Zustand von Bit 3 bestimmt

Bit 3 = 0: der CA2-Ausgang ist Null (low)
1: der CA2-Ausgang ist Eins (high)

Programmierung der Steueraussaenge CA2 und CB2 fuer den "HANDSHAKE-MODE"

Da der PIA-Teil A fuer Lesen und der PIA-Teil B fuer Schreiben besonders ausgestattet sind, haben beide Teile abhaengig von Bit 3 im Kontrollregister unterschiedliche Funktionen.

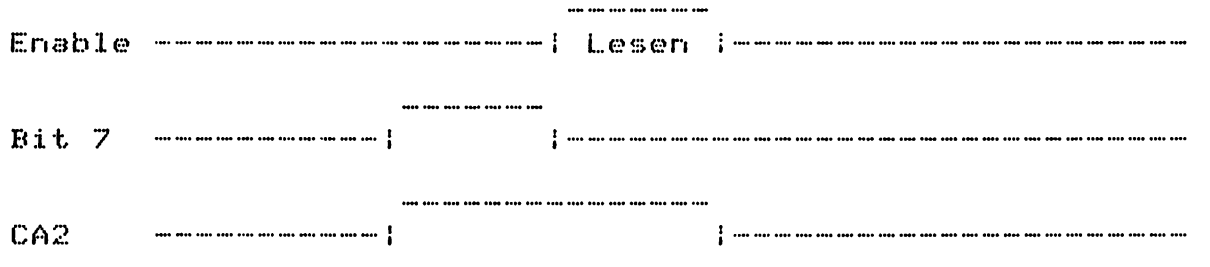
Programmierung des Steuerausganges CA2 fuer den "HANDSHAKE-MODE"

Bit 5 = 1: CA2 ist ein Steuerausgang

Bit 4 = 0: der PIA ist im "HANDSHAKE-MODE"

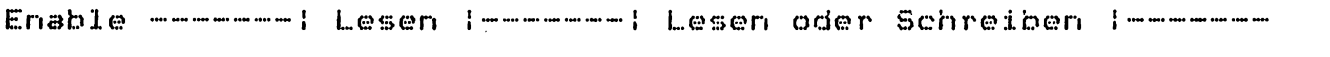
Bit 3 = 0: der CA2-Ausgang gibt den logischen Zustand von Bit 7 im Kontrollregister A wieder

Wird Bit 7 im Kontrollregister A durch die programmierte Flanke am CA1-Eingang gesetzt, wird der CA2-Ausgang ebenfalls high. Der Impuls am CA1-Eingang kann die Anmeldung zur Datenuebergabe eines externen Geraetes (z. B. Lochstreifenleser) sein. Der High-Pegel am CA2-Ausgang kann das Geraet anhalten, bis das anstehende Datenwort vom Mikrokomputer eingelesen wurde. Geht nach dem Lesen des Datenregisters A Bit 7 im Kontrollregister A wieder auf 0, wird der CA2-Ausgang automatisch ebenfalls auf 0 gesetzt (genauer: wenn der Enable-Eingang nach dem Lesen des Datenregisters A wieder auf 0 geht, wird CA2 ebenfalls wieder 0). Das angeschlossene Geraet ist wieder freigegeben und kann das naechste Datenwort uebermitteln.



Bit 3 = 1: der CA2-Ausgang wird durch den Enable-Eingang gesteuert

Der CA2-Ausgang ist im Ruhezustand high. Nach dem Lesen des Datenregisters geht der CA2-Ausgang solange auf low, bis der PIA ein weiteres mal adressiert wurde (CA2 wird wieder high nach der naechsten negativen Flanke von Enable). Dies wird verwendet, wenn nach dem Lesen eines Datenwortes eine weitere Ein- oder Ausgabe als Reaktion erforderlich ist. Ein externes Geraet kann auf das Low-Signal hin ein weiteres Datenwort an den PIA ausgeben oder ein Datenwort von dem PIA uebernehmen.



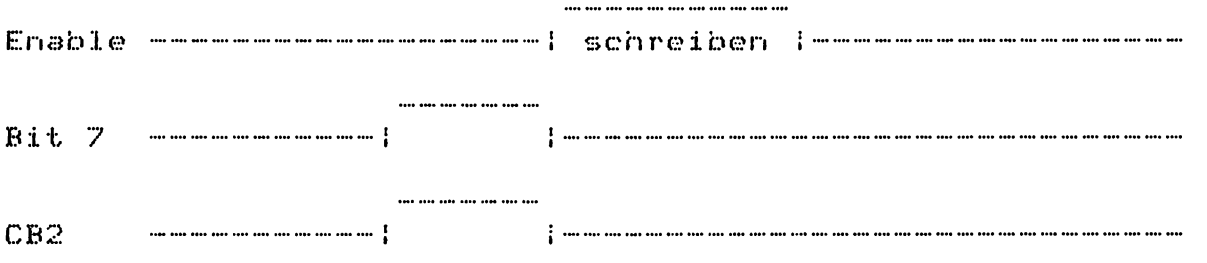
Programmierung des Steuerausganges CB2 fuer den "HANDSHAKE-MODE"

Bit 5 = 1: CB2 ist ein Steuerausgang

Bit 4 = 0: der PIA ist im "HANDSHAKE-MODE"

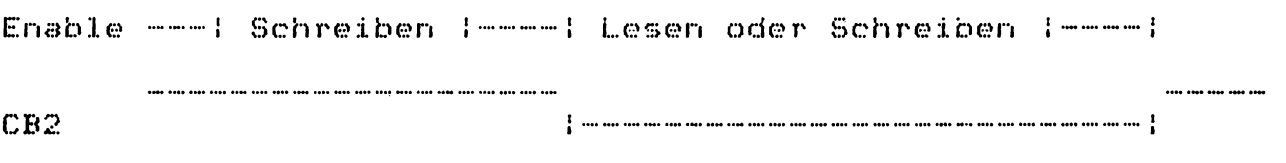
Bit 3 = 0: der CB2-Ausgang gibt den logischen Zustand von Bit 7 im Kontrollregister B wieder

Wird Bit 7 im Kontrollregister B durch die programmierte Flanke am CB1-Eingang gesetzt, wird der CB2-Ausgang ebenfalls high. Der Impuls am CB1-Eingang kann die Daten-Anforderung eines externen Gerätes (z. B. Drucker) sein. Der High-Pegel vom CB2-Ausgang kann das Gerät anhalten, bis der Mikrokomputer das naechste Datenwort ausgegeben hat. Geht beim Schreiben des Datenregisters B Bit 7 im Kontrollregister B wieder auf 0, wird der CB2-Ausgang automatisch ebenfalls auf 0 gesetzt (genauer: wenn der Enable-Eingang vor dem Schreiben des Daten-registers B auf 1 geht, wird CB2 wieder 0). Das angeschlossene Gerät ist wieder freigegeben, bis das Gerät durch einen Impuls am CB1-Eingang neue Daten anfordert.



Bit 3 = 1: der CB2-Ausgang wird durch den Enable-Eingang gesteuert

Der CB2-Ausgang ist im Ruhezustand high. Wird nach dem Schreiben des Datenregisters der PIA erneut adressiert, geht der CB2-Ausgang solange auf low, bis der PIA ein weiteres mal adressiert wurde (CB2 wird wieder high nach der naechsten positiven Flanke von Enable). Dies wird verwendet, wenn nach dem Schreiben eines Datenwortes eine weitere Ein- oder Ausgabe als Reaktion erforderlich ist. Ein externes Gerät kann auf das Low-Signal hin ein weiteres Datenwort von dem PIA uebernehmen oder ein Datenwort an den PIA ausgeben.



```
0001                        NAM    EUROCOM1
0002                *
0003                *
0004                *
0005 F800                   ORG    $F800
0006                *

0008                * * * * *
0009                *  PIA  *
0010                * * * * *

0012       8010     PIA     EQU    $8010

0014                * * * * *
0015                * ACIA  *
0016                * * * * *

0018       8008     ACIA    EQU    $8008
0019       0055     MON     EQU    $55
0020       0015     MOFF    EQU    $15

0022                * * * * *
0023                * STACK *
0024                * * * * *

0026       A7A8     S       EQU    $A7A8
0027       A7A7     STACK   EQU    S-1

0029                * * * * *
0030                * TABLE *
0031                * * * * *

0033       A7D3     BPTABL  EQU    S+43
0034       A800     ENDBPT  EQU    15*3+BPTABL

0036       A7A8     DISPL   EQU    S
0037       A7B8     CARRY   EQU    S+16
0038       A7B9     KEY     EQU    S+17
0039       A7BA     SAVEX   EQU    S+18
0040       A7BC     INDEX   EQU    S+20
0041       A7BE     NMIV    EQU    S+22
0042       A7C0     SWIV    EQU    S+24
0043       A7C2     IRQV    EQU    S+26
0044       A7C4     LENGTH  EQU    S+28
0045       A7C5     HEADER  EQU    S+29
0046       A7C8     BEGA    EQU    S+32
0047       A7CA     ENDA    EQU    S+34
0048       A7CC     ACTADD  EQU    S+36
0049       A7CE     CKSM    EQU    S+38
0050       A7CF     RG      EQU    S+39
0051       A7CF     BCONT   EQU    S+39
0052       A7D0     MCONT   EQU    S+40
0053       A7D1     SWITCH  EQU    S+41
0054       A7D2     HDRSW   EQU    S+42
```

PAGE 002 EUROCOM1

```
0056                    * * * * *
0057                    * KEYS  *
0058                    * * * * *

0060        0010        K0      EQU     $10
0061        0011        K1      EQU     $11
0062        0012        K2      EQU     $12
0063        0013        K3      EQU     $13
0064        0014        K4      EQU     $14
0065        0015        K5      EQU     $15
0066        0016        K6      EQU     $16
0067        0017        K7      EQU     $17
0068        0020        K8      EQU     $20
0069        0021        K9      EQU     $21
0070        0022        KA      EQU     $22
0071        0023        KB      EQU     $23
0072        0024        KC      EQU     $24
0073        0025        KD      EQU     $25
0074        0026        KE      EQU     $26
0075        0027        KF      EQU     $27
0076                    *
0077        0045        UA      EQU     $45
0078        0046        DA      EQU     $46
0079        0044        LA      EQU     $44
0080                    *
0081        0047        KK      EQU     $47
0082        0041        KM      EQU     $41
0083        0042        KP      EQU     $42
0084        0040        KG      EQU     $40
0085        0043        KS      EQU     $43
0086        0082        KR      EQU     $82
0087        0083        KL      EQU     $83
0088        0081        KJ      EQU     $81
```

```
PAGE  003  EUROCOM1


0090                    *
0091                    *LOESCHEN DER INTERRUPT-MASKE
0092                    *
0093 F800 0E            START   CLI

0095                    *
0096                    *SWI-VERARBEITUNG
0097                    *
0098 F801 CE FAEB               LDX     #SWIR
0099 F804 FF A7C0               STX     SWIV

0101                    *
0102                    *IRQ+NMI-VERARBEITUNG
0103                    *
0104 F807 CE FB4C               LDX     #SSLOOP
0105 F80A FF A7C2               STX     IRQV
0106 F80D FF A7BE               STX     NMIV

0108                    *
0109                    *PIA INITIALISIERUNG
0110                    *
0111 F810 CE 8010               LDX     #PIA
0112 F813 6F 02                 CLR     2,X
0113 F815 6F 03                 CLR     3,X
0114 F817 86 0F                 LDAA    #$F
0115 F819 A7 01                 STAA    1,X
0116 F81B 86 FF                 LDAA    #$FF
0117 F81D A7 00                 STAA    X
0118 F81F 4A                    DECA
0119 F820 A7 02                 STAA    2,X
0120 F822 A7 03                 STAA    3,X

0122                    *
0123                    *ACIA INITIALISIERUNG
0124                    *
0125 F824 86 03                 LDAA    #3
0126 F826 B7 8008               STAA    ACIA

0128                    *
0129                    *AUSGABE-FORMAT VOREINSTELLUNG
0130                    *
0131 F829 86 18                 LDAA    #24
0132 F82B B7 A7C4               STAA    LENGTH

0134                    *
0135                    *LOESCHEN DES HALTPUNKTE-SPEICHERS
0136                    *START DER HAUPTKONTROLLSCHLEIFE
0137                    *
0138 F82E 7E FA90               JMP     GOTOK
```

8010 -> IX
00 -> 8012
00 -> 8013
0F -> ACCA
<ACCA> -> 8011
FF -> ACCA
<ACCA> -> 8010
<ACCA> - 1 -> ACCA
<ACCA> -> 8012
<ACCA> -> 8013

Programmier. der Daten-register

0F -> 8011 } Port B 4Eing. 4 Port A 8Ausg

FF -> 8010

PAGE 004 EUROCOM1

```
0140                    *
0141                    *ANZEIGE UND EINGABE MULTIPLEXER
0142                    *
0143 F831 8D 00    DISP4   BSR    DISP2
0144 F833 8D 00    DISP2   BSR    DISP
0145 F835 4F       DISP    CLRA
0146 F836 CE A7A8          LDX    #DISPL
0147 F839 C6 FF    GOON    LDAB   #$FF
0148 F83B F7 8010          STAB   PIA
0149 F83E B7 8011          STAA   PIA+1
0150 F841 8D 25            BSR    TSTKEY
0151 F843 26 13            BNE    KPSH
0152 F845 E6 00            LDAB   X
0153 F847 F7 8010          STAB   PIA
0154 F84A 8D 23            BSR    DEL
0155 F84C 08               INX
0156 F84D 4C               INCA
0157 F84E 81 08            CMPA   #8
0158 F850 26 E7            BNE    GOON
0159 F852 86 FF            LDAA   #$FF
0160 F854 B7 8010          STAA   PIA
0161 F857 39               RTS
0162                    *
0163                    *WARTEN AUF LOESEN DER TASTE
0164                    *
0165 F858 1B       KPSH    ABA
0166 F859 C6 FF            LDAB   #$FF
0167 F85B F7 8010          STAB   PIA
0168 F85E B7 A7B9          STAA   KEY
0169 F861 8D 0C    KPSH1   BSR    DEL
0170 F863 8D 03            BSR    TSTKEY
0171 F865 26 FA            BNE    KPSH1
0172 F867 39               RTS
0173                    *
0174                    *EINGABEABFRAGE
0175                    *
0176 F868 F6 8011  TSTKEY  LDAB   PIA+1
0177 F86B CA 0F            ORAB   #$F
0178 F86D 53               COMB
0179 F86E 39               RTS
0180                    *
0181                    *ZEITVERZOEGERUNG
0182                    *
0183 F86F 36       DEL     PSHA
0184 F870 4F               CLRA
0185 F871 4A       DEL1    DECA
0186 F872 26 FD            BNE    DEL1
0187 F874 32               PULA
0188 F875 39               RTS
0189                    *
0190                    *WARTEN AUF EINGABE
0191                    *
0192 F876 8D 0B    WAIT    BSR    BUILD
0193 F878 7F A7B9          CLR    KEY
```

8010 und 8011 sind Datenregister

8010 = DRA = Anzeige

0 → ACCA

A7A8 in IX

FF in ACCB

⟨ACCB⟩ → 8010 → Keine Anzeige

⟨ACCA⟩ → 8011

Springe → TSTKEY

Springe, wenn ≠ 0

Lade B indiziert,

⟨ACCB⟩ → 8010

Springe → DEL

⟨IX⟩+1 → IX

⟨A⟩+1 → A

⟨A⟩ -8

Springe, wenn ≠ 0

FF → ACCA

⟨A⟩ → 8010 → Keine Anzeige

von 1,56 ms

ACCA ← ⟨ACCA⟩ - 1

Daten aus Kellerspeicher holen

PAGE 005 EUROCOM1

```
0194 F87B 8D B8    WAIT1  BSR   DISP
0195 F87D 7D A7B9         TST   KEY
0196 F880 27 F9           BEQ   WAIT1
0197 F882 39              RTS
0198                *
0199                *ERZEUGEN DER SEGMENTANZEIGE
0200                *AUS ASCII-SPEICHER
0201                *
0202 F883 CE A7A8  BUILD  LDX   #DISPL
0203 F886 E6 08    BUILD1 LDAB  8,X
0204 F888 FF A7BA         STX   SAVEX
0205 F88B CE FE82         LDX   #CTABLE
0206 F88E A6 00    ACODE1 LDAA  X
0207 F890 11              CBA
0208 F891 26 04           BNE   ACODE2
0209 F893 E6 01           LDAB  1,X
0210 F895 20 0A           BRA   ADONE
0211 F897 08       ACODE2 INX
0212 F898 08              INX
0213 F899 08              INX
0214 F89A 8C FED6         CPX   #ENDTAB
0215 F89D 26 EF           BNE   ACODE1
0216 F89F C6 FF           LDAB  #$FF
0217 F8A1 FE A7BA  ADONE  LDX   SAVEX
0218 F8A4 E7 00           STAB  X
0219 F8A6 08              INX
0220 F8A7 8C A7B0         CPX   #DISPL+8
0221 F8AA 26 DA           BNE   BUILD1
0222 F8AC 39              RTS
0223                *
0224                *CARRY IN ASCII-SPEICHER SCHIEBEN
0225                *NEUE TASTENEINGABE NACH CARRY
0226                *
0227 F8AD 8D 00    INDIS4 BSR   INDIS2
0228 F8AF 8D 00    INDIS2 BSR   INDIS1
0229 F8B1 8D 55    INDIS1 BSR   SHIFT8
0230 F8B3 8D 09    INDISO BSR   OUTDIS
0231 F8B5 27 04           BEQ   CNTRL2
0232 F8B7 F7 A7B8         STAB  CARRY
0233 F8BA 39              RTS
0234                *
0235 F8BB 7E FA57  CNTRL2 JMP   CNTRLA
0236                *
0237                *NEUE TASTENEINGABE NACH CARRY
0238                *
0239 F8BE 8D B6    OUTDIS BSR   WAIT
0240 F8C0 F6 A7B9         LDAB  KEY   ACCB <(A7B9>
0241                *
0242                *UMWANDLUNG TASTEN-CODE IN ASCII
0243                *
0244 F8C3 CE FE82         LDX   #CTABLE IX=
0245 F8C6 A6 02    KCODE1 LDAA  2,X   ACCA = <(IX)+2
0246 F8C8 11              CBA          ACC - ACCB>
0247 F8C9 27 08           BEQ   KDONE  Status if Equal
```

PAGE 006 EUROCOM1

```
0248 F8CB 08       KCODE2 INX      <IX> ← <IX> + 0001
0249 F8CC 08              INX
0250 F8CD 08              INX
0251 F8CE 8C FED6         CPX   #ENDTAB  <IX+1> - <M+1> / <IX+1> - <M>
0252 F8D1 26 F3           BNE   KCODE1   Branch if Not Equal → F8C6
0253 F8D3 E6 00    KDONE  LDAB  X        ACC B ← (M), indiziert
0254                      *
0255                      *ABFRAGE OB TASTENEINGABE HEXADEZIMAL
0256                      *
0257 F8D5 C1 30           CMPB  #$30
0258 F8D7 2B 0C           BMI   NO
0259 F8D9 C1 46           CMPB  #$46
0260 F8DB 2E 08           BGT   NO
0261 F8DD C1 3A           CMPB  #$3A
0262 F8DF 2B 05           BMI   YES
0263 F8E1 C1 40           CMPB  #$40
0264 F8E3 2E 01           BGT   YES
0265 F8E5 5F       NO     CLRB
0266 F8E6 39       YES    RTS
0267                      *
0268                      *UMWANDLUNG ACIA-DATEN IN HEXADEZIMAL
0269                      *
0270 F8E7 BD FD72  INHEX  JSR   INACIA
0271 F8EA 80 30           SUBA  #$30
0272 F8EC 2B 0F           BMI   DERR
0273 F8EE 81 09           CMPA  #9
0274 F8F0 2F 0A           BLE   INDONE
0275 F8F2 81 11           CMPA  #$11
0276 F8F4 2B 07           BMI   DERR
0277 F8F6 81 16           CMPA  #$16
0278 F8F8 2E 03           BGT   DERR
0279 F8FA 80 07           SUBA  #7
0280 F8FC 39       INDONE RTS
0281                      *
0282 F8FD C6 35    DERR   LDAB  #'5
0283 F8FF 7E F9CF         JMP   ERROR
0284                      *
0285                      *VERSCHIEBUNG DES ASCII-SPEICHERINHALTS
0286                      *
0287 F902 8D 00    SHIF88 BSR   SHIF48
0288 F904 8D 00    SHIF48 BSR   SHIF28
0289 F906 8D 00    SHIF28 BSR   SHIFT8
0290 F908 CE A7B0  SHIFT8 LDX   #DISPL+8
0291 F90B A6 01    SHIFT  LDAA  1,X
0292 F90D A7 00           STAA  X
0293 F90F 08              INX
0294 F910 8C A7B8         CPX   #DISPL+16
0295 F913 26 F6           BNE   SHIFT
0296 F915 6F 00           CLR   X
0297 F917 39              RTS
0298                      *
0299                      *UMWANDLUNG ASCII IN HEXADEZIMAL
0300                      *
0301 F918 7D A7D1  GETHEX TST   SWITCH
```

PAGE 007 EUROCOM1

```
0302 F91B 26 CA          BNE   INHEX
0303 F91D A6 00          LDAA  X
0304 F91F 80 30          SUBA  #$30
0305 F921 81 09          CMPA  #9
0306 F923 2F 02          BLE   OUTHEX
0307 F925 80 07          SUBA  #7
0308 F927 08       OUTHEX INX
0309 F928 39              RTS
0310                 *
0311                 *UMWANDLUNG 2 HEX IN 1 BYTE
0312                 *
0313 F929 8D ED    BYTE  BSR   GETHEX
0314 F92B 48             ASLA
0315 F92C 48             ASLA
0316 F92D 48             ASLA
0317 F92E 48             ASLA
0318 F92F 37             PSHB
0319 F930 16             TAB
0320 F931 8D E5          BSR   GETHEX
0321 F933 1B             ABA
0322 F934 16             TAB
0323 F935 FB A7CE        ADDB  CKSM
0324 F938 F7 A7CE        STAB  CKSM
0325 F93B 33             PULB
0326 F93C 39             RTS
0327                 *
0328                 *EINLESEN VON BEFEHL UND ADRESSE
0329                 *
0330 F93D 8D C3    DISADD BSR   SHIF88
0331 F93F F7 A7B8  DISAD1 STAB  CARRY
0332 F942 8D C4          BSR   SHIFT8
0333 F944 BD F8AD  DIS    JSR   INDIS4
0334 F947 8D BF          BSR   SHIFT8
0335 F949 7F A7B9        CLR   KEY
0336 F94C BD FA00        JSR   TIMEDF
0337                 *
0338                 * 2 BYTE NACH INDEX
0339                 *
0340 F94F CE A7B4        LDX   #DISPL+12
0341 F952 8D D5    BADDR BSR   BYTE
0342 F954 B7 A7BC        STAA  INDEX
0343 F957 8D D0          BSR   BYTE
0344 F959 B7 A7BD        STAA  INDEX+1
0345 F95C FE A7BC        LDX   INDEX
0346 F95F 39             RTS
0347                 *
0348                 *SPRUNG ZUR GEWAEHLTEN ADRESSE
0349                 *
0350 F960 C6 47    GOTOG LDAB  #'G
0351 F962 8D D9          BSR   DISADD
0352 F964 6E 00          JMP   X
0353                 *
0354                 *HALBES BYTE ALS ASCII-ZEICHEN NACH CARRY
0355                 *
```

```
PAGE   008   EUROCOM1


0356 F966 44        ASCIH   LSRA
0357 F967 44                LSRA
0358 F968 44                LSRA
0359 F969 44                LSRA
0360 F96A 84 0F     ASCIL   ANDA   #$F
0361 F96C 8B 30             ADDA   #$30
0362 F96E 81 39             CMPA   #$39
0363 F970 23 02             BLS    OK
0364 F972 8B 07             ADDA   #7
0365 F974 39        OK      RTS
0366                *
0367                *CARRY NACH ASCII-SPEICHER
0368                *
0369 F975 B7 A7B8   LOADC   STAA   CARRY
0370 F978 20 8E             BRA    SHIFT8
0371                *
0372                * 1 ODER 2 BYTE NACH ASCII-SPEICHER
0373                *
0374 F97A 8D 00     INBYT2  BSR    INBYTE
0375 F97C A6 00     INBYTE  LDAA   X
0376 F97E FF A7BA           STX    SAVEX
0377 F981 8D E3             BSR    ASCIH
0378 F983 8D F0             BSR    LOADC
0379 F985 FE A7BA           LDX    SAVEX
0380 F988 A6 00             LDAA   X
0381 F98A 8D DE             BSR    ASCIL
0382 F98C 8D E7             BSR    LOADC
0383 F98E FE A7BA           LDX    SAVEX
0384 F991 08                INX
0385 F992 39                RTS
0386                *
0387                *MEMORY ABFRAGE UND AENDERUNG
0388                *
0389 F993 C6 4D     GOTOM   LDAB   #'M
0390 F995 8D A6             BSR    DISADD
0391 F997 BD F906   GOTOM1  JSR    SHIF28
0392 F99A FE A7BC           LDX    INDEX
0393 F99D 8D DD             BSR    INBYTE
0394 F99F BD F8BE   NEW     JSR    OUTDIS
0395 F9A2 27 0B             BEQ    GORD
0396 F9A4 F7 A7B8           STAB   CARRY
0397 F9A7 CE A7B6           LDX    #DISPL+14
0398 F9AA BD F90B           JSR    SHIFT
0399 F9AD 20 F0             BRA    NEW
0400 F9AF 8D 7F     GORD    BSR    ORDER
0401 F9B1 BD FA44           JSR    CHANGE
0402 F9B4 FE A7BC           LDX    INDEX
0403 F9B7 F6 A7B9           LDAB   KEY
0404 F9BA C1 46             CMPB   #DA
0405 F9BC 27 06             BEQ    INCR
0406 F9BE C1 45             CMPB   #UA
0407 F9C0 26 20             BNE    CNTRL3
0408 F9C2 09                DEX
0409 F9C3 09                DEX
```

<X> -> M u. M+1 = SAVEX
SAVE(X+1)

```
PAGE  009  EUROCOM1


0410 F9C4 08       INCR   INX
0411 F9C5 FF A7BC         STX   INDEX
0412 F9C8 CE A7BC         LDX   #INDEX
0413 F9CB 8D AD           BSR   INBYT2
0414 F9CD 20 C8           BRA   GOTOM1
0415                 *
0416                 *FEHLERAUSGABE
0417                 *FEHLERNUMMER IN ACCB
0418                 *
0419 F9CF 86 15    ERROR  LDAA  #MOFF
0420 F9D1 B7 8008         STAA  ACIA
0421 F9D4 37              PSHB
0422 F9D5 CE FE54         LDX   #TERROR
0423 F9D8 8D 41           BSR   LTEXT
0424 F9DA 33              PULB
0425 F9DB CE A7B7         LDX   #DISPL+15
0426 F9DE E7 00           STAB  X
0427 F9E0 8D 05           BSR   BLINK1
0428 F9E2 20 73    CNTRL3 BRA   CNTRLA
0429                 *
0430                 *BLINKENDES ZEICHEN
0431                 *
0432 F9E4 CE A7B0  BLINK  LDX   #DISPL+8
0433 F9E7 7F A7B9  BLINK1 CLR   KEY
0434 F9EA A6 00    BLINK2 LDAA  X
0435 F9EC 36              PSHA
0436 F9ED 86 0A           LDAA  #10
0437 F9EF 6F 00           CLR   X
0438 F9F1 8D 0F           BSR   TIMED1
0439 F9F3 32              PULA
0440 F9F4 A7 00           STAA  X
0441 F9F6 86 0A           LDAA  #10
0442 F9F8 8D 08           BSR   TIMED1
0443 F9FA 7D A7B9         TST   KEY
0444 F9FD 27 EB           BEQ   BLINK2
0445 F9FF 39              RTS
0446                 *
0447                 * 1 SEC FESTSTEHENDE ANZEIGE
0448                 *
0449 FA00 86 32    TIMEDP LDAA  #50
0450 FA02 36       TIMED1 PSHA
0451 FA03 FF A7BC         STX   INDEX
0452 FA06 BD F883         JSR   BUILD
0453 FA09 7D A7B9         TST   KEY
0454 FA0C 26 08           BNE   STOP
0455 FA0E BD F835  TIME1  JSR   DISP
0456 FA11 32              PULA
0457 FA12 4A              DECA
0458 FA13 36              PSHA
0459 FA14 26 F8           BNE   TIME1
0460 FA16 FE A7BC  STOP   LDX   INDEX
0461 FA19 32              PULA
0462 FA1A 39              RTS
0463                 *
```

PAGE 010 EUROCOM1

```
0464                    *LADEN VON TEXT NACH ASCII-SPEICHER
0465                    *
0466 FA1B C6 08    LTEXT  LDAB  #8
0467 FA1D A6 00    LTEXT1 LDAA  X
0468 FA1F B7 A7B8         STAA  CARRY
0469 FA22 FF A7BA         STX   SAVEX
0470 FA25 BD F908         JSR   SHIFT8
0471 FA28 FE A7BA         LDX   SAVEX
0472 FA2B 08              INX
0473 FA2C 5A              DECB
0474 FA2D 26 EE           BNE   LTEXT1
0475 FA2F 39              RTS
0476                    *
0477                    *BEFEHLSARTABFRAGE
0478                    *
0479 FA30 F6 A7B9 ORDER   LDAB  KEY
0480 FA33 C1 44           CMPB  #LA
0481 FA35 27 08           BEQ   ENDORD
0482 FA37 C1 45           CMPB  #UA
0483 FA39 27 04           BEQ   ENDORD
0484 FA3B C1 46           CMPB  #DA
0485 FA3D 26 18           BNE   CNTRLA
0486 FA3F 39        ENDORD RTS
0487                    *
0488                    *VERAENDERUNG DES SPEICHERINHALTS
0489                    *
0490 FA40 C6 31    MERR1  LDAB  #'1
0491 FA42 20 8B    ERRORO BRA   ERROR
0492                    *
0493 FA44 CE A7B6 CHANGE  LDX   #DISPL+14
0494 FA47 BD F929 CHANG1  JSR   BYTE
0495 FA4A FE A7BC         LDX   INDEX
0496 FA4D A7 00           STAA  X
0497 FA4F A1 00           CMPA  X
0498 FA51 26 ED           BNE   MERR1
0499 FA53 39              RTS
0500                    *
0501                    *HAUPTKONTROLLSCHLEIFE
0502                    *
0503 FA54 7F A7B9 CNTRL   CLR   KEY
0504 FA57 8E A7A7 CNTRLA  LDS   #STACK
0505 FA5A 7F A7D1         CLR   SWITCH
0506 FA5D 86 15           LDAA  #MOFF
0507 FA5F B7 8008         STAA  ACIA
0508 FA62 CE FE5A CLOOP   LDX   #TEUROC
0509 FA65 8D B4           BSR   LTEXT
0510 FA67 8D 97           BSR   TIMEDP
0511 FA69 CE FE62         LDX   #TCONTR
0512 FA6C 8D AD           BSR   LTEXT
0513 FA6E 8D 90           BSR   TIMEDP
0514 FA70 BD F902         JSR   SHIF88
0515 FA73 8D 8B           BSR   TIMEDP
0516 FA75 B6 A7B9         LDAA  KEY
0517 FA78 CE FED5         LDX   #FCTABL
```

!! LDS !

PAGE 011 EUROCOM1

```
0518 FA7B A1 00    TSTNXT CMPA  X
0519 FA7D 27 0D           BEQ   GOOD
0520 FA7F 08              INX
0521 FA80 08              INX
0522 FA81 08              INX
0523 FA82 8C FEED         CPX   #FCTBEN
0524 FA85 26 F4           BNE   TSTNXT
0525 FA87 7F A7B9         CLR   KEY
0526 FA8A 20 D6           BRA   CLOOP
0527 FA8C EE 01    GOOD   LDX   1,X
0528 FA8E 6E 00           JMP   X
0529                      *
0530                      *ALLE HALTPUNKTE LOESCHEN
0531                      *
0532 FA90 CE A7D3  GOTOK  LDX   #BPTABL
0533 FA93 A6 02    KILL   LDAA  2,X
0534 FA95 27 08           BEQ   NOBPT
0535 FA97 6F 02           CLR   2,X
0536 FA99 EE 00           LDX   X
0537 FA9B A7 00           STAA  X
0538 FA9D 20 F1           BRA   GOTOK
0539 FA9F 08       NOBPT  INX
0540 FAA0 08              INX
0541 FAA1 08              INX
0542 FAA2 8C A800         CPX   #ENDBPT
0543 FAA5 26 EC           BNE   KILL
0544 FAA7 20 AB    CNTRL4 BRA   CNTRL
0545                      *
0546                      *HALTPUNKT SETZEN
0547                      *
0548 FAA9 C6 34    PERR   LDAB  #'4
0549 FAAB 20 95    ERROR2 BRA   ERRORO
0550 FAAD C6 32    MERR   LDAB  #'2
0551 FAAF 20 91           BRA   ERRORO
0552 FAB1 C6 50    GOTOP  LDAB  #'P
0553 FAB3 BD F93D         JSR   DISADD
0554 FAB6 A6 00           LDAA  X
0555 FAB8 C6 3F           LDAB  #$3F
0556 FABA 11              CBA
0557 FABB 27 EC           BEQ   PERR
0558 FABD E7 00           STAB  X
0559 FABF E6 00           LDAB  X
0560 FAC1 C1 3F           CMPB  #$3F
0561 FAC3 26 E8           BNE   MERR
0562 FAC5 CE A7D3         LDX   #BPTABL
0563 FAC8 E6 02    NXTTRY LDAB  2,X
0564 FACA 26 0E           BNE   FILLED
0565 FACC A7 02           STAA  2,X
0566 FACE F6 A7BC         LDAB  INDEX
0567 FAD1 E7 00           STAB  X
0568 FAD3 F6 A7BD         LDAB  INDEX+1
0569 FAD6 E7 01           STAB  1,X
0570 FAD8 20 CD    CNTRL5 BRA   CNTRL4
0571 FADA 08       FILLED INX
```

PAGE 012 EUROCOM1

```
0572 FADB 08              INX
0573 FADC 08              INX
0574 FADD 8C A800         CPX   #ENDBPT
0575 FAE0 26 E6           BNE   NXTTRY
0576 FAE2 FE A7BC         LDX   INDEX
0577 FAE5 A7 00           STAA  X
0578 FAE7 C6 33           LDAB  #'3
0579 FAE9 20 C0           BRA   ERROR2
0580                  *
0581                  *SOFTWARE-INTERRUPT ABLAUF
0582                  *
0583 FAEB 30          SWIR  TSX
0584 FAEC 6D 06           TST   6,X
0585 FAEE 26 02           BNE   SWIR1
0586 FAF0 6A 05           DEC   5,X
0587 FAF2 6A 06       SWIR1 DEC   6,X
0588 FAF4 34              DES
0589 FAF5 34              DES
0590 FAF6 34              DES
0591 FAF7 8D 72           BSR   REGDIS
0592 FAF9 31              INS
0593 FAFA 31              INS
0594 FAFB 31              INS
0595 FAFC 30              TSX
0596 FAFD EE 05           LDX   5,X
0597 FAFF FF A7BC         STX   INDEX
0598 FB02 CE A7D3         LDX   #BPTABL
0599 FB05 A6 00       NXTBPT LDAA X
0600 FB07 B1 A7BC         CMPA  INDEX
0601 FB0A 26 19           BNE   NOFIT
0602 FB0C A6 01           LDAA  1,X
0603 FB0E B1 A7BD         CMPA  INDEX+1
0604 FB11 26 12           BNE   NOFIT
0605 FB13 A6 02           LDAA  2,X
0606 FB15 6F 02           CLR   2,X
0607 FB17 FE A7BC         LDX   INDEX
0608 FB1A A7 00           STAA  X
0609 FB1C C1 47           CMPB  #KK
0610 FB1E 27 0D           BEQ   CNTRL6
0611 FB20 C1 43           CMPB  #KS
0612 FB22 27 3E           BEQ   ST
0613 FB24 3B              RTI
0614 FB25 08          NOFIT INX
0615 FB26 08              INX
0616 FB27 08              INX
0617 FB28 8C A800         CPX   #ENDBPT
0618 FB2B 26 D8           BNE   NXTBPT
0619 FB2D 20 A9       CNTRL6 BRA  CNTRL5
0620                  *
0621                  *EINZELSCHRITTSTEUERUNG
0622                  *
0623 FB2F 4F          GOTOS CLRA
0624 FB30 36              PSHA
0625 FB31 BF A7CF         STS   RG
```

PAGE 013 EUROCOM1

```
0626 FB34 36              PSHA
0627 FB35 36              PSHA
0628 FB36 36              PSHA
0629 FB37 36              PSHA
0630 FB38 36              PSHA
0631 FB39 36              PSHA
0632 FB3A C6 35           LDAB  #'5
0633 FB3C BD F93D         JSR   DISADD
0634 FB3F FE A7CF         LDX   RG
0635 FB42 B6 A7BC         LDAA  INDEX
0636 FB45 A7 00           STAA  X
0637 FB47 B6 A7BD         LDAA  INDEX+1
0638 FB4A A7 01           STAA  1,X
0639 FB4C 86 3E    SSLOOP LDAA  #%00111110
0640 FB4E B7 8012         STAA  PIA+2
0641 FB51 B6 8010         LDAA  PIA
0642 FB54 34              DES
0643 FB55 34              DES
0644 FB56 34              DES
0645 FB57 8D 12           BSR   REGDIS
0646 FB59 27 D2           BEQ   CNTRL6
0647 FB5B 31              INS
0648 FB5C 31              INS
0649 FB5D 31              INS
0650 FB5E C1 40           CMPB  #KG
0651 FB60 27 08           BEQ   GO
0652 FB62 86 37    ST     LDAA  #%00110111
0653 FB64 B7 8012         STAA  PIA+2
0654 FB67 01              NOP
0655 FB68 01              NOP
0656 FB69 01              NOP
0657 FB6A 3B       GO     RTI
0658                      *
0659                      *REGISTERANZEIGE
0660                      *
0661 FB6B 30       REGDIS TSX
0662 FB6C 08              INX
0663 FB6D 08              INX
0664 FB6E EF 00           STX   X
0665 FB70 6F 02           CLR   2,X
0666 FB72 FF A7CF         STX   RG
0667 FB75 86 05           LDAA  #5
0668 FB77 36       REGCH  PSHA
0669 FB78 FE A7CF         LDX   RG
0670 FB7B FF A7BC         STX   INDEX
0671 FB7E CE FE6A         LDX   #TRGSTR
0672 FB81 4A       SEARCH DECA
0673 FB82 27 12           BEQ   REGOUT
0674 FB84 08              INX
0675 FB85 08              INX
0676 FB86 FF A7BA         STX   SAVEX
0677 FB89 FE A7BC         LDX   INDEX
0678 FB8C 08              INX
0679 FB8D 08              INX
```

PAGE 014 EUROCOM1

```
0680 FB8E FF A7BC          STX   INDEX
0681 FB91 FE A7BA          LDX   SAVEX
0682 FB94 20 EB            BRA   SEARCH
0683 FB96 A6 01    REGOUT  LDAA  1,X
0684 FB98 36               PSHA
0685 FB99 A6 00            LDAA  X
0686 FB9B BD F975          JSR   LOADC
0687 FB9E 32               PULA
0688 FB9F BD F975          JSR   LOADC
0689 FBA2 86 3D            LDAA  #'=
0690 FBA4 BD F975          JSR   LOADC
0691 FBA7 BD F908          JSR   SHIFT8
0692 FBAA FE A7BC          LDX   INDEX
0693 FBAD BD F97A          JSR   INBYT2
0694 FBB0 BD F8BE  RNEW    JSR   OUTDIS
0695 FBB3 27 0B            BEQ   RORD
0696 FBB5 F7 A7B8          STAB  CARRY
0697 FBB8 CE A7B4          LDX   #DISPL+12
0698 FBBB BD F90B          JSR   SHIFT
0699 FBBE 20 F0            BRA   RNEW
0700 FBC0 F6 A7B9  RORD    LDAB  KEY
0701 FBC3 C1 40            CMPB  #KG
0702 FBC5 27 0B            BEQ   RON
0703 FBC7 C1 43            CMPB  #KS
0704 FBC9 27 07            BEQ   RON
0705 FBCB C1 47            CMPB  #KK
0706 FBCD 27 03            BEQ   RON
0707 FBCF BD FA30          JSR   ORDER
0708 FBD2 CE A7B4  RON     LDX   #DISPL+12
0709 FBD5 BD FA47          JSR   CHANGI
0710 FBD8 7C A7BD          INC   INDEX+1
0711 FBDB 26 03            BNE   NOCARR
0712 FBDD 7C A7BC          INC   INDEX
0713 FBE0 BD FA44  NOCARR  JSR   CHANGE
0714 FBE3 32               PULA
0715 FBE4 C1 46            CMPB  #DA
0716 FBE6 27 06            BEQ   UP
0717 FBE8 C1 45            CMPB  #UA
0718 FBEA 26 10            BNE   STEP
0719 FBEC 4A               DECA
0720 FBED 4A               DECA
0721 FBEE 4C       UP      INCA
0722 FBEF 26 02            BNE   POS
0723 FBF1 86 05            LDAA  #5
0724 FBF3 81 06    POS     CMPA  #6
0725 FBF5 26 02            BNE   SUIT
0726 FBF7 86 01            LDAA  #1
0727 FBF9 7E FB77  SUIT    JMP   REGCH
0728 FBFC 39       STEP    RTS
0729               *
0730               *EINZELZEICHEN ZUM ACIA
0731               *
0732 FBFD 36       OUT     PSHA
0733 FBFE 37               PSHB
```

PAGE 015 EUROCOM1

```
0734 FBFF FF A7BC          STX   INDEX
0735 FC02 20 03            BRA   OUT2
0736 FC04 BD F835 OUT1     JSR   DISP
0737 FC07 B6 8008 OUT2     LDAA  ACIA
0738 FC0A 47               ASRA
0739 FC0B 47               ASRA
0740 FC0C 24 F6            BCC   OUT1
0741 FC0E 33               PULB
0742 FC0F 32               PULA
0743 FC10 B7 8009          STAA  ACIA+1
0744 FC13 FE A7BC          LDX   INDEX
0745 FC16 39               RTS
0746                  *
0747                  * 4 HEXZEICHEN ZUM ACIA
0748                  *
0749 FC17 8D 00    OUT4H  BSR   OUT2H
0750 FC19 EB 00    OUT2H  ADDB  X
0751 FC1B A6 00           LDAA  X
0752 FC1D 36       OUT2HA PSHA
0753 FC1E BD F966          JSR   ASCIH
0754 FC21 8D DA            BSR   OUT
0755 FC23 32               PULA
0756 FC24 08               INX
0757 FC25 BD F96A          JSR   ASCIL
0758 FC28 20 D3            BRA   OUT
0759                  *
0760                  *AUSGABE UEBER ACIA
0761                  *
0762 FC2A BD FCBA GOTOR    JSR   AREA
0763 FC2D C6 72            LDAB  #'r
0764 FC2F BD FCFB          JSR   HDRIN
0765 FC32 8D 7B            BSR   DELAY
0766 FC34 C6 0A            LDAB  #10
0767 FC36 4F               CLRA
0768 FC37 8D C4    COUNT   BSR   OUT
0769 FC39 5A               DECB
0770 FC3A 26 FB            BNE   COUNT
0771 FC3C 7D A7D2          TST   HDRSW
0772 FC3F 26 0E            BNE   REC
0773 FC41 BD FCCB          JSR   OUTST1
0774 FC44 86 30            LDAA  #'0
0775 FC46 8D B5            BSR   OUT
0776 FC48 CE A7C5          LDX   #HEADER
0777 FC4B 8D CA            BSR   OUT4H
0778 FC4D 8D CA            BSR   OUT2H
0779 FC4F FE A7C8 REC      LDX   BEGA
0780 FC52 FF A7CC          STX   ACTADD
0781 FC55 B6 A7CB RECO     LDAA  ENDA+1
0782 FC58 B0 A7CD          SUBA  ACTADD+1
0783 FC5B F6 A7CA          LDAB  ENDA
0784 FC5E F2 A7CC          SBCB  ACTADD
0785 FC61 26 05            BNE   REC1
0786 FC63 B1 A7C4          CMPA  LENGTH
0787 FC66 25 04            BCS   REC2
```

```
PAGE  016  EUROCOM1


0788 FC68 B6 A7C4 REC1   LDAA  LENGTH
0789 FC6B 4A             DECA
0790 FC6C 8B 04   REC2   ADDA  #4
0791 FC6E B7 A7D0        STAA  MCONT
0792 FC71 80 03          SUBA  #3
0793 FC73 B7 A7CF        STAA  BCONT
0794 FC76 8D 53          BSR   OUTST1
0795 FC78 86 31          LDAA  #'1
0796 FC7A 8D 81          BSR   OUT
0797 FC7C 5F             CLRB
0798 FC7D CE A7D0        LDX   #MCONT
0799 FC80 8D 97          BSR   OUT2H
0800 FC82 CE A7CC        LDX   #ACTADD
0801 FC85 8D 90          BSR   OUT4H
0802 FC87 FE A7CC        LDX   ACTADD
0803 FC8A 8D 8D   REC3   BSR   OUT2H
0804 FC8C 7A A7CF        DEC   BCONT
0805 FC8F 26 F9          BNE   REC3
0806 FC91 FF A7CC        STX   ACTADD
0807 FC94 53             COMB
0808 FC95 17             TBA
0809 FC96 8D 85          BSR   OUT2HA
0810 FC98 FE A7CC        LDX   ACTADD
0811 FC9B 09             DEX
0812 FC9C BC A7CA        CPX   ENDA
0813 FC9F 26 B4          BNE   RECO
0814 FCA1 8D 28          BSR   OUTST1
0815 FCA3 86 39          LDAA  #'9
0816 FCA5 BD FBFD        JSR   OUT
0817 FCA8 86 22          LDAA  #34
0818 FCAA 8D 05          BSR   DELAY1
0819 FCAC 7E FA54        JMP   CNTRL
0820                *
0821                * 10 SEC VORSPANN
0822                *
0823 FCAF 86 AA   DELAY  LDAA  #170
0824 FCB1 36      DELAY1 PSHA
0825 FCB2 BD F831        JSR   DISP4
0826 FCB5 32             PULA
0827 FCB6 4A             DECA
0828 FCB7 26 F8          BNE   DELAY1
0829 FCB9 39             RTS
0830                *
0831                *ADRESSBEREICH EINLESEN
0832                *
0833 FCBA CE FE74 AREA   LDX   #TBEG
0834 FCBD 8D 2E          BSR   CHRIN
0835 FCBF FF A7C8        STX   BEGA
0836 FCC2 CE FE77        LDX   #TEND
0837 FCC5 8D 26          BSR   CHRIN
0838 FCC7 FF A7CA        STX   ENDA
0839 FCCA 39             RTS
0840                *
0841                *TEXTSTRING FUER ACIA
```

PAGE 017 EUROCOM1

```
0842                    *
0843 FCCB CE FE7A OUTST1 LDX    #TCRLFS
0844 FCCE A6 00   OUTSTR LDAA   X
0845 FCD0 81 04          CMPA   #4
0846 FCD2 27 06          BEQ    OUTEND
0847 FCD4 BD FBFD        JSR    OUT
0848 FCD7 08             INX
0849 FCD8 20 F4          BRA    OUTSTR
0850 FCDA 39      OUTEND RTS
0851                    *
0852                    *TEXTSTRING-AUSGABE
0853                    *
0854 FCDB 8D 00   NXTCH2 BSR    NXTCHR
0855 FCDD F7 A7B8 NXTCHR STAB   CARRY
0856 FCE0 BD F908        JSR    SHIFT8
0857 FCE3 FE A7BC NXTC1  LDX    INDEX
0858 FCE6 E6 00          LDAB   X
0859 FCE8 08             INX
0860 FCE9 FF A7BC        STX    INDEX
0861 FCEC 39             RTS
0862                    *
0863                    * 3 ZEICHEN AUSGABE + ADRESSEINGABE
0864                    *
0865 FCED FF A7BC CHRIN  STX    INDEX
0866 FCF0 BD F902        JSR    SHIF88
0867 FCF3 8D EE          BSR    NXTC1
0868 FCF5 8D E4          BSR    NXTCH2
0869 FCF7 BD F93F        JSR    DISAD1
0870 FCFA 39             RTS
0871                    *
0872                    *HEADEREINGABE
0873                    *
0874 FCFB 7F A7D2 HDRIN  CLR    HDRSW
0875 FCFE BD F902        JSR    SHIF88
0876 FD01 F7 A7B8        STAB   CARRY
0877 FD04 BD F906        JSR    SHIF28
0878 FD07 BD F8BE        JSR    OUTDIS
0879 FD0A 26 12          BNE    HDR1
0880 FD0C F6 A7B9        LDAB   KEY
0881 FD0F C1 44          CMPB   #LA
0882 FD11 26 51          BNE    CNTRL7
0883 FD13 7C A7D2        INC    HDRSW
0884 FD16 BD F904        JSR    SHIF48
0885 FD19 BD F906        JSR    SHIF28
0886 FD1C 20 15          BRA    NOHDR
0887 FD1E F7 A7B8 HDR1   STAB   CARRY
0888 FD21 BD F8B1        JSR    INDIS1
0889 FD24 BD F944        JSR    DIS
0890 FD27 FF A7C6        STX    HEADER+1
0891 FD2A CE A7B2        LDX    #DISPL+10
0892 FD2D BD F929        JSR    BYTE
0893 FD30 B7 A7C5        STAA   HEADER
0894 FD33 86 15   NOHDR  LDAA   #MOFF
0895 FD35 B7 8008        STAA   ACIA
```

PAGE 018 EUROCOM1

```
0896 FD38 BD F876          JSR   WAIT
0897 FD3B 20 08            BRA   OF
0898 FD3D 86 55    MOON    LDAA  #MON
0899 FD3F B7 8008          STAA  ACIA
0900 FD42 BD F9E4          JSR   BLINK
0901 FD45 F6 A7B9  OF      LDAB  KEY
0902 FD48 C1 46            CMPB  #DA
0903 FD4A 27 E7            BEQ   NOHDR
0904 FD4C C1 45            CMPB  #UA
0905 FD4E 27 ED            BEQ   MOON
0906 FD50 C1 44            CMPB  #LA
0907 FD52 26 10            BNE   CNTRL7
0908 FD54 86 55            LDAA  #MON
0909 FD56 B7 8008          STAA  ACIA
0910 FD59 7C A7D1          INC   SWITCH
0911 FD5C 86 2D            LDAA  #'-
0912 FD5E B7 A7B1          STAA  DISPL+9
0913 FD61 7E F883          JMP   BUILD
0914                       *
0915 FD64 7E FA57  CNTRL7  JMP   CNTRLA
0916                       *
0917                       *EINZELZEICHEN-EINGABE UEBER ACIA
0918                       *
0919 FD67 37       INA1    PSHB
0920 FD68 FF A7BA          STX   SAVEX
0921 FD6B BD F835          JSR   DISP
0922 FD6E FE A7BA          LDX   SAVEX
0923 FD71 33               PULB
0924 FD72 B6 8008  INACIA  LDAA  ACIA
0925 FD75 47               ASRA
0926 FD76 24 EF            BCC   INA1
0927 FD78 B6 8009          LDAA  ACIA+1
0928 FD7B 84 7F            ANDA  #$7F
0929 FD7D 39               RTS
0930                       *
0931                       *LADEN UEBER ACIA
0932                       *
0933 FD7E C6 4C    GOTOL   LDAB  #'L
0934 FD80 BD FCFB          JSR   HDRIN
0935 FD83 8D ED    LD1     BSR   INACIA
0936 FD85 81 53            CMPA  #'S
0937 FD87 26 FA            BNE   LD1
0938 FD89 7D A7D2          TST   HDRSW
0939 FD8C 26 1A            BNE   NOSLCT
0940 FD8E 8D E2            BSR   INACIA
0941 FD90 81 30            CMPA  #'0
0942 FD92 26 EF            BNE   LD1
0943 FD94 CE A7B2          LDX   #DISPL+10
0944 FD97 E6 00    LD2     LDAB  X
0945 FD99 8D D7    LD3     BSR   INACIA
0946 FD9B 27 FC            BEQ   LD3
0947 FD9D 11               CBA
0948 FD9E 26 E3            BNE   LD1
0949 FDA0 08               INX
```

PAGE 019 EUROCOM1

```
0950 FDA1 8C A7B8          CPX   #DISPL+16
0951 FDA4 26 F1            BNE   LD2
0952 FDA6 20 1B            BRA   LOAD
0953 FDA8 8D C8    NOSLCT  BSR   INACIA
0954 FDAA 81 30            CMPA  #'0
0955 FDAC 26 21            BNE   LOAD0
0956 FDAE C6 06            LDAB  #6
0957 FDB0 8D C0    NOS1    BSR   INACIA
0958 FDB2 B7 A7B8          STAA  CARRY
0959 FDB5 CE A7B2          LDX   #DISPL+10
0960 FDB8 BD F90B          JSR   SHIFT
0961 FDBB 37               PSHB
0962 FDBC BD F883          JSR   BUILD
0963 FDBF 33               PULB
0964 FDC0 5A               DECB
0965 FDC1 26 ED            BNE   NOS1
0966 FDC3 8D AD    LOAD    BSR   INACIA
0967 FDC5 81 53            CMPA  #'S
0968 FDC7 26 FA            BNE   LOAD
0969 FDC9 8D A7            BSR   INACIA
0970 FDCB 81 39            CMPA  #'9
0971 FDCD 27 95            BEQ   CNTRL7
0972 FDCF 81 31    LOAD0   CMPA  #'1
0973 FDD1 26 F0            BNE   LOAD
0974 FDD3 7F A7CE          CLR   CKSM
0975 FDD6 BD F929          JSR   BYTE
0976 FDD9 80 02            SUBA  #2
0977 FDDB B7 A7CF          STAA  BCONT
0978 FDDE BD F952          JSR   BADDR
0979 FDE1 FE A7BC          LDX   INDEX
0980 FDE4 BD F929  LOAD1   JSR   BYTE
0981 FDE7 7A A7CF          DEC   BCONT
0982 FDEA 27 09            BEQ   LOAD2
0983 FDEC A7 00            STAA  X
0984 FDEE A1 00            CMPA  X
0985 FDF0 26 0D            BNE   MERR2
0986 FDF2 08               INX
0987 FDF3 20 EF            BRA   LOAD1
0988 FDF5 7C A7CE  LOAD2   INC   CKSM
0989 FDF8 27 C9            BEQ   LOAD
0990 FDFA C6 36            LDAB  #'6
0991 FDFC 7E F9CF  ERROR1  JMP   ERROR
0992 FDFF C6 37    MERR2   LDAB  #'7
0993 FE01 20 F9            BRA   ERROR1
0994                       *
0995                       *BRANCHBERECHNUNG
0996                       *
0997 FE03 C6 42    GOTOB   LDAB  #'B
0998 FE05 BD F93D          JSR   DISADD
0999 FE08 FF A7C8          STX   BEGA
1000 FE0B BD F908          JSR   SHIFT8
1001 FE0E 86 74            LDAA  #'t
1002 FE10 BD F975          JSR   LOADC
1003 FE13 C6 6F            LDAB  #'o
```

PAGE 020 EUROCOM1

```
1004 FE15 BD F93F          JSR   DISAD1
1005 FE18 FF A7CA          STX   ENDA
1006 FE1B BD F902          JSR   SHIF88
1007 FE1E 86 3D            LDAA  #'=
1008 FE20 BD F975          JSR   LOADC
1009 FE23 BD F908          JSR   SHIFT8
1010 FE26 FE A7C8          LDX   BEGA
1011 FE29 86 7E            LDAA  #$7E
1012 FE2B 09         B1    DEX
1013 FE2C 4A               DECA
1014 FE2D 26 FC            BNE   B1
1015 FE2F C6 80            LDAB  #$80
1016 FE31 BC A7CA    B2    CPX   ENDA
1017 FE34 27 0A            BEQ   BOUT
1018 FE36 08               INX
1019 FE37 5C               INCB
1020 FE38 C1 80            CMPB  #$80
1021 FE3A 26 F5            BNE   B2
1022 FE3C C6 38            LDAB  #'8
1023 FE3E 20 BC            BRA   ERROR1
1024 FE40 CE A7B9  BOUT    LDX   #KEY
1025 FE43 E7 00            STAB  X
1026 FE45 BD F97C          JSR   INBYTE
1027 FE48 BD F906          JSR   SHIF28
1028 FE4B 7F A7B9          CLR   KEY
1029 FE4E BD F876          JSR   WAIT
1030 FE51 7E FA57          JMP   CNTRLA
```

```
PAGE   021   EUROCOM1


1032                    *
1033                    *ANZUZEIGENDE TEXTE
1034                    *
1035 FE54 20            TERROR FCC    # Error#
1036 FE5A 20            TEUROC FCC    # Eurocon#
1037 FE62 20            TCONTR FCC    # Control#
1038 FE6A 35            TRGSTR FCC    #5FCCBAIDFC#
1039 FE74 42            TBEG   FCC    #BEG#
1040 FE77 45            TEND   FCC    #EnD#
1041 FE7A 0D            TCRLFS FCB    $D,$A,,,,,,'S,4



1043                    *                                  Taste
1044                    *CONVERTIERUNGSTABELLE                    ASCII - Code
1045                    *                                         7-Segment - Code
1046 FE82 30            CTABLE FCB    '0,%11000000,K0             Taste Code
1047 FE85 31                   FCB    '1,%11111001,K1
1048 FE88 32                   FCB    '2,%10100100,K2
1049 FE8B 33                   FCB    '3,%10110000,K3
1050 FE8E 34                   FCB    '4,%10011001,K4
1051 FE91 35                   FCB    '5,%10010010,K5
1052 FE94 36                   FCB    '6,%10000010,K6
1053 FE97 37                   FCB    '7,%11111000,K7
1054 FE9A 38                   FCB    '8,%10000000,K8
1055 FE9D 39                   FCB    '9,%10010000,K9
1056 FEA0 41                   FCB    'A,%10001000,KA
1057 FEA3 42                   FCB    'B,%10000011,KB
1058 FEA6 43                   FCB    'C,%11000110,KC
1059 FEA9 44                   FCB    'D,%10100001,KD
1060 FEAC 45                   FCB    'E,%10000110,KE
1061 FEAF 46                   FCB    'F,%10001110,KF
1062 FEB2 47                   FCB    'G,%11000010,
1063 FEB5 4C                   FCB    'L,%11000111,
1064 FEB8 4D                   FCB    'M,%11001000,
1065 FEBB 50                   FCB    'P,%10001100,
1066 FEBE 3D                   FCB    '=,%10110111,
1067 FEC1 2D                   FCB    '-,%10111111,
1068 FEC4 75                   FCB    'u,%11100011,
1069 FEC7 74                   FCB    't,%10000111,
1070 FECA 72                   FCB    'r,%10101111,
1071 FECD 63                   FCB    'c,%10100111,
1072 FED0 6E                   FCB    'n,%10101011,
1073 FED3 6F                   FCB    'o,%10100011
1074       FED6     ENDTAB EQU    *+1
```

PAGE 022 EUROCOM1

```
1076                    *
1077                    *SPRUNGTABELLE
1078                    *
1079 FED5 40       FCTABL FCB   KG
1080 FED6 F960            FDB   GOTOG
1081 FED8 47              FCB   KK
1082 FED9 FA90            FDB   GOTOK
1083 FEDB 41              FCB   KM
1084 FEDC F993            FDB   GOTOM
1085 FEDE 42              FCB   KP
1086 FEDF FAB1            FDB   GOTOF
1087 FEE1 43              FCB   KS
1088 FEE2 FB2F            FDB   GOTOS
1089 FEE4 83              FCB   KL
1090 FEE5 FD7E            FDB   GOTOL
1091 FEE7 82              FCB   KR
1092 FEE8 FC2A            FDB   GOTOR
1093 FEEA 81              FCB   KJ
1094 FEEB FE03            FDB   GOTOB
1095      FEED     FCTBEN EQU   *
1096                    *
1097                    *INTERRUPT (IRQ)
1098                    *
1099 FEED FE A7C2 IRQ     LDX   IRQV
1100 FEF0 6E 00           JMP   X
1101                    *
1102                    *SOFTWARE-INTERRUPT
1103                    *
1104 FEF2 FE A7C0 SWI     LDX   SWIV
1105 FEF5 6E 00           JMP   X
1106                    *
1107                    *INTERRUPT (NMI)
1108                    *
1109 FEF7 FE A7BE NMI     LDX   NMIV
1110 FEFA 6E 00           JMP   X
1111                    *
1112                    *INTERRUPT SPRUNGZIELE
1113                    *
1114 FFF8                 ORG   $FFF8

1116 FFF8 FEED            FDB   IRQ
1117 FFFA FEF2            FDB   SWI
1118 FFFC FEF7            FDB   NMI
1119 FFFE F800            FDB   START


1121                      END
```

: Label F960 = Adresse

: Label

FEFC } frei
FFF7 }

```
        NAM     STPUHR.SRC
*
* PROGRAMM ZUR VERWENDUNG DES EUROCOM-MIKROKOMPUTERS
* ALS STOPP-UHR.
*
* UNTER VERWENDUNG EINER GENAU ABGESTIMMTEN
* 10 MILLISEKUNDEN LANGEN PROGRAMMSCHLEIFE WIRD
* DIE 8-STELLIGE SIEBEN-SEGMENT-ANZEIGE ZUR ANZEIGE
* VON STUNDEN, MINUTEN, SEKUNDEN, ZEHNTEL- UND
* HUNDERTSTEL-SEKUNDEN BENUTZT. NACH STARTEN DES
* PROGRAMMES GEHT DIE ANZEIGE DER UHR AUF NULL, NACH
* BETAETIGEN EINER BELIEBIGEN TASTE BEGINNT DIE UHR
* ZU LAUFEN. DIE UHR KANN JEDERZEIT MIT DER TASTE
* 'PFEIL NACH LINKS' (LEFT ARROW) ANGEHALTEN WERDEN.
* BETAETIGEN DER TASTE 'PFEIL NACH UNTEN' (DOWN ARROW)
* LAESST DIE UHR WIEDER WEITERLAUFEN. BETAETIGEN DER
* TASTE 'PFEIL NACH OBEN' (UP ARROW) FUEHRT WIEDER
* AN DEN PROGRAMM-ANFANG, D.H., DIE UHR WIRD WIEDER
* AUF NULL GESETZT, UND DAS PROGRAMM WARTET AUF DIE
* EINGABE EINES BELIEBIGEN ZEICHENS ZUM STARTEN DER
* UHR
*
* DIE GENAU ABGESTIMMTE PROGRAMMSCHLEIFE WIRD VON
* ZWEI SPEICHERZELLEN, DEREN INHALTE ALS ZAEHLER INS
* INDEX-REGISTER GELADEN WERDEN, GESTEUERT. DA DIE
* PROGRAMM-LAUFZEIT NICHT AUF ALLEN MIKROKOMPUTERN
* GENAU GLEICH IST, KANN DURCH VERAENDERN DER BEIDEN
* ZELLEN DIE UHR NACH-JUSTIERT WERDEN. DIE ZELLE
* A583 BEINHALTET DIE HOEHERWERTIGEN 8 BIT DES
* 16-BIT-ZAEHLERS, DIE ZELLE A584 DIE NIEDERWERTIGEN
* 8 BIT.
*
* PROGRAMM-SPEICHER:  152 ZELLEN
* DATEN-SPEICHER:       4 ZELLEN
* STACK-BELASTUNG:      8 ZELLEN
*
* VERSION V02-3-79                          KA
*
```

STPUHR PAGE : 2

```
                *
0030                    ORG     $0030
                *
                * DATEN-SPEICHER-ZELLEN
                *
0030 0001       STUNDE  RMB     1           SPEICHER FUER STUNDEN-ZAEHLER
0031 0001       MINUTE  RMB     1           SPEICHER FUER MINUTEN-ZAEHLER
0032 0001       SEKUND  RMB     1           SPEICHER FUER SEKUNDEN-ZAEHLER
0033 0001       ZEHNTL  RMB     1           ZEHNTEL-HUNDERTSTEL SEKUNDEN
                *
                * SPEICHER-ZELLEN UND UNTERPRROGRAMM-ADRESSEN
                * DES EUROCOM-BETRIEBS-SYSTEMS
                *
     0044       LA      EQU     $44         CODE FUER TASTE 'PFEIL NACH LINKS'
     0045       UA      EQU     $45         CODE FUER TASTE 'PFEIL NACH OBEN'
     0046       DA      EQU     $46         CODE FUER TASTE 'PFEIL NACH UNTEN'
     A7A8       DISPL   EQU     $A7A8       ANFANG DES ANZEIGE-SPEICHERS
     A7B9       KEY     EQU     $A7B9       SPEICHER FUER TASTATUR-EINGABE
     F835       DISP    EQU     $F835       GIBT ANZEIGE-SPEICHER AUF ANZEIGE
     F883       BUILD   EQU     $F883       WANDELT ASCII-BUFFER IN BINAER-MUSTER
     F97A       INBYT2  EQU     $F97A       SCHIEBT 4 ZEICHEN IN ANZEIGE-BUFFER
     FA75       CNTRLB  EQU     $FA75       BEARBEITET FUNKTIONS-TASTEN-EINGABE
```

STPUHR PAGE : 3

```
                    *
A500                        ORG     $A500
                    *
                    * HAUPT-PROGRAMM-TEIL MIT UEBERPRUEFUNG DER
                    * TASTEN-EINGABEN UND ERHOEHEN DER ANZEIGE-
                    * ZAEHLER
                    *
A500 4F             START   CLR A           ALLE ZAEHLER-
A501 97 33                  STA A   ZEHNTL  ZELLEN FUER STUNDEN,
A503 97 32                  STA A   SEKUND  MINUTEN, SEKUNDEN UND
A505 97 31                  STA A   MINUTE  ZEHNTEL-
A507 97 30                  STA A   STUNDE  SEKUNDEN LOESCHEN
A509 8D 7B                  BSR     EINGAB  AUF BELIEBIGE EINGABE WARTEN
                    *
A50B 8D 54          ANZEIG  BSR     OUT8HX  8 ZIFFERN AN ANZEIGE AUSGEBEN
A50D B6 A7B9                LDA A   KEY     WURDE EINE EINGABE-TASTE BETAETIGT?
A510 27 13                  BEQ     CONTIN  NEIN, HUNDERTSTEL-SEKUNDE WARTEN
                    *
A512 81 45          TSTKEY  CMP A   #UA     WURDE 'PFEIL NACH OBEN' EINGEGEBEN?
A514 27 EA                  BEQ     START   JA, ZAEHLER LOESCHEN, NEU STARTEN
A516 81 46                  CMP A   #DA     WURDE 'PFEIL NACH UNTEN' EINGEGEBEN?
A518 27 0B                  BEQ     CONTIN  JA, UHR WEITER-LAUFEN LASSEN
A51A 81 44                  CMP A   #LA     WURDE 'PFEIL NACH LINKS' EINGEGEBEN?
A51C 27 03                  BEQ     STOP    JA, UHR ANHALTEN
A51E 7E FA75                JMP     CNTRLB  ANDERE EINGABE => SPRUNG INS SYSTEM
                    *
A521 8D 63          STOP    BSR     EINGAB  AUF BELIEBIGE EIBGABE WARTEN
A523 20 ED                  BRA     TSTKEY  EINGABE UEBERPRUFEN
                    *
A525 7F A7B9        CONTIN  CLR     KEY     EINGABE-ZELLE LOESCHEN
A528 86 06                  LDA A   #$06    ANZEIGE-SCHLEIFEN-ZAEHLER LADEN
                    *
A52A 36             WARTEN  PSH A           SCHLEIFEN-ZAEHLER KONSERVIEREN
A52B BD F835                JSR     DISP    ZEIT AN ANZEIGE AUSGEBEN
A52E 32                     PUL A           SCHLEIFEN-ZAEHLER WIEDER LADEN
A52F 4A                     DEC A           IST SCHLEIFEN-ZAEHLER ABGELAUFEN?
A530 26 F8                  BNE     WARTEN  NEIN, ANZEIGE NOCHMAL AUSGEBEN
A532 B6 A7B9                LDA A   KEY     WURDE EINE EINGABE-TASTE BETAETIGT?
A535 26 DB                  BNE     TSTKEY  JA, EINGABE UEBERPRUEFEN
A537 FE A596                LDX     DELAY   ZAEHLER FUER JUSTIER-SCHLEIFE
                    *
A53A 09             WARTE1  DEX             IST ZEIT-ZAEHLER ABGELAUFEN?
A53B 26 FD                  BNE     WARTE1  NEIN, ZAEHLER NOCHMAL ERNIEDRIGEN
A53D CE 0032                LDX     #SEKUND  POINTER AUF SEKUNDEN-ZAEHLER
A540 A6 01                  LDA A   1,X     ZEHNTEL-SEKUNDEN LADEN
A542 8B 10                  ADD A   #$10    ZEHNTEL-SEKUNDEN UM 1 ERHOEHEN
A544 19                     DAA             IN DEZIMALE DARSTELLUNG UMWANDELN
A545 A7 01                  STA A   1,X     UEBERLAUF AUF SEKUNDEN-ZAEHLER?
A547 26 C2                  BNE     ANZEIG  NEIN, NEUE UHR-ZEIT ANZEIGEN
A549 C6 02                  LDA B   #2      NAECHSTEN TEIL 2 MAL DURCHLAUFEN
                    *
```

STPUHR PAGE : 4

```
                    *
A54B 8D 41      INCMIN BSR     ADDONE    ZAEHLER UM 1 ERHOEHEN
A54D 81 60             CMP A   #$60      HAT DER ZAEHLER DIE 60 ERREICHT?
A54F 26 BA             BNE     ANZEIG    NEIN, NEUE UHR-ZEIT ANZEIGEN
A551 6F 00             CLR     0,X       JA, WIDER AUF NULL SETZEN
A553 09                DEX               POINTER AUF NAECHSTEN ZAEHLER
A554 5A                DEC B             SCHLEIFE ZWEIMAL DURCHLAUFEN?
A555 26 F4             BNE     INCMIN    NEIN, MINUTEN-ZAEHLER AUCH ERHOEHEN
A557 8D 35             BSR     ADDONE    JA, STUNDEN-ZAEHLER UM 1 ERHOEHEN
A559 81 24             CMP A   #$24      HAT DER ZAEHLER DIE 24 ERREICHT?
A55B 26 AE             BNE     ANZEIG    NEIN, NEUE UHR-ZEIT AUSGEBEN
A55D 6F 00             CLR     0,X       JA, ZAEHLER BEGINNT WIEDER BEI NULL
A55F 20 AA             BRA     ANZEIG    NEUE UHR-ZEIT AUSGEBEN



                    *
                    * UNTERPROGRAMM ZUR AUSGABE DER VIER ZAEHLER-STAENDE
                    * (8 ZIFFERN)
                    *
A561 7F A7B9    OUT8HX CLR     KEY       EINGABE-ZELLE WIEDER LOESCHEN
A564 CE 0030           LDX     #STUNDE    POINTER AUF HOECHSTE ZAEHLER-ADRESSE
A567 BD F97A           JSR     INBYT2    STUNDEN UND MINUTEN IN ANZEIGE-BUFFER
A56A BD F97A           JSR     INBYT2    SEKUNDEN UND HUNDERTSTEL IN BUFFER
A56D BD F883           JSR     BUILD     ASCII-BUFFER IN BINAER-MUSTER WANDELN
A570 CE A7A8           LDX     #DISPL    ANFANG DES BINAER-ANZEIGE-BUFFERS
A573 86 7F             LDA A   #$7F      MASKE, UM IN DER ANZEIGE DEN DEZIMAL-
A575 16                TAB               PUNKT ANZUMACHEN IN BEIDE ACCUS
A576 A4 01             AND A   1,X       HINTER DER ZWEITEN STELLE DEN
A578 A7 01             STA A   1,X       DEZIMAL-PUNKT ANMACHEN
A57A 17                TBA               MASKE NOCHMAL IN BEIDE ACCUS
A57B A4 03             AND A   3,X       HINTER DER VIERTEN STELLE DEN
A57D A7 03             STA A   3,X       DEZIMAL-PUNKT ANMACHEN
A57F E4 05             AND B   5,X       HINTER DER SECHSTEN STELLE DEN
A581 E7 05             STA B   5,X       DEZIMAL-PUNKT ANMACHEN
A583 7E F835           JMP     DISP      AUSGABE DER 8-STELLIGEN ANZEIGE
                    *
                    * UNTERPROGRAMM ZUM ANZEIGEN DER UHR-ZEIT BIS AUF
                    * DER TASTATUR EINE EINGABE ERFOLGT
                    *
A586 8D D9      EINGAB BSR     OUT8HX    8 ZIFFERN AN ANZEIGE AUSGEBEN
A588 B6 A7B9           LDA A   KEY       WURDE EINE EINGABE-TASTE BETAETIGT?
A58B 27 F9             BEQ     EINGAB    NEIN, ANZEIGE ERNEUT AUSGEBEN
A58D 39                RTS               EINGABE IM ACCU A ABLIEFERN
                    *
                    * UNTERPROGRAMM ZUM ERHOEHEN DES ZAEHLERS, AUF DEN
                    * DAS INDEX-REGISTER ZEIGT UM 1
                    *
A58E A6 00      ADDONE LDA A   0,X       ZAEHLER-STAND LADEN
A590 8B 01             ADD A   #1        ZAEHLER-STAND UM 1 ERHOEHEN
A592 19                DAA               DEZIMALE DARSTELLUNG ERZEUGEN
A593 A7 00             STA A   0,X       NEUEN ZAEHLERSTAND WIEDER ABSPEICHERN
A595 39                RTS               RUECKSPRUNG
                    *
                    * WARTE-SCHLEIFEN-ZAEHLER ZUR ERZEUGUNG DER
                    * GENAUEN PROGRAMM-LAUFZEIT
                    *
A596 03A0       DELAY  FDB     $03A0     ZAEHLER BEI BEDARF AENDERN
```

```
        NAM     WEKUHR.SRC
*
* PROGRAMM ZUR VERWENDUNG DES EUROCOM-MIKROKOMPUTERS
* ALS DIGITAL-UHR MIT WECKER
*
* UNTER VERWENDUNG EINER GENAU ABGESTIMMTEN
* 1 SEKUNDE LANGEN PROGRAMM-SCHLEIFE WIRD DIE
* 8-STELLIGE SIEBEN-SEGMENT-ANZEIGE ZUR ANZEIGE
* VON STUNDEN, MINUTEN UND SEKUNDEN BENUTZT.
* NACH STARTEN DES PROGRAMMES BEGINNT DIE  UHR
* MIT DER ANZEIGE 00.00.00 (NULL UHR, NULL MINUTEN
* UND NULL SEKUNDEN) ZU LAUFEN. DAS STELLEN DER
* UHR ERFOLGT FOLGENDERMASSEN:
*
* TASTE 'PFEIL NACH UNTEN' (DOWN ARROW) SETZT DEN
* SEKUNDEN-ZAEHLER AUF NULL
*
* TASTE 'PFEIL NACH LINKS' (LEFT ARROW) ERHOEHT
* DEN MINUTEN-ZAEHLER UM 1
*
* TASTE 'PFEIL NACH OBEN' (UP ARROW) ERHOEHT DEN
* STUNDEN-ZAEHLER UM 1
*
* HIERBEI IST ZU BEACHTEN, DASS DIE OBEN ANGEGEBENE
* REIHENFOLGE EINGEHALTEN WIRD. FAENGT MAN MIT
* DEM STUNDEN-ZAEHLER AN, FUEHRT EIN UEBERLAUF
* BEIM STELLEN DES MINUTEN-ZAEHLERS ZU EINER
* WEITEREN ERHOEHUNG DES STUNDEN-ZAEHLERS.
* WEITERHIN IST ZU BEACHTEN, DASS DIE STELL-
* FUNKTIONEN ERST NACH LOSLASSEN DER TASTE AUS-
* GEFUEHRT WERDEN. ERHOEHEN DES MINUTEN-ZAEHLERS
* UM BEISPIELSWEISE 25 MINUTEN ERFORDERT ALSO
* EIN 25-MALIGES BETAETIGEN DER TASTE 'PFEIL NACH
* LINKS' (UP ARROW).
*
* DIE GENAU ABGESTIMMTE PROGRAMMSCHLEIFE WIRD VON
* ZWEI SPEICHERZELLEN, DEREN INHALTE ALS ZAEHLER
* INS INDEX-REGISTER GELADEN WERDEN, GESTEUERT. DA
* DIE PROGRAMM-LAUFZEIT NICHT AUF ALLEN MIKROKOM-
* PUTERN GENAU GLEICH IST, KANN DURCH VERAENDERN
* DER BEIDEN ZELLEN DIE UHR NACH-JUSTIERT WERDEN.
* DIE ZELLE A5C1 BEINHALTET DIE HOEHERWERTIGEN 8
* BIT DES 16-BIT-ZAEHLERS, DIE ZELLE A5C2 DIE NIEDER-
* WERTIGEN 8 BIT.
*
```

```
*
* DIE WECK-FUNKTION KANN WAEHREND DES PROGRAMM-
* LAUFES JEDERZEIT DURCH BETAETIGEN DER M-TASTE
* (MEMORY-TASTE) EINGESCHALTET WERDEN. NACH BETAE-
* TIGEN DER TASTE GIBT DAS PROGRAMM DIE ZULETZT EIN-
* GEGEBENE WECK-ZEIT AUF DIE ANZEIGE (BEIM ERSTEN
* EINGEBEN ERSCHEINT DIE ANZEIGE 00.00 = NULL UHR UND
* NULL MINUTEN). DIE ZAEHLER-STAENDE DER NORMALEN
* ZEIT-ANZEIGE BLEIBEN DAVON UNBERUEHRT. DURCH
* BETAETIGEN DER GLEICHEN TASTEN WIE ZUM STELLEN
* DER STUNDEN- UND MINUTEN-ANZEIGE FUER DIE NORMALE
* UHR, KANN DIE ANZEIGE SCHRITTWEISE ERHOET
* WERDEN, BIS DIE GEWUENSCHTE WECK-ZEIT AUF
* DER ANZEIGE ERSCHEINT. DURCH BETAETIGEN DER
* TASTE 'PFEIL NACH UNTEN' (DOWN ARROW) WIRD DIE
* ANGEZEIGTE ZEIT (STUNDEN UND MINUTEN) ALS WECK-
* ZEIT VOM PROGRAMM UEBERNOMMEN. VORHER EINGEGEBENE
* WECK-ZEITEN WERDEN DURCH DIE NEUE UEBERSCHRIEBEN.
* DAS PROGRAMM FAEHRT NUN MIT DER ANZEIGE DER NORMALEN
* UHR-ZEIT FORT UND UEBERPRUEFT STAENDIG DIE MOMENTANE
* ZEIT MIT DER WECK-ZEIT. BEI UEBEREINSTIMMUNG WIRD
* ALS WECK-SIGNAL DER CA2-KONTROLL-AUSGANG DES PARALLEL-
* INTERFACE-ADAPTERS (PIA) VOM EUROCOM-BETRIEBS-SYSTEM
* VON LOW- AUF HIGH-PEGEL UMGESCHALTET. DER BENUTZER
* HAT DIE MOEGLICHKEIT UEBER EIN TTL-ANSTEUERBARES
* RELAIS ODER EINE TRANSISTOR-STUFE MIT DEM HIGH-PEGEL
* DES CA2-AUSGANGES BELIEBIGE SCHALTFUNKTIONEN AUSZU-
* LOESEN. NACH AUSLOESEN DER WECK-FUNKTION KANN
* DURCH BETAETIGEN DER K-TASTE (BREAKPOINT-LOESCH-
* TASTE) DER CA2-AUSGANG WIEDER AUF NULL GESETZT UND
* DIE SCHALTFUNKTION RUECKGAENGIG GEMACHT WERDEN. ES
* IST NOCH ZU BEACHTEN, DASS WAEHREND DER EINGABE DER
* WECKZEIT DIE NORMALE UHR ANHAELT. DIE EINGABE SOLLTE
* DAHER NICHT ZU LANGE DAUERN, ANSONSTEN DIE UHR
* ETWAS NACHGEHT.
*
* PROGRAMM-SPEICHER:  223 ZELLEN
* DATEN-SPEICHER:       6 ZELLEN
* STACK-BELASTUNG:     10 ZELLEN
*
* VERSION V05-3-79                    KA
*
```

WEKUHR PAGE : 3

```
                *
0030            ORG     $0030
                *
                * DATEN-SPEICHER-ZELLEN
                *
0030 0001       STUNDE  RMB     1       SPEICHER FUER STUNDEN-ZAEHLER
0031 0001       MINUTE  RMB     1       SPEICHER FUER MINUTEN-ZAEHLER
0032 0001       SEKUND  RMB     1       SPEICHER FUER SEKUNDEN-ZAEHLER
0033 0001       WEKFLG  RMB     1       FLAG-ZELLE FUER DIE WECK-FUNKTION
0034 0001       WEKSTD  RMB     1       WECK-ZEIT STUNDEN-SPEICHER
0035 0001       WEKMIN  RMB     1       WECK-ZEIT MINUTEN-SPEICHER
                *
                * SPEICHER-ZELLEN UND UNTERPRROGRAMM-ADRESSEN
                * DES EUROCOM-BETRIEBS-SYSTEMS
                *
        0041    KM      EQU     $41     CODE FUER MEMORY-TASTE
        0044    LA      EQU     $44     CODE FUER TASTE 'PFEIL NACH LINKS'
        0045    UA      EQU     $45     CODE FUER TASTE 'PFEIL NACH OBEN'
        0046    DA      EQU     $46     CODE FUER TASTE 'PFEIL NACH UNTEN'
        0047    KK      EQU     $47     CODE FUER BREAK-POINT-LOESCH-TASTE
        A7A8    DISPL   EQU     $A7A8   ANFANG DES ANZEIGE-SPEICHERS
        A7B9    KEY     EQU     $A7B9   SPEICHER FUER TASTATUR-EINGABE
        A7BA    SAVEX   EQU     $A7BA   ZWISCHENSPEICHER FUER INDEX-REGISTER
        F835    DISP    EQU     $F835   GIBT ANZEIGE-SPEICHER AUF ANZEIGE
        F883    BUILD   EQU     $F883   WANDELT ASCII-BUFFER IN BINAER-MUSTER
        F902    SHIF88  EQU     $F902   LOESCHT DIE 8-STELLIGE ANZEIGE
        F97A    INBYT2  EQU     $F97A   SCHIEBT 4 ZEICHEN IN ANZEIGE-BUFFER
        F97C    INBYTE  EQU     $F97C   SCHIEBT 2 ZEICHEN IN ANZEIGE-BUFFER
        FA75    CNTRLB  EQU     $FA75   BEARBEITET FUNKTIONS-TASTEN-EINGABE
                *
                * ADRESSE DES PIA-KONTROLL-REGISTERS AUF DEM
                * EUROCOM-MIKROKOMPUTER
                *
        8006    PIAACR  EQU     $8006   PIA-TEIL A KONTROLL-REGISTER
                *
```

WEKUHR PAGE : 4

```
                    *
A500                        ORG     $A500
                    *
                    * HAUPT-PROGRAMM-TEIL MIT UEBERPRUEFUNG DER
                    * TASTEN-EINGABEN UND ERHOEHEN DER ANZEIGE-
                    * ZAEHLER
                    *
A500 7F A7B9 START   CLR     KEY        DIE EINGABE-SPEICHER-ZELLE
A503 CE 0000        LDX     #0000      UND ALLE ZAEHLER-ZELLEN
A506 DF 30          STX     STUNDE     FUER STUNDEN, MINUTEN,
A508 DF 32          STX     SEKUND     SEKUNDEN, DIE WECK-ZEIT UND
A50A DF 34          STX     WEKSTD     DIE WECKER-FLAG-ZELLE LOESCHEN
                    *
A50C 86 36   CLRCA2 LDA A   #$36       DEN CA2-AUSGANG DES PIA
A50E B7 8006        STA A   PIAACR     ZUNAECHST AUF LOW (NULL) SETZEN
                    *
A511 CE 0030 ANZEIG LDX     #STUNDE     POINTER AUF STUNDEN-ZAEHLER-ADRESSE
A514 BD A59E        JSR     OUT6HX     6 ZIFFERN AN ANZEIGE AUSGEBEN
A517 86 4C          LDA A   #$4C       ANZEIGE-SCHLEIFEN-ZAEHLER LADEN
                    *
A519 36      WARTEN PSH A               SCHLEIFEN-ZAEHLER KONSERVIEREN
A51A BD F835        JSR     DISP       ZEIT AN ANZEIGE AUSGEBEN
A51D 32             PUL A              SCHLEIFEN-ZAEHLER WIEDER LADEN
A51E 4A             DEC A              IST SCHLEIFEN-ZAEHLER ABGELAUFEN?
A51F 26 F8          BNE     WARTEN     NEIN, ANZEIGE NOCHMAL AUSGEBEN
A521 FE A5DE        LDX     DELAY      ZAEHLER FUER JUSTIER-SCHLEIFE
                    *
A524 09      WARTE1 DEX                IST ZEIT-ZAEHLER ABGELAUFEN?
A525 26 FD          BNE     WARTE1     NEIN, ZAEHLER NOCHMAL ERNIEDRIGEN
A527 96 33          LDA A   WEKFLG     WURDE WECK-ZEIT EINGEGEBEN?
A529 27 0E          BEQ     NOWECK     NEIN, KEINE UEBERPRUEFUNG NOETIG
A52B DE 30          LDX     STUNDE     IST DER STUNDEN- UND MINUTEN-
A52D 9C 34          CPX     WEKSTD     ZAEHLER BEI DER WECK-ZEIT ANGELANGT?
A52F 26 08          BNE     NOWECK     NEIN, WECK-ZEIT NOCH NICHT ERREICHT
A531 86 3E          LDA A   #$3E       JA, WECK-ZEIT ERREICHT, CA2-
A533 B7 8006        STA A   PIAACR     AUSGANG DES PIA AUF HIGH SETZEN
A536 7F 0033        CLR     WEKFLG     WECKEN ERLEDIGT, FLAG LOESCHEN
                    *
A539 CE 0032 NOWECK LDX     #SEKUND     POINTER AUF SEKUNDEN-ZAEHLER SETZEN
A53C BD A5CE        JSR     COMP60     SEKUNDEN UM 1 ERHOEHEN, UEBERLAUF?
A53F 26 0A          BNE     TSTKEY     NEIN, AUF EINGABE UEBERPRUEFEN
                    *
A541 BD A5CE INCMIN JSR     COMP60     SEKUNDEN UM 1 ERHOEHEN, UEBERLAUF?
A544 26 05          BNE     TSTKEY     NEIN, AUF EINGABE UEBERPRUEFEN
                    *
A546 C6 24   INCSTD LDA B   #$24       STUNDEN-ZAEHLER MIT 24 VERGLEICHEN
A548 BD A5D0        JSR     COMPAR     STUNDEN-ZAEHLER UM 1 ERHOEHEN
                    *
```

WEKUHR PAGE : 5

```
                    *
A54B B6 A7B9 TSTKEY LDA A   KEY       WURDE EINE EINGABE-TASTE BETAETIGT?
A54E 27 C1          BEQ     ANZEIG    NEIN, NEUE UHR-ZEIT AUSGEBEN
A550 7F A7B9        CLR     KEY       EINGABE-ZELLE WIEDER LOESCHEN
A553 CE 0031        LDX     #MINUTE    POINTER AUF STUNDEN-ZAEHLER SETZEN
A556 81 46          CMP A   #DA       WURDE 'PFEIL NACH UNTEN' EINGEGEBEN?
A558 26 04          BNE     NCLEAR    NEIN, SEKUNDEN-ZAEHLER NICHT LOESCHEN
A55A 6F 01          CLR     1,X       JA, SEKUNDEN-ZAEHLER AUF NULL SETZEN
A55C 20 B3          BRA     ANZEIG    NEUE UHR-ZEIT AUSGEBEN
                    *
A55E 81 44  NCLEAR  CMP A   #LA       WURDE 'PFEIL NACH LINKS' EINGEGEBEN?
A560 27 DF          BEQ     INCMIN    JA, MINUTEN-ZAEHLER UM 1 ERHOEHEN
A562 09             DEX               POINTER AUF STUNDEN-ZAEHLER RUECKEN
A563 81 45          CMP A   #UA       WURDE 'PFEIL NACH OBEN' EINGEGEBEN?
A565 27 DF          BEQ     INCSTD    JA, STUNDEN-ZAEHLER UM 1 ERHOEHEN
A567 81 41          CMP A   #KM       WURDE DIE MEMORY-TASTE GEDRUECKT?
A569 27 0A          BEQ     WECKEN    JA, WECK-ZEIT EINLESEN
A56B 81 47          CMP A   #KK       WURDE BREAKPOINT-LOESCH-TASTE GEDR.?
A56D 27 9D          BEQ     CLRCA2    JA, CA2-AUSGANG WIEDER LOESCHEN
                    *
A56F B7 A7B9 RETURN STA A   KEY       EINGABE-ZELLE WIEDER HERSTELLEN
A572 7E FA75        JMP     CNTRLB    ANDERE EINGABE => SPRUNG INS SYSTEM
                    *
                    * PROGRAMM-TEIL ZUM EINLESEN UND ANZEIGEN DER WECK-ZEIT
                    *
A575 CE 0033 WECKEN LDX     #WEKFLG    POINTER AUF WECK-FLAG SETZEN
A578 6F 00          CLR     0,X       LOESCHEN, DAMIT ANZEIGE BLANK
A57A 8D 2A          BSR     OUT4HX    2 BLANK UND 4 ZIFFERN AUSGEBEN
A57C B6 A7B9        LDA A   KEY       WURDE EINE EINGABE-TASTE BETAETIGT?
A57F 27 F4          BEQ     WECKEN    NEIN, ERNEUT ANZEIGEN UND WARTEN
A581 7F A7B9        CLR     KEY       JA, EINGABE-ZELLE WIEDER LOESCHEN
A584 97 33          STA A   WEKFLG    WECK-FUNKTION EIN => FLAG SETZEN
A586 81 46          CMP A   #DA       WURDE 'PFEIL NACH UNTEN' EINGEGEBEN?
A588 27 87          BEQ     ANZEIG    JA, NORMALE UHR-ZEIT WIEDER ANZEIGEN
A58A CE 0035        LDX     #WEKMIN    POINTER AUF WECK-MINUTEN-SPEICHER
A58D C6 60          LDA B   #$60      MINUTEN-ZAEHLER MIT 60 VERGLEICHEN
A58F 81 44          CMP A   #LA       WURDE 'PFEIL NACH LINKS' EINGEGEBEN?
A591 27 07          BEQ     INCREM    JA, WECK-MINUTEN-SPEICHER ERHOEHEN
A593 09             DEX               POINTER AUF WECK-STUNDEN-SPEICHER
A594 81 45          CMP A   #UA       WURDE 'PFEIL NACH OBEN' EINGEGEBEN?
A596 26 D7          BNE     RETURN    ANDERE EINGABE => SPRUNG INS SYSTEM
A598 C6 24          LDA B   #$24      STUNDEN-ZAEHLER MIT 24 VERGLEICHEN
                    *
A59A 8D 34  INCREM  BSR     COMPAR    WECK-SPEICHER UM EINS ERHOEHEN
A59C 20 D7          BRA     WECKEN    NEUE WECK-ZEIT ANZEIGEN
```

WEKUHR PAGE : 6

```
                    *
                    * UNTERPROGRAMM ZUR AUSGABE DER 8-STELLIGEN ANZEIGE
                    * (ZWEI BZW. VIER BLANK UND SECHS BZW. VIER ZIFFERN)
                    *
A59E 8D 11    OUT6HX BSR     BUFFER    6 ZEICHEN IN ANZEIGE-BUFFER BRINGEN
A5A0 A4 03           AND A   3,X       HINTER DER VIERTEN STELLE DEN
A5A2 A7 03           STA A   3,X       DEZIMAL-PUNKT ANMACHEN
A5A4 20 08           BRA     DISPLY    ANZEIGE DER 6 ZEICHEN
                    *
A5A6 8D 09    OUT4HX BSR     BUFFER    4 ZEICHEN IN ANZEIGE-BUFFER BRINGEN
A5A8 86 FF           LDA A   #$FF      DIE DRITTE UND
A5AA A7 02           STA A   2,X       VIERTE ANZEIGEN-
A5AC A7 03           STA A   3,X       STELLE LOESCHEN
                    *
A5AE 7E F835  DISPLY JMP     DISP      AUSGABE DER 8-STELLIGEN ANZEIGE
                    *
A5B1 FF A7BA  BUFFER STX     SAVEX     AUSGABE-POINTER ZWISCHEN-SPEICHERN
A5B4 BD F902         JSR     SHIF88    DIE ACHT-STELLIGE ANZEIGE LOESCHEN
A5B7 FE A7BA         LDX     SAVEX     AUSGABE-POINTER WIEDER LADEN
A5BA BD F97A         JSR     INBYT2    ERSTEN 4 ZIFFERN IN ANZEIGE-BUFFER
A5BD BD F97C         JSR     INBYTE    NAECHSTEN 2 ZIFFERN IN ANZEIGE-BUFFER
A5C0 BD F883         JSR     BUILD     ASCII-BUFFER IN BINAER-MUSTER WANDELN
A5C3 CE A7A8         LDX     #DISPL    ANFANG DES BINAER-ANZEIGE-BUFFERS
A5C6 86 7F           LDA A   #$7F      MASKE, UM IN DER ANZEIGE DEN DEZIMAL-
A5C8 16              TAB               PUNKT ANZUMACHEN IN BEIDE ACCUS
A5C9 E4 05           AND B   5,X       HINTER DER SECHSTEN STELLE DEN
A5CB E7 05           STA B   5,X       DEZIMAL-PUNKT ANMACHEN
A5CD 39              RTS               RUECKSPRUNG
                    *
                    * UNTERPROGRAMM ZUM ERHOEHEN DES ZAEHLERS, AUF DEN
                    * DAS INDEX-REGISTER ZEIGT UND UEBERPRUEFUNG AUF
                    * UEBERLAUF
                    *
A5CE C6 60    COMP60 LDA B   #$60      ZAEHLER-STAND MIT 60 VERGLEICHEN
                    *
A5D0 A6 00    COMPAR LDA A   0,X       ZAEHLER-STAND LADEN
A5D2 8B 01           ADD A   #1        ZAEHLER-STAND UM 1 ERHOEHEN
A5D4 19              DAA               DEZIMALE DARSTELLUNG ERZEUGEN
A5D5 A7 00           STA A   0,X       NEUEN ZAEHLERSTAND WIEDER ABSPEICHERN
A5D7 09              DEX               POINTER AUF NAECHSTEN ZAEHLER RUECKEN
A5D8 11              CBA               HAT ZAEHLER DEN END-STAND ERREICHT?
A5D9 26 02           BNE     NOCARY    NEIN, KEIN UEBERLAUF AUFGETRETEN
A5DB 6F 01           CLR     1,X       JA, WIEDER AUF NULL SETZEN
                    *
A5DD 39       NOCARY RTS               RUECKSPRUNG INS RUFENDE PROGRAMM
                    *
                    * WARTE-SCHLEIFEN-ZAEHLER ZUR ERZEUGUNG DER PROGRAMM-
                    * LAUFZEIT VON 1 SEKUNDE
                    *
A5DE 0096     DELAY  FDB     $0096     ZAEHLER BEI BEDARF AENDERN
```

```
        NAM     RECHT4.SRC
*
* PROGRAMM ZUM ERZEUGEN EINER IN DER FREQUENZ UND
* IM TAKT-VERHAELTNIS STEUERBAREN RECHTECK-FUNKTION
*
* DAS PROGRAMM GIBT AUS DEM CA2-KONTROLL-AUSGANG
* DES PARALLEL-INTERFACE-ADAPTERS (PIA) VOM EUROCOM-
* MIKROKOMPUTER ABWECHSELND EINEN HIGH- UND
* EINEN LOW-PEGEL AUS, SODASS DER CA2-AUSGANG ALS
* RECHTECK-GENERATOR VERWENDET WERDEN KANN. DAS
* TAKT-VERHAELTNIS (VERHAELTNIS DER ZEIT MIT LOW-
* PEGEL ZUR ZEIT MIT HIGH-PEGEL) UND DIE TAKT-
* FREQUENZ WERDEN NACH STARTEN DES PROGRAMMES
* ZUNAECHST VORGEGEBEN. BEIDES KANN WAEHREND
* DES PROGRAMM-LAUFES DURCH ENTSPRECHENDE TASTEN-
* EINGABE BELIEBIG VERAENDERT WERDEN. FOLGENDE
* EINGABEN SIND MOEGLICH:
*
* TASTE 'PFEIL NACH OBEN' (UP ARROW) ERHOEHT DIE
* FREQUENZ:
* HIERBEI WERDEN BEI JEDEM TAKT DIE LAUFZEITEN
* DER WARTE-SCHLEIFEN FUER DEN HIGH- UND
* FUER DEN LOW-PEGEL UM JEWEILS 8 MIKRO-SEKUNDEN
* VERKUERZT, BIS DIE WARTE-SCHLEIFEN-ZAEHLER DEN
* KLEINST-MOEGLICHEN WERT (0001) ERREICHT HABEN.
* EINE WEITERE ERHOEHUNG DER FREQUENZ IST MIT DIESEM
* PROGRAMM NICHT MOEGLICH. DAS PROGRAMM "RECHT3.SRC"
* ERMOEGLICHT HOEHERE FREQUENZEN, JEDOCH IST DIE
* FREQUENZ NICHT VON DER TASTATUR AUS STEUERBAR.
* WIRD VOR ERREICHEN DES MINIMALEN ZAEHLER-
* STANDES DIE TASTE 'PFEIL NACH LINKS' (LEFT
* ARROW) BETAETIGT, WIRD DIE FREQUENZ-VERAENDERUNG
* ANGEHALTEN.
*
* TASTE 'PFEIL NACH UNTEN' (DOWN ARROW) ERNIEDRIGT
* DIE FREQUENZ:
* HIERBEI WERDEN BEI JEDEM TAKT DIE LAUFZEITEN
* DER WARTE-SCHLEIFEN FUER DEN HIGH- UND FUER
* DEN LOW-PEGEL UM JEWEILS 6 MIKRO-SEKUNDEN
* VERLAENGERT, BIS DIE WARTE-SCHLEIFEN-ZAEHLER
* IHREN GROESST-MOEGLICHEN WERT (FFFF) ERREICHT
* HABEN. EINE WEITERE ERNIEDRIGUNG DER FREQUENZ
* IST NICHT MOEGLICH. WIRD VOR ERREICHEN DES
* MAXIMALEN ZAEHLER-STANDES DIE TASTE 'PFEIL
* NACH LINKS' (LEFT ARROW) BETAETIGT, WIRD DIE
* FREQUENZ-VERAENDERUNG ANGEHALTEN.
*
* TASTE 'PFEIL NACH LINKS' (LEFT ARROW) BEENDET DIE
* JEWEILS ZUVOR GEWAEHLTE FREQUENZ-VERAENDERUNG,
* WONACH DIE FREQURENZ WIEDER KONSTANT BLEIBT.
*
```

RECHT4 PAGE : 2

```
* TASTE 'MEMORY' FUEHRT ZUM VERAENDERN DES TAKT-
* VERHAELTNISSES MIT HILFE DER 'PFEIL-TASTEN';
* DA DIE 'PFEIL-TASTEN' BEREITS ZUR FREQUENZ-
* VERAENDERUNG VERWENDET WERDEN, MUSS MIT HILFE
* EINER ANDEREN TASTE (HIER MIT DER 'MEMORY-
* TASTE') DIE WIRKUNG DER 'PFEIL-TASTEN' AUF DIE
* TAKT-VERHAELTNIS-AENDERUNG UMGESCHALTET WERDEN.
* NACH BETAETIGEN DER 'MEMORY-TASTE' HABEN DIE
* PFEIL-TASTEN ALSO ANDERE FUNKTIONEN ALS BEI
* DER FREQUENZ-VERAENDERUNG.
*
* TASTE 'PFEIL NACH OBEN' (UP ARROW) ERHOEHT DEN
* HIGH-ANTEIL DES RECHTECK-SIGNALS;
* HIERBEI WIRD BEI JEDEM TAKT DIE LAUFZEIT DER
* WARTE-SCHLEIFE FUER DEN HIGH-PEGEL UM 8 MIKRO-
* SEKUNDEN VERLAENGERT UND DIE LAUFZEIT DER WARTE-
* SCHLEIFE FUER DEN LOW-PEGEL UM 8 MIKRO-SEKUNDEN
* VERKUERZT. DIE FREQUENZ DES RECHTECK-SIGNALS
* BLEIBT UNVERAENDERT, NUR DAS TAKT-VERHAELTNIS
* WIRD ZUGUNSTEN DES HIGH-PEGELS VERAENDERT. ERST
* WENN DER ZAEHLER FUER DEN HIGH-PEGEL SEIN MAXIMUM
* (FFFF) ODER DER ZAEHLER FUER DEN LOW-PEGEL SEIN
* MINIMUM (0001) ERREICHT HAT, WIRD NUR NOCH DER
* ANDERE WARTE-SCHLEIFEN-ZAEHLER VERAENDERT, WAS
* AUCH EINE FREQUENZ-AENDERUNG ZUR FOLGE HAT. SIND
* BEIDE ZAEHLER 'AM ANSCHLAG', BLEIBT DAS BETAE-
* TIGEN DER TASTE OHNE AUSWIRKUNG. WIRD VOR ERREI-
* CHEN DES MAXIMALEN BZW. MINIMALEN ZAEHLER-STANDES
* DIE TASTE 'PFEIL NACH LINKS' (LEFT ARROW) BETAETIGT,
* WIRD DIE TAKT-VERHAELTNIS-AENDERUNG ABGESCHLOSSEN,
* UND DIE 'PFEIL-TASTEN' BEWIRKEN WIEDER EINE
* FREQUENZ-VERAENDERUNG.
*
* TASTE 'PFEIL NACH UNTEN' (DOWN ARROW) ERHOEHT
* DEN LOW-ANTEIL DES RECHTECK-SIGNALS;
* HIERBEI WIRD BEI JEDEM TAKT DIE LAUFZEIT
* DER WARTE-SCHLEIFE FUER DEN LOW-PEGEL UM
* 8 MIKRO-SEKUNDEN VERLAENGERT UND DIE LAUF-
* ZEIT DER WARTE-SCHLEIFE FUER DEN HIGH-PEGEL
* UM 8 MIKRO-SEKUNDEN VERKUERZT. DIE FREQUENZ DES
* RECHTECK-SIGNALS BLEIBT UNVERAENDERT, NUR DAS
* TAKT-VERHAELTNIS WIRD ZUGUNSTEN DES LOW-PEGELS
* VERAENDERT. ERST WENN DER ZAEHLER FUER DEN LOW-
* PEGEL SEIN MAXIMUM (FFFF) ODER DER ZAEHLER FUER
* DEN HIGH-PEGEL SEIN MINIMUM (0001) ERREICHT HAT,
* WIRD NUR NOCH DER ANDERE WARTE-SCHLEIFEN-ZAEHLER
* VERAENDERT, WAS AUCH EINE FREQUENZ-AENDERUNG ZUR
* FOLGE HAT. SIND BEIDE ZAEHLER 'AM ANSCHLAG',
* BLEIBT DAS BETAETIGEN DER TASTE OHNE AUSWIRKUNG.
* WIRD VOR ERREICHEN DES MAXIMALEN BZW. MINI-
* MALEN ZAEHLER-STANDES DIE TASTE 'PFEIL NACH
* LINKS' (LEFT ARROW) BETAETIGT, WIRD DIE TAKT-
* VERHAELTNIS-AENDERUNG ABGESCHLOSSEN, UND DIE
* 'PFEIL-TASTEN' BEWIRKEN WIEDER EINE FREQUENZ-
* VERAENDERUNG.
```

RECHT4 PAGE : 3

```
                *
                * TASTE 'PFEIL NACH LINKS' (LEFT ARROW) SCHALTET
                * WIEDER VON TAKT-VERHAELTNIS-AENDERUNG AUF FREQUENZ-
                * VERAENDERUNG ZURUECK;
                * NACH BETAETIGEN DER TASTE 'PFEIL NACH LINKS' WIRD DIE
                * VERAENDERUNG DES TAKT-VERHAELTNISSES ABGESCHLOSSEN
                * UND DIE 'PFEIL-TASTEN' BEWIRKEN WIEDER EINE FREQUENZ-
                * VERAENDERUNG.
                *
                * JEDE ANDERE EINGABE BLEIBT UNBERUECKSICHTIGT.
                *
                * PROGRAMM-SPEICHER:   146 ZELLEN
                * DATEN-SPEICHER:        7 ZELLEN
                * STACK-BELASTUNG:       4 ZELLEN
                *
                * VERSION V09-3-79                          KA
                *
0030                    ORG     $0030
                *
                * DATEN-SPEICHER-ZELLEN
                *
0030 0002       DELHGH  RMB     2       WARTE-SCHLEIFEN-ZAEHLER HIGH-PEGEL
0032 0002       DELLOW  RMB     2       WARTE-SCHLEIFEN-ZAEHLER LOW-PEGEL
0034 0001       VERHLT  RMB     1       FLAG FUER TAKT-VERHAELTNIS-AENDERUNG
0035 0001       UPFLAG  RMB     1       FLAG FUER FREQUENZ-ERHOEHUNG
0036 0001       DWNFLG  RMB     1       FLAG FUER FREQUENZ-ERNIEDRIGUNG
                *
                * SPEICHER-ZELLEN UND UNTERPROGRAMM-ADRESSEN
                * DES EUROCOM-BETRIEBS-SYSTEMS
                *
     F868       TSTKEY  EQU     $F868   EINLESEN DER TASTEN-EINGABE
                *
                * ADRESSE DES PIA-KONTROLL-REGISTERS AUF DEM
                * EUROCOM-MIKROKOMPUTER
                *
     8006       PIAACR  EQU     $8006   PIA-TEIL A KONTROLL-REGISTER
     8010       SYSPIA  EQU     $8010   ADRESSE DES BETRIEBS-SYSTEM-PIA
                *
```

ECHT4 PAGE : 4

```
                    *
00                  ORG     $A500
                    *
                    * HAUPT-PROGRAMM-TEIL MIT UEBERPRUEFUNG DER
                    * TASTEN-EINGABEN UND VERAENDERN DER WARTE-
                    * SCHLEIFEN-ZAEHLERS
                    *
00 4F               START   CLR A           ALLE FLAG-ZELLEN FUER
01 97 35                    STA A   UPFLAG  FREQUENZ- UND VERHAELTNIS-
03 97 36                    STA A   DWNFLG  AENDERUNG
05 97 34                    STA A   VERHLT  LOESCHEN
07 CE 0100                  LDX     #$0100  DIE WARTE-SCHLEIFEN-ZAEHLER
0A DF 30                    STX     DELHGH  FUER DEN HIGH-PEGEL UND DEN
0C DF 32                    STX     DELLOW  LOW-PEGEL VORSETZEN
                    *
0E DE 32            TAKT    LDX     DELLOW  ZAEHLER FUER LOW-PEGEL-SCHLEIEFE
10 86 36                    LDA A   #$36    DEN CA2-AUSGANG DES PIA LOESCHEN
12 8D 2E                    BSR     WARTEN  LOW-PEGEL AUSGEBEN, WARTE-SCHLEIFE
14 DE 30                    LDX     DELHGH  ZAEHLER FUER HIGH-PEGEL-SCHLEIFE
16 86 3E                    LDA A   #$3E    DEN CA2-AUSGANG DES PIA SETZEN
18 8D 28                    BSR     WARTEN  HIGH-PEGEL AUSGEBEN, WARTE-SCHLEIFE
1A 96 34                    LDA A   VERHLT  WIRD TAKT-VERHAELTNIS GEAENDERT?
1C 27 14                    BEQ     FREQUZ  NEIN, FREQUENZ-VERAENDERUNG
                    *
                    * PROGRAMM-TEIL ZUM VERAENDERN DES TAKT-
                    * VERHAELTNISSES
                    *
1E 96 35                    LDA A   UPFLAG  SOLL HIGH-ANTEIL ERHOEHT WERDEN?
20 26 0A                    BNE     MORHGH  JA, HIGH-ANTEIL WIRD GROESSER
22 96 36                    LDA A   DWNFLG  SOLL LOW-ANTEIL ERHOEHT WERDEN?
24 27 E8                    BEQ     TAKT    NEIN, NAECHSTEN TAKT AUSGEBEN
26 8D 4E                    BSR     DECHGH  JA, HIGH-PEGEL-ZAEHLER ERNIEDRIGEN
                    *
28 8D 5A            INXLOW  BSR     INCLOW  LOW-PEGEL-ZAEHLER ERHOEHEN
2A 20 E2                    BRA     TAKT    NAECHSTEN TAKT AUSGEBEN
                    *
2C 8D 50            MORHGH  BSR     INCHGH  HIGH-PEGEL-ZAEHLER ERHOEHEN
                    *
2E 8D 5C            DEXLOW  BSR     DECLOW  LOW-PEGEL-ZAEHLER ERNIEDRIGEN
                    *
30 20 DC                    BRA     TAKT    NAECHSTEN TAKT AUSGEBEN
                    *
```

ECHT4 PAGE : 5

```
                    *
                    * PROGRAMM-TEIL ZUM VERAENDERN DER TAKT-
                    * FREQUENZ
                    *
32 96 35    FREQUZ  LDA A   UPFLAG    SOLL FREQUENZ ERHOEHT WERDEN?
34 26 08            BNE     DECREM    JA, SCHLEIFEN-ZAEHLER ERNIEDRIGEN
36 96 36            LDA A   DWNFLG    SOLL FREQUENZ ERNIEDRIGT WERDEN?
38 27 D4            BEQ     TAKT      NEIN, NAECHSTEN TAKT AUSGEBEN
3A 8D 42            BSR     INCHGH    HIGH-PEGEL-ZAEHLER UM 1 ERHOEHEN
3C 20 EA            BRA     INXLOW    LOW-PEGEL-ZAEHLER UM 1 ERHOEHEN
                    *
3E 8D 36    DECREM  BSR     DECHGH    HIGH-PEGEL-ZAEHLER UM 1 ERNIEDRIGEN
40 20 EC            BRA     DEXLOW    LOW-PEGEL-ZAEHLER UM 1 ERNIEDRIGEN
                    *
                    * UNTERPROGRAMM FUER DEN ABLAUF DER
                    * WARTE-SCHLEIFE
                    *
42 B7 8006  WARTEN  STA A   PIAACR    CODE-WORT INS PIA-KONTROLL-REGISTER
                    *
45 09       WARTE1  DEX               IST ZEIT-ZAEHLER ABGELAUFEN?
46 26 FD            BNE     WARTE1    NEIN, ZAEHLER NOCHMAL ERNIEDRIGEN
48 CE 0034          LDX     #VERHLT    POINTER AUF ERSTE FLAG SETZEN
4B 86 01            LDA A   #1        ANWAHL-CODE FUER 'MEMORY-TASTE'
4D 8D 1E            BSR     TASTE     IST 'MEMORY-TASTE' GEDRUECKT?
4F 27 12            BEQ     SETFLG    JA, FLAG VERHAELTNIS-AENDERUNG SETZEN
51 86 04            LDA A   #4        ANWAHL-CODE FUER 'LEFT-ARROW-TASTE'
53 8D 18            BSR     TASTE     TASTE 'PFEIL NACH LINKS' GEDRUECKT?
55 27 0F            BEQ     CLRFLG    JA, ALLE FREQUENZ-FLAGS LOESCHEN
57 08               INX               POINTER AUF NAECHSTE FLAG SETZEN
58 4C               INC A             ANWAHL-CODE FUER 'UP-ARROW-TASTE'
59 8D 12            BSR     TASTE     TASTE 'PFEIL NACH OBEN' GEDRUECKT?
5B 27 06            BEQ     SETFLG    JA, FLAG 'FREQUENZ ERHOEHEN' SETZEN
5D 08               INX               POINTER AUF NAECHSTE FLAG SETZEN
5E 4C               INC A             ANWAHL-CODE FUER 'DOWN-ARROW-TASTE'
5F 8D 0C            BSR     TASTE     TASTE 'PFEIL NACH UNTEN' GEDRUECKT?
61 26 02            BNE     RETURN    NEIN, KEINE GUELTIGE EINGABE ERFOLGT
                    *
63 E7 00    SETFLG  STA B   0,X       ENTSPRECHENDE FREQUENZ-FLAG SETZEN
                    *
65 39       RETURN  RTS               RUECKSPRUNG
                    *
66 6F 00    CLRFLG  CLR     0,X       ALLE FLAGS FUER
68 6F 01            CLR     1,X       FREQUENZ- UND VERHAELTNIS-
6A 6F 02            CLR     2,X       AENDERUNG LOESCHEN
6C 39               RTS               RUECKSPRUNG
                    *
                    * UNTERPROGRAMM ZUM EINHOLEN DER TASTEN-EINGABE
                    *
6D B7 8011  TASTE   STA A   SYSPIA+1   ANWAHL INS PIA-DATEN-REGISTER DES
70 BD F868          JSR     TSTKEY    SYSTEM-PIA. TASTEN-EINGABE HOLEN
73 C1 40            CMP B   #$40       ACCU B ENTHAELT $40 BEI GUELTIGER
75 39               RTS               EINGABE. RUECKSPRUNG
```

```
ECHT4                                                    PAGE :    6

                        *
                        * UNTERPROGRAMM ZUM ERNIEDRIGEN DES WARTE-SCHLEIFEN-
                        * ZAEHLERS FUER DEN HIGH-PEGEL
                        *
176 DE 30    DECHGH LDX     DELHGH    HIGH-PEGEL-ZAEHLER UM 1 ERNIEDRIGEN
178 09              DEX               IST DER ZAEHLER-STAND JETZT 0000?
179 27 02           BEQ     NOTDEX    JA, ZAEHLER BLEIBT AUF 0001
                        *
17B DF 30    STXHGH STX     DELHGH    NEIN, NEUEN ZAEHLER-STAND ABSPEICHERN
                        *
17D 39       NOTDEX RTS               RUECKSPRUNG
                        *
                        * UNTERPROGRAMM ZUM ERHOEHEN DES WARTE-SCHLEIFEN-
                        * ZAEHLERS FUER DEN HIGH-PEGEL
                        *
17E DE 30    INCHGH LDX     DELHGH    HIGH-PEGEL-ZAEHLER UM 1 ERHOEHEN
180 08              INX               IST DER ZAEHLER-STAND JETZT 0000?
181 26 F8           BNE     STXHGH    NEIN, NEUEN ZAEHLER-STAND ABSPEICHERN
183 39              RTS               JA, ZAEHLER BLEIBT FFFF, RUECKSPRUNG
                        *
                        * UNTERPROGRAMM ZUM ERHOEHEN DES WARTE-SCHLEIFEN-
                        * ZAEHLERS FUER DEN LOW-PEGEL
                        *
184 DE 32    INCLOW LDX     DELLOW    LOW-PEGEL-ZAEHLER UM 1 ERHOEHEN
186 08              INX               IST DER ZAEHLER-STAND JETZT 0000?
187 27 02           BEQ     NOTINX    JA, ZAEHLER BLEIBT AUF FFFF
                        *
189 DF 32    STXLOW STX     DELLOW    NEIN, NEUEN ZAEHLER-STAND ABSPEICHERN
                        *
18B 39       NOTINX RTS               RUECKSPRUNG
                        *
                        * UNTERPROGRAMM ZUM ERNIEDRIGEN DES WARTE-SCHLEIFEN-
                        * ZAEHLERS FUER DEN LOW-PEGEL
                        *
18C DE 32    DECLOW LDX     DELLOW    LOW-PEGEL-ZAEHLER UM 1 ERNIEDRIGEN
18E 09              DEX               IST DER ZAEHLER-STAND JETZT 0000?
18F 26 F8           BNE     STXLOW    NEIN, NEUEN ZAEHLER-STAND ABSPEICHERN
191 39              RTS               JA, ZAEHLER BLEIBT 0001, RUECKSPRUNG
```

```
        NAM     MULTIP.SRC
*
* UNTERPROGRAMM ZUR MULTIPLIKATION ZWEIER VORZEICHEN-
* LOSER GANZER 16-BIT-BINAER-ZAHLEN
*
* FUER DIE BENUTZUNG DES UNTERPROGRAMMES IN EINEM
* UEBERGEORDNETEN VERARBEITUNGS-PROGRAMM SIND
* FOLGENDE EINGANGS-BEDINGUNGEN ZU ERFUELLEN:
*
* 1: DER MULTIPLIKAND MUSS IN DEN ZELLEN 'MKANDH'
*    ($0040 = HOHER-WERTIGE 8 BIT) UND 'MKANDL' ($0041
*     = NIEDER-WERTIGE 8 BIT) STEHEN.
* 2: DER MULTIPLIKATOR MUSS IN DEN ZELLEN 'KATORH'
*    ($0042 = HOEHER-WERTIGE 8 BIT) UND 'KATORL' ($0043
*     = NIEDER-WERTIGE 8 BIT) STEHEN.
*
* DAS 32-BIT-LANGE ERGEBNIS WIRD MIT DEN HOECHST-
* WERTIGEN 8 BIT IN DER ZELLE 'MKANDH' ($0040),
* MIT DEN ZWEIT-HOECHST-WERTIGEN 8 BIT IN DER
* ZELLE 'MKANDL' ($0041), MIT DEN DRITT-HOECHST-
* WERTIGEN 8 BIT IN DER ZELLE 'KATORH' ($0042)
* UND MIT DEN NIEDER-WERTIGSTEN 8 BIT IN DER ZELLE
* 'KATORL' ($0043) ABGESPEICHERT. DER MULTIPLIKATOR
* UND DER MULTIPLIKAND WERDEN ALSO VOM ERGEBNIS
* UEBERSPEICHERT. DIE INHALTE DER ACCUMULATOREN 'A'
* UND 'B' UND DES INDEX-REGISTERS WERDEN ZERSTOERT.
*
* DAS PROGRAMM LAEUFT FOLGENDER-MASSEN AB:
* DAS INDEX-REGISTER WIRD MIT 16 GELADEN UND ALS ZAEHLER
* FUER DIE PROGRAMM-SCHLEIFE ZUM 16-MALIGEN DURCHLAUF
* BENUTZT (16-BIT-MULTIPLIKATOR). DARAUFHIN WIRD DER
* MULTIPLIKATOR BIT FUER BIT UEBERPRUEFT, BEGINNEND MIT
* DEM NIEDER-WERTIGSTEN BIT (LSB = LEAST SIGNIFICANT BIT).
* IST DAS LSB GESETZT, WIRD DER MULTIPLIKAND IN DIE
* BEIDEN HOECHST-WERTIGEN BYTE (ACUUMULATOR 'A' UND 'B')
* GELADEN. DURCH 16-FACHES ROTIEREN (EINMAL PRO DURCHLAUF)
* STEHT DER MULTIPLIKAND DANN ENTSPRECHEND DER WERTIGKEIT
* DES MULTIPLIKATOR-BITS (= 1) IN DEN NIEDER-WERTIGSTEN
* BEIDEN BYTE DES 32-BIT-ERGEBNISSES. BEIM ZWEITEN DURCH-
* LAUF WIRD DAS ZWEIT-NIEDER-WERTIGSTE BIT DES MULTIPLI-
* KATORS UEBERPRUEFT UND, FALLS DAS BIT GESETZT IST, DER
* MULTIPLIKAND ZU ACCUMULATOR 'A' UND 'B' ADDIERT. DA DANN
* NUR NOCH 15-MAL ROTIERT WIRD, STEHT DER MULTIPLIKAND UM
* 1 BIT NACH LINKS VERSCHOBEN (GLEICH EINER MULTIPLIKATION
* MIT 2) IM 32-BIT-ERGEBNIS, ENTSPRECHEND DER WERTIGKEIT
* (= 2) DES MULTIPLIKATOR-BITS. BEIM LETZTEN DURCHLAUF
* WIRD, FALLS DAS HOECHST-WERTIGSTE BIT (MSB = MOST
* SIGNIFICANT BIT) BESETZT IST, DER MULTIPLIKAND ZU
* ACCUMULATOR 'A' UND 'B' ADDIERT UND NOCH EINMAL ROTIERT.
* DER MULTIPLIKAND STEHT DANN UM 15 BIT VERSCHOBEN (GLEICH
* EINER MULTIPLIKATION MIT 2 HOCH 15) IM 32-BIT-ERGEBNIS.
* TRITT BEI DER LETZTEN ADDITION EIN UEBERTRAG AUF, WIRD
* BEI DEM ROTIEREN DAS HOECHST-WERTIGSTE BIT IM ERGEBNIS
* GESETZT. IST EIN MULTIPLIKATOR-BIT NICHT GESETZT, WIRD
* NUR ROTIERT UND NICHT ADDIERT.
```

```
                    *
                    * PROGRAMM-SPEICHER:   33 ZELLEN
                    * DATEN-SPEICHER:      4 ZELLEN
                    * STACK-BELASTUNG:   2+0 ZELLEN
                    *
                    * VERSION 20-3-79                                  KA
                    *
0040                        ORG     $0040
                    *
                    * DATEN-SPEICHER-ZELLEN
                    *
0040 0001           MKANDH  RMB     1           HIGH-BYTE DES MULTIPLIKAND
                    *                           UND 1. HIGH-BYTE DES ERGEBNISSES
0041 0001           MKANDL  RMB     1           LOW-BYTE DES MULTIPLIKAND
                    *                           UND 2. HIGH-BYTE DES ERGEBNISSES
0042 0001           KATORH  RMB     1           HIGH-BYTE DES MULTIPLIKATORS
                    *                           UND 3. HIGH-BYTE DES ERGEBNISSES
0043 0001           KATORL  RMB     1           LOW-BYTE DES MULTIPLIKATORS
                    *                           UND LOW-BYTE DES ERGEBNISSES
```

```
                    *
A500                        ORG     $A500
                    *
                    * UNTERPROGRAMM ZUR MULTIPLIKATION ZWEIER VORZEICHEN-
                    * LOSER GANZER 16-BIT-BINAER-ZAHLEN
                    *
A500 CE 0010 MULTIP LDX     #0016       SCHLEIFEN-ZAEHLER AUF 16 SETZEN
A503 4F             CLR A               BEIDE ACCUS WEGEN DER FOLGENDEN
A504 5F             CLR B               ADDITION ZU BEGINN LOESCHEN
A505 76 0042        ROR     KATORH      BEIDE NIEDER-WERTIGEN BYTES ROTIEREN
A508 76 0043        ROR     KATORL      NAECHSTES MULTIPLIKATOR-BIT IN CARRY
                    *
A50B 24 04    AGAIN BCC     ROTATE      CARRY NICHT GESETZT, NUR ROTIEREN
A50D DB 41          ADD B   MKANDL      ZU ACCU A UND B HIGH- UND LOW-
A50F 99 40          ADC A   MKANDH      BYTE DES MULTIPLIKAND ADDIEREN
                    *
A511 46       ROTATE ROR A              DIE BEIDEN HOEHER-WERTIGEN
A512 56              ROR B              ERGEBNIS-BYTE EINMAL ROTIEREN
A513 76 0042         ROR    KATORH      BEIDE NIEDER-WERTIGEN BYTES ROTIEREN
A516 76 0043         ROR    KATORL      NAECHSTES MULTIPLIKATOR-BIT IN CARRY
A519 09              DEX                SCHLEIFE 16-MAL DURCHLAUFEN?
A51A 26 EF           BNE    AGAIN       NEIN, NOCHMAL DURCHLAUFEN
A51C 97 40           STA A  MKANDH      DIE BEIDEN HOECHSTEN BYTE DES
A51E D7 41           STA B  MKANDL      ERGEBNISSES ABSPEICHERN
A520 39              RTS                RUECKSPRUNG INS RUFENDE PROGRAMM
```

```
        NAM     DIVIDE.SRC
*
* UNTERPROGRAMM ZUR DIVISION ZWEIER VORZEICHEN-
* LOSER GANZER 16-BIT-BINAER-ZAHLEN
*
* FUER DIE BENUTZUNG DES UNTERPROGRAMMES IN EINEM
* UEBERGEORDNETEN VERARBEITUNGS-PROGRAMM SIND
* FOLGENDE EINGANGS-BEDINGUNGEN ZU ERFUELLEN:
*
* 1: DER DIVIDEND MUSS IN DEN ZELLEN 'DIVIDH'
*    ($0040 = HOHER-WERTIGE 8 BIT) UND 'DIVIDL' ($0041
*     = NIEDER-WERTIGE 8 BIT) STEHEN.
* 2: DER DIVISOR MUSS IN DEN ZELLEN 'DIVISH'
*    ($0042 = HOEHER-WERTIGE 8 BIT) UND 'DIVISL' ($0043
*     = NIEDER-WERTIGE 8 BIT) STEHEN.
*
* DAS 16-BIT-LANGE ERGEBNIS WIRD MIT DEN HOEHER-
* WERTIGEN 8 BIT IN DER ZELLE 'RESLTH' ($0044)
* UND MIT DEN NIEDER-WERTIGEN 8 BIT IN DER ZELLE
* 'RESLTL' ($0045) ABGESPEICHERT. DIE INHALTE DER
* ACCUMULATOREN 'A' UND 'B' WERDEN ZERSTOERT, DER
* INHALT DES INDEX-REGISTERS BLEIBT UNVERAENDERT.
*
* DAS PROGRAMM LAEUFT FOLGENDER-MASSEN AB:
* NACH LOESCHEN DER ZELLEN FUER DAS ERGEBNIS WIRD DER
* DIVISOR SOLANGE NACH LINKS ROTIERT, BIS DAS HOECHST-
* WERTIGE BIT (MSB = MOST SIGNIFICANT BIT) GESETZT IST.
* DIE ANZAHL, WIE OFT ROTIERT WORDEN IST, WIRD IN EINEM
* ZAEHLER ABGESPEICHERT. DANACH WIRD DER ROTIERTE DIVISOR
* VOM DIVIDEND ABGEZOGEN.
* IST DER DIVIDEND GROESSER ALS DER ROTIERTE DIVISOR,
* WIRD DAS NIEDER-WERTIGSTE BIT (LSB = LEAST SIGNIFICANT
* BIT) IM LOW-BYTE DES ERGEBNISSES GESETZT. DADURCH,
* DASS DAS ERGEBNIS GENAUSO OFT ROTIERT WIRD, WIE VORHER
* DER DIVISOR ROTIERT WORDEN IST, GIBT DAS BIT NACHHER
* AN, WIEOFT DER DIVISOR IM DIVIDEND ENTHALTEN IST.
* WURDE DER DIVISOR NICHT ROTIERT, WIRD DAS ERGEBNIS
* EBENFALLS NICHT ROTIERT UND DER DIVISOR IST 1-MAL
* (2 HOCH 0) IM DIVIDEND ENTHALTEN. WURDE DER DIVISOR
* 1-MAL ROTIERT, STEHT DAS ERGEBNIS-BIT AM PROGRAMM-
* ENDE AN DER ZWEIT-NIEDER-WERTIGSTEN STELLE, UND
* DER DIVISOR IST 2-MAL (2 HOCH 1) IM DIVIDEND ENT-
* HALTEN. WURDE DER DIVISOR 2-MAL ROTIERT, RUECKT
* DAS ERGEBNIS-BIT AN DIE DRITT-NIEDER-WERTIGSTE
* STELLE, UND DER DIVISOR IST 4-MAL (2 HOCH 2) IM
* DIVIDEND ENTHALTEN, USW. BEI DER WEITEREN RECHNUNG
* WIRD DER REST, DER BEI DER SUBTRAKTION VON DIVI-
* DEND UND DIVISOR ENTSTEHT, ALS NEUER DIVIDEND
* VERWENDET.
*
```

DIVIDE PAGE 2

```
                *
                * IST DER DIVIDEND KLEINER ALS DER ROTIERTE DIVISOR,
                * WIRD DAS NIEDER-WERTIGSTE BIT IM ERGEBNIS GELOESCHT
                * UND DER DIVISOR EINMAL NACH RECHTS ROTIERT (DURCH
                * 2 GETEILT). DER DIVIDEND BLEIBT UNVERAENDERT.
                * DIE RECHNUNG IST BEENDET, WENN DER DIVISOR IM
                * VERLAUF DER RECHNUNG SO OFT NACH RECHTS ROTIERT
                * WORDEN IST, WIE ER ZU BEGINN NACH LINKS ROTIERT
                * WURDE. DIE LAUFZEIT DES UNTERPROGRAMMES IST ALSO
                * UM SO GROESSER, JE KLEINER DER DIVISOR IST. IST
                * DER DIVISOR GLEICH NULL (DIVISION DURCH 0), ENT-
                * HAELT DAS ERGEBNIS LAUTER EINSEN. WICHTIG IST NOCH,
                * DASS DAS ERGEBNIS NICHT GERUNDET, SONDERN DIE STELLE
                * HINTER DEM KOMMA ABGESCHNITTEN WIRD. DIE DIVISION
                * 29:10 HAT ALSO ALS ERGEBNIS EINE 2, OBWOHL DER
                * QUOTIENT NAHEZU 3 IST.
                *
                * PROGRAMM-SPEICHER:   62 ZELLEN
                * DATEN-SPEICHER:       7 ZELLEN
                * STACK-BELASTUNG:    2+0 ZELLEN
                *
                * VERSION 21-3-79                              KA
                *
0040                    ORG     $0040
                *
                * DATEN-SPEICHER-ZELLEN
                *
0040 0001       DIVIDH  RMB     1       HIGH-BYTE DES DIVIDEND
0041 0001       DIVIDL  RMB     1       LOW-BYTE DES DIVIDEND
0042 0001       DIVISH  RMB     1       HIGH-BYTE DES DIVISORS
0043 0001       DIVISL  RMB     1       LOW-BYTE DES DIVISORS
0044 0001       RESLTH  RMB     1       HIGH-BYTE DES 16-BIT-ERGEBNISSES
0045 0001       RESLTL  RMB     1       LOW-BYTE DES 16-BIT-ERGEBNISSES
0046 0001       COUNT   RMB     1       ROTIER- UND PROGRAMM-SCHLEIFENZAEHLER
```

DIVIDE PAGE : 3

```
                *
A500            ORG     $A500
                *
                * UNTERPROGRAMM ZUR DIVISION ZWEIER VORZEICHEN-
                * LOSER GANZER 16-BIT-BINAER-ZAHLEN
                *
A500 4F         DIVIDE CLR A            HIGH- UND LOW-BYTE
A501 97 44             STA A  RESLTH    FUER DAS 16-BIT-ERGEBNIS
A503 97 45             STA A  RESLTL    ZUNAECHST LOESCHEN
A505 4C                INC A            ROTIER-ZAEHLER AUF 1 VORSETZEN
A506 7D 0042           TST    DIVISH    IST MSB VOM DIVISOR GESETZT?
A509 2B 0D             BMI    GENUG     JA, KEIN ROTIEREN ERFORDERLICH
                *
A50B 4C         AGAIN  INC A            ROTIER-ZAEHLER UM 1 ERHOEHEN
A50C 78 0043           ASL    DIVISL    DIVISOR 1-MAL NACH LINKS ROTIREN
A50F 79 0042           ROL    DIVISH    IST MSB VOM DIVISOR GESETZT?
A512 2B 04             BMI    GENUG     JA,  DIVISOR GENUG ROTIERT
A514 81 11             CMP A  #17       NEIN, WURDE SCHON 16-MAL ROTIERT?
A516 26 F3             BNE    AGAIN     NEIN, DIVISOR NOCHMAL ROTIEREN
                *
A518 97 46      GENUG  STA A  COUNT     ROTIER-ZAEHLER KONSERVIEREN
A51A 96 40             LDA A  DIVIDH    HIGH- UND LOW-BYTE VOM
A51C D6 41             LDA B  DIVIDL    DIVIDEND LADEN
                *
A51E D0 43      SUBTRA SUB B  DIVISL    DIVIDEND MINUS ROTIERTER DIVISOR
A520 92 42             SBC A  DIVISH    IST DIVIDEND GROESSER ALS DIVISOR?
A522 24 07             BCC    SETBIT    JA, BIT IM ERGEBNIS SETZEN
A524 DB 43             ADD B  DIVISL    NEIN, KLEINER, DIVIDEND
A526 99 42             ADC A  DIVISH    WIEDER RESTAURIEREN
A528 0C                CLC              LOESCHT BIT IM ERGEBNIS
A529 20 01             BRA    ROTATE    CARRY INS ERGEBNIS ROTIEREN
                *
A52B 0D         SETBIT SEC              SETZT BIT IM ERGEBNIS
                *
A52C 79 0045    ROTATE ROL     RESLTL   CARRY WIRD LSB IM ERGEBIS
A52F 79 0044           ROL     RESLTH   HIGH- UND LOW-BYTE ROTIEREN
A532 74 0042           LSR     DIVISH   HIGH- UND LOW-BYTE DES DIVISORS
A535 76 0043           ROR     DIVISL   EINMAL NACH RECHTS ROTIEREN
A538 7A 0046           DEC     COUNT    SO OFT ZURUECK WIE HIN ROTIERT?
A53B 26 E1             BNE     SUBTRA   NEIN, DIVISOR VOM REST ABZIEHEN
A53D 39                RTS              JA, PROGRAMM BEENDET, RUECKSPRUNG
```

PAGE : 1

```
                    NAM     BCHXBC
                   *
                   * PROGRAMME ZUR UMWANDLUNG VON 16-BIT-
                   * HEXADEZIMAL-ZAHLEN IN DEZIMALE DARSTELLUNG
                   * UND DEZIMALER ZAHLEN (MAXIMAL 65535) IN
                   * 16-BIT-HEXADEZIMAL-DARSTELLUNG. BEIDE
                   * PROGRAMME GREIFEN AUF GEMEINSAME PROGRAMM-
                   * TEILE UND UNTERPROGRAMME ZU UND SIND DAHER
                   * IMMER GEMEINSAM ZU LADEN.
                   *
                   * PROGRAMM-SPEICHER: 338 ZELLEN
                   * DATEN-SPEICHER:      5 ZELLEN
                   *
                   * VERSION V03-3/79                  KA
                   *
0030                       ORG     $0030
                   *
0030 0001          BIN1    RMB     1          HOEHER-WERTIGES HEXADEZIMAL-BYTE
0031 0001          BIN0    RMB     1          NIEDER-WERTIGES HEXADEZIMAL-BYTE
0032 0001          DEZI2   RMB     1          SPEICHER FUER ZEHNTAUSENDER-STELLE
0033 0001          DEZI1   RMB     1          SPEICHER FUER TAUSENDER + HUNDERTER
0034 0001          DEZI0   RMB     1          SPEICHER FUER ZEHNER UND EINER-STELLE
                   *
                   * SPEICHER-ZELLEN UND UNTERPRROGRAMM-ADRESSEN
                   * DES EUROCOM-BETRIEBS-SYSTEMS
                   *
           0044    LA      EQU     $44        CODE FUER TASTE 'PFEIL NACH LINKS'
           A7B9    KEY     EQU     $A7B9      SPEICHER FUER TASTATUR-EINGABE
           A7BA    SAVEX   EQU     $A7BA      ZWISCHENSPEICHER FUER INDEX-REGISTER
           F8B3    INDISO  EQU     $F8B3      HOLT EIN HEX-ZEICHEN VON DER TASTATUR,
           F902    SHIF88  EQU     $F902      LOESCHT DIE 8-STELLIGE ANZEIGE
           F944    DIS     EQU     $F944      HOLT 4 HEX-ZEICHEN VON DER TASTATUR
           F97A    INBYT2  EQU     $F97A      SCHIEBT 4 ZEICHEN IN ANZEIGE-BUFFER
           F985    INBYTL  EQU     $F985      SCHIEBT HALB-BYTE IN ANZEIGE-BUFFER
           FA00    TIMEDP  EQU     $FA00      GIBT ANZEIGE 1 SEK. AUS, HOLT EINGABE
           FA1D    LTEXT1  EQU     $FA1D      SCHIEBT TEXT IN ANZEIGE-BUFFER
           FA75    CNTRLB  EQU     $FA75      BEARBEITET FUNKTIONS-TASTEN-EINGABE
```

```
                *
A500                    ORG     $A500
                *
                *
                * TEIL 1: UMWANDLUNG EINER 5-STELLIGEN DEZIMAL-ZAHL
                *         (MAXIMAL 65535) IN 16-BIT HEXADEZIMALE
                *         DARSTELLUNG
                *
                * PROGRAMMABLAUF: NACH STARTEN DES PROGRAMMES BEI DER
                * ADRESSE $A500 GIBT DER RECHNER DEN TEXT 'D= ' AUF DIE
                * ANZEIGE AUS UND ERWARTET DIE EINGABE DER UMZUWANDELNDEN
                * 5-STELLIGEN DEZIMAL-ZAHL, WOBEI FUEHRENDE NULLEN
                * MIT EINGEGEBEN WERDEN MUESSEN. BEI NICHT HEXADEZIMALER
                * EINGABE (FUNKTIONS-TASTEN) ERFOLGT EIN SPRUNG IN DAS
                * EUROCOM-BETRIEBS-SYSTEM. BEI HEXADEZIMALER ABER NICHT
                * DEZIMALER EINGABE (HEXA A BIS F) ERFOLGT EIN SPRUNG
                * ZUM PROGRAMM-ANFANG. NACH EINGABE DER FUENFTEN
                * DEZIMAL-STELLE WIRD UEBERPRUEFT, OB DIE MAXIMAL
                * ZULAESSIGE DEZIMAL-ZAHL 65535 UEBERSCHRITTEN WURDE.
                * IST DIES DER FALL ERFOLGT EIN SPRUNG ZUM PROGRAMM-
                * ANFANG. ANSONSTEN WIRD SOFORT DIE UMWANDLUNG VOR-
                * GENOMMEN UND MIT DEM TEXT 'B=  ' DAS 4-STELLIGE
                * HEXADEZIMALE ERGEBNIS AUSGEGEBEN. SOLL EINE WEITERE
                * ZAHL UMGEWANDELT WERDEN, FUEHRT DIE EINGABE DER
                * TASTE 'PFEIL NACH LINKS' (LEFT ARROW) WIEDER AN DEN
                * PROGRAMM-ANFANG. JEDE ANDERE EINGABE FUEHRT ZUM
                * RUECKSPRUNG IN DAS EUROCOM-BETRIEBS-SYSTEM.
                *
                * STACK-BELASTUNG: 16 ZELLEN
                *
A500 CE A64E START1 LDX     #DEZTXT   POINTER AUF TEXT-ANFANG 'D= '
A503 BD A5A8        JSR     TXTOUT    ANZEIGE LOESCHEN, TEXT AUSGEBEN
A506 BD FA00        JSR     TIMEDP    TEXT 1 SEKUNDE AUSGEBEN
A509 BD F8B3        JSR     INDISO    ZEHNTAUSENDER-STELLE VON TASTATUR
A50C C1 36          CMP B   #$36      IST EINGEGEBENE ZAHL GROESSER ALS 6?
A50E 22 F0          BHI     START1    JA, FALSCHE EINGABE, NEUER START
A510 C4 0F          AND B   #$0F      ZAHL STEHT IM UNTEREN HALB-
A512 D7 32          STA B   DEZI2     BYTE. IN DEZIMAL-ZELLE ABSPEICHERN
A514 BD F944        JSR     DIS       ZAHL IN DEN ANZEIGE-BUFFER BRINGEN
A517 DF 33          STX     DEZI1     NOCH 4 DEZIMAL-STELLEN VON TASTATUR
A519 CE 0030        LDX     #DEZI2-2  POINTER 2 ZELLEN VOR 1. DEZIMALZAHL
A51C C6 02          LDA B   #2        ZAEHLER FUER UEBERPRUEFUNGS-SCHLEIFE
```

```
                    *
A51E A6 03    TESTEN LDA A  3,X       DIE NAECHSTEN 2 DEZIMAL-STELLEN LADEN
A520 81 99           CMP A  #$99      IST DIE ZAHL GROESSER ALS 99?
A522 22 DC           BHI    START1    JA, FALSCHE EINAGEB, NEUER START
A524 84 0F           AND A  #$0F      IST DAS UNTERE HALB-BYTE DER
A526 81 09           CMP A  #$09      DEZIMAL-ZAHL GROESSER ALS 9?
A528 22 D6           BHI    START1    JA, FALSCHE EINGABE, NEUER START
A52A 08              INX              POINTER AUF NAECHSTE DEZIMAL-STELLEN
A52B 5A              DEC B            ALLE DEZIMAL-STELLEN UEBERPRUEFT?
A52C 26 F0           BNE    TESTEN    NEIN, DIE NAECHSTEN 2 STELLEN TESTEN
A52E A6 00           LDA A  0,X       ZEHNTAUSENDER-STELLE LADEN
A530 81 06           CMP A  #$06      IST SIE GROESSER ALS 6?
A532 26 0E           BNE    OKAY      NEIN, KLEINER, ALLES GUELTIG
A534 A6 01           LDA A  1,X       JA, SIND DIE MITTLEREN BEIDEN
A536 81 55           CMP A  #$55      DEZIMAL-STELLEN GROESSER ALS 55?
A538 22 C6           BHI    START1    JA, GROESSER, FALSCHE EINGABE
A53A 26 06           BNE    OKAY      NEIN, KLEINER, ALLES OKAY
A53C A6 02           LDA A  2,X       GLEICH, WEITER TESTEN, SIND DIE
A53E 81 35           CMP A  #$35      LETZTEN 2 STELLEN GROESSER ALS 35?
A540 22 BE           BHI    START1    JA, GROESSER, FALSCHE EINGABE
                    *
A542 8D 12    OKAY   BSR    BCDHEX    NEIN, KLEINER ODER GLEICH, UMWANDELN
A544 CE A64A         LDX    #BINTXT+1   POINTER AUF TEXT-ANFANG 'B=  '
A547 8D 6B           BSR    TXTBUF    ANZEIGE LOESCHEN, TEXT AUSGEBEN
A549 CE 0030         LDX    #BIN1     POINTER AUF ERSTE ERGEBNIS-ZELLE
A54C BD F97A         JSR    INBYT2    4 HEX-ZEICHEN IN AUSGABE-BUFFER
A54F 8D 68           BSR    DISPLY    HEXADEZIMAL-ERGEBNIS ANZEIGEN
A551 27 AD           BEQ    START1    'LEFT-ARROW' => NEUER PROGRAMM-LAUF
A553 7E FA75         JMP    CNTRLB    ANDERE EINGABE => SPRUNG INS SYSTEM
```

BCHXBC PAGE : 4

```
                *
                * BCDHEX: EIGENSTAENDIGES UNTERPROGRAMM ZUR UMWANDLUNG
                *         EINER 5-STELLIGEN DEZIMAL-ZAHL (MAXIMAL 65535)
                *         IN HEXADEZIMALE DARSTELLUNG
                *
                * PROGRAMM-ABLAUF: DAS UNTERPROGRAMM ERWARTET DIE
                * UMZUWANDELNDE ZAHL IN DEN ZELLEN 'DEZI2' (OBERES
                * HALB-BYTE = 0, UNTERES HALB-BYTE = ZEHNTAUSENDER-
                * STELLE), 'DEZI1' (OBERES HALB-BYTE = TAUSENDER-
                * STELLE, UNTERES HALB-BYTE = HUNDERTER-STELLE)
                * UND 'DEZIO' (OBERES HALB-BYTE = ZEHNER-STELLE,
                * UNTERES HALB-BYTE = EINER-STELLE). DAS HEXADEZIMALE
                * ERGEBNIS WIRD IN DEN ZELLEN 'BIN1' (HOEHER-WERTIGES
                * BYTE) UND 'BINO' (NIEDER-WERTIGES BYTE) ABGESPEICHERT.
                * DIE INHALTE DER ACCUMULATOREN A UND B UND DES
                * INDEX-REGISTERS WERDEN ZERSTOERT. IN DEM UNTER-
                * PROGRAMM WERDEN KEINE WEITEREN UNTERPROGRAMME
                * AUFGERUFEN.
                *
                * PROGRAMM-SPEICHER: 82 ZELLEN
                * DATEN-SPEICHER:     5 ZELLEN
                * STACK-BELASTUNG:  2+0 ZELLEN
                *
A556 CE 0032 BCDHEX LDX     #DEZI2    POINTER AUF ZEHNTAUSENDER-STELLE
A559 A6 02          LDA A   2,X       ZEHNER- UND EINER-STELLE
A55B 16             TAB               IN BEIDE ACCUMULATOREN LADEN
A55C C4 0F          AND B   #$0F      DIE EINER-STELLE BLEIBT UNVERAENDERT
A55E 44             LSR A             DIE ZEHNER-STELLE (OBERES
A55F 44             LSR A             HALB-BYTE) IN DIE UNTERE
A560 44             LSR A             HAELFTE VON ACCU A ROTIEREN
A561 44             LSR A             IST DIE ZEHNER-STELLE = 0?
A562 27 05          BEQ     HUNDER    JA, HUNDERTER-STELLE BEARBEITEN
                *
A564 CB 0A    NXTZEH ADD B  #10        NEIN, ZEHN ZUM ACCU B DAZU-ADDIEREN
A566 4A             DEC A             ZEHNER-STELLE FERTIG BEARBEITET?
A567 26 FB          BNE     NXTZEH    NEIN, NOCHMAL ZEHN ADDIEREN
                *
A569 97 30    HUNDER STA A  BIN1       OBERES ERGEBNIS-BYTE LOESCHEN
A56B A6 01          LDA A   1,X       HUNDERTER-STELLE RECHTS IN ACCU A
A56D 84 0F          AND A   #$0F      IST DIE HUNDERTER-STELLE = 0?
A56F 27 0A          BEQ     TAUSEN    JA, TAUSENDER-STELLE BEARBEITEN
                *
A571 CB 64    NXTHUN ADD B  #100       ACCU B + 100. ACCU B UEBERGELAUFEN?
A573 24 03          BCC     NOCARY    NEIN, KEIN UEBERTRAG AUF OBERES BYTE
A575 7C 0030        INC     BIN1      JA, UEBERLAUF, OBERES BYTE PLUS 1
                *
A578 4A       NOCARY DEC A             HUNDERTER-STELLE FERTIG ABGEARBEITET?
A579 26 F6          BNE     NXTHUN    NEIN, NOCHMAL HUNDERT ADDIEREN
                *
A57B A6 01    TAUSEN LDA A  1,X        DIE TAUSENDER-STELLE (OBERES
A57D 44             LSR A             HALB-BYTE) IN DIE UNTERE
A57E 44             LSR A             HAELFTE VON ACCUMU-
A57F 44             LSR A             LATOR A ROTIEREN
A580 44             LSR A             UND ALS ZAEHLER ZWISCHENSPEICHERN
A581 97 31          STA A   BINO       IST DIE TAUSENDER-STELLE = 0?
```

```
A583 27 0D          BEQ     ZEHTAU   JA, ZEHNTAUSENDER-STELLE BEARBEITEN
A585 96 30          LDA A   BIN1     OBERES ERGEBNIS-BYTE LADEN
                    *
A587 CB E8    NXTTAU ADD B  #$E8     1000 = HEXA 03E8 => ZUM UNTEREN BYTE
A589 89 03          ADC A   #$03     $E8, ZUM OBEREN BYTE $03 ADDIEREN
A58B 7A 0031        DEC     BIN0     TAUSENDER-STELLE FERTIG ABGEARBEITET?
A58E 26 F7          BNE     NXTTAU   NEIN, NOCHMAL TAUSEND ADDIEREN
A590 97 30          STA A   BIN1     JA, OBERES ERGEBNIS-BYTE KONSERVIEREN
                    *
A592 A6 00    ZEHTAU LDA A  0,X      IST DIE ZEHNTAUSENDER-STELLE = 0?
A594 27 0F          BEQ     FERTIG   JA, ERGEBNIS ABSPEICHERN
A596 97 31          STA A   BIN0     NEIN, ALS ZAEHLER ZWISCHENSPEICHERN
A598 96 30          LDA A   BIN1     OBERES ERGEBNIS-BYTE WIEDER LADEN
                    *
A59A CB 10    NXTZTA ADD B  #$10     10000 = HEXA 2710 => ZUM UNTEREN BYTE
A59C 89 27          ADC A   #$27     $10, ZUM OBEREN BYTE $27 ADDIEREN
A59E 7A 0031        DEC     BIN0     ZEHNTAUSENDER-STELLE ERLEDIGT?
A5A1 26 F7          BNE     NXTZTA   NEIN, NOCHMAL ZEHNTAUSEND ADDIEREN
A5A3 97 30          STA A   BIN1     JA, OBERES ERGEBNIS-BYTE ABSPEICHERN
                    *
A5A5 D7 31    FERTIG STA B  BIN0     UNTERES ERGEBNIS-BYTE ABSPEICHERN
A5A7 39             RTS              RUECKSPRUNG INS RUFENDE PROGRAMM
```



```
                    *
                    * UNTERPROGRAMM ZUM LOESCHEN DER ANZEIGE UND
                    * AUSGEBEN DES TEXTES (3 ZEICHEN), WOBEI DAS
                    * INDEX-REGISTER AUF DAS ERSTE ZEICHEN ZEIGEN
                    * MUSS. DIE EINGABE-FLAG WIRD GELOESCHT, WODURCH
                    * JEDE ZUVOR GETAETIGTE EINGABE VERLORENGEHT.
                    *
A5A8 7F A7B9 TXTOUT CLR     KEY      EINGABE-FLAG WIRD GELOESCHT
A5AB FF A7BA        STX     SAVEX    TEXT-ADRESSE KONSERVIEREN
A5AE BD F902        JSR     SHIF88   DIE ACHT-STELLIGE ANZEIGE LOESCHEN
A5B1 FE A7BA        LDX     SAVEX    TEXT-ANFANGS-ADRESSE WIEDER LADEN
                    *
                    * UNTERPROGRAMM ZUM SHIFTEN VON DREI
                    * ASCII-ZEICHEN IN DEN ANZEIGE-BUFFER,
                    * WOBEI DAS INDEX-REGISTER AUF DAS ERSTE
                    * ZEICHEN ZEIGEN MUSS.
                    *
A5B4 C6 04    TXTBUF LDA B  #4       ZEICHEN-ANZAHL IN ACCU B LADEN
A5B6 7E FA1D        JMP     LTEXT1   ZEICHEN IN DEN ANZEIGE-BUFFER BRINGEN
                    *
                    * UNTERPROGRAMM ZUR STAENDIGEN ANZEIGE DES
                    * UMWANDLUNGS-ERGEBNISSES, BIS AUF DER TASTATUR
                    * EINE WEITERE EINGABE ERFOLGT. DIE EINGABE
                    * WIRD MIT DER TASTE 'PFEIL NACH LINKS' (LEFT
                    * ARROW) VERGLICHEN UND IM RUFENDEN PROGRAMM
                    * ABGEFRAGT.
                    *
A5B9 BD FA00 DISPLY JSR     TIMEDP   1 SEKUNDE ANZEIGEN, EINGABE ERWARTEN
A5BC B6 A7B9        LDA A   KEY      WURDE EINE EINGABE GETAETIGT?
A5BF 27 F8          BEQ     DISPLY   NEIN, ERNEUT ANZEIGEN UND WARTEN
A5C1 81 44          CMP A   #LA      JA, DIE EINGABE 'PFEIL NACH LINKS'?
A5C3    27          NOP
```

```
                *
                * TEIL 2: UMWANDLUNG EINER 16-BIT-HEXADEZIMAL-
                *         ZAHL IN DEZIMALE DARSTELLUNG
                *
                * PROGRAMM-ABLAUF: NACH STARTEN DES PROGRAMMES
                * BEI DER ADRESSE $A5C2 GIBT DER RECHNER DEN
                * TEXT 'B=   ' AUF DIE ANZEIGE AUS UND ERWARTET
                * DIE EINGABE DER UMZUWANDELNDEN 4-STELLIGEN
                * HEXADEZIMALEN ZAHL AUF DER TASTATUR, WOBEI
                * FUEHRENDE NULLEN MIT EINGEGEBEN WERDEN
                * MUESSEN. BEI NICHT HEXA-DEZIMALER EINGABE
                * (FUNKTIONS-TASTEN) ERFOLGT EIN SPRUNG IN DAS
                * EUROCOM-BETRIEBS-SYSTEM. NACH EINGABE DES
                * VIERTEN HEX-ZEICHENS WIRD SOFORT DIE UMWAND-
                * LUNG VORGENOMMEN UND MIT DEM TEXT 'D= ' ALS
                * 5-STELLIGE DEZIMAL-ZAHL ZUR ANZEIGE GEBRACHT.
                * SOLL EINE WEITERE ZAHL UMGEWANDELT WERDEN,
                * FUEHRT DIE EINGABE DER TASTE 'PFEIL NACH
                * LINKS' (LEFT ARROW) WIEDER AN DEN PROGRAMM-
                * ANFANG. JEDE ANDERE EINGABE FUEHRT ZUM
                * RUECKSPRUNG IN DAS EUROCOM-BETRIEBS-SYSTEM.
                *
                * STACK-BELASTUNG: 16 ZELLEN
                *
A5C4 CE A649 START2 LDX     #BINTXT   POINTER AUF TEXT-ANFANG ' B= '
A5C7 8D DF          BSR     TXTOUT    ANZEIGE LOESCHEN, TEXT AUSGEBEN
A5C9 BD F944        JSR     DIS       4 HEX-ZEICHEN VON TASTATUR HOLEN
A5CC DF 30          STX     BIN1      ZEICHEN STEHEN IM INDEX-REGISTER
A5CE 8D 18          BSR     HEXBCD    4 HEX-ZEICHEN IN DEZIMAL UMWANDELN
A5D0 CE A64E        LDX     #DEZTXT   POINTER AUF TEXT-ANFANG 'D= '
A5D3 8D DF          BSR     TXTBUF    TEXT 'D ' IN DEN ANZEIGE-BUFFER
A5D5 CE 0032        LDX     #DEZI2    POINTER AUF ZEHNTAUSENDER-STELLE
A5D8 FF A7BA        STX     SAVEX     FUER AUSGABE-PROGRAMM KONSERVIEREN
A5DB BD F985        JSR     INBYTL    ZEHNTAUSENDER-STELLE STEHT IM UNTEREN
                *                     HALB-BYTE, IN ANZEIGE-BUFFER BRINGEN
A5DE BD F97A        JSR     INBYT2    UEBRIGE 4 DEZIMAL-STELLEN IN BUFFER
A5E1 8D D6          BSR     DISPLY    DEZIMAL-ERGEBNIS ZUR ANZEIGE BRINGEN
A5E3 27 DF          BEQ     START2    'LEFT-ARROW' => NEUER PROGRAMM-LAUF
A5E5 7E FA75        JMP     CNTRLB    ANDERE EINGABE => SPRUNG INS SYSTEM
```

```
                * HEXBCD: EIGENSTAENDIGES UNTERPROGRAMM ZUR UMWANDLUNG
                *         EINER 16-BIT-HEXADEZIMAL-ZAHL IN DEZIMALE
                *         DARSTELLUNG.
                *
                * PROGRAMM-ABLAUF: DAS UNTERPROGRAMM ERWARTET DIE
                * UMZUWANDELNDE ZAHL IN DEN ZELLEN 'BIN1' (HOEHER-
                * WERTIGES BYTE) UND 'BIN0' (NIEDER-WERTIGES BYTE).
                * DAS DEZIMALE ERGEBNIS WIRD IN DEN ZELLEN 'DEZI2'
                * (OBERES HALB-BYTE = 0, UNTERES HALB-BYTE =
                * ZEHNTAUSENDER-STELLE), 'DEZI1' (OBERES HALB-
                * BYTE = TAUSENDER-STELLE, UNTERES HALB-BYTE =
                * HUNDERTER-STELLE) UND 'DEZI0' (OBERES HALB-
                * BYTE = ZEHNER-STELLE, UNTERES HALB-BYTE = EINER-
                * STELLE) ABGESPEICHERT. DIE INHALTE DER ACCUMULATOREN
                * A UND B UND DES INDEX-REGISTERS WERDEN ZERSTOERT.
                * IN DEM UNTERPROGRAMM WERDEN 2 WEITERE UNTERPROGRAMM-
                * EBENEN AUFGERRUFEN.
                *
                * PROGRAMM-SPEICHER: 97 ZELLEN
                * DATEN-SPEICHER:     5 ZELLEN
                * STACK-BELASTUNG:  2+4 ZELLEN
                *
A5E8 4F         HEXBCD  CLR A           ZELLEN FUER DAS
A5E9 97 33              STA A  DEZI1    DEZIMALE ERGEBNIS
A5EB 97 32              STA A  DEZI2    ZUNAECHST LOESCHEN
A5ED D6 31              LDA B  BIN0     UNTERES HALB-BYTE DES
A5EF C4 0F              AND B  #$0F     NIEDERWERTIGEN BINAER-
A5F1 1B                 ABA             BYTES IN BCD-DARSTELLUNG
A5F2 19                 DAA             UMWANDELN
A5F3 97 34              STA A  DEZI0    ERGIBT UNTERSTE DEZIMAL-STELLE
A5F5 CE A622            LDX    #BIT07-3   DEZIMAL-LISTE DES LOW-BINAER-BYTES
A5F8 D6 31              LDA B  BIN0     OBERES HALB-BYTE DES
A5FA C4 F0              AND B  #$F0     NIEDERWERTIGEN BINAER-BYTES
A5FC 8D 05              BSR    BITSR    DEZIMAL-WERTE AUFSUCHEN
A5FE CE A62E            LDX    #BIT815-3   DEZIMAL-LISTE DES HIGH-BINAER-BYTE
A601 D6 30              LDA B  BIN1     OBERES BINAER-BYTE LADEN
                *
A603 08         BITSR   INX             POINTER AUF DEZIMAL-
A604 08                 INX             WERT DER NAECHSTEN
A605 08                 INX             BINAER-STELLE
A606 58                 ASL B           NAECHSTES BIT IM BINAER-WORT GESETZT?
A607 24 02              BCC    NEXTB    NEIN, RUECKSPRUNG PRUEFEN
A609 8D 04              BSR    ADDSR    JA, DEZIMAL-WERT ADDIEREN
                *
A60B 5D         NEXTB   TST B           SIND ALLE BITS BEARBEITET?
A60C 26 F5              BNE    BITSR    NEIN, NAECHSTEN WERT ADDIEREN
A60E 39                 RTS             RUECKSPRUNG

A60F 96 34      ADDSR   LDA A  DEZI0    ERSTEN DEZIMAL-WERT
A611 AB 00              ADD A  0,X      IN DER TABELLE ZUR
A613 19                 DAA             UNTERSTEN DEZIMAL-STELLE
A614 97 34              STA A  DEZI0    DAZU-ADDIEREN
A616 96 33              LDA A  DEZI1    ZWEITEN DEZIMAL-WERT
A618 A9 01              ADC A  1,X      IN DER TABELLE ZUR
A61A 19                 DAA             MITTLEREN DEZIMAL-STELLE
A61B 97 33              STA A  DEZI1    DAZU-ADDIEREN
A61D 96 32              LDA A  DEZI2    DRITTEN DEZIMAL-WERT
A61F A9 02              ADC A  2,X      IN DER TABELLE ZUR
A621 19                 DAA             OBERSTEN DEZIMAL-STELLE
A622 97 32              STA A  DEZI2    DAZU-ADDIEREN
A624 39                 RTS             RUECKSPRUNG
```

```
BCHXBC

                    *
                    * TABELLEN ZUR HEXADEZIMAL-BCD-UMWANDLUNG
                    *
A625 28     BIT07   FCB     $28,1,0,$64,,,$32,,,$16,,0
A626 01
A627 00
A628 64
A629 00
A62A 00
A62B 32
A62C 00
A62D 00
A62E 16
A62F 00
A630 00
A631 68     BIT815  FCB     $68,$27,3,$84,$63,1,$92,$81,0
A632 27
A633 03
A634 84
A635 63
A636 01
A637 92
A638 81
A639 00
A63A 96             FCB     $96,$40,,$48,$20,,$24,$10,,$12,5
A63B 40
A63C 00
A63D 48
A63E 20
A63F 00
A640 24
A641 10
A642 00
A643 12
A644 05
A645 00             FCB     ,$56,2,0
A646 56
A647 02
A648 00
                    *
                    * ANZEIGE-TEXTE
                    *
A649 20     BINTXT  FCC     ' B=  '
A64A 42
A64B 3D
A64C 20
A64D 20
A64E 20     DEZTXT  FCC     ' D= '
A64F 44
A650 3D
A651 20
```

S6802

MICROPROCESSOR WITH CLOCK AND RAM

Features

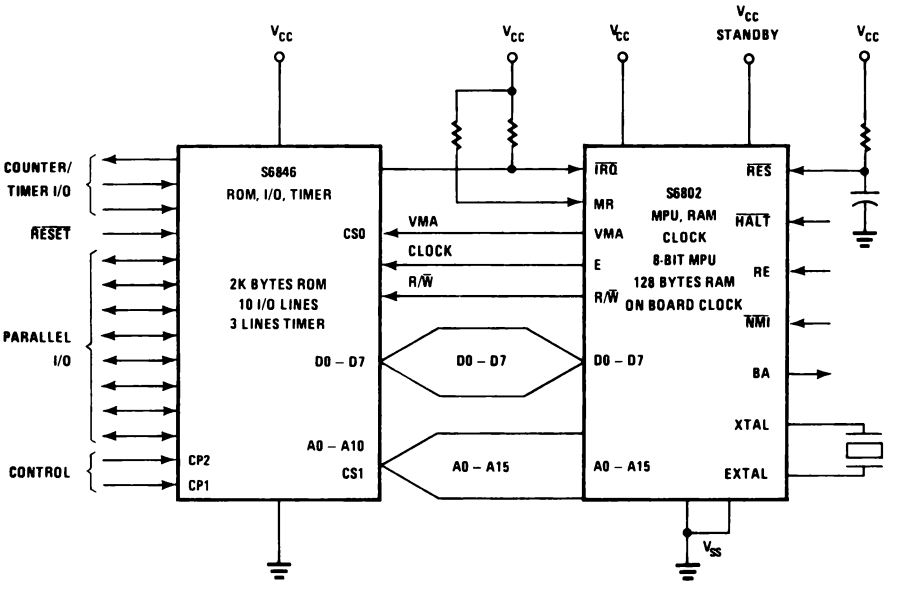
- On-Chip Clock Circuit
- 128 x 8 Bit On-Chip RAM
- 32 Bytes of RAM Are Retainable
- Software-Compatible with the S6800
- Expandable to 65K Words
- Standard TTL-Compatible Inputs and Outputs
- 8-Bit Word Size
- 16-Bit Memory Addressing
- Interrupt Capability

General Description

The S6802 is a monolithic 8-bit microprocessor that contains all the registers and accumulators of the present S6800 plus an internal clock oscillator and driver on the same chip. In addition, the S6802 has 128 bytes of RAM on board located at hex addresses 0000 to 007F. The first 32 bytes of RAM, at hex addresses 0000 to 001F, may be retained in a low power mode by utilizing V_{CC} standby, thus facilitating memory retention during a power-down situation.

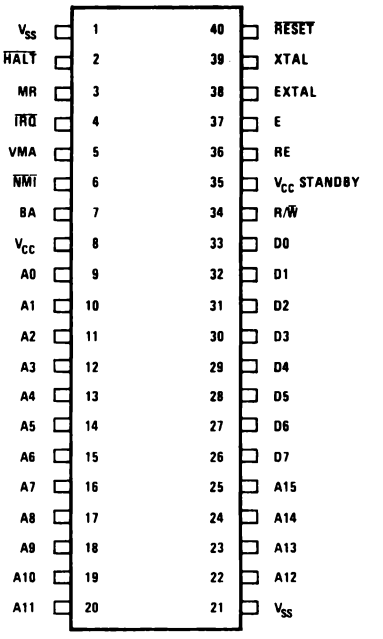
The S6802 is completely software compatible with the S6800 as well as the entire S6800 family of parts. Hence, the S6802 is expandable to 65K words. When the S6802 is interfaced with the S6846 ROM — I/O — Timer chip, as shown in the Block Diagram below, a basic 2-chip microcomputer system is realized.

Typical Microcomputer Block Diagram



BLOCK DIAGRAM OF A TYPICAL COST EFFECTIVE MICROCOMPUTER. THE MPU IS THE CENTER OF THE MICROCOMPUTER SYSTEM AND IS SHOWN IN A MINIMUM SYSTEM INTERFACING WITH A ROM COMBINATION CHIP. IT IS NOT INTENDED THAT THIS SYSTEM BE LIMITED TO THIS FUNCTION BUT THAT IT BE EXPANDABLE WITH OTHER PARTS IN THE S6800 MICROCOMPUTER FAMILY.

Pin Configuration



This is advance information and specifications are subject to change without notice

S6802

Absolute Maximum Ratings

Supply Voltage	-0.3V to + 7.0V
Input Voltage	-0.3V to + 7.0V
Operating Temperature Range	0°C to + 70°C
Storage Temperature Range	-55°C to + 150°C
Thermal Resistance	70°C/W

*COMMENT: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other condition above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

D.C. Electrical Characteristics (V_{CC} = 5.0V ± 5%, V_{SS} = 0, T_A = 0°C to +70°C unless otherwise noted.)

Symbol	Parameter		Min.	Typ.	Max.	Unit
V_{IH}	Input High Voltage	Logic, EXtal	V_{SS} +2.0	—	V_{CC}	V
		$\overline{Reset}$	V_{SS} +4.0	—	V_{CC}	
V_{IL}	Input Leakage Voltage	Logic, EXtal, $\overline{Reset}$	V_{SS} - 0.3	—	V_{SS} + 0.8	V
I_{IN}	Input Leakage Current (V_{IN} = 0 to 5.25V, V_{CC} = Max)	Logic*	—	1.0	2.5	μA
V_{OH}	Output High Voltage					V
	(I_{LOAD} = - 205μA, V_{CC} = Min)	D0 — D7	V_{SS} +2.4	—	—	V
	(I_{LOAD} = - 145μA, V_{CC} = Min)	A0—A15, R/$\overline{W}$, VMA,E	V_{SS} +2.4	—	—	V
	(I_{LOAD} = - 100μA, V_{CC} = Min)	BA	V_{SS} +2.4	—	—	V
V_{OL}	Output Low Voltage (I_{LOAD} = 1.6mA, V_{CC} = Min)		—	—	V_{SS} + 0.4	V
P_D**	Power Dissipation		—	0.600	1.2	W
C_{IN}	Capacitance #					pF
	(V_{IN} = 0, T_A=25°C, f=1.0MHz	D0 — D7	—	10	12.5	
		Logic Inputs, EXtal	—	6.5	10	
C_{OUT}		A0 — A15, R/$\overline{W}$, VMA	—	—	12	pF

Clock Timing (V_{CC} = 5.0V ± 5%, V_{SS} = 0, T_A = 0°C to +70°C unless otherwise noted)

Symbol	Parameter		Min.	Typ.	Max.	Unit
f	Frequency of Operation	Input Clock÷4	0.1	—	1.0	MHz
f_{Xtal}		Crystal Frequency	1.0	—	4.0	
t_{CYC}	Cycle Time		1.0	—	10	μs
$PW_{\phi Hs}$	Clock Pulse Width		450	—	4500	ns
$PW_{\phi L}$	Measured at 2.4V					
t_ϕ	Fall Time		—	—	25	ns
	Measured between V_{SS} + 0.4V and V_{SS} -2.4V					

*Except $\overline{IRQ}$ and $\overline{NMI}$, which require 3KΩ pullup load resistors for wire-OR capability at optimum operation. Does not include EXtal and Xtal, which are crystal inputs.

**In power-down mode, maximum power dissipation is less than 40mW.

#Capacitances are periodically sampled rather than 100% tested.

S6802

Read/Write Timing (Figures 1 through 5; Load Circuit of Figure 3).
(V_{CC} = 5.0V ± 5%, V_{SS} = 0, T_A = 0°C to + 70°C unless otherwise noted.)

Symbol	Parameters	Min.	Typ.	Max.	Unit
t_{AD}	Address Delay	—	—	270	ns
t_{ACC}	Peripheral Read Access Time $t_{ACC} = t_{ut} - t_{DSR}$)	—	—	530	ns
t_{DSR}	Data Setup Time (Read)	100	—	—	ns
t_H	Input Data Hold Time	10	—	—	ns
t_{AH}	Address Hold Time (Address, R/$\overline{W}$, VMA)	20	—	—	ns
t_{DDW}	Data Delay Time (Write)	—	165	225	ns
	Processor Controls:				
t_{PCS}	Processor Control Setup Time	200	—	—	ns
t_{PCr}, t_{PCf}	Processor Control Rise and Fall Time (Measured between 0.8V and 2.0V)	—	—	100	ns

Figure 1. Read Data From Memory or Peripherals

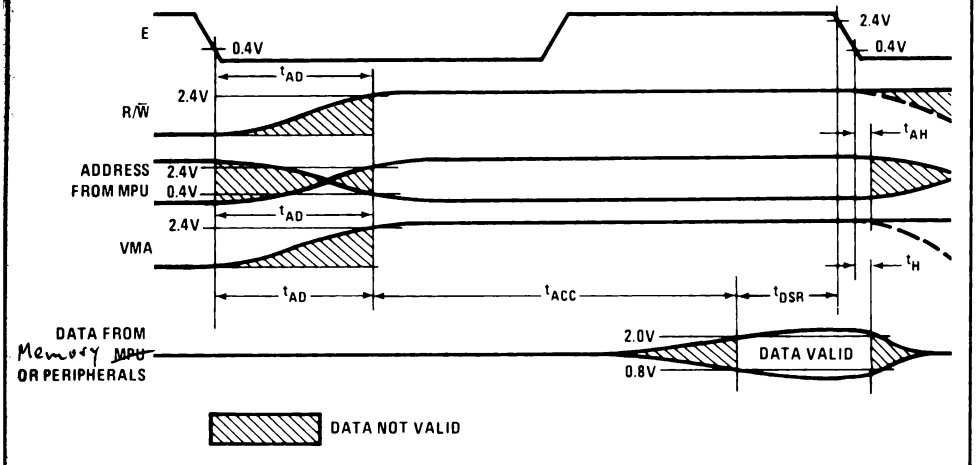


Figure 2. Write Data In Memory or Peripherals

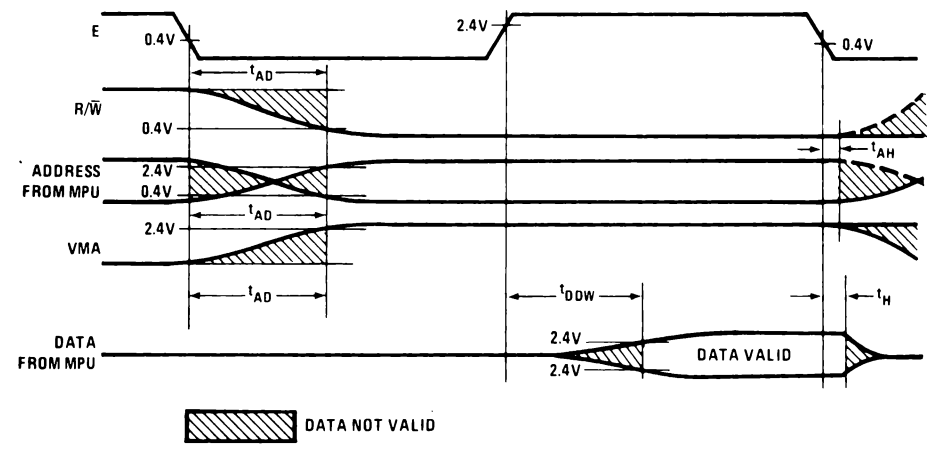
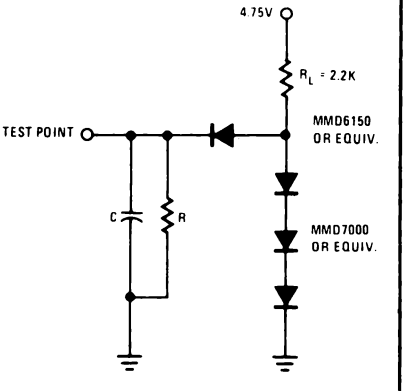


Figure 3. Bus Timing Test Load



C = 130pF FOR D0 – D7, E
= 90pF FOR A0 – A15, R/$\overline{W}$, AND VMA
= 30pF FOR BA
R = 11.7 KΩ FOR D0 – D7, E
= 16.5KΩ FOR A0 – A15, R/$\overline{W}$, AND VMA
= 24KΩ FOR BA

Figure 4. Typical Data Bus Output Delay Versus Capacitive Loading

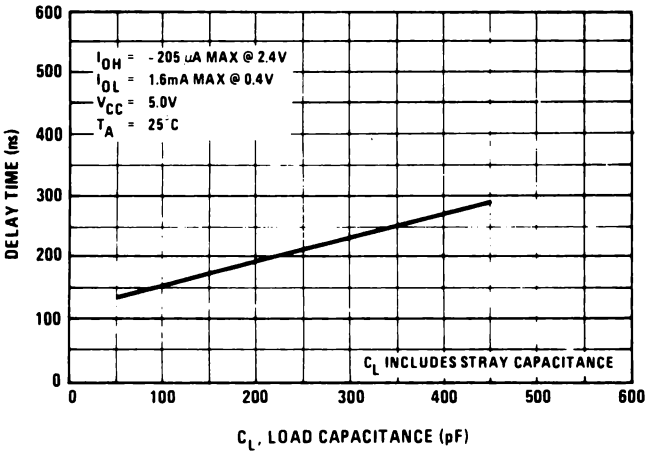


Figure 5. Typical Read/Write, VMA, and Address Output Delay Versus Capacitive Loading

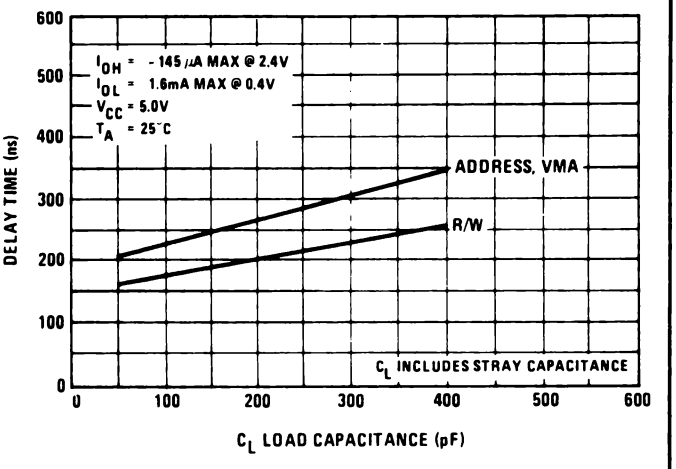
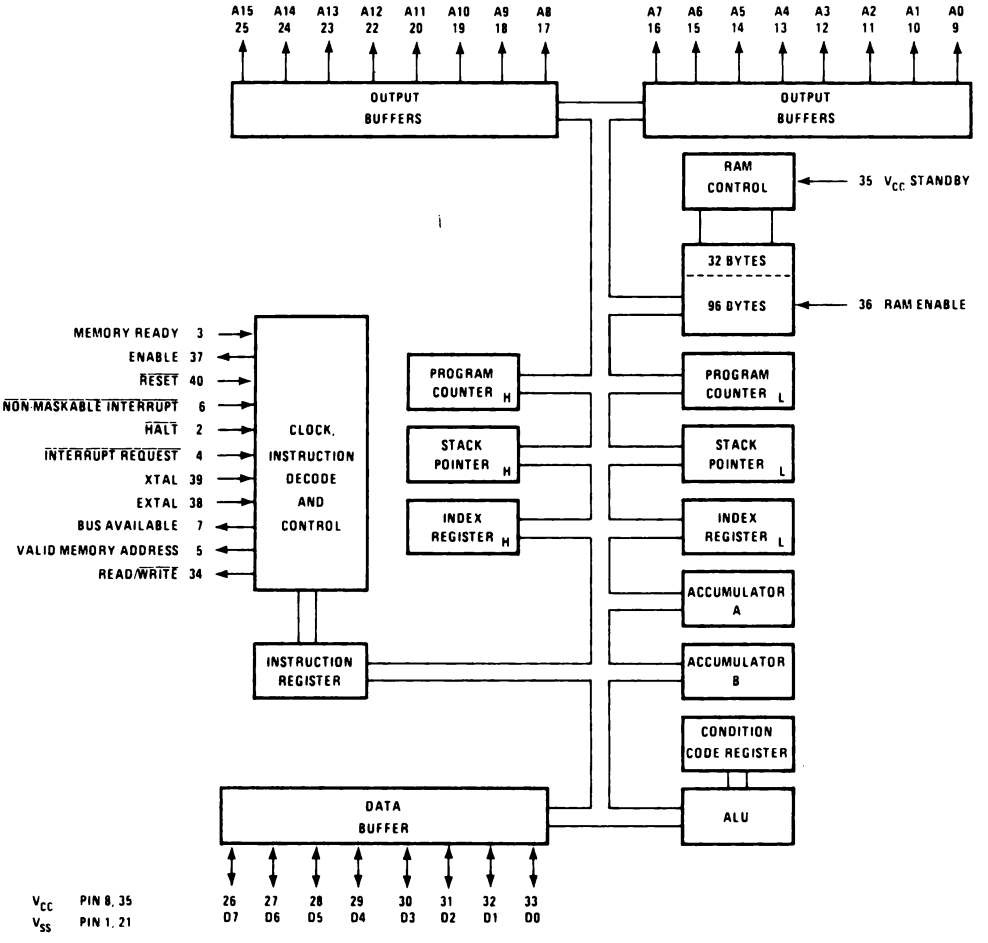


Figure 6. S6802 Expanded Block Diagram



Functional Description

MPU Registers

A general block diagram of the S6802 is shown in Figure 6. As shown, the number and configuration of the registers are the same as for the S6800. The 128 x 8 bit RAM has been added to the basic MPU. The first 32 bytes may be operated in a low power mode via a V_{CC} standby. These 32 bytes can be retained during power-up and power-down conditions via the RE signal.

The MPU has three 16-bit registers and three 8-bit registers available for use by the programmer (Figure 7).

Program Counter — The program counter is a two byte (16-bits) register that points to the current program address.

Stack Pointer — The stack pointer is a two byte register that contains the address of the next available location in an external push-down/pop-up stack. This stack is normally a random access Read/Write memory that may have any location (address) that is convenient. In those applications that require storage of information in the stack when power is lost, the stack must be non-volatile.

Index Register — The index register is a two byte register that is used to store data or a sixteen-bit memory address for the Indexed mode of memory addressing.

Accumulators — The MPU contains two 8-bit accumulators that are used to hold operands and results from an arithmetic logic unit (ALU).

Condition Code Register — The condition code register indicates the results of an Arithmetic Logic Unit operation: Negative (N), Zero (Z), Overflow (V), Carry from bit 7 (C), and Half Carry from bit 3 (H). These bits of the Condition Code Register are used as testable conditions for the conditional branch instructions. Bit 4 is the interrupt mask bit (I). The used bits of the Condition Code Register (b6 and b7) are ones.

Figure 8 shows the order of saving the microprocessor status within the stack.

Figure 7. Programming Model of the Microprocessing Unit

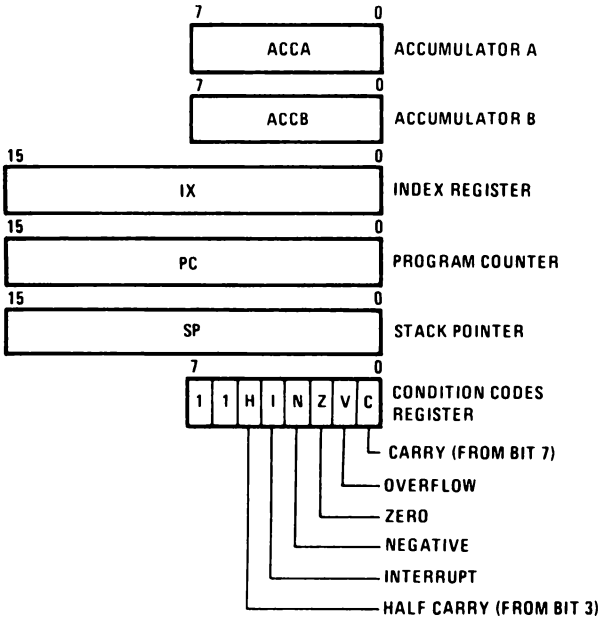
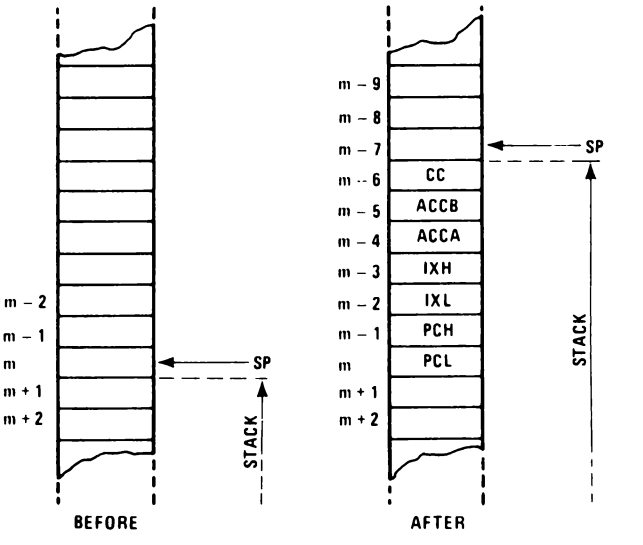


Figure 8. Saving the Status of the Microprocessor in the Stack



SP = STACK POINTER
CC = CONDITION CODES (ALSO CALLED THE PROCESSOR STATUS BYTE)
ACCB = ACCUMULATOR B
ACCA = ACCUMULATOR A
IXH = INDEX REGISTER, HIGHER ORDER 8 BITS
IXL = INDEX REGISTER, LOWER ORDER 8 BITS
PCH = PROGRAM COUNTER, HIGHER ORDER 8 BITS
PCL = PROGRAM COUNTER, LOWER ORDER 8 BITS

S6802 MPU Signal Description

Proper operation of the MPU requires that certain control and timing signals be provided to accomplish specific functions and that other signal lines be monitored to determine the state of the processor. These control and timing signals for the S6802 are identical to those of the S6800 except that TSC, DBE, $\phi 1$, $\phi 2$ input, and two unused pins have been eliminated, and the following signal and timing lines have been added:

RAM Enable (RE)

Crystal Connections EXtal and Xtal

Memory Ready (MR)

V_{CC} Standby

Enable $\phi 2$ Output (E)

The following is a summary of the S6802 MPU signals:

Address Bus (A0 — A15) — Sixteen pins are used for the address bus. The outputs are capable of driving one standard TTL load and 130pF.

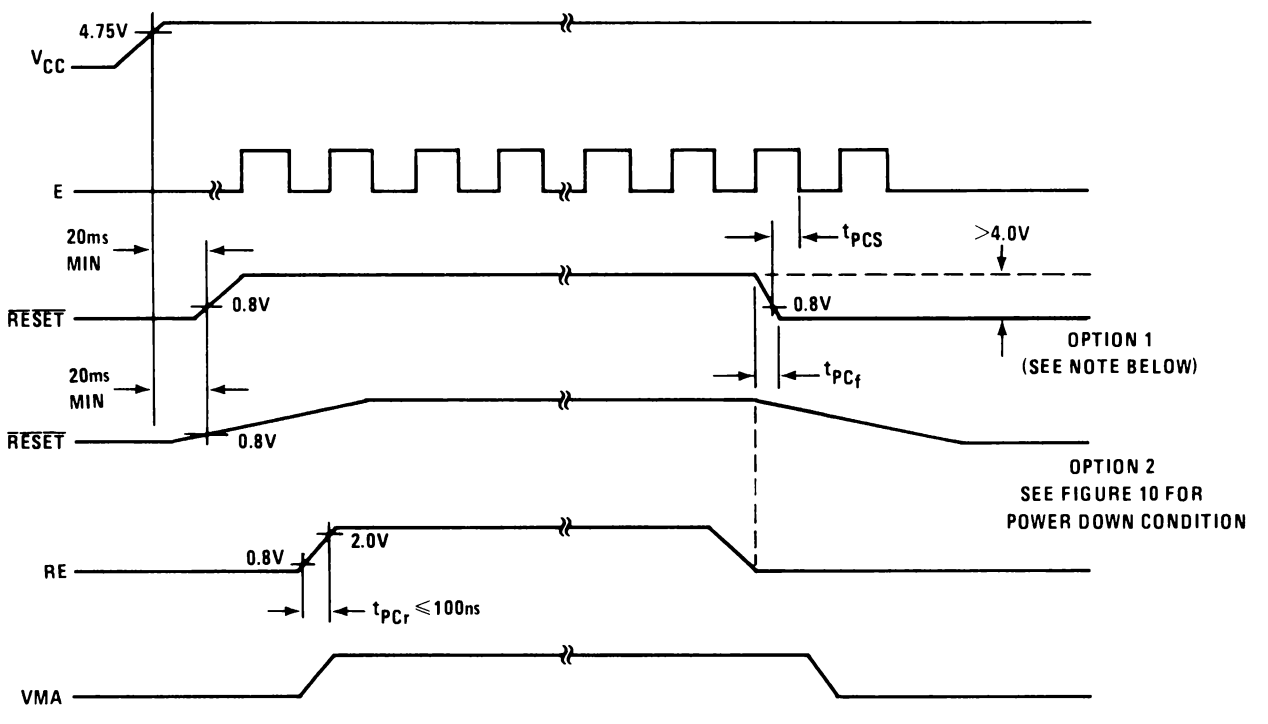
Data Bus (D0 — D7) — Eight pins are used for the data bus. It is bi-directional, transferring data to and from the memory and peripheral devices. It also has three-state-output buffers capable of driving one standard TTL load and 130pF.

$\overline{\text{Halt}}$ — When this input is in the low state, all activity in the machine will be halted. This input is level sensitive. In the halt mode, the machine will stop at the end of an instruction, Bus Available will be at a high state, Valid Memory Address will be at a low state, and all other three-state lines will be in the three-state mode. The address bus will display the address of the next instruction.

To insure single instruction operation, transition of the $\overline{\text{Halt}}$ line must not occur during the last 250ns of E and the $\overline{\text{Halt}}$ line must go high for one Clock cycle.

Read/$\overline{\text{Write}}$ (R/$\overline{\text{W}}$) — This TTL compatible output signals the peripherals and memory devices whether the MPU is in a Read (high) or Write (low) state. The normal standby state of this signal is Read (high).

Figure 9. Power-up and Reset Timing



NOTE: IF OPTION 1 IS CHOSEN, $\overline{\text{RESET}}$ AND RE PINS CAN BE TIED TOGETHER.

hen the processor is halted, it will be in the logical ne state. This output is capable of driving one stanard TTL load and 90pF.

alid Memory Address (VMA) — This output indicates peripheral devices that there is a valid address on he address bus. In normal operation, this signal hould be utilized for enabling peripheral interfaces uch as the PIA and ACIA. This signal is not threeate. One standard TTL load and 90pF may be irectly driven by this active high signal.

us Available (BA) — The Bus Available signal will ormally be in the low state; when activated, it will o to the high state indicating that the microprocessor as stopped and that the address bus is available. This ill occur if the $\overline{\text{Halt}}$ line is in the low state or the rocessor is in the WAIT state as a result of the exeution of a WAIT instruction. At such time, all threeate output drivers will go to their off state and ther outputs to their normally inactive level. The rocessor is removed from the WAIT state by the ccurrence of a maskable (mask bit I = 0) or nonaskable interrupt. This output is capable of driving ne standard TTL load and 30pF.

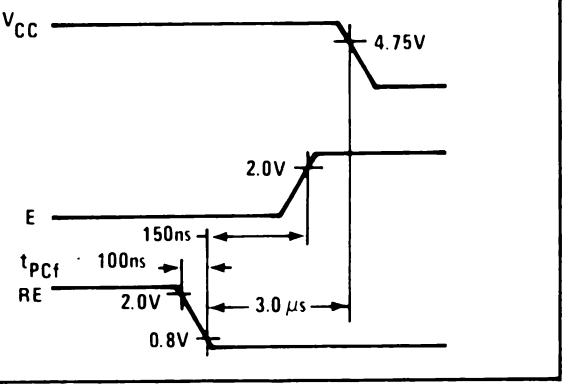
$\overline{\text{nterrupt Request (IRQ)}}$ — This level sensitive input equests that an interrupt sequence be generated ithin the machine. The processor will wait until it ompletes the current instruction that is being exeuted before it recognizes the request. At that time, if he interrupt mask bit in the Condition Code Register i not set, the machine will begin an interrupt sequence. he Index Register, Program Counter, Accumulators, nd Condition Code Register are stored away on the tack. Next the MPU will respond to the interrupt equest by setting the interrupt mask bit high so that o further interrupts may occur. At the end of the ycle, a 16-bit address will be loaded that points to vectoring address which is located in memory locaons FFF8 and FFF9. An address loaded at these ocations causes the MPU to branch to an interrupt outine in memory.

he $\overline{\text{Halt}}$ line must be in the high state for interrupts o be serviced. Interrupts will be latched internally hile $\overline{\text{Halt}}$ is low.

he $\overline{\text{IRQ}}$ has a high impedance pull-up device internal o the chip; however a 3kΩ external resistor to V_{CC} hould be used for wire-OR and optimum control of nterrrupts.

$\overline{\text{eset}}$ — This input is used to reset and start the MPU rom a power down condition, resulting from a power ilure or an initial start-up of the processor. When his line is low, the MPU is inactive and the information in the registers will be lost. If a high level is detected on the input, this will signal the MPU to begin the restart sequence. This will start execution of a routine to initialize the processor from its reset condition. All the higher order address lines will be forced high. For the restart, the last two (FFFE, FFFF) locations in memory will be used to load the program that is addressed by the program counter. During the restart routine, the interrupt mask bit is set and must be reset before the MPU can be interrupted by $\overline{\text{IRQ}}$. Power-up and reset timing and power-down sequences are shown in Figures 9 and 10, respectively.

Figure 10. Power-Down Sequence



Non-Maskable Interrupt (NMI) — A low-going edge on this input requests that a non-mask-interrupt sequence be generated within the processor. As with the $\overline{\text{Interrupt Request}}$ signal, the processor will complete the current instruction that is being executed before it recognizes the $\overline{\text{NMI}}$ signal. The interrupt mask bit in the Condition Code Register has no effect on $\overline{\text{NMI}}$.

The Index Register, Program Counter, Accumulators, and Condition Code Register are stored away on the stack. At the end of the cycle, a 16-bit address will be loaded that points to a vectoring address which is located in memory locations FFFC and FFFD. An address loaded at these locations caused the MPU to branch to a non-maskable interrupt routine in memory.

$\overline{\text{NMI}}$ has a high impedance pull-up resistor internal to the chip; however a 3kΩ external resistor to V_{CC} should be used for wire-OR and optimum control of interrupts.

Inputs $\overline{\text{IRQ}}$ and $\overline{\text{NMI}}$ are hardware interrupt lines that are sampled when E is high and will start the interrupt routine on a low E following the completion of an instruction.

Figure 11 is a flow chart describing the major decision paths and interrupt vectors of the microprocessor. Table 1 gives the memory map for interrupt vectors.

RAM Enable (RE) — A TTL-compatible RAM enable input controls the on-chip RAM of the S6802. When placed in the high state, the on-chip memory is enabled to respond to the MPU controls. In the low state, RAM is disabled. This pin may also be utilized to disable reading and writing the on-chip RAM during power-down situation. RAM enable must be low three μs before V_{CC} goes below 4.75V during power-down.

EXtal and Xtal — The S6802 has an internal oscillator that may be crystal controlled. These connections are for a series resonant fundamental crystal. (AT out.) A divide-by four circuit has been added to the S6802 so that a 4MHz crystal may be used in lieu of a 1MHz crystal for a more cost effective system. Pin 38 of the S6802 may be driven externally by a TTL input signal if a separate clock is required. Pin 39 is to be left open in this mode.

Memory Ready (MR) — MR is a TTL compatible input control signal which allows stretching of E. When MR is high, E will be in normal operation. When MR is low, it may be stretched integral multiples of half periods, thus allowing interface to slow memories. Memory Ready timing is shown in Figure 12.

Enable (E) — This pin supplies the clock for the MPU and the rest of the system. This is a single phase, TTL compatible clock. This clock may be conditioned by a Memory Ready Signal. This is equivalent to $\phi 2$ on the S6800.

V_{CC} Standby — This pin supplies the dc voltage to the first 32 bytes of RAM as well as the RAM Enable (RE) control logic. Thus retention of data in this portion of the RAM on a power-up, power-down, or standby condition is guaranteed. Maximum current drain at 5.25V is 8mA.

Table 1. Memory Map for Interrupt Vectors

VECTOR		DESCRIPTION
MS	LS	
FFFE	FFFF	RESTART
FFFC	FFFD	NON-MASKABLE INTERRUPT
FFFA	FFFB	SOFTWARE INTERRUPT
FFF8	FFF9	INTERRUPT REQUEST

Figure 11. MPU Flow Chart

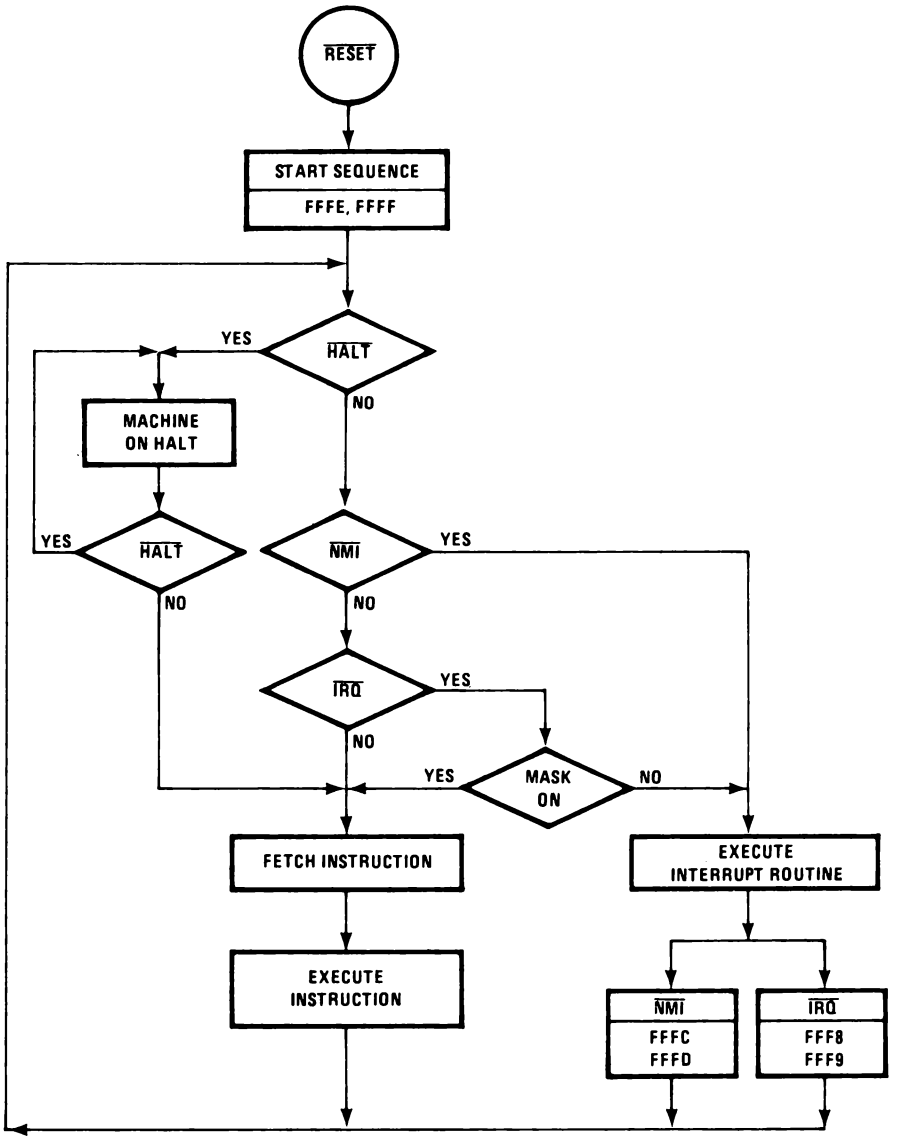
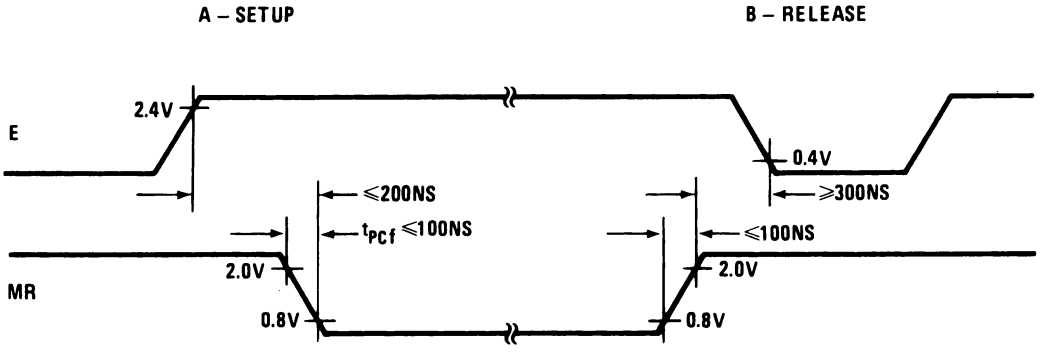


Figure 12. Memory Ready Control Function



MPU Instruction Set

The S6802 has a set of 72 different instructions. Included are binary and decimal arithmetic, logical, shift, rotate, load, store, conditional or unconditional branch, interrupt and stack manipulation instructions (Tables 2 thru 6). This instruction set is the same as that for the S6800.

MPU Addressing Modes

The S6802 eight-bit microprocessing unit has seven address modes that can be used by a programmer, with the addressing mode a function of both the type of instruction and the coding within the instruction. A summary of the addressing modes for a particular instruction can be found in Table 7 along with the associated instruction execution time that is given in machine cycles. With a clock frequency of 1MHz, these times would be microseconds.

Accumulator (ACCX) Addressing — In accumulator only addressing, either accumulator A or accumulator B is specified. These are one-byte instructions.

Immediate Addressing — In immediate addressing, the operand is contained in the second byte of the instruction except LDS and LDX which have the operand in the second and third bytes of the instruction. The MPU addresses this location when it fetches the immediate instruction for execution. These are two or three-byte instructions.

Direct Addressing — In direct addressing, the address of the operand is contained in the second byte of the instruction. Direct addressing allows the user to directly address the lowest 256 bytes in the machine i.e., locations zero through 255. Enhanced execution times are achieved by storing data in these locations. In most configurations, it should be a random access memory. These are two-byte instructions.

Extended Addressing — In extended addressing, the address contained in the second byte of the instruction is used as the higher eight-bits of the address of the operand. The third byte of the instruction is used as the lower eight-bits of the address for the operand. This is an absolute address in memory. These are three-byte instructions.

Indexed Addressing — In indexed addressing, the address contained in the second byte of the instruction is added to the index register's lowest eight bits in the MPU. The carry is then added to the higher order eight bits of the index register. This result is then used to address memory. The modified address is held in a temporary address register so there is no change to the index register. These are two-byte instructions.

Implied Addressing — In the implied addressing mode the instruction gives the address (i.e., stack pointer, index register, etc.). These are one-byte instructions.

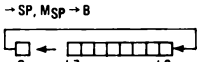
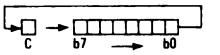
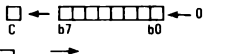
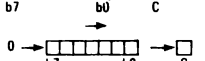
Relative Addressing — In relative addressing, the address contained in the second byte of the instruction is added to the program counter's lowest eight bits plus two. The carry or borrow is then added to the high eight bits. This allows the user to address data within a range of -125 to +129 bytes of the present instruction. These are two-byte instructions.

Table 2. Microprocessor Instruction Set — Alphabetic Sequence

ABA	ADD ACCUMULATORS	CLV	CLEAR OVERFLOW	PSH	PUSH DATA
ADC	ADD WITH CARRY	CMP	COMPARE	PUL	PULL DATA
ADD	ADD	COM	COMPLEMENT		
AND	LOGICAL AND	CPX	COMPARE INDEX REGISTER	ROL	ROTATE LEFT
ASL	ARITHMETIC SHIFT LEFT			ROR	ROTATE RIGHT
ASR	ARITHMETIC SHIFT RIGHT	DAA	DECIMAL ADJUST	RTI	RETURN FROM INTERRUPT
		DEC	DECREMENT	RTS	RETURN FROM SUBROUTINE
BCC	BRANCH IF CARRY CLEAR	DES	DECREMENT STACK POINTER		
BCS	BRANCH IF CARRY SET	DEX	DECREMENT INDEX REGISTER	SBA	SUBTRACT ACCUMULATORS
BEQ	BRANCH IF EQUAL TO ZERO			SBC	SUBTRACT WITH CARRY
BGE	BRANCH IF GREATER OR EQUAL ZERO	EOR	EXCLUSIVE OR	SEC	SET CARRY
BGT	BRANCH IF GREATER THAN ZERO			SEI	SET INTERRUPT MASK
BHI	BRANCH IF HIGHER	INC	INCREMENT	SEV	SET OVERFLOW
BIT	BIT TEST	INS	INCREMENT STACK POINTER	STA	STORE ACCUMULATOR
BLE	BRANCH IF LESS OR EQUAL	INX	INCREMENT INDEX REGISTER	STS	STORE STACK REGISTER
BLS	BRANCH IF LOWER OR SAME			STX	STORE INDEX REGISTER
BLT	BRANCH IF LESS THAN ZERO	JMP	JUMP	SUB	SUBTRACT
BMI	BRANCH IF MINUS	JSR	JUMP TO SUBROUTINE	SWI	SOFTWARE INTERRUPT
BNE	BRANCH IF NOT EQUAL TO ZERO				
BPL	BRANCH IF PLUS	LDA	LOAD ACCUMULATOR	TAB	TRANSFER ACCUMULATORS
BRA	BRANCH ALWAYS	LDS	LOAD STACK POINTER	TAP	TRANSFER ACCUMULATORS TO CONDITION CODE REG.
BSR	BRANCH TO SUBROUTINE	LDX	LOAD INDEX REGISTER	TBA	TRANSFER ACCUMULATORS
BVC	BRANCH IF OVERFLOW CLEAR	LSR	LOGICAL SHIFT RIGHT	TPA	TRANSFER CONDITION CODE REG. TO ACCUMULATOR
BVS	BRANCH IF OVERFLOW SET			TST	TEST
		NEG	NEGATE	TSX	TRANSFER STACK POINTER TO INDEX REGISTER
CBA	COMPARE ACCUMULATORS	NOP	NO OPERATION	TXS	TRANSFER INDEX REGISTER TO STACK POINTER
CLC	CLEAR CARRY				
CLI	CLEAR INTERRUPT MASK	ORA	INCLUSIVE OR ACCUMULATOR	WAI	WAIT FOR INTERRUPT
CLR	CLEAR				

S6802

Table 3. Accumulator and Memory Instructions

OPERATIONS	MNEMONIC	IMMED OP	IMMED ~	IMMED #	DIRECT OP	DIRECT ~	DIRECT #	INDEX OP	INDEX ~	INDEX #	EXTND OP	EXTND ~	EXTND #	IMPLIED OP	IMPLIED ~	IMPLIED #	BOOLEAN/ARITHMETIC OPERATION (All register labels refer to contents)	5 H	4 I	3 N	2 Z	1 V	0 C
ADD	ADDA	8B	2	2	9B	3	2	AB	5	2	BB	4	3				A + M → A	↕	•	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	5	2	FB	4	3				B + M → B	↕	•	↕	↕	↕	↕
ADD ACMLTRS	ABA													1B	2	1	A + B → A	↕	•	↕	↕	↕	↕
ADD WITH CARRY	ADCA	89	2	2	99	3	2	A9	5	2	B9	4	3				A + M + C → A	↕	•	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	5	2	F9	4	3				B + M + C → B	↕	•	↕	↕	↕	↕
AND	ANDA	84	2	2	94	3	2	A4	5	2	B4	4	3				A • M → A	•	•	↕	↕	R	•
	ANDB	C4	2	2	D4	3	2	E4	5	2	F4	4	3				B • M → B	•	•	↕	↕	R	•
BIT TEST	BITA	85	2	2	95	3	2	A5	5	2	B5	4	3				A • M	•	•	↕	↕	R	•
	BITB	C5	2	2	D5	3	2	E5	5	2	F5	4	3				B • M	•	•	↕	↕	R	•
CLEAR	CLR							6F	7	2	7F	6	3				00 → M	•	•	R	S	R	R
	CLRA													4F	2	1	00 → A	•	•	R	S	R	R
	CLRB													5F	2	1	00 → B	•	•	R	S	R	R
COMPARE	CMPA	81	2	2	91	3	2	A1	5	2	B1	4	3				A - M	•	•	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	5	2	F1	4	3				B - M	•	•	↕	↕	↕	↕
COMPARE ACMLTRS	CBA													11	2	1	A - B	•	•	↕	↕	↕	↕
COMPLEMENT, 1'S	COM							63	7	2	73	6	3				$\overline{M} \rightarrow M$	•	•	↕	↕	R	S
	COMA													43	2	1	$\overline{A} \rightarrow A$	•	•	↕	↕	R	S
	COMB													53	2	1	$\overline{B} \rightarrow B$	•	•	↕	↕	R	S
COMPLEMENT, 2'S (NEGATE)	NEG							60	7	2	70	6	3				00 - M → M	•	•	↕	↕	1	2
	NEGA													40	2	1	00 - A → A	•	•	↕	↕	1	2
	NEGB													50	2	1	00 - B → B	•	•	↕	↕	1	2
DECIMAL ADJUST, A	DAA													19	2	1	CONVERTS BINARY ADD. OF BCD CHARACTERS INTO BCD FORMAT	•	•	↕	↕	↕	3
DECREMENT	DEC							6A	7	2	7A	6	3				M - 1 → M	•	•	↕	↕	4	•
	DECA													4A	2	1	A - 1 → A	•	•	↕	↕	4	•
	DECB													5A	2	1	B - 1 → B	•	•	↕	↕	4	•
EXCLUSIVE OR	EORA	88	2	2	98	3	2	A8	5	2	B8	4	3				A ⊕ M → A	•	•	↕	↕	R	•
	EORB	C8	2	2	D8	3	2	E8	5	2	F8	4	3				B ⊕ M → B	•	•	↕	↕	R	•
INCREMENT	INC							6C	7	2	7C	6	3				M + 1 → M	•	•	↕	↕	5	•
	INCA													4C	2	1	A + 1 → A	•	•	↕	↕	5	•
	INCB													5C	2	1	B + 1 → B	•	•	↕	↕	5	•
LOAD ACMLTR	LDAA	86	2	2	96	3	2	A6	5	2	B6	4	3				M → A	•	•	↕	↕	R	•
	LDAB	C6	2	2	D6	3	2	E6	5	2	F6	4	3				M → B	•	•	↕	↕	R	•
OR, INCLUSIVE	ORAA	8A	2	2	9A	3	2	AA	5	2	BA	4	3				A + M → A	•	•	↕	↕	R	•
	ORAB	CA	2	2	DA	3	2	EA	5	2	FA	4	3				B + M → B	•	•	↕	↕	R	•
PUSH DATA	PSHA													36	4	1	$A \rightarrow M_{SP}$, SP - 1 → SP	•	•	•	•	•	•
	PSHB													37	4	1	$B \rightarrow M_{SP}$, SP - 1 → SP	•	•	•	•	•	•
PULL DATA	PULA													32	4	1	SP + 1 → SP, $M_{SP} \rightarrow A$	•	•	•	•	•	•
	PULB													33	4	1	SP + 1 → SP, $M_{SP} \rightarrow B$	•	•	•	•	•	•
ROTATE LEFT	ROL							69	7	2	79	6	3				M	•	•	↕	↕	6	↕
	ROLA													49	2	1	A 	•	•	↕	↕	6	↕
	ROLB													59	2	1	B (C b7 b0)	•	•	↕	↕	6	↕
ROTATE RIGHT	ROR							66	7	2	76	6	3				M	•	•	↕	↕	6	↕
	RORA													46	2	1	A 	•	•	↕	↕	6	↕
	RORB													56	2	1	B (C b7 b0)	•	•	↕	↕	6	↕
SHIFT LEFT, ARITHMETIC	ASL							68	7	2	78	6	3				M	•	•	↕	↕	6	↕
	ASLA													48	2	1	A 	•	•	↕	↕	6	↕
	ASLB													58	2	1	B (C b7 b0)	•	•	↕	↕	6	↕
SHIFT RIGHT, ARITHMETIC	ASR							67	7	2	77	6	3				M	•	•	↕	↕	6	↕
	ASRA													47	2	1	A 	•	•	↕	↕	6	↕
	ASRB													57	2	1	B (b7 b0 C)	•	•	↕	↕	6	↕
SHIFT RIGHT, LOGIC	LSR							64	7	2	74	6	3				M	•	•	R	↕	6	↕
	LSRA													44	2	1	A	•	•	R	↕	6	↕
	LSRB													54	2	1	B (0 b7 b0 C)	•	•	R	↕	6	↕
STORE ACMLTR.	STAA				97	4	2	A7	6	2	B7	5	3				A → M	•	•	↕	↕	R	•
	STAB				D7	4	2	E7	6	2	F7	5	3				B → M	•	•	↕	↕	R	•
SUBTRACT	SUBA	80	2	2	90	3	2	A0	5	2	B0	4	3				A - M → A	•	•	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	5	2	F0	4	3				B - M → B	•	•	↕	↕	↕	↕
SUBTRACT ACMLTRS.	SBA													10	2	1	A - B → A	•	•	↕	↕	↕	↕
SUBTR. WITH CARRY	SBCA	82	2	2	92	3	2	A2	5	2	B2	4	3				A - M - C → A	•	•	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	5	2	F2	4	3				B - M - C → B	•	•	↕	↕	↕	↕
TRANSFER ACMLTRS.	TAB													16	2	1	A → B	•	•	↕	↕	R	•
	TBA																B → A	•	•	↕	↕	R	•
TEST, ZERO OR MINUS	TST							6D	7	2	7D	6	3				M - 00	•	•	↕	↕	R	R
	TSTA													4D	2	1	A - 00	•	•	↕	↕	R	R
	TSTB													5D	2	1	B - 00	•	•	↕	↕	R	R
																		H	I	N	Z	V	C

LEGEND:

- **OP** Operation Code (Hexadecimal);
- ~ Number of MPU Cycles;
- # Number of Program Bytes;
- + Arithmetic Plus;
- - Arithmetic Minus;
- • Boolean AND;
- M_{SP} Contents of memory location pointed to be Stack Pointer;
- + Boolean Inclusive OR;
- ⊕ Boolean Exclusive OR;
- $\overline{M}$ Complement of M;
- → Transfer Into;
- 0 Bit = Zero
- 00 Byte = Zero

NOTE: Accumulator addressing mode instructions are included in the column for IMPLIED addressing

CONDITION CODE SYMBOLS:

- H Half-carry from bit 3;
- I Interrupt Mask
- N Negative (sign bit)
- Z Zero (byte)
- V Overflow, 2's complement
- C Carry from bit 7
- R Reset Always
- S Set Always
- ↕ Test and set if true, cleared otherwise
- • Not Affected

S6802

Table 4. Index Register and Stack Manipulation Instructions

POINTER OPERATIONS	MNEMONIC	IMMED OP	IMMED ~	IMMED #	DIRECT OP	DIRECT ~	DIRECT #	INDEX OP	INDEX ~	INDEX #	EXTND OP	EXTND ~	EXTND #	IMPLIED OP	IMPLIED ~	IMPLIED #	BOOLEAN/ARITHMETIC OPERATION	COND. CODE REG. 5 H	4 I	3 N	2 Z	1 V	0 C
COMPARE INDEX REG	CPX	8C	3	3	9C	4	2	AC	6	2	BC	5	3				$X_H - M, X_L - (M+1)$	•	•	7	↕	8	•
DECREMENT INDEX REG	DEX													09	4	1	$X - 1 \rightarrow X$	•	•	•	↕	•	•
DECREMENT STACK PNTR	DES													34	4	1	$SP - 1 \rightarrow SP$	•	•	•	•	•	•
INCREMENT INDEX REG	INX													08	4	1	$X + 1 \rightarrow X$	•	•	•	↕	•	•
INCREMENT STACK PNTR	INS													31	4	1	$SP + 1 \rightarrow SP$	•	•	•	•	•	•
LOAD INDEX REG	LDX	CE	3	3	DE	4	2	EE	6	2	FE	5	3				$M \rightarrow X_H, (M+1) \rightarrow X_L$	•	•	9	↕	R	•
LOAD STACK PNTR	LDS	8E	3	3	9E	4	2	AE	6	2	BE	5	3				$M \rightarrow SP_H, (M+1) \rightarrow SP_L$	•	•	9	↕	R	•
STORE INDEX REG	STX				DF	5	2	EF	7	2	FF	6	3				$X_H \rightarrow M, X_L \rightarrow (M+1)$	•	•	9	↕	R	•
STORE STACK PNTR	STS				9F	5	2	AF	7	2	BF	6	3				$SP_H \rightarrow M, SP_L \rightarrow (M+1)$	•	•	9	↕	R	•
INDEX REG → STACK PNTR	TXS													35	4	1	$X - 1 \rightarrow SP$	•	•	•	•	•	•
STACK PNTR → INDEX REG	TSX													30	4	1	$SP + 1 \rightarrow X$	•	•	•	•	•	•

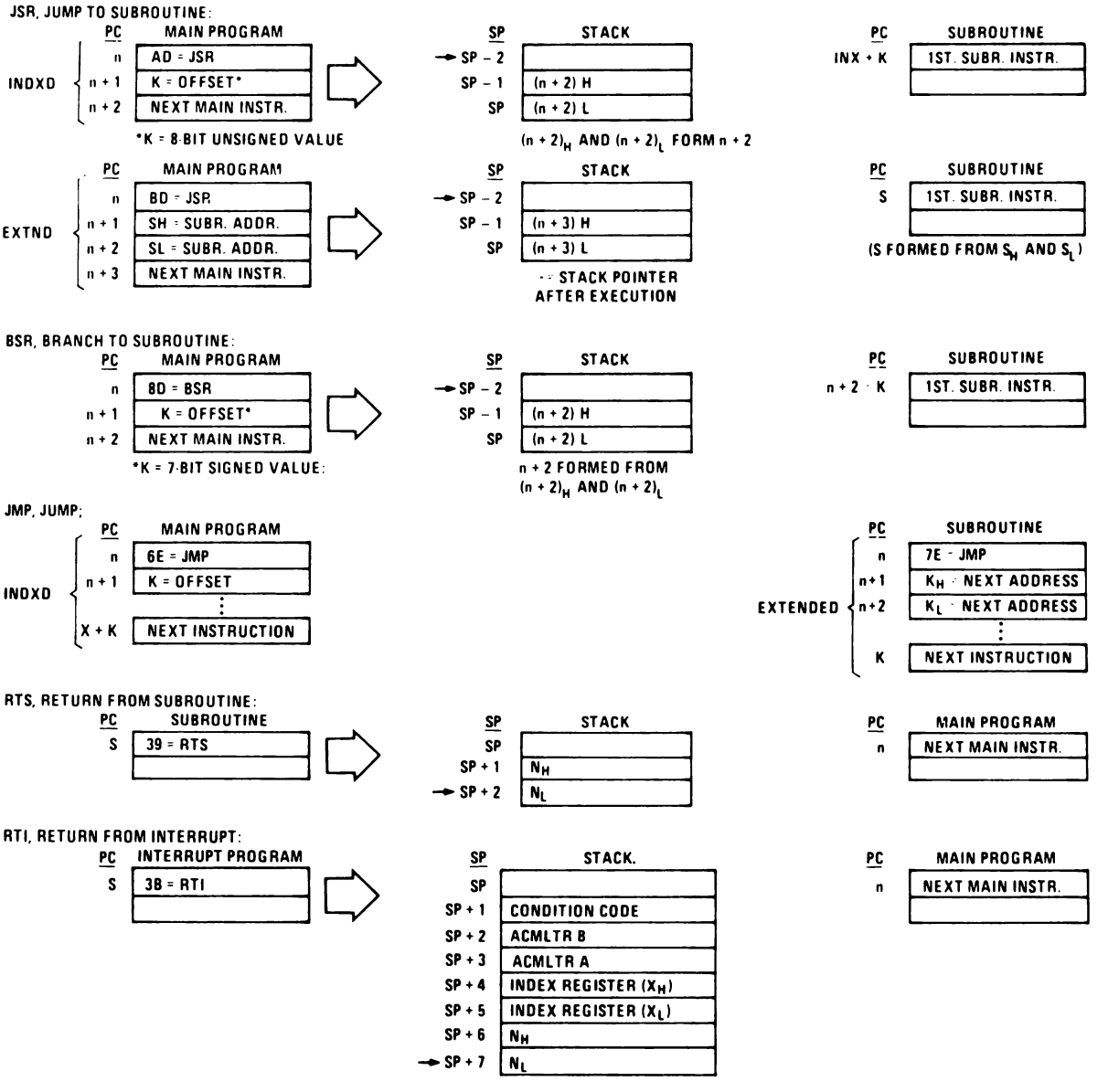
Table 5. Jump and Branch Instructions

OPERATIONS	MNEMONIC	RELATIVE OP	RELATIVE ~	RELATIVE #	INDEX OP	INDEX ~	INDEX #	EXTND OP	EXTND ~	EXTND #	IMPLIED OP	IMPLIED ~	IMPLIED #	BRANCH TEST	COND. CODE REG. 5 H	4 I	3 N	2 Z	1 V	0 C
BRANCH ALWAYS	BRA	20	4	2										NONE	•	•	•	•	•	•
BRANCH IF CARRY CLEAR	BCC	24	4	2										C = 0	•	•	•	•	•	•
BRANCH IF CARRY SET	BCS	25	4	2										C = 1	•	•	•	•	•	•
BRANCH IF = ZERO	BEQ	27	4	2										Z = 1	•	•	•	•	•	•
BRANCH IF ≥ ZERO	BGE	2C	4	2										N ⊕ V = 0	•	•	•	•	•	•
BRANCH IF > ZERO	BGT	2E	4	2										Z + (N ⊕ V) = 0	•	•	•	•	•	•
BRANCH IF HIGHER	BHI	22	4	2										C + Z = 0	•	•	•	•	•	•
BRANCH IF ≤ ZERO	BLE	2F	4	2										Z + (N ⊕ V = 1	•	•	•	•	•	•
BRANCH IF LOWER OR SAME	BLS	23	4	2										C + Z = 1	•	•	•	•	•	•
BRANCH IF < ZERO	BLT	2D	4	2										N ⊕ V = 1	•	•	•	•	•	•
BRANCH IF MINUS	BMI	2B	4	2										N = 1	•	•	•	•	•	•
BRANCH IF NOT EQUAL ZERO	BNE	26	4	2										Z = 0	•	•	•	•	•	•
BRANCH IF OVERFLOW CLEAR	BVC	28	4	2										V = 0	•	•	•	•	•	•
BRANCH IF OVERFLOW SET	BVS	29	4	2										V = 1	•	•	•	•	•	•
BRANCH IF PLUS	BPL	2A	4	2										N = 0	•	•	•	•	•	•
BRANCH TO SUBROUTINE	BSR	8D	8	2											•	•	•	•	•	•
JUMP	JMP				6E	4	2	7E	3	3				SEE SPECIAL OPERATIONS	•	•	•	•	•	•
JUMP TO SUBROUTINE	JSR				AD	8	2	BD	9	3					•	•	•	•	•	•
NO OPERATION	NOP										01	2	1	ADVANCES PROG. CNTR. ONLY	•	•	•	(10)	•	•
RETURN FROM INTERRUPT	RTI										3B	10	1							
RETURN FROM SUBROUTINE	RTS										39	5	1		•	•	•	•	•	•
SOFTWARE INTERRUPT	SWI										3F	12	1	SEE SPECIAL OPERATIONS	•	•	•	•	•	•
WAIT FOR INTERRUPT*	WAI										3E	9	1		•	(11)	•	•	•	•

*WAI puts Address Bus, R/W, and Data Bus in the three-state mode while VMA is held low.

S6802

Special Operations



S6802

Table 6. Condition Code Register Manipulation Instructions

OPERATIONS	MNEMONIC	IMPLIED OP	~	#	BOOLEAN OPERATION	COND. CODE REG. 5 H	4 I	3 N	2 Z	1 V	0 C
CLEAR CARRY	CLC	0C	2	1	0 → C	•	•	•	•	•	R
CLEAR INTERRUPT MASK	CLI	0E	2	1	0 → I	•	R	•	•	•	•
CLEAR OVERFLOW	CLV	0A	2	1	0 → V	•	•	•	•	R	•
SET CARRY	SEC	0D	2	1	1 → C	•	•	•	•	•	S
SET INTERRUPT MASK	SEI	0F	2	1	1 → I	•	S	•	•	•	•
SET OVERFLOW	SEV	0B	2	1	1 → V	•	•	•	•	S	•
ACMLTR A → CCR	TAP	06	2	1	A → CCR	(12)					
CCR → ACMLTR A	TPA	07	2	1	CCR → A	•	•	•	•	•	•

CONDITION CODE REGISTER NOTES: (Bit set if test is true and cleared otherwise)

1 (Bit V) Test: Result = 10000000?
2 (Bit C) Test: Result = 00000000?
3 (Bit C) Test: Decimal value of most significant BCD Character greater than nine? (Not cleared if previously set).
4 (Bit V) Test: Operand = 10000000 prior to execution?
5 (Bit V) Test: Operand = 01111111 prior to execution?
6 (Bit V) Test: Set equal to result of N ⊕ C after shift has occurred.
7 (Bit N) Test: Sign bit of most significant (MS) byte = 1?
8 (Bit V) Test: 2's complement overflow from subtraction of MS bytes?
9 (Bit N) Test: Result less than zero? (Bit 15 = 1)
10 (All) Load Condition Code Register from Stack. (See Special Operations)
11 (Bit I) Set when interrupt occurs. If previously set, a Non Maskable Interrupt is required to exit the wait state.
12 (All) Set according to the contents of Accumulator A.

Table 7. Instruction Addressing Modes and Associated Execution Times (Times in Machine Cycles)

	(DUAL OPERAND)	ACCX	IMMEDIATE	DIRECT	EXTENDED	INDEXED	IMPLIED	RELATIVE
ABA		•	•	•	•	•	2	•
ADC	x	•	2	3	4	5	•	•
ADD	x	•	2	3	4	5	•	•
AND	x	•	2	3	4	5	•	•
ASL		2	•	•	6	7	•	•
ASR		2	•	•	6	7	•	•
BCC		•	•	•	•	•	•	4
BCS		•	•	•	•	•	•	4
BEA		•	•	•	•	•	•	4
BGE		•	•	•	•	•	•	4
BGT		•	•	•	•	•	•	4
BHI		•	•	•	•	•	•	4
BIT	x	•	2	3	4	5	•	•
BLE		•	•	•	•	•	•	4
BLS		•	•	•	•	•	•	4
BLT		•	•	•	•	•	•	4
BMI		•	•	•	•	•	•	4
BNE		•	•	•	•	•	•	4
BPL		•	•	•	•	•	•	4
BRA		•	•	•	•	•	•	4
BSR		•	•	•	•	•	•	8
BVC		•	•	•	•	•	•	4
BVS		•	•	•	•	•	•	4
CBA		•	•	•	•	•	2	•
CLC		•	•	•	•	•	2	•
CLI		•	•	•	•	•	2	•
CLR		2	•	•	6	7	•	•
CLV		•	•	•	•	•	2	•
CMP	x	•	2	3	4	5	•	•
COM		2	•	•	6	7	•	•
CPX		•	3	4	5	6	•	•
DAA		•	•	•	•	•	2	•
DEC		2	•	•	6	7	•	•
DES		•	•	•	•	•	4	•
DEX		•	•	•	•	•	4	•
EOR	x	•	2	3	4	5	•	•

	(DUAL OPERAND)	ACCX	IMMEDIATE	DIRECT	EXTENDED	INDEXED	IMPLIED
INC		2	•	•	6	7	•
INS		•	•	•	•	•	4
INX		•	•	•	•	•	4
JMP		•	•	•	3	4	•
JSR		•	•	•	9	8	•
LDA	x	•	2	3	4	5	•
LDS		•	3	4	5	6	•
LDX		•	3	4	5	6	•
LSR		2	•	•	6	7	•
NEG		2	•	•	6	7	•
NOP		•	•	•	•	•	2
ORA	x	•	2	3	4	5	•
PSH		•	•	•	•	•	4
PUL		•	•	•	•	•	4
ROL		2	•	•	6	7	•
ROR		2	•	•	6	7	•
RTI		•	•	•	•	•	10
RTS		•	•	•	•	•	5
SBA		•	•	•	•	•	2
SBC	x	•	2	3	4	5	•
SEC		•	•	•	•	•	2
SEI		•	•	•	•	•	2
SEV		•	•	•	•	•	2
STA	x	•	•	4	5	6	•
STS		•	•	5	6	7	•
STX		•	•	5	6	7	•
SUB	x	•	2	3	4	5	•
SWI		•	•	•	•	•	12
TAB		•	•	•	•	•	2
TAP		•	•	•	•	•	2
TBA		•	•	•	•	•	2
TPA		•	•	•	•	•	2
TST		2	•	•	6	7	•
TSX		•	•	•	•	•	4
TSX		•	•	•	•	•	4
WAI		•	•	•	•	•	9

NOTE: Interrupt time is 12 cycles from the end of the instruction being executed, except following a WAI instruction. Then it is 4 cycles.

Summary of Cycle by Cycle Operation

Table 8 provides a detailed description of the information present on the Address Bus, Data Bus, Valid Memory Address line (VMA), and the Read/Write line (R/W) during each cycle for each instruction.

This information is useful in comparing actual with expected results during debug of both software and hardware as the control program is executed. The information is categorized in groups according to Addressing Mode and Number of Cycles per instruction. (In general, instructions with the same Addressing mode and Number of Cycles execute in the same manner; exceptions are indicated in the table.).

Table 8. Operation Summary

ADDRESS MODE AND INSTRUCTIONS	CYCLES	CYCLE #	VMA LINE	ADDRESS BUS	R/W LINE	DATA BUS
IMMEDIATE						
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	2	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	OPERAND DATA
CPX LDS LDX	3	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	OPERAND DATA (High Order Byte)
		3	1	OP CODE ADDRESS + 2	1	OPERAND DATA (Low Order Byte)
DIRECT						
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	3	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	ADDRESS OF OPERAND
		3	1	ADDRESS OF OPERAND	1	OPERAND DATA
CPX LDS LDX	4	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS +1	1	ADDRESS OF OPERAND
		3	1	ADDRESS OF OPERAND	1	OPERAND DATA (High Order Byte)
		4	1	OPERAND ADDRESS +1	1	OPERAND DATA (Low Order Byte)
STA	4	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	DESTINATION ADDRESS
		3	0	DESTINATION ADDRESS	1	IRRELEVANT DATA (Note 1)
		4	1	DESTINATION ADDRESS	0	DATA FROM ACCUMULATOR
STS STX	5	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	ADDRESS OF OPERAND
		3	0	ADDRESS OF OPERAND	1	IRRELEVANT DATA (Note 1)
		4	1	ADDRESS OF OPERAND	0	REGISTER DATA (High Order Byte)
		5	1	ADDRESS OF OPERAND + 1	0	REGISTER DATA (Low Order Byte)
INDEXED						
JMP	4	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	OFFSET
		3	0	INDEX REGISTER	1	IRRELEVANT DATA (Note 1)
		4	0	INDEX REGISTER PLUS OFFSET (W/O CARRY)	1	IRRELEVANT DATA (note 1)
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	5	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	OFFSET
		3	0	INDEX REGISTER	1	IRRELEVANT DATA (Note 1)
		4	0	INDEX REGISTER PLUS OFFSET (W/O CARRY)	1	IRRELEVANT DATA (Note 1)
		5	1	INDEX REGISTER PLUS OFFSET	1	OPERAND DATA

Table 8. Operation Summary (Continued)

ADDRESS MODE AND INSTRUCTIONS	CYCLES	CYCLE #	VMA LINE	ADDRESS BUS	R/W LINE	DATA BUS
INDEXED (Continued)						
CPX LDS LDX	6	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	OFFSET
		3	0	INDEX REGISTER	1	IRRELEVANT DATA (Note 1)
		4	0	INDEX REGISTER PLUS OFFSET (W O CARRY)	1	IRRELEVANT DATA (Note 1)
		5	1	INDEX REGISTER PLUS OFFSET	1	OPERAND DATA (High Order Byte)
		6	1	INDEX REGISTER PLUS OFFSET + 1	1	OPERAND DATA (Low Order Byte)
STA	6	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	OFFSET
		3	0	INDEX REGISTER	1	IRRELEVANT DATA (Note 1)
		4	0	INDEX REGISTER PLUS OFFSET (W O CARRY)	1	IRRELEVANT DATA (Note 1)
		5	0	INDEX REGISTER PLUS OFFSET	1	IRRELEVANT DATA (Note 1)
		6	1	INDEX REGISTER PLUS OFFSET	0	OPERAND DATA
ASL LSR ASR MEG CLR ROL COM ROR DEC TST INC	7	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	OFFSET
		3	0	INDEX REGISTER	1	IRRELEVANT DATA (Note 1)
		4	0	INDEX REGISTER PLUS OFFSET (W/O CARRY)	1	IRRELEVANT DATA (Note 1)
		5	1	INDEX REGISTER PLUS OFFSET	1	CURRENT OPERAND DATA
		6	0	INDEX REGISTER PLUS OFFSET	1	IRRELEVANT DATA (Note 1)
		7	1/0[4]	INDEX REGISTER PLUS OFFSET	0	NEW OPERAND DATA (Note 3)
STS STX	7	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	OFFSET
		3	0	INDEX REGISTER	1	IRRELEVANT DATA (Note 1)
		4	0	INDEX REGISTER PLUS OFFSET (W/O CARRY)	1	IRRELEVANT DATA (Note 1)
		5	0	INDEX REGISTER PLUS OFFSET	1	IRRELEVANT DATA (Note 1)
		6	1	INDEX REGISTER PLUS OFFSET	0	OPERAND DATA (High Order Byte)
		7	1	INDEX REGISTER PLUS OFFSET + 1	0	OPERAND DATA (Low Order Byte)
JSR	8	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	OFFSET
		3	0	INDEX REGISTER	1	IRRELEVANT DATA (Note 1)
		4	1	STACK POINTER	0	RETURN ADDRESS (Low Order Byte)
		5	1	STACK POINTER – 1	0	RETURN ADDRESS (High Order Byte)
		6	0	STACK POINTER – 2	1	IRRELEVANT DATA (Note 1)
		7	0	INDEX REGISTER	1	IRRELEVANT DATA (Note 1)
		8	0	INDEX REGISTER PLUS OFFSET (W/O CARRY)	1	IRRELEVANT DATA (Note 1)
EXTENDED						
JMP	3	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	JUMP ADDRESS (High Order Byte)
		3	1	OP CODE ADDRESS + 2	1	JUMP ADDRESS (Low Order Byte)
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB		1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	ADDRESS OF OPERAND (High Order Byte)
		3	1	OP CODE ADDRESS + 2	1	JUMP ADDRESS (Low Order Byte)
		4	1	ADDRESS OF OPERAND	1	OPERAND DATA

Table 8. Operation Summary (Continued)

ADDRESS MODE AND INSTRUCTIONS	CYCLES	CYCLE #	VMA LINE	ADDRESS BUS	R/W LINE	DATA BUS
EXTENDED (Continued)						
CPX LDS LDX	5	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	ADDRESS OF OPERAND (High Order Byte)
		3	1	OP CODE ADDRESS + 2	1	ADDRESS OF OPERAND (Low Order Byte)
		4	1	ADDRESS OF OPERAND	1	OPERAND DATA (High Order Byte)
		5	1	ADDRESS OF OPERAND + 1	1	OPERAND DATA (Low Order Byte)
STA A STA B	5	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	DESTINATION ADDRESS (High Order Byte)
		3	1	OP CODE ADDRESS + 2	1	DESTINATION ADDRESS (Low Order Byte)
		4	0	OPERAND DESTINATION ADDRESS	1	IRRELEVANT DATA (Note 1)
		5	1	OPERAND DESTINATION ADDRESS	0	DATA FROM ACCUMULATOR
ASL LSR ASR NEG CLR ROL COM ROR DEC TST INC	6	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	ADDRESS OF OPERAND (High Order Byte)
		3	1	OP CODE ADDRESS + 2	1	ADDRESS OF OPERAND (Low Order Byte)
		4	1	ADDRESS OF OPERAND	1	CURRENT OPERAND DATA
		5	0	ADDRESS OF OPERAND	1	IRRELEVANT DATA (Note 1)
		6	1/0[3]	ADDRESS OF OPERAND	0	NEW OPERAND DATA (Note 3)
STS STX	6	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	ADDRESS OF OPERAND (High Order Byte)
		3	1	OP CODE ADDRESS + 2	1	ADDRESS OF OPERAND (Low Order Byte)
		4	0	ADDRESS OF OPERAND	1	IRRELEVANT DATA (Note 1)
		5	1	ADDRESS OF OPERAND	0	OPERAND DATA (High Order Byte)
		6	1	ADDRESS OF OPERAND + 1	0	OPERAND DATA (Low Order Byte)
JSR	9	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	ADDRESS OF SUBROUTINE (High Order Byte)
		3	1	OP CODE ADDRESS + 2	1	ADDRESS OF SUBROUTINE (Low Order Byte)
		4	1	SUBROUTINE STARTING ADDRESS	1	OP CODE OF NEXT INSTRUCTION
		5	1	STACK POINTER	0	RETURN ADDRESS (Low Order Byte)
		6	1	STACK POINTER – 1	0	RETURN ADDRESS (High Order Byte)
		7	0	STACK POINTER - 2	1	IRRELEVANT DATA (Note 1)
		8	0	OP CODE ADDRESS + 2	1	IRRELEVANT DATA (Note 1)
		9	1	OP CODE ADDRESS + 2	1	ADDRESS OF SUBROUTINE (Low Order Byte)

Table 8. Operation Summary (Continued)

ADDRESS MODE AND INSTRUCTIONS	CYCLES	CYCLE #	VMA LINE	ADDRESS BUS	R/W LINE	DATA BUS
INHERENT						
ABA DAA SEC ASL DEC SEI ASR INC SEV CBA LSR TAB CLC NEG TAP CLI NOP TBA CLR ROL TPA CLV ROR TST COM SBA	2	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	OP CODE OF NEXT INSTRUCTION
DES DEX INS INX	4	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	OP CODE OF NEXT INSTRUCTION
		3	0	PREVIOUS REGISTER CONTENTS	1	IRRELEVANT DATA (Note 1)
		4	0	NEW REGISTER CONTENTS	1	IRRELEVANT DATA (Note 1)
PSH	4	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	OP CODE OF NEXT INSTRUCTION
		3	1	STACK POINTER	0	ACCUMULATOR DATA
		4	0	STACK POINTER – 1	1	ACCUMULATOR DATA
PUL	4	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	OP CODE OF NEXT INSTRUCTION
		3	0	STACK POINTER	1	IRRELEVANT DATA (Note 1)
		4	1	STACK POINTER + 1	1	OPERAND DATA FROM STACK
TSX	4	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	OP CODE OF NEXT INSTRUCTION
		3	0	STACK POINTER	1	IRRELEVANT DATA (Note 1)
		4	0	NEW INDEX REGISTER	1	IRRELEVANT DATA (Note 1)
TXS	4	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	OP CODE OF NEXT INSTRUCTION
		3	0	INDEX REGISTER	1	IRRELEVANT DATA
		4	0	NEW STACK POINTER	1	IRRELEVANT DATA
RTS	5	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	IRRELEVANT DATA (Note 2)
		3	0	STACK POINTER	1	IRRELEVANT DATA (Note 1)
		4	1	STACK POINTER + 1	1	ADDRESS OF NEXT INSTRUCTION (High Order Byte)
		5	1	STACK POINTER + 2	1	ADDRESS OF NEXT INSTRUCTION (Low Order Byte)
WAI	9	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	OP CODE OF NEXT INSTRUCTION
		3	1	STACK POINTER	0	RETURN ADDRESS (Low Order Byte)
		4	1	STACK POINTER – 1	0	RETURN ADDRESS (High Order Byte)
		5	1	STACK POINTER – 2	0	INDEX REGISTER (Low Order Byte)
		6	1	STACK POINTER – 3	0	INDEX REGISTER (High Order Byte)
		7	1	STACK POINTER – 4	0	CONTENTS OF ACCUMULATOR A
		8	1	STACK POINTER – 5	0	CONTENTS OF ACCUMULATOR B
		9	1	STACK POINTER – 6 (Note 4)	1	CONTENTS OF COND. CODE REGISTER

Table 8. Operation Summary (Continued)

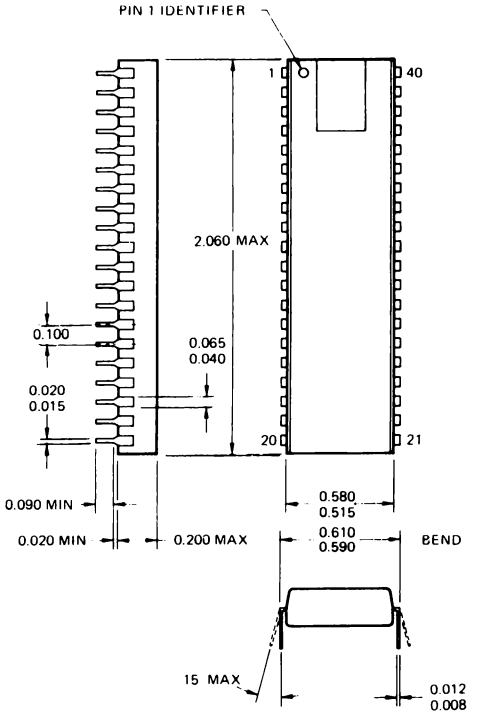
ADDRESS MODE AND INSTRUCTIONS	CYCLES	CYCLE #	VMA LINE	ADDRESS BUS	R/W LINE	DATA BUS
INHERENT (Continued)						
RTI	10	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	IRRELEVANT DATA (Note 2)
		3	0	STACK POINTER	1	IRRELEVANT DATA (Note 1)
		4	1	STACK POINTER + 1	1	CONTENTS OF COND. CODE REGISTER FROM STACK
		5	1	STACK POINTER + 2	1	CONTENTS OF ACCUMULATOR B FROM STACK
		6	1	STACK POINTER + 3	1	CONTENTS OF ACCUMULATOR A FROM STACK
		7	1	STACK POINTER + 4	1	INDEX REGISTER FROM STACK (High Order Byte)
		8	1	STACK POINTER + 5	1	INDEX REGISTER FROM STACK (Low Order Byte)
		9	1	STACK POINTER + 6	1	NEXT INSTRUCTION ADDRESS FROM STACK (High Order Byte)
		10	1	STACK POINTER + 7	1	NEXT INSTRUCTION ADDRESS FROM STACK (Low Order Byte)
SWI	12	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	IRRELEVANT DATA (Note 1)
		3	1	STACK POINTER	0	RETURN ADDRESS (Low Order Byte)
		4	1	STACK POINTER – 1	0	RETURN ADDRESS (High Order Byte)
		5	1	STACK POINTER – 2	0	INDEX REGISTER (Low Order Byte)
		6	1	STACK POINTER – 3	0	INDEX REGISTER (High Order Byte)
		7	1	STACK POINTER – 4	0	CONTENTS OF ACCUMULATOR A
		8	1	STACK POINTER – 5	0	CONTENTS OF ACCUMULATOR B
		9	1	STACK POINTER – 6	0	CONTENTS OF COND. CODE REGISTER
		10	0	STACK POINTER – 7	1	IRRELEVANT DATA (Note 1)
		11	1	VECTOR ADDRESS FFFA (HEX)	1	ADDRESS OF SUBROUTINE (High Order Byte)
		12	1	VECTOR ADDRESS FFFB (HEX)	1	ADDRESS OF SUBROUTINE (Low Order Byte)
RELATIVE						
BCC BHI BNE BCS BLE BPL BEQ BLS BRA BGE BLT BVC BGT BMI BVS	4	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	BRANCH OFFSET
		3	0	OP CODE ADDRESS + 2	1	IRRELEVANT DATA (Note 1)
		4	0	BRANCH ADDRESS	1	IRRELEVANT DATA (Note 1)
BSR	8	1	1	OP CODE ADDRESS	1	OP CODE
		2	1	OP CODE ADDRESS + 1	1	BRANCH OFFSET
		3	0	RETURN ADDRESS OF MAIN PROGRAM	1	IRRELEVANT DATA (Note 1)
		4	1	STACK POINTER	0	RETURN ADDRESS (Low Order Byte)
		5	1	STACK POINTER – 1	0	RETURN ADDRESS (High Order Byte)
		6	0	STACK POINTER – 2	1	IRRELEVANT DATA (Note 1)
		7	0	RETURN ADDRESS OF MAIN PROGRAM	1	IRRELEVANT DATA (Note 1)
		8	0	SUBROUTINE	1	IRRELEVANT DATA (Note 1)

Note 1. If device which is addressed during this cycle uses VMA, then the Data Bus will go to the high impedance three-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the Data Bus.
Note 2. Data is ignored by the MPU.
Note 3. For TST, VMA = 0 and Operand data does not change.
Note 4. While the MPU is waiting for the interrupt, Bus Available will go high indicating the following states of the control lines: VMA is low; Address Bus, R/W, and Data Bus are all in the high impedance state.

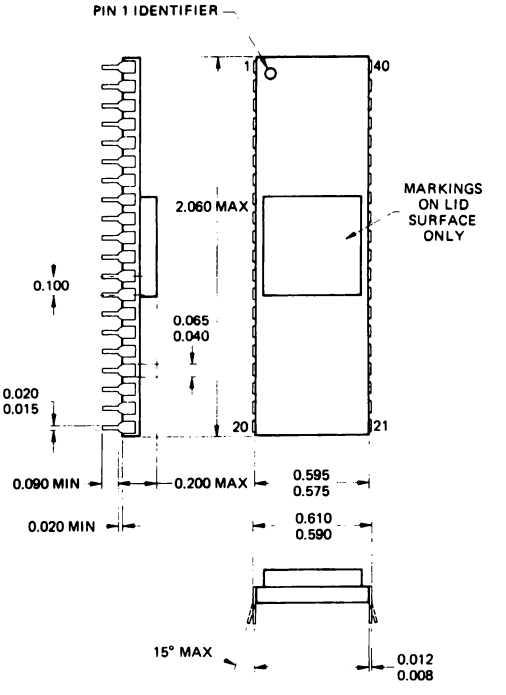
S6802

Package Dimensions

40-Pin Plastic



40-Pin Ceramic

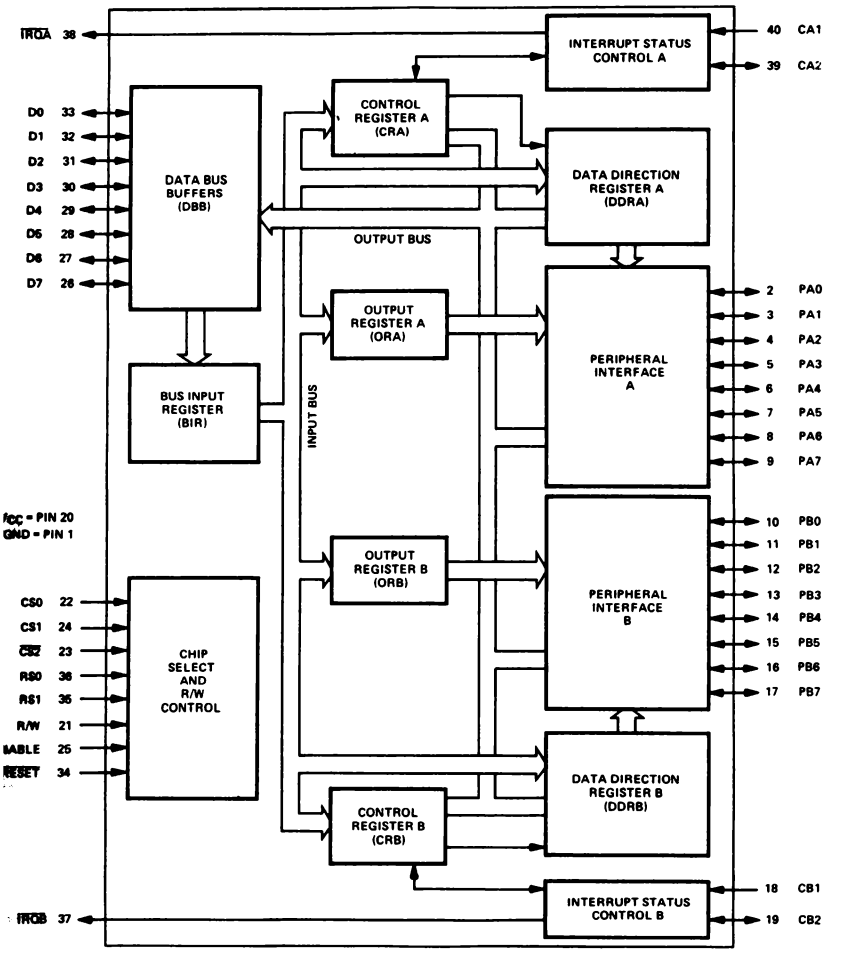


Ordering Information

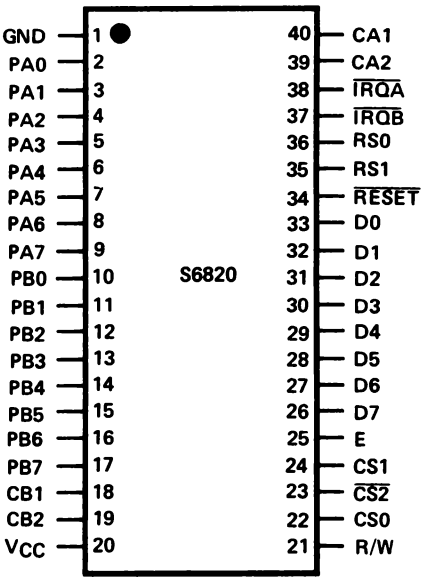
Order No.	No Pins	Package	Temp. Range	Description
S6802P	40	Plastic	0°C — + 70°C	NMOS MPU with 128 Byte RAM and Clock
S6802	40	Ceramic	0°C — + 70°C	NMOS MPU with 128 Byte RAM and Clock

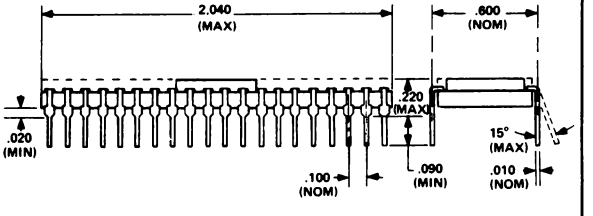
S6820I

PERIPHERAL INTERFACE ADAPTER (PIA)



BLOCK DIAGRAM





PIN/PACKAGE CONFIGURATION

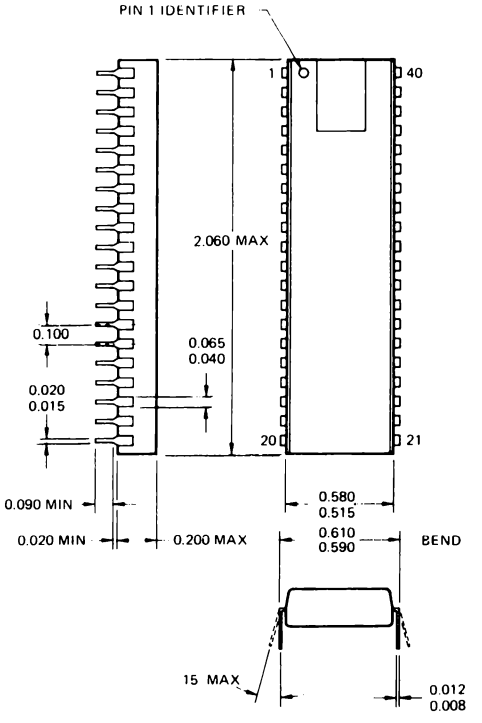
FEATURES

- 8-Bit Bidirectional Data Bus for Communication with the MPU
- Two Bidirectional 8-Bit Buses for Interface to Peripherals
- Two Programmable Control Registers
- Two Programmable Data Direction Registers
- Four Individually-Controlled Interrupt Input Lines; Two Usable as Peripheral Control Outputs
- Handshake Control Logic for Input and Output Peripheral Operation
- High-Impedance 3-State and Direct Transistor Drive Peripheral Lines
- Program Controlled Interrupt and Interrupt Disable Capability
- CMOS Compatible Peripheral Lines

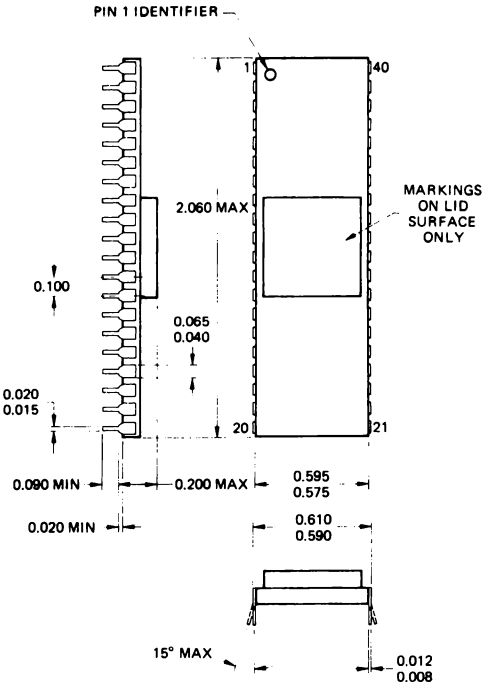
S6802

Package Dimensions

40-Pin Plastic



40-Pin Ceramic

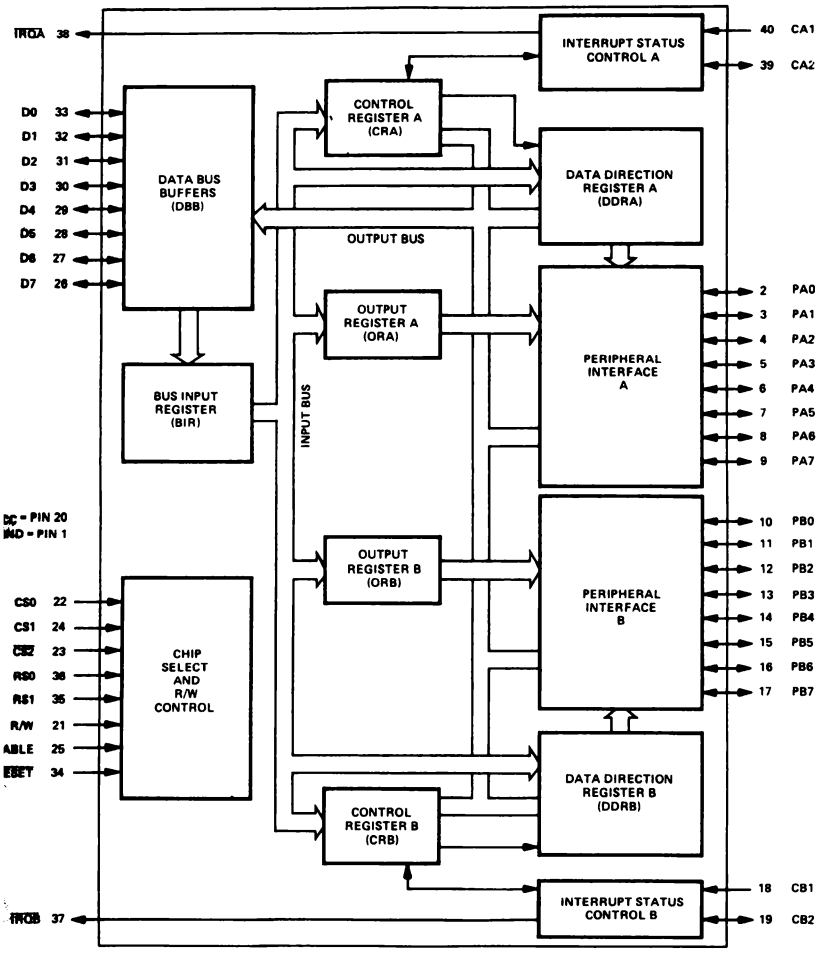


Ordering Information

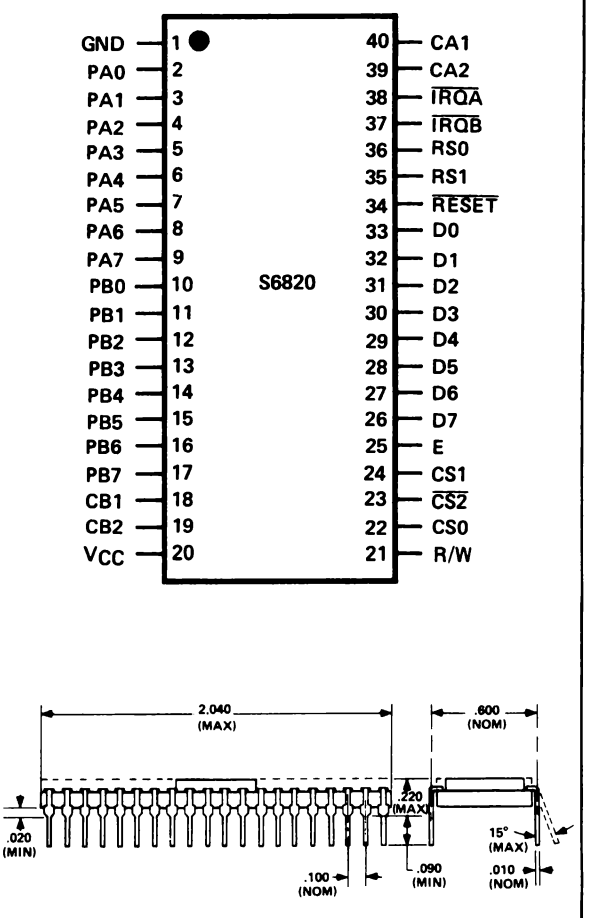
Order No.	No Pins	Package	Temp. Range	Description
S6802P	40	Plastic	0°C — + 70°C	NMOS MPU with 128 Byte RAM and Clock
S6802	40	Ceramic	0°C — + 70°C	NMOS MPU with 128 Byte RAM and Clock

S6820I

PERIPHERAL INTERFACE ADAPTER (PIA)



BLOCK DIAGRAM



PIN/PACKAGE CONFIGURATION

FEATURES

- 8-Bit Bidirectional Data Bus for Communication with the MPU
- Two Bidirectional 8-Bit Buses for Interface to Peripherals
- Two Programmable Control Registers
- Two Programmable Data Direction Registers
- Four Individually-Controlled Interrupt Input Lines; Two Usable as Peripheral Control Outputs
- Handshake Control Logic for Input and Output Peripheral Operation
- High-Impedance 3-State and Direct Transistor Drive Peripheral Lines
- Program Controlled Interrupt and Interrupt Disable Capability
- CMOS Compatible Peripheral Lines

S6820I

IPHERAL INTERFACE ADAPTER (PIA)

FUNCTIONAL DESCRIPTION

The S6820 Peripheral Interface Adapter provides the universal means of interfacing peripheral equipment to the S6800 Microprocessing Unit (MPU). This device is capable of interfacing the MPU to peripherals through two 8-bit bidirectional peripheral data buses and four control lines. No external logic is required for interfacing to most peripheral devices.

The functional configuration of the PIA is programmed by the MPU during system initialization. Each of the peripheral data lines can be programmed to act as an input or output, and each of the four control/interrupt lines may be programmed for one of several control modes. This allows a high degree of flexibility in the over-all operation of the interface.

The PIA interfaces to the S6800 MPU with an eight-bit bidirectional data bus, three chip select lines, two register select lines, two interrupt request lines, read/write line, enable line and reset line. These signals, in conjunction with the S6800 VMA output, permit the MPU to have complete control over the PIA. VMA may be utilized to gate the input signals to the PIA.

ABSOLUTE MAXIMUM RATINGS

Supply Voltage V_{CC}	-0.3 to +7.0V
Input Voltage V_{in}	-0.3 to +7.0V
Operating Temperature Range T_A	- 40°C to + 85°C
Storage Temperature Range T_{stg}	-55 to +150°C

NOTE: This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields, however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high-impedance circuit.

S6820I
PERIPHERAL INTERFACE ADAPTER (PIA)

DC (STATIC) CHARACTERISTICS
(V_{CC} = 5.0V ± 5%; T_A = - 40°C to + 85°C (unless otherwise noted)

Characteristic	Symbol	Min.	Typ.	Max.	Unit
Input High Voltage (Normal Operating Levels)	V_{IH}	+2.4	–	V_{CC}	Vdc
Input Low Voltage (Normal Operating Levels)	V_{IL}	-0.3	–	+0.4	Vdc
Input High Threshold Voltage All Inputs Except Enable	V_{IHT}	+2.0	–	–	Vdc
Input Low Threshold Voltage All Inputs Except Enable	V_{ILT}	–	–	+0.8	Vdc
Input Leakage Current (V_{in} = 0 to 5.0 Vdc) R/W, $\overline{\text{Reset}}$, RS0, RS1, CS0, CS1, $\overline{\text{CS2}}$, CA1, CB1, Enable	I_{in}	–	1.0	2.5	μAdc
Three-State (Off State) Input Current (V_{in} = 0.4 to 2.4 Vdc, V_{CC} = max) D0-D7, PB0-PB7, CB2	I_{TSI}	–	2.0	10	μAdc
Input High Current PA0-PA7, CA2 (V_{IH} = 2.4 Vdc)	I_{IH}	100	250	–	μAdc
Input Low Current PA0-PA7, CA2 (V_{IL} = 0.4 Vdc)	I_{IL}	–	1.0	1.6	mAdc
Output High Voltage (V_{CC} = min, I_{Load} = -100 μAdc, Enable Pulse Width < 25 μs)	V_{OH}	+2.4	–	–	Vdc
Output Low Voltage (V_{CC} = min, I_{Load} = 1.6 mAdc)	V_{OL}	–	–	+0.4	Vdc
Output High Current (Sourcing) (V_{OH} = 2.4 Vdc) (V_{OH} = 1.5 Vdc, the current for driving other than TTL, e.g., Darlington Base) PB0-PB7, CB2	I_{OH}	-100 -1.0	-1000 -2.5	– –	μAdc mAdc
Output Low Current (Sinking) (V_{OL} = 0.4 Vdc)	I_{OL}	1.6	–	–	mAdc
Output Leakage Current (Off State) $\overline{\text{IRQA}}$, $\overline{\text{IRQB}}$	I_{off}	–	1.0	10	μAdc
Supply Current	I_{CC}	–	56	112	mA
Input Capacitance (V_{in} = 0, T_A = 25°C, f = 1.0 MHz) D0-D7, PA0-PA7, PB0-PB7, CA2, CB2 R/W, $\overline{\text{Reset}}$, RS0, RS1, CS0, CS1, $\overline{\text{CS2}}$, CA1, CB1 Enable	C_{in}	 – – –	 – – –	 10 7.0 20	pF
Output Capacitance (V_{in} = 0, T_A = 25°C, f = 1.0 MHz)	C_{out}	–	–	10	pF

S6820I
PERIPHERAL INTERFACE ADAPTER (PIA)

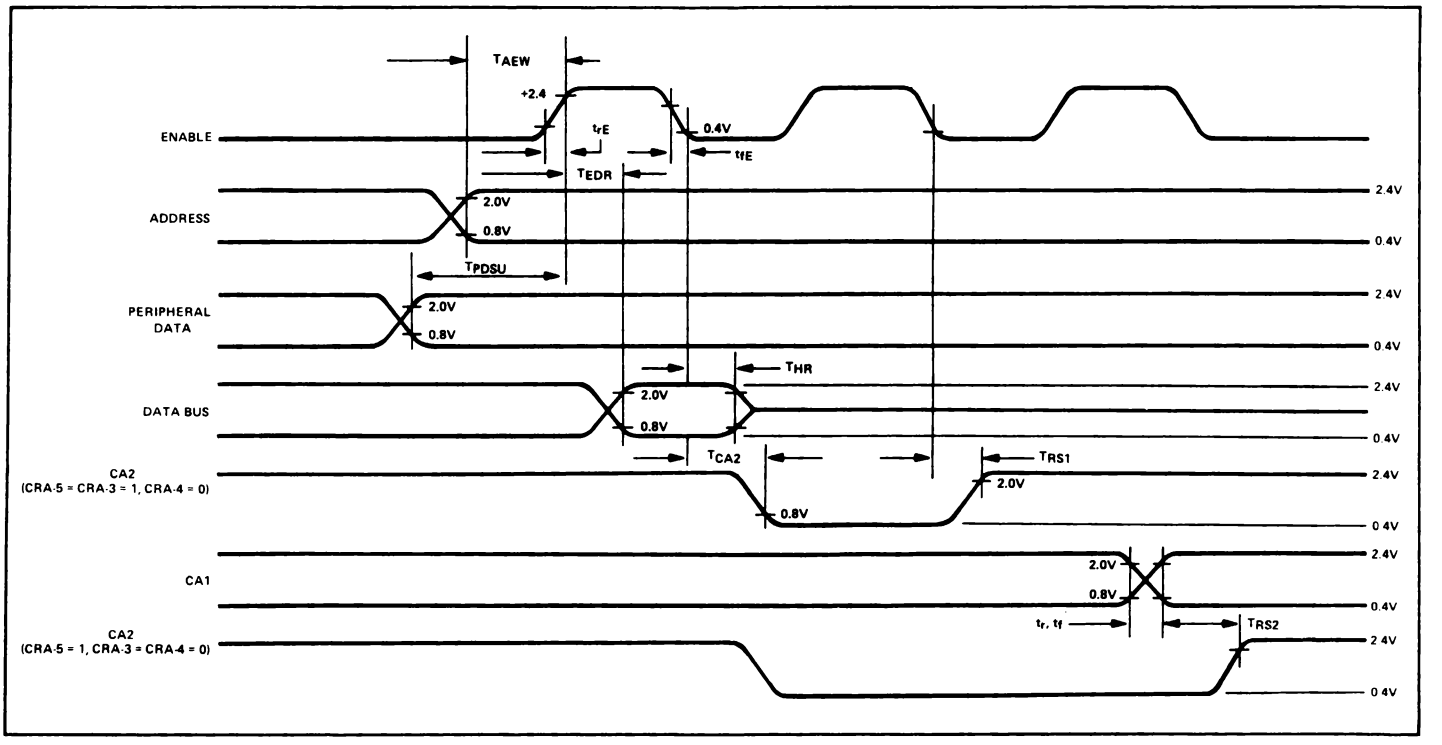
AC (DYNAMIC) CHARACTERISTICS Loading = 30 pF and one TTL load for PA0-PA7, PB0-PB7, CA2, CB2
= 130 pF and one TTL load for D0-D7, $\overline{IRQA}$, $\overline{IRQB}$

(V_{CC} = 5.0V ± 5%; T_A = - 40°C to + 85°C (unless otherwise noted)

READ TIMING CHARACTERISTICS (Figure 1)

Characteristic	Symbol	Min.	Typ.	Max.	Unit
Delay Time, Address valid to Enable positive transition	T_{AEW}	180	–	–	ns
Delay Time, Enable positive transition to Data valid on bus	T_{EDR}	–	–	395	ns
Peripheral Data Setup Time	T_{PDSU}	300	–	–	ns
Data Bus Hold Time	T_{HR}	10	–	–	ns
Delay Time, Enable negative transition to CA2 negative transition	T_{CA2}	–	–	1.0	µs
Delay Time, Enable negative transition to CA2 positive transition	T_{RS1}	–	–	1.0	µs
Rise and Fall Time for CA1 and CA2 input signals	t_r, t_f	–	–	1.0	µs
Delay Time from CA1 active transition to CA2 positive transition	T_{RS2}	–	–	2.0	µs
Rise and Fall Time for Enable input	t_{rE}, t_{fE}	–	–	25	µs

FIGURE 1 – READ TIMING CHARACTERISTICS

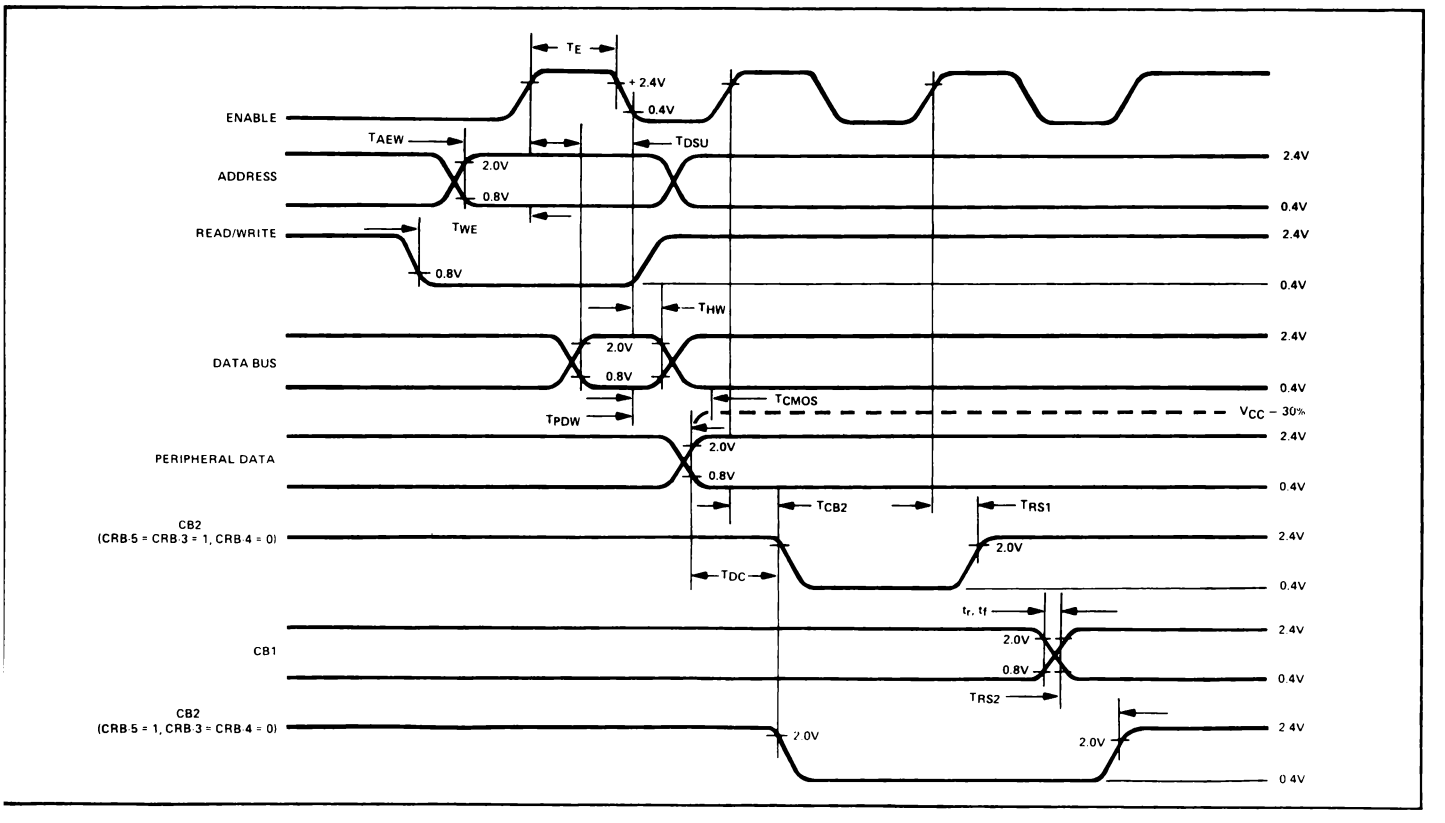


S6820I
PERIPHERAL INTERFACE ADAPTER (PIA)

WRITE TIMING CHARACTERISTICS (Figure 2)

Characteristic	Symbol	Min.	Typ.	Max.	Unit
Enable Pulse Width	T_E	0.470	–	25	µs
Delay Time, Address valid to Enable positive transition	T_{AEW}	180	–	–	ns
Delay Time, Data valid to Enable negative transition	T_{DSU}	300	–	–	ns
Delay Time, Read/Write negative transition to Enable positive transition	T_{WE}	130	–	–	ns
Data Bus Hold Time.	T_{HW}	10	–	–	ns
Delay Time, Enable negative transition to Peripheral Data valid	T_{PDW}	–	–	1.0	µs
Delay Time, Enable negative transition to Peripheral Data valid, CMOS (V_{CC} – 30%) PA0-PA7, CA2	T_{CMOS}	–	–	2.0	µs
Delay Time, Enable positive transition to CB2 negative transition	T_{CB2}	–	–	1.0	µs
Delay Time, Peripheral Data valid to CB2 negative transition	T_{DC}	0	–	1.5	µs
Delay Time, Enable positive transition to CB2 positive transition	T_{RS1}	–	–	1.0	µs
Rise and Fall Time for CB1 and CB2 input signals	t_r, t_f	–	–	1.0	µs
Delay Time, CB1 active transition to CB2 positive transition	T_{RS2}	–	–	2.0	µs

FIGURE 2 – WRITE TIMING CHARACTERISTICS



S6820I
PERIPHERAL INTERFACE ADAPTER (PIA)

INTERFACE DESCRIPTION

MPU/PIA INTERFACE

Pin	Label	Function
(33)	D0	**Bi-Directional Data** – The bi-directional data lines (D0-D7) allow the transfer of data between the MPU and the PIA. The data bus output drivers are three-state devices that remain in the high-impedance (off) state except when the MPU performs a PIA read operation. The Read/Write line is in the Read (high) state when the PIA is selected for a Read operation.
(32)	D1	
(31)	D2	
(30)	D3	
(29)	D4	
(28)	D5	
(27)	D6	
(26)	D7	
(25)	E	**Enable** – The enable pulse, E, is the only timing signal that is supplied to the PIA. Timing of all other signals is referenced to the leading and trailing edges of the E pulse. This signal will normally be a derivative of the S6800 $\phi 2$ Clock. The E pulse is used to condition the interrupt/control lines CA1, CA2, CB1, and CB2. At least one E pulse must occur from the inactive edge to the active edge of the input signal to set the interrupt flag, when the lines are used as inputs.
(21)	R/W	**Read/Write** – This signal is generated by the MPU to control the direction of data transfers on the Data Bus. A low state on the PIA Read/Write line enables the input buffers and data is transferred from the MPU to the PIA on the E signal if the device has been selected. A high on the Read/Write line sets up the PIA for a transfer of data to the bus. The PIA output buffers are enabled when the proper address and the enable pulse E are present.
(34)	$\overline{\text{RESET}}$	**$\overline{\text{Reset}}$** – The active low $\overline{\text{Reset}}$ line is used to reset all register bits in the PIA to a logical zero (low). This line can be used as a power-on reset and as a master reset during system operation.
(22)	CS0	**Chip Select** – These three input signals are used to select the PIA. CS0 and CS1 must be high and $\overline{\text{CS2}}$ must be low for selection of the device. Data transfers are then performed under the control of the Enable and Read/Write signals. The chip select lines must be stable for the duration of the E pulse.
(24)	CS1	
(23)	$\overline{\text{CS2}}$	
(36)	RS0	**PIA Register Select** – The two register select lines are used to select the various registers inside the PIA. These two lines are used in conjunction with internal Control Registers to select a particular register that is to be written or read. The Register select lines should be stable for the duration of the E pulse while in the read or write cycle.
(35)	RS1	

S6820I
PERIPHERAL INTERFACE ADAPTER (PIA)

(38)	$\overline{\text{IRQA}}$	**Interrupt Request** – The active low Interrupt Request lines ($\overline{\text{IRQA}}$ and $\overline{\text{IRQB}}$) act to
(37)	$\overline{\text{IRQB}}$	interrupt the MPU either directly or through interrupt priority circuitry. These lines are "open source" (no load device on the chip) and are capable of sinking a current of 1.6 mA from an external source. This permits all interrupt request lines to be tied together in a wire-OR configuration.

Each Interrupt Request line has two internal interrupt flag bits that will cause the Interrupt Request line to go low. Each flag bit is associated with a particular peripheral interrupt line. Also four interrupt enable bits are provided in the PIA which may be used to inhibit a particular interrupt from a peripheral device.

Servicing an interrupt by the MPU is accomplished by a software routine that, on a prioritized basis, sequentially reads and tests the two control registers in each PIA for interrupt flag bits that are set.

The Interrupt Flag is cleared (zeroed) as a result of an MPU Read Peripheral Data Operation.

PIA/PERIPHERAL INTERFACE

Pin	Label	Function
(2)	PA0	**Section A Peripheral Data** – Each of the peripheral data lines can be programmed
(3)	PA1	to act as an input or output. This is accomplished by setting a "1" in the corresponding
(4)	PA2	Data Direction Register bit for those lines which are to be outputs. A "0" in a bit of the
(5)	PA3	Data Direction Register causes the corresponding peripheral data line to act as an input.
(6)	PA4	During an MPU Read Peripheral Data Operation, the data on peripheral lines programmed to
(7)	PA5	act as inputs appears directly on the corresponding MPU Data Bus lines. In the input mode
(8)	PA6	the internal pullup resistor on these lines represents a maximum of one standard TTL load.
(9)	PA7	The data in Output Register A will appear on the data lines that are programmed to be outputs. A logical "1" written into the register will cause a "high" on the corresponding data line while a "0" results in a "low". Data in Output Register A may be read by an MPU "Read Peripheral Data A" operation when the corresponding lines are programmed as outputs. This data will be read properly if the voltage on the peripheral data lines is greater than 2.0 volts for a logic "1" output and less than 0.8 volt for a logic "0" output. Loading the output lines such that the voltage on these lines does not reach full voltage causes the data transferred into the MPU on a Read operation to differ from that contained in the respective bit of Output Register A.
(10)	PB0	**Section B Peripheral Data** – The peripheral data lines in the B Section of the PIA can
(11)	PB1	be programmed to act as either inputs or outputs in a similar manner to PA0-PA7. How-
(12)	PB2	ever, the output buffers driving these lines differ from those driving lines PA0-PA7. They
(13)	PB3	have three-state capability, allowing them to enter a high impedance state when the peripher-
(14)	PB4	al data line is used as an input. In addition, data on the peripheral data lines PB0-PB7 will
(15)	PB5	be read properly from those lines programmed as outputs even if the voltages are below 2.0
(16)	PB6	volts for a "high". As outputs, these lines are compatible with standard TTL and may also be
(17)	PB7	used as a source of up to 1 milliampere at 1.5 volts to directly drive the base of a transistor switch.

S6820I
PERIPHERAL INTERFACE ADAPTER (PIA)

Pin	Label	Function
(40) (18)	CA1 CB1	**Interrupt Input** – Peripheral Input lines CA1 and CB1 are input-only lines that set the interrupt flags of the control registers. The active transition for these signals is also programmed by the two control registers.
(39)	CA2	**Peripheral Control** – The peripheral control line CA2 can be programmed to act as an interrupt input or as a peripheral control output. As an output, this line is compatible with standard TTL; as an input the internal pullup resistor on this line represents one standard TTL load. The function of this signal line is programmed with Control Register A.
(19)	CB2	**Peripheral Control** – Peripheral Control line CB2 may also be programmed to act as an interrupt input or peripheral control output. As an input, this line has high input impedance and is compatible with standard TTL. As an output it is compatible with standard TTL and may also be used as a source of up to 1 milliampere at 1.5 volts to directly drive the base of a transistor switch. This line is programmed by Control Register B.
(1)	GND	**Ground**
(20)	V_{CC}	+5 Volts ± 5%

S6820I
PERIPHERAL INTERFACE ADAPTER (PIA)

APPLICATION INFORMATION

INITIALIZATION

A low reset line has the effect of zeroing all PIA registers. This will set PA0-PA7, PB0-PB7, CA2 and CB2 as inputs, and all interrupts disabled. The PIA must be configured during the restart program which follows the reset.

REGISTER ADDRESSING

There are six locations within the PIA accessible to the MPU data bus; two Peripheral Registers, two Data Direction Registers, and two Control Registers, Selection of these locations is controlled by the RS0 and RS1 inputs together with bit 2 in the Control Register, as shown in Table 1.

TABLE 1 – INTERNAL ADDRESSING

RS1	RS0	Control Register Bit CRA-2	Control Register Bit CRB-2	Location Selected
0	0	1	X	Peripheral Register A
0	0	0	X	Data Direction Register A
0	1	X	X	Control Register A
1	0	X	1	Peripheral Register B
1	0	X	0	Data Direction Register B
1	1	X	X	Control Register B

X = Don't Care

DATA DIRECTION REGISTERS (DDRA and DDRB)

The two Data Direction Registers allow the MPU to control the direction of data through each corresponding peripheral data line. All Data Direction Register bits set at "0" configure the corresponding peripheral data line as an input; all "1s" result in an output.

CONTROL REGISTERS (CRA and CRB)

The two Control Registers (CRA and CRB) allow the MPU to control the operation of the four peripheral control lines CA1, CA2, CB1 and CB2. In addition they allow the MPU to enable the interrupt lines and monitor the status of the interrupt flags. Bits 0 through 5 of the two registers may be written or read by the MPU when the proper chip select and register select signals are applied. Bits 6 and 7 of the two registers are read only and are modified by external interrupts occurring on control lines CA1, CA2, CB1 or CB2. The format of the control words is shown in Table 2.

TABLE 2 – CONTROL WORD FORMAT

	7	6	5	4	3	2	1	0
CRA	IRQA1	IRQA2	CA2 Control			DDRA Access	CA1 Control	
CRB	IRQB1	IRQB2	CB2 Control			DDRB Access	CB1 Control	

Data Direction Access Control Bit (CRA-2 and CRB-2) – Bit 2 in each Control register (CRA and CRB) allows selection of either a Peripheral Interface Register or the Data Direction Register when the proper register select signals are applied to RS0 and RS1.

Control of CA1 and CB1 Interrupt Input Lines (CRA-0, CRB-0, CRA-1, and CRB-1) – The two lowest order bits of the control registers are used to control the interrupt input lines CA1 and CB1. Bits CRA-0 and CRB-0 are used to enable the MPU interrupt signals $\overline{IRQA}$ and $\overline{IRQB}$, respectively. Bits CRA-1 and CRB-1 determine the active transition of the interrupt input signals CA1 and CB1 (Table 3).

TABLE 3 – CONTROL OF INTERRUPT INPUTS CA1 AND CB1

CRA-1 (CRB-1)	CRA-0 (CRB-0)	Interrupt Input CA1 (CB1)	Interrupt Flag CRA-7 (CRB-7)	MPU Interrupt Request $\overline{\text{IRQA}}$ ($\overline{\text{IRQB}}$)
0	0	↓ Active	Set high on ↓ of CA1 (CB1)	Disabled – $\overline{\text{IRQ}}$ remains high
0	1	↓ Active	Set high on ↓ of CA1 (CB1)	Goes low when the interrupt flag bit CRA-7 (CRB-7) goes high
1	0	↑ Active	Set high on ↑ of CA1 (CB1)	Disabled – $\overline{\text{IRQ}}$ remains high
1	1	↑ Active	Set high on ↑ of CA1 (CB1)	Goes low when the interrupt flag bit CRA-7 (CRB-7) goes high

NOTES: 1. ↑ indicates positive transition (low to high)
2. ↓ indicates negative transition (high to low)
3. The Interrupt flag bit CRA-7 is cleared by an MPU Read of the A Data Register, and CRB-7 is cleared by an MPU Read of the B Data Register.
4. If CRA-0 (CRB-0) is low when an interrupt occurs (Interrupt disabled) and is later brought high, $\overline{\text{IRQA}}$ ($\overline{\text{IRQB}}$) occurs on the positive transition of CRA-0 (CRB-0).

Control of CA2 and CB2 Peripheral Control Lines (CRA-3, CRA-4, CRA-5, CRB-3, CRB-4, and CRB-5) – Bits 3, 4, and 5 of the two control registers are used to control the CA2 and CB2 Peripheral Control lines. These bits determine if the control lines will be an interrupt input or an output control signal. If bit CRA-5 (CRB-5) is low, CA2 (CB2) is an interrupt input line similar to CA1 (CB1) (Table 4). When CRA-5 (CRB-5) is high, CA2 (CB2) becomes an output signal that may be used to control peripheral data transfers. When in the output mode, CA2 and CB2 have slightly different characteristics (Tables 5 and 6).

TABLE 4 – CONTROL OF CA2 AND CB2 AS INTERRUPT INPUTS
CRA5 (CRB5) is low

CRA-5 (CRB-5)	CRA-4 (CRB-4)	CRA-3 (CRB-3)	Interrupt Input CA2 (CB2)	Interrupt Flag CRA-6 (CRB-6)	MPU Interrupt Request $\overline{\text{IRQA}}$ ($\overline{\text{IRQB}}$)
0	0	0	↓ Active	Set high on ↓ of CA2 (CB2)	Disabled – $\overline{\text{IRQ}}$ remains high
0	0	1	↓ Active	Set high on ↓ of CA2 (CB2)	Goes low when the interrupt flag bit CRA-6 (CRB-6) goes high
0	1	0	↑ Active	Set high on ↑ of CA2 (CB2)	Disabled – $\overline{\text{IRQ}}$ remains high
0	1	1	↑ Active	Set high on ↑ of CA2 (CB2)	Goes low when the interrupt flag bit CRA-6 (CRB-6) goes high

NOTES: 1. ↑ indicates positive transition (low to high)
2. ↓ indicates negative transition (high to low)
3. The Interrupt flag bit CRA-6 is cleared by an MPU Read of the A Data Register and CRB-6 is cleared by an MPU Read of the B Data Register.
4. If CRA-3 (CRB-3) is low when an interrupt occurs (Interrupt disabled) and is later brought high, $\overline{\text{IRQA}}$ ($\overline{\text{IRQB}}$) occurs on the positive transition of CRA-3 (CRB-3).

TABLE 5 – CONTROL OF CA2 AS AN OUTPUT
CRA-5 is high

CRA-5	CRA-4	CRA-3	CA2 Cleared	CA2 Set
1	0	0	Low on negative transition of E after an MPU Read "A" Data operation.	High on an active transition of the CA1 signal
1	0	1	Low immediately after an MPU Read "A" Data operation.	High on the negative edge of the next "E" pulse.
1	1	0	Low when CRA-3 goes low as a result of an MPU Write in Control Register "A".	Always low as long as CRA-3 is low.
1	1	1	Always high as long as CRA-3 is high	High when CRA-3 goes high as a result of a Write in Control Register "A".

TABLE 6 – CONTROL OF CB2 AS AN OUTPUT
CRB-5 is high

CRB-5	CRB-4	CRB-3	CB2 Cleared	CB2 Set
1	0	0	Low on the positive transition of the first E pulse following an MPU Write "B" Data Register operation.	High when the interrupt flag bit CRB-7 is set by an active transition of the CB1 signal
1	0	1	Low on the positive transition of the first E pulse following an MPU Write "B" Data Register operation.	High on the positive transition of the next "E" pulse.
1	1	0	Low when CRB-3 goes low as a result of an MPU Write in Control Register "B".	Always low as long as CRB-3 is low. Will go high on an MPU Write in Control Register "B" that changes CRB-3 to "one".
1	1	1	Always high as long as CRB-3 is high. Will be cleared when an MPU Write Control Register "B" results in clearing CRB-3 to "zero".	High when CRB-3 goes high as a result of an MPU write into control register "B".

Interrupt Flags (CRA-6, CRA-7, CRB-6, and CRB-7) – ; four interrupt flag bits are set by active transitions of ıals on the four Interrupt and Peripheral Status lines en those lines are programmed to be interrupt inputs. These bits cannot be set directly from the MPU Data Bus and are reset indirectly by a Read Peripheral Data Operation on the appropriate section.

S6820I
PERIPHERAL INTERFACE ADAPTER (PIA)

BASIC SYSTEM CONFIGURATION

The Microprocessing Unit (MPU) may be configured with a Read Only Memory (ROM), Random Access Memory (RAM), a Peripheral Interface Adapter (PIA), restart circuitry and clock circuitry to form a minimum functional system (Figure 3). Such a system can easily be adapted for a number of small scale applications by simply changing the content of the ROM.

TWO-PHASE CLOCK CIRCUITRY AND TIMING–The MPU requires a two-phase non-overlapping clock which has a frequency range as high as 1 MHz. In addition to the two phases, this circuit should also generate an enable Signal E, and its complement E, to enable ROMs, RAMs, PIAs and ACIAs. This Enable signal and its complement is obtained by ANDing $\phi 2$ and VMA (Valid Memory Address).

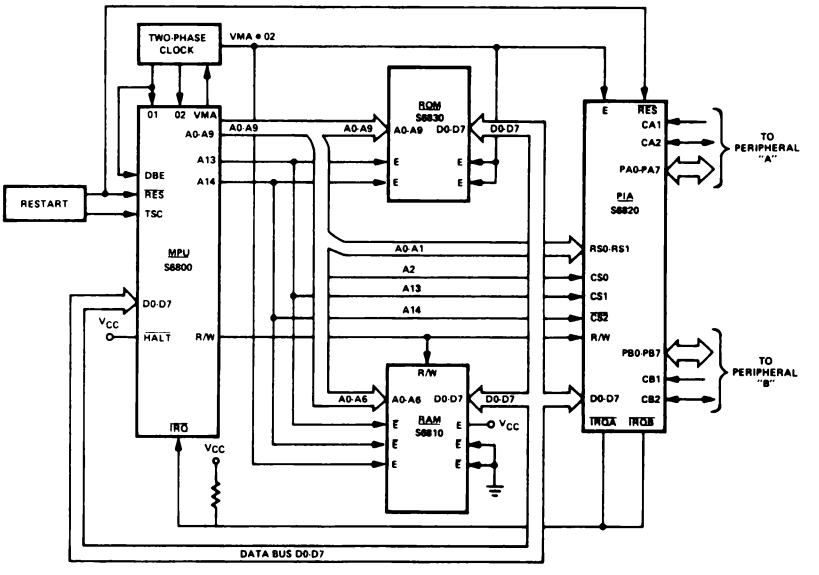
CHIP SELECTION AND ADDRESSING–The Minimum system configuration permits direct selection of the ROM, RAM, ACIA and PIA without the use of special TTL select logic. This is accomplished by simply wiring the address lines A13 A14 to the Enable or chip select lines on the memories and PIA. This permits the devices to be addressed as follows:

Device	A14	A13	Hex Addresses
RAM	0	0	0000–007F
PIA	0	1	2004–2007 (Registers)
ROM	1	1	6000–63FF

Other addressing schemes can be utilized which use any combination of two of the lines A10 through A14 for chip selection.

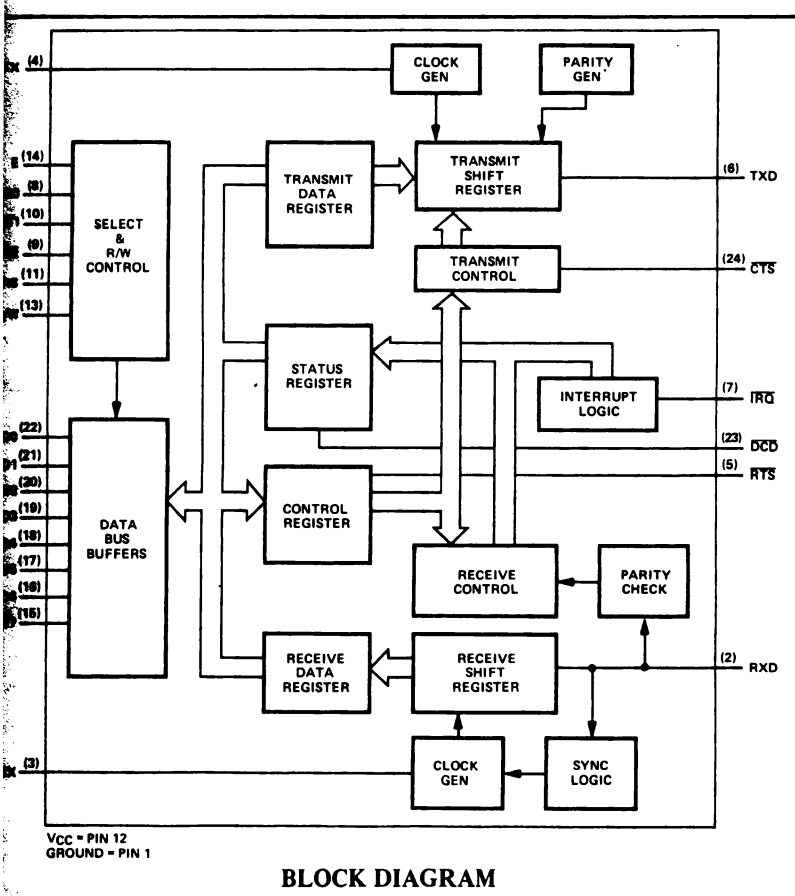
PERIPHERAL CONTROL–All control and timing for the peripherals that are connected to the PIA is accomplished by software routines under the control of the MPU.

FIGURE 3. MINIMUM SYSTEM IMPLEMENTATION

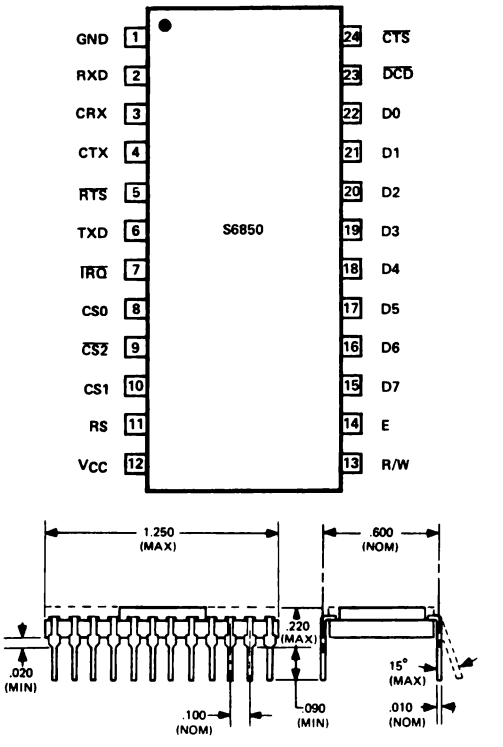


S6850

ASYNCHRONOUS COMMUNICATION INTERFACE ADAPTER (ACIA)



BLOCK DIAGRAM



PIN/PACKAGE CONFIGURATION

(Available in Pkgs. 2L, 1W - see Sec. 1)

FEATURES

- 8 Bit Bidirectional Data Bus for Communication with MPU.
- False start bit deletion.
- Peripheral/modem control functions.
- Double buffered Receiver and Transmitter
- One or two stop bit operation.
- Eight and nine-bit transmission with optional even and odd parity.
- Parity, overrun and framing error checking.
- Programmable control register.
- Optional ÷1, ÷16, and ÷64 clock modes.
- Up to 500,000 bps transmission.

NCTIONAL DESCRIPTION

The S6850 Asynchronous Communications Interface pter (ACIA) provides the data formatting and control to face serial asynchronous data communications to bus nized systems such as the S6800 Microprocessing Unit.

The S6850 includes select enable, read/write, interrupt bus interface logic to allow data transfer over an eight bit bi-directional data bus. The parallel data of the bus system is serially transmitted and received by the asynchronous data interface, with proper formatting and error checking. The functional configuration of the ACIA is programmed via the data bus during system initialization. Word lengths, clock division ratios and transmit control through the Request to Send output may be programmed. For modem operation three control lines are provided. These lines allow the ACIA to interface directly with the S6860 0-600 bps digital modem.

S6850
ASYNCHRONOUS COMMUNICATION INTERFACE ADAPTER (ACIA)

ABSOLUTE MAXIMUM RATINGS

Supply Voltage V_{CC}	−0.3 to +7.0V	Operating Temperature Range T_A	0 to +70°C
Input Voltage V_{in}	−0.3 to +7.0V	Storage Temperature Range T_{stg}	−55 to +150°C

NOTE: This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields, however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high-impedance circuit.

DC (STATIC) CHARACTERISTICS

(V_{CC} = 5.0V ± 5%, T_A = 25°C unless otherwise noted.)

Characteristic	Symbol	Min.	Typ.	Max.	Unit
Input High Voltage (Normal Operating Levels)	V_{IH}	+2.4	–	V_{CC}	Vdc
Input Low Voltage (Normal Operating Levels)	V_{IL}	−0.3	–	+0.4	Vdc
Input High Threshold Voltage All Inputs Except Enable	V_{IHT}	+2.0	-	–	Vdc
Input Low Threshold Voltage All Inputs Except Enable	V_{ILT}	–	–	+0.8	Vdc
Input Leakage Current (V_{in} = 0 to 5.0 Vdc) R/W, RS, CS0, CS1, CS2, Enable	I_{in}	–	1.0	2.5	μAdc
Three-State (Off State) Input Current (V_{in} = 0.4 to 2.4 Vdc, V_{CC} = max) D0-D7,	I_{TSI}	–	2.0	10	μAdc
Output High Voltage (I_{Load} = −100 μAdc, Enable Pulse Width < 25 μs) All Outputs Except $\overline{IRQ}$	V_{OH}	+2.4		–	Vdc
Output Low Voltage (I_{Load} = 1.6 mAdc) Enable Pulse Width < 25 μs	V_{OL}	–	–	+0.4	Vdc
Output Leakage Current (Off State) $\overline{IRQ}$	I_{LOH}	–	1.0	10	μAdc
Power Dissipation	P_D	–	300	525	mW
Input Capacitance (V_{in} = 0, T_A = 25°C, f = 1.0 MHz) D0-D7 R/W, RS, CS0, CS1, $\overline{CS2}$, RXD, $\overline{CTS}$, $\overline{DCD}$, CTX, CRX Enable,	C_{in}	 – – –	 – – 	 10 10	pF
Output Capacitance (V_{in} = 0, T_A = 25°C, f = 1.0 MHz)	C_{out}	–	–	10	pF

S6850
ASYNCHRONOUS COMMUNICATION INTERFACE ADAPTER (ACIA)

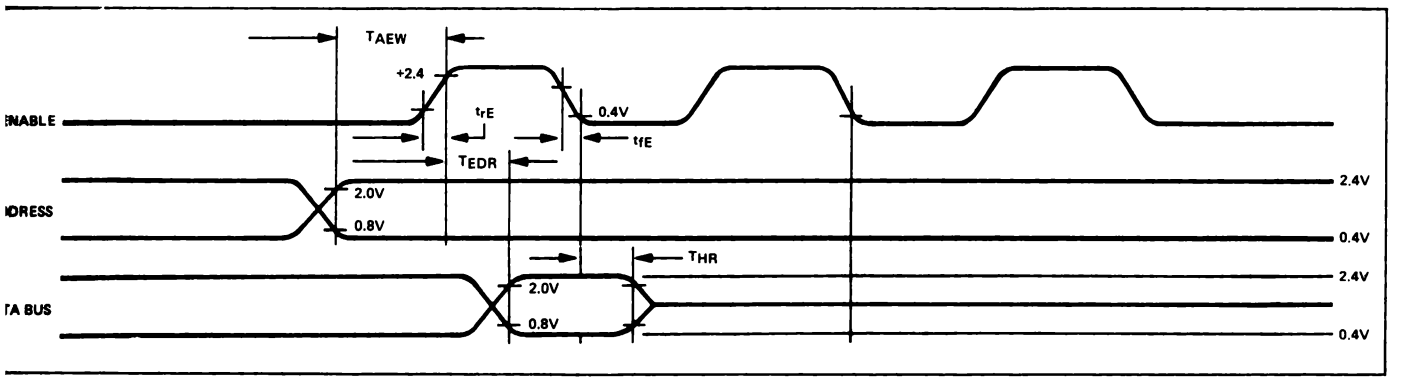
(DYNAMIC) CHARACTERISTICS

ding = 130 pF and one TTL load for D0-D7 = 20pF and 1 TTL load for $\overline{RTS}$ and TXD = 100pF and 3KΩ to V_{CC} for $\overline{IRQ}$.

AD TIMING CHARACTERISTICS (Figure 1)

Characteristic	Symbol	Min.	Typ.	Max.	Unit
Setup Time, Address valid to Enable positive transition	T_{AEW}	180	–	–	ns
Setup Time, Enable positive transition to Data valid on bus	T_{EDR}	–	–	395	ns
Data Bus Hold Time	T_{HR}	10	–	–	ns
Rise and Fall Time for Enable input	t_{rE}, t_{fE}	–	–	25	μs

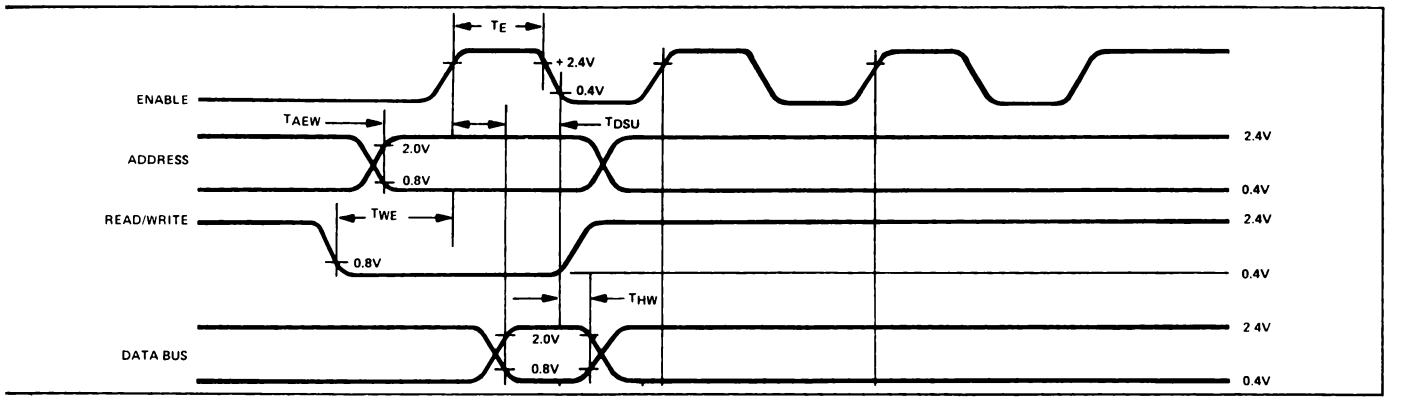
URE 1 – READ TIMING CHARACTERISTICS



ITE TIMING CHARACTERISTICS (Figure 2)

Characteristic	Symbol	Min.	Typ.	Max.	Unit
Enable Pulse Width	T_E	0.470	–	25	μs
Setup Time, Address valid to Enable positive transition	T_{AEW}	180	–	–	ns
Setup Time, Data valid to Enable netative transition	T_{DSU}	300	–	–	ns
Setup time, Read/Write negative transition to Enable positive transition	T_{WE}	130	–	–	ns
Data Bus Hold Time	T_{HW}	10	–	–	ns

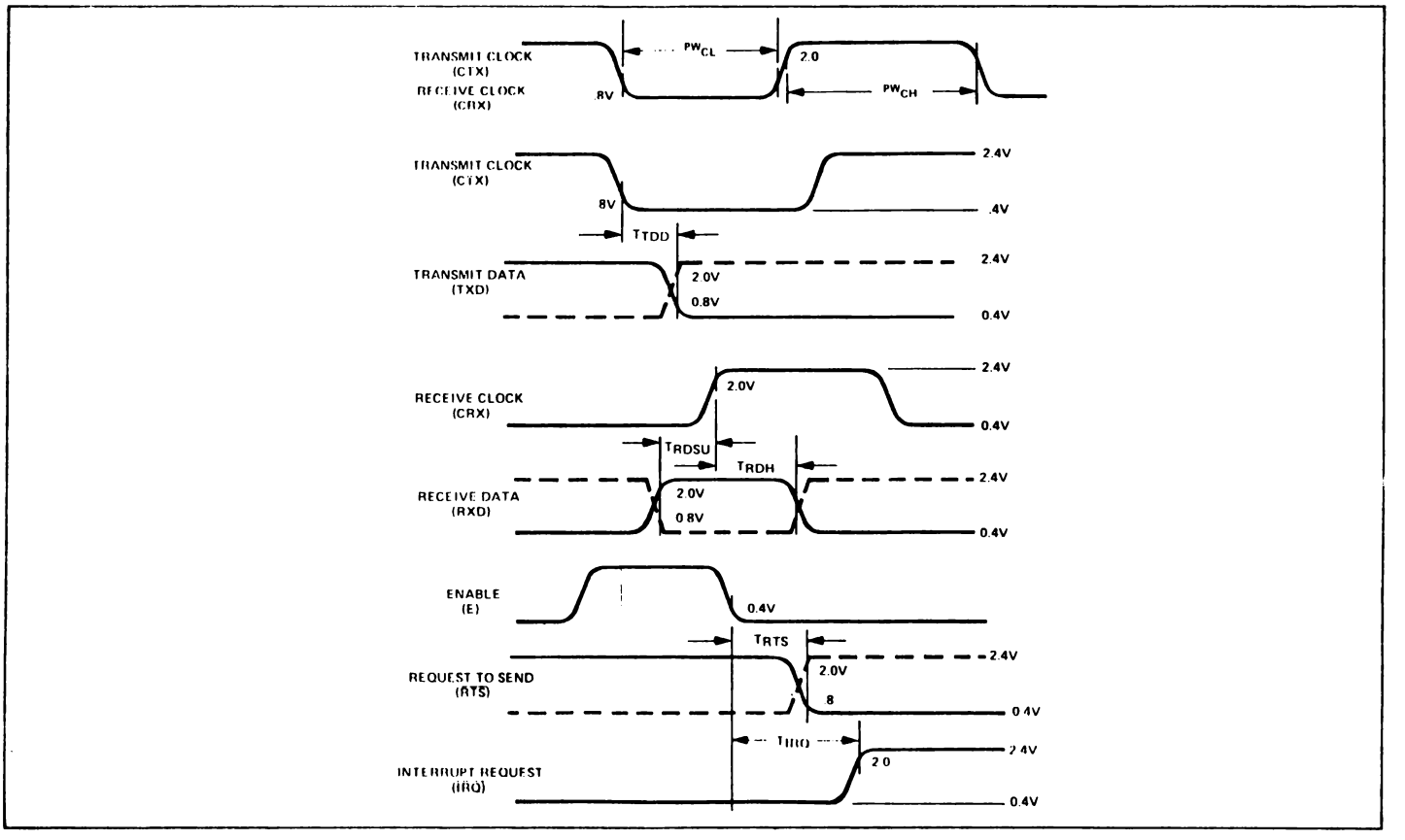
URE 2 – WRITE TIMING CHARACTERISTICS



TRANSMIT/RECEIVE CHARACTERISTICS (Figure 3)

Characteristic	Symbol	Min.	Typ.	Max.	Unit
Clock Frequency ÷ 1 mode ÷ 16 mode ÷ 64 mode	f_C			 500 800 800	 KHz KHz KHz
Clock Pulse Width, Low State	PW_{CL}	600			nsec
Clock Pulse Width, High State	PW_{CH}	600			nsec
Delay Time, Transmit Clock to Data Out	T_{TDD}			1.0	μsec
Set up Time, Receive Data	T_{RDSU}	500			nsec
Hold Time, Receive Data	T_{RDH}	500			nsec
Delay Time, Enable to $\overline{IRQ}$ Reset	T_{IRQ}			1.2	μsec
Delay Time, Enable to RTS	T_{RTS}			1.0	μsec

FIGURE 3 - TRANSMIT/RECEIVE TIMING



S6850
ASYNCHRONOUS COMMUNICATION INTERFACE ADAPTER (ACIA)

/ACIA INTERFACE

Pin	Label	FUNCTION
(22) (21) (20) (19) (18) (17) (16) (15)	D0 D1 D2 D3 D4 D5 D6 D7	**ACIA BI-DIRECTIONAL DATA LINES**–The bi-directional data lines (D0-D7) allow for data transfer between the ACIA and the MPU. The data bus output drivers are three-state devices that remain in the high-impedance (off) state except when the MPU performs an ACIA read operation. The Read/Write line is in the read (high) state when the ACIA is selected for a read operation.
(14)	E	**ACIA ENABLE SIGNAL**–The Enable signal (E) is a high impedance TTL compatible input that enables the bus input/output data buffers and clocks data to and from the ACIA. This signal will normally be a derivative of the S6800 $\phi 2$ clock.
(13)	R/W	**READ/WRITE CONTROL SIGNAL**–The Read/Write line is a high impedance input that is TTL compatible and is used to control the direction of data flow through the ACIA's input/output data bus interface. When Read/Write is high (MPU Read cycle), the ACIA output driver is turned on and a selected register is read. When it is low, the ACIA output driver is turned off and the MPU writes into a selected register. Thus, the Read/Write signal is used to select the Read Only or Write Only registers within the ACIA.
(8) (10) (9)	CS0 CS1 $\overline{CS2}$	**CHIP SELECT SIGNALS**–These three high impedance TTL compatible input lines are used to address an ACIA. A particular ACIA is selected when CS0 and CS1 are high and $\overline{CS2}$ is low. Transfers of data to and from ACIA are then performed under the control of Enable, Read/Write, and Register Select.
(11)	RS	**REGISTER SELECT SIGNAL**–The Register Select line is a high impedance input that is TTL compatible and is used to select the Transmit/Receive Data or Control/Status registers in the ACIA. The Read/Write signal line is used in conjunction with Register Select to select the Read Only or Write Only register in each register pair.
(7)	$\overline{IRQ}$	**INTERRUPT REQUEST SIGNAL**–Interrupt request is a TTL compatible, open drain active low output that is used to interrupt the MPU. The Interrupt Request remains low as long as the cause of the interrupt is present and the appropriate interrupt enable within the ACIA is set.

/MODEM OR PERIPHERAL INTERFACE

Pin	Label	FUNCTION
(4)	CTX	**TRANSMIT CLOCK**–The Transmit Clock is a high impedance TTL compatible input used for the clocking of transmitted data. The transmitter initiates data on the negative transition of the clock. Clock frequency of 1, 16, or 64 times the data rate may be selected.

Pin	Label	FUNCTION
(3)	CRX	**RECEIVE CLOCK**—The Receive Clock is a high impedance TTL compatible input used for synchronization of received data. (In the ÷ 1 mode, the clock and data must be synchronized externally.) The receiver strobes the data on the positive transition of the clock. Clock frequency of 1, 16, or 64 times the data rate may be selected.
(2)	RXD	**RECEIVED DATA**—The Received Data line is a high impedance TTL compatible input through which data is received in a serial NRZ(Non Return to Zero) format. Synchronization with a clock for detection of data is accomplished internally when clock rates of 16 or 64 times the bit rate are used. Data rates are in the range of 0 to 500 Kbps when external synchronization is utilized.
(6)	TXD	**TRANSMIT DATA** The Transmit Data output line transfers serial NRZ data to a modem or other peripheral device. Data rates are in the range of 0 to 500Kbps when external synchronization is utilized.
(24)	$\overline{CTS}$	**CLEAR-TO-SEND**—This high impedance TTL compatible input provides automatic control of the transmitting end of a communications link via the modem's "clear-to-send" active low output by inhibiting the Transmitter Data Register Empty status bit (TDRE).
(5)	RTS	**REQUEST-TO-SEND**—The Request-to-Send output enables the MPU to control a peripheral or modem via the data bus. The active state is low. The Request-to-Send output is controlled by the contents of the ACIA control register.
(23)	$\overline{DCD}$	**DATA CARRIER DETECTED** This high impedance TTL compatible input provides automatic control of the receiving end of a communications link by means of the modem "Data-Carrier-Detect" or "Received-Line-Signal Detect" output. The $\overline{DCD}$ input inhibits and initializes the receiver section of the ACIA when high. A low to high transition of the Data Carrier Detect initiates an interrupt to the MPU to indicate the occurrence of a loss of carrier when the Receiver Interrupt Enable (RIE) is set.
(12)	V_{CC}	**+5 volts ± 5%**
(1)	GND	**GROUND**

S6850
ASYNCHRONOUS COMMUNICATION INTERFACE ADAPTER (ACIA)

PPLICATION INFORMATION

ITERNAL REGISTERS–The ACIA has four internal registers ilized for status, control, receiving data, and transmitting ıta. The register addressing by the R/W and RS lines and the t definitions for each register are shown in Figure 4.

GURE 4 – DEFINITION OF ACIA REGISTERS

	BUFFER ADDRESS			
	RS • $\overline{R/W}$	**RS • R/W**	**$\overline{RS}$ • $\overline{R/W}$**	**$\overline{RS}$ • R/W**
Data Bus Line Number	Transmit Data Register	Receiver Date Register	Control Register	Status Register
	(Write Only)	(Read Only)	(Write Only)	(Read Only)
0	Data Bit 0*	Data Bit 0*	Clk. Divide Sel. (CR0)	Rx Data Reg. Full (RDRF)
1	Data Bit 1	Data Bit 1	Clk. Divide Sel. (CR1)	Tx Data Reg. Empty (TDRE)
2	Data Bit 2	Data Bit 2	Word Sel. 1 (CR2)	Data Carrier Det. loss ($\overline{DCD}$)
3	Data Bit 3	Data Bit 3	Word Sel. 2 (CR3)	Clear-to-Send (CTS)
4	Data Bit 4	Data Bit 4	Word Sel. 3 (CR4)	Framing Error (FE)
5	Data Bit 5	Data Bit 5	Tx Control 1 (CR5)	Overrun (OVRN)
6	Data Bit 6	Data Bit 6	Tx Control 2 (CR6)	Parity Error (PE)
7	Data Bit 7***	Data Bit 7**	Rx Interrupt Enable (CR7)	Interrupt Request (IRQ)

Notes:
* Leading bit = LSB = Bit 0
** Unused data bits in received character will be "0's."
*** Unused data bits for transmission are "don't care's."

S6850
ASYNCHRONOUS COMMUNICATION INTERFACE ADAPTER (ACIA)

ACIA STATUS REGISTER–Information on the status of the ACIA is available to the MPU by reading the ACIA Status Register. This Read Only register is selected when RS is low and R/W is high. Information stored in this register indicates the status of: transmitting data register, the receiving data register and error status and the modem status inputs of the ACIA.

Receiver Data Register Full (RDRF) [Bit 0] – Receiver Data Register Full indicates that received data has been transferred to the Receiver Data Register. RDRF is cleared after an MPU read of the Receiver Data Register or by a Master Reset. The cleared or empty state indicates that the contents of the Receiver Data Register are not current. Data Carrier Detect being high also causes RDRF to indicate empty.

Transmit Data Register Empty (TDRE) [Bit 1]–The Transmit Data Register Empty bit being set high indicates that the Transmit Data Register contents have been transferred and that new data may be entered. The low state indicates that the register is full and that transmission of a new character has not begun since the last write data command.

Data Carrier Detect ($\overline{DCD}$) [Bit 2] – The Data Carrier Detect bit will be high when the $\overline{DCD}$ input from a modem has gone high to indicate that a carrier is not present. This bit going high causes an Interrupt Request to be generated if the Receiver Interrupt Enable (RIE) is set. It remains high until the interrupt is cleared by reading the Status Register and the data register or a Master Reset occurs. If the $\overline{DCD}$ input remains high after Read Status and Read Data or Master Reset have occurred, the $\overline{DCD}$ Status bit remains high and will follow the $\overline{DCD}$ input.

Clear-to-Send ($\overline{CTS}$) [Bit 3] – The Clear-to-Send bit indicates the state of the Clear-to-Send input from a modem. A low $\overline{CTS}$ indicates that there is a Clear-to-Send from the modem. In the high state, the Transmit Data Register Empty bit is inhibited and the Clear-to-Send status bit will be high. Master Reset does not affect the Clear-to-Send status bit.

Framing Error (FE) [Bit 4] – Framing error indicates that the received character is improperly framed by the start and stop bit and is detected by the absence of the 1st stop bit. This error indicates a synchronization error, faulty transmission, or a break condition. The framing error flag is set or reset during the receiver data transfer time. Therefore, this error indicator is present throughout the time that the associated character is available.

Receiver Overrun (OVRN) [Bit 5] – Overrun is an error flag that indicates that one or more characters in the data stream were lost. That is, a character or a number of characters were received but not read from the Receiver Data Register (RDR) prior to subsequent characters being received. The overrun condition begins at the midpoint of the last bit of the second character received in succession without a read of the RDR having occurred. The Overrun does not occur in the Status Register until the valid character prior to Overrun has been read. The RDRF bit remains set until Overrun is reset. Character synchronization is maintained during the Overrun condition. The overrun indication is reset after the reading of data from the Receive Data Register. Overrun is also reset by the Master Reset.

Parity Error (PE) [Bit 6]–The parity error flag indicates that the number of highs (ones) in the character does not agree with the preselected odd or even parity. Odd parity is defined to be when the total number of ones is odd. The parity error indication will be present as long as the data character is in the RDR. If no parity is selected, then both the transmitter parity generator output and the receiver parity check results are inhibited.

Interrupt Request ($\overline{IRQ}$) [Bit 7]–The IRQ bit indicates the state of the $\overline{IRQ}$ output. Any interrupt that is set and enabled will be indicated in the status register. Any time the $\overline{IRQ}$ output is low the IRQ bit will be high to indicate the interrupt or service request status.

CONTROL REGISTER–The ACIA control Register consists of eight bits of write only buffer that are selected when RS and R/W are low. This register controls the function of the receiver, transmitter, interrupt enables, and the Request-to-Send modem control output.

Counter Divide Select Bits (CR0 and CR1)–The Counter Divide Select Bits (CR0 and CR1) determine the divide ratios utilized in both the transmitter and receiver sections of the ACIA. Additionally, these bits are used to provide a Master Reset for the ACIA which clears the Status Register and initializes both the receiver and transmitter. Note that after a power-on or a power-fail restart, these bits must be set High to reset the ACIA. After resetting, the clock divide ratio may be selected. These counter select bits provide for the following clock divide ratios:

CR1	CR0	Function
0	0	÷ 1
0	1	÷16
1	0	÷64
1	1	Master Reset

S6850
ASYNCHRONOUS COMMUNICATION INTERFACE ADAPTER (ACIA)

Word Select Bits (CR2, CR3, and CR4)—The Word Select bits are used to select word length, parity, and the number of stop bits. The encoding format is as follows:

CR4	CR3	CR2	Function
0	0	0	7 Bits + Even Parity + 2 Stop Bits
0	0	1	7 Bits + Odd Parity + 2 Stop Bit
0	1	0	7 Bits + Even Parity + 1 Stop Bit
0	1	1	7 Bits + Odd Parity + 1 Stop Bit
1	0	0	8 Bits + 2 Stop Bits
1	0	1	8 Bits + 1 Stop Bit
1	1	0	8 Bits + Even Parity + 1 Stop Bit
1	1	1	8 Bits + Odd Parity + 1 Stop Bit

Word length, Parity Select, and Stop Bit changes are not double-buffered and therefore become effective immediately.

Transmitter Control Bits (CR5 and CR6)—Two Transmitter Control bits provide for the control of the Transmitter Buffer Empty interrupt output, the Request-to-Send output and the transmission of a BREAK level (space). The following encoding format is used:

CR6	CR5	Function
0	0	$\overline{RTS}$ = low, Transmitting Interrupt Disabled
0	1	$\overline{RTS}$ = low, Transmitting Interrupt Enabled
1	0	$\overline{RTS}$ = high, Transmitting Interrupt Disabled
1	1	$\overline{RTS}$ = low, Transmitting Interrupt Disabled and Transmits a BREAK level on the Transmit Data Output.

Receiver Interrupt Enable Bit (RIE) (CR7)--Interrupts will be enabled by a high level in bit position 7 of the Control Register (CR7). Interrupts caused by the Receiver Data Register Full being high or by a low to high transition on the Data Carrier Detect signal line are enabled or disabled by the Receiver Interrupt Enable Bit.

TRANSMIT DATA REGISTER (TDR)—Data is written in the Transmit Data Register *during* the peripheral enable time (E) when the ACIA has been addressed and RS · R/W is selected. Writing data into the register causes the Transmit Data Register Empty bit in the status register to go low. Data can then be transmitted. If the transmitter is idling and no character is being transmitted, then the transfer will take place within one bit time of the trailing edge of the Write command. If a character is being transmitted, the new data character will commence as soon as the previous character is complete. The transfer of data causes the Transmit Data Register Empty (TDRE) bit to indicate empty.

RECEIVE DATA REGISTER (RDR)—Data is automatically transferred to the empty Receive Data Register (RDR) from the receiver deserializer (a shift register) upon receiving a complete character. This event causes the Receiver Data Register Full bit (RDRF) (in the status buffer) to go high (full). Data may then be read through the bus by addressing the ACIA and selecting the Receiver Data Register with RS and R/W high when the ACIA is enabled. The non-destructive read cycle causes the RDRF bit to be cleared to empty although the data is retained in the RDR. The status is maintained by RDRF as to whether or not the data is current. When the Receiver Data Register is full, the automatic transfer of data from the Receiver Shift Register to the Data Register is inhibited and the RDR contents remain valid with its current status stored in the Status Register.

OPERATIONAL DESCRIPTION

From the MPU Bus interface the ACIA appears as two addressable RAM memory locations. Internally, there are four registers; two read-only and two write-only registers. The read-only registers are status and receive data, and the write only registers are control and transmit data. The serial interface consists of serial transmit and receive lines and three modem/peripheral control lines.

During a power-on sequence, the ACIA is internally latched in a reset condition to prevent erroneous output transitions. This power-on reset latch can only be released by the master reset function via the control register; bits b_0 and b_1 are set "high" for a master reset. After master resetting the ACIA, the programmable control register can be set for a number of options such as variable clock divider ratios, variable word length, one or two stop bits, parity (even, odd, or none) and etc.

TRANSMITTER—A typical transmitting sequence consists of reading the ACIA status register either as a result of an interrupt or in the ACIA's turn in a polling sequence. A character may be written into the Transmitter Data Register if the status read operation has indicated that the Transmit Data

Register is empty. This character is transferred to a shift register where it is serialized and transmitted from the Tx Data output preceded by a start bit and followed by one or two stop bits. Internal parity (odd or even) can be optionally added to the character and will occur between the last data bit and the first stop bit. After the first character is written in the data register, the status register can be read again to check for a Transmit Data Register Empty condition and current peripheral status. If the register is empty, another character can be loaded for transmission even though the first character is in the process of being transmitted. This second character will be automatically transferred into the shift register when the first character transmission is completed. The above sequence continues until all the characters have been transmitted.

RECEIVER--Data is received from a peripheral by means of the Rx Data input. A divide by one clock ratio is provided for an externally synchronized clock (to its data) while the divide by 16 and 64 ratios are provided for internal synchronization. Bit synchronization in the divide by 16 and 64 modes is obtained by the detection of the leading mark-to-space transition of the start bit. False start bit deletion capability insures that a full half bit of a start bit has been received before the internal clock is synchronized to the bit time. As a character is being received, parity (odd or even) will be checked and the error indication will be available in the status register along with framing error, overrun error, and receiver data register full. In a typical receiving sequence, the status register is read to determine if a character has been received from a peripheral. If the receiver data register is full, the character is placed on the 8-bit ACIA bus when a Read Data command is received from the MPU. The status register can be read again to determine if another character is available in the receiver data register. The receiver is also double buffered so that a character can be read from the data register as another character is being received in the shift register. The above sequence continues until all characters have been received.