oettle & reichler
datentechnik

Schießgrabenstr. 28a
8900 Augsburg 1

Tel.: (0821) 15 46 32

C P U

Zentrale Rechnerbaugruppe

H A N D B U C H

Dezember 1983

Inhaltsverzeichnis

Schaltungsbeschreibung

Die CPU-Baugruppe eignet sich für den universellen Einsatz als Zentraleinheit in Geräten aller Größen vom Stand-alone System für Steuerungsaufgaben bis hin zum beliebig erweiterbaren Großsystem.
Mit einer segmentierten Speicherverwaltungseinheit und einem Arithmetikprozessor weist die CPU-Baugruppe Strukturen modernster Rechnereinheiten auf.

Das Betriebssystem CP/M wird durch eine Vielzahl von I/O-Schnittstellen mit den Peripherieeinheiten verbunden. Insbesondere das Betriebssystem CP/M 3.0 wird durch die Echtzeituhr und die segmentierte Speicherverwaltungseinheit optimal unterstützt.

Besonderen Wert wurde auf die bidirektionale BUS-Pufferung gelegt. Die CPU Karte läßt sich somit nicht nur als Master, sondern auch als Slave oder I/O Baugruppe einsetzen.

Insbesondere der Einsatz als I/O und RAM/ROM Baugruppe ist äußerst hilfreich bei der Entwicklung von Steuereinheiten mit der CPU Karte. Die Software läßt sich somit komfortabel und zeitsparend auf einem ECB Entwicklungssystem entwickeln und testen.

Stromversorgung

Die Karte benötigt zwei Betriebsspannungen: +5 und +12 Volt. Die +12 V Versorgung ist nur für die RS-232 Schnittstelle und den Arithmetikprozessor erforderlich. Die für die RS-232 notwendigen -12 V werden von einem Spannungsinverter ICL 7660 (IC 25) erzeugt. Dem RS-232 Treiber werden somit ca. +/- 9 Volt bereit gestellt.

Takt

Auf der Karte befinden sich 2 Quarzoszillatoren, aufgebaut aus einem LS 04 Baustein und zwei Quarzen. Der Baudrate Takt wird über einen LS 393 Zähler von der Quarzfrequenz abgeleitet. Teilerfaktoren von 2, 4, 8 und 16 sind möglich (Grundeinstellung 4).

Der zweite Oszillator schwingt auf doppelter Systemfrequenz, die am BUS-Stecker an Pin 16a (2xClk) zur Verfügung gestellt wird. Nach Teilung durch 2 (LS 393) und Pufferung über eine symmetrische Transistorstufe wird daraus der Systemtakt abgeleitet. Der Ausgang QB des LS 393 versorgt den Arithmetikprozessor mit halber Systemfrequenz.

Reset

Ein Power-up Reset wird über ein RC-Gliedes und zwei LS 14 (IC 8) Gatter erzeugt und über eine Open-Collector-Stufe auf PWRCLR (Pin 26c am BUS-Stecker) geführt. Dieses Signal wird zum Zurücksetzen der CPU, STI, DART, AM9511 und 8255 benutzt. Der Eingang RESET am BUS (31c) kann zum Anschluß eines Reset Schalters verwendet werden.

Wait

Die Wait Schaltung (IC 7) kann bei I/O- und bei Speicherzugriffen Wait-Staits einfügen. Normalerweise wird ein Wait-Stait (1 Takt-Zyklus) eingefügt. Prinzipiell sind aber auch zwei bzw. drei Wait-Staits möglich. Mittels des I/O PROM's wird festgelegt, bei welchem der insgesamt 256 I/O Adressen ein Wait-Zyklus generiert wird. Der Memory PROM erlaubt es jedem 4 K-Byte Speicherblock innerhalb des 1 M-Byte Speicherbereichs ein Wait zuzuordnen.

Standardmäßig wird bei jedem I/O-Zugriff und bei Speicherzugriffen auf den Boot-Eprom jeweils ein Wait-State eingefügt. Speicherzugriffe auf externe Speicherbaugruppen (dynamischer RAM) erzeugen keine Wait's, um den Systemdurchsatz nicht unnötig herabzusetzen.

Bussteuerung

Die Steuerung des Daten- und Adreßbus werden von einem 256x4 PROM (IC 15) durchgeführt. In Verbindung mit der bidirektionalen BUS-Pufferung ermöglicht dies einen vielseitigen Einsatz der CPU-Baugruppe.

Signalbelegung des BUS PROMs IC 15 (24 S 10):

A7	Pin 15	BUSAK	D3	Pin 9	EN 367
A6	Pin 1	RD	D2	Pin 10	DIR Adress
A5	Pin 2	IEI	D1	Pin 11	DIR Data
A4	Pin 3	IORQ	D0	Pin 12	EN System BUS
A3	Pin 4	Memory Select			
A2	Pin 7	M1			
A1	Pin 6	I/O Select			
A0	Pin 5	IEO			

Memory Select

Zur Adreßdekodierung der beiden 28 poligen Byte-Wide Sockel wird ein 256 x 4 PROM (IC 05) verwendet. Durch entsprechende Programmierung kann die Adreßlage und Größe der beiden Speicher bestimmt werden. Die Adreßlage beschränkt sich dabei auf das untere halbe M-Byte des maximal möglichen Adreßraums von 1 M-Byte. Die Größe kann aus 4 kByte Blöcken zusammengesetzt werden. Der BOOT-Eingang am PROM erlaubt das Ein- und Ausschalten einzelner Speicherbereiche über die System-Software. Der Ausgang M-SEL ist die ODER Verknüpfung der beiden CE-Signale und dient zur BUS-Steuerung. Dieses Signal wird auch als Open-Collector-Ausgang an den BUS-Stecker DESLCT (Pin 26a) geführt. Der Ausgang WAIT erlaubt die Generierung von Wait-Staits.

Signalbelegung des Memory PROMs IC 05 (24 SA 10):

A7	Pin 15	BOOT	D3	Pin 9	WAIT Request
A6	Pin 1	A12/BUSAK	D2	Pin 10	Memory Select
A5	Pin 2	A14	D1	Pin 11	CE IC 13
A4	Pin 3	A13	D0	Pin 12	CE IC 12
A3	Pin 4	A15			
A2	Pin 7	A16			
A1	Pin 6	A17			
A0	Pin 5	A18+A19			

I/O Select

Die Dekodierung der I/O-Bausteine übernimmt ein 512 x 8 PROM (IC 02). Mit ihm läßt sich jeder einzelne I/O-Baustein beliebig im maximalen I/O Adreßbereich (max. 256) anordnen. In Abhängigkeit der neun Eingangssignale M1 und A0 -A7 werden sieben Ausgangssignale erzeugt. Davon werden fünf direkt als Chip Enable Signale für die I/O-Bausteine verwendet. Das Signal I/O-SEL ist die ODER-Verknüpfung der CE-Signale und dient zur BUS-Steuerung. Das Ausgangssignal WAIT ist an die WAIT-Schaltung geführt und dient zur Anforderung von WAIT-States bei I/O-Zugriffen.

Signalbelegung des I/O PROMs IC 02 (TBP 28 SA 42):

A8	Pin 19	A6	D7	Pin 14	CE DART
A7	Pin 18	A7	D6	Pin 13	unbelegt
A6	Pin 17	A1	D5	Pin 12	CE 8255
A5	Pin 16	A2	D4	Pin 11	I/O Select
A4	Pin 5	A3	D3	Pin 9	WAIT Request
A3	Pin 4	A4	D2	Pin 8	CE 9511
A2	Pin 3	A5	D1	Pin 7	CE MMU
A1	Pin 2	A0	D0	Pin 6	CE STI
A0	Pin 1	M1			

Standardadreßbelegung

Standardmäßig belegt das CPU-Modul folgende 27 I/O-Addressen:

50h - 5Fh STI Serial Timer Interrupt Controller
50h Indirect Data Register
...
5Fh Usart Data Register

60h - 63h DART Dual Asynchronous Receive/Transmitter
60h Channel A Data
61h Channel A Ctr
62h Channel B Data
63h Channel B Ctr

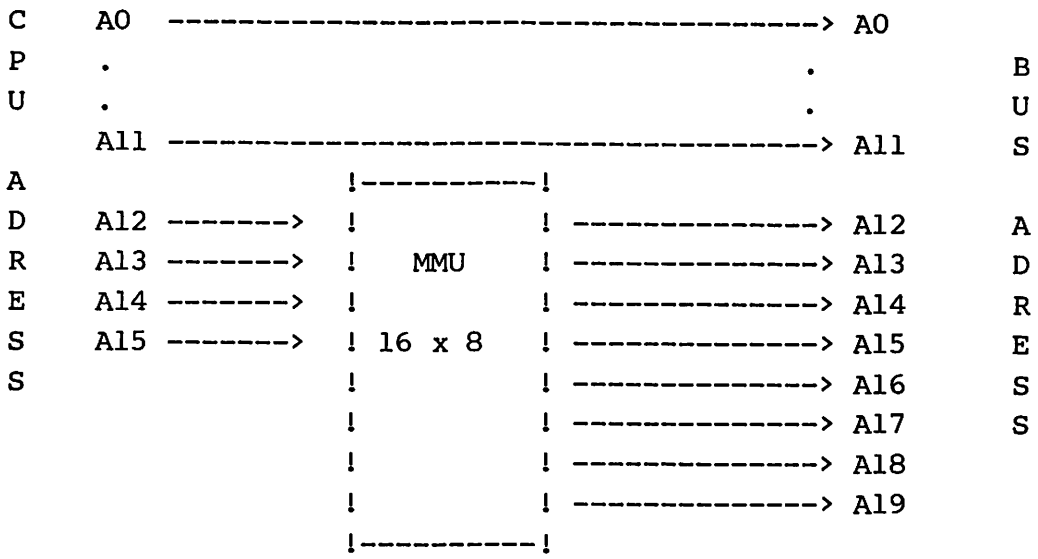
64h - 67h PIO 8255 Peripheral I/O Controller
64h Port A
65h Port B
66h Port C
67h Control Port

68h - 69h APU 9511/12 Arithmetik Processor
68h Data-Port
69h Command/Control Port

6Ah MMU Memory Management Unit

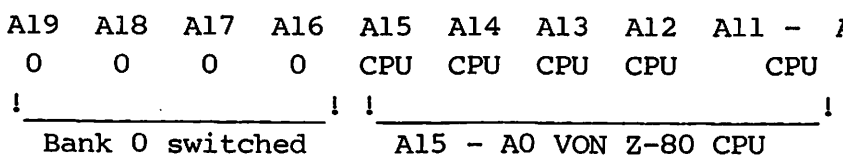
Segmentierte Speicher-Verwaltung

Die segmentierte MMU erlaubt es der CPU 1 M-Byte Speicher direkt zu adressieren. Der maximale physikalische Adreßraum wird dabei in 256 vier kByte große Segmente aufgeteilt. Durch Beschreiben eines 16 Byte großen Speichers über I/O wird der 64 K-Byte große CPU Adreßraum definiert. Durch einen I/O-Befehl ist es somit möglich ein beliebiges 4 k-Byte Speichersegment aus dem 1 M-Byte Hintergrundspeicher an beliebige Stelle in den CPU-Adreßraum zu schalten. Ein aktiv werden des CPU-Signals BUSAK (z.B. bei DMA-Transfers) setzt die Ausgänge A12 - A19 in den Tri-State-Zustand.



MMU nach dem Reset

Nach einem Reset ist die MMU deaktiviert. Das Ausgangssignal IO ("MMU") des STI Bausteins führt, bedingt durch den Pull-Up-Widerstand R2, High Pegel. Dies versetzt die beiden 16x4 TTL RAMs (IC 10 und IC 9) in den Tri-State Zustand. Stellvertretend fÜr die MMU erzeugt IC 04 (74 LS 245) die obersten acht Adressen A12 - A19. A16 - A19 führen Low Pegel und A12 - A15 der CPU werden direkt auf den System-BUS geführt. D.h. die unterste 64 K-Byte Page wird angesprochen.



Programmieren der MMU

Bevor die MMU aktiv geschaltet werden kann, muß der 16 Byte große MMU Speicher (IC 9,10) beschrieben werden. Bei der Adressierung der 16 Bytes wird ausgenützt, daß die Z-80 CPU bei indirekten I/O-Befehlen (OUTP X) den Inhalt des Registers B auf die oberen acht Adressen A8 - A15 legt, den Inhalt des Registers C auf die unteren acht Adressen A7 - A0. Da A0 - A3 der MMU-Speicher mit A12 - A15 der CPU verbunden sind, ist es möglich, mit den oberen vier Bits des B Registers eines der 16 MMU-Register anzuwählen. Bei einem Outp-Befehl an die MMU adressieren die 4 High-Bits des B-Registers 1 der 16 MMU-Register (4k Segment). Die beim Output-Befehl ausgegebene Daten bestimmen den Inhalt (A19 - A12) des MMU-Registers. Dabei ist zu beachten, daß die Daten durch die MMU-Speicher invertiert auf dem Adressbus erscheinen.

MMU aktivieren

Sind alle 16 MMU-Register beschrieben, so kann die MMU über den Ausgang IO ("MMU") des STI Bausteins aktiviert werden. Führt das Signal "MMU" Low Pegel, geht IC 04 in den Tri-State Zustand und die Ausgänge (A12 - A19) der MMU werden aktiv geschaltet.

Ein Segment in den CPU Adressraum schalten

Die oberen vier Bits des B-Registers bestimmen, welches 4 kByte Speichersegment des 64 K-Byte CPU-Adressraum umgeschaltet wird. 00H im B Register entspricht z.B. dem CPU Adressraum 0000H - 0FFFH und F0H in B dem Adressraum F000H - FFFFH. Die Daten des OUTP-Befehl bestimmen, welches der ingesamt 256 4-kByte Segmente des maximalen physikalischen Adressraums von einem M-Byte an die zuvor bestimmte Position geschaltet werden soll. Dabei ist der invertierte Wert auszugeben. Das Beschreiben des MMU Speichers mit FFH hätte zur Folge, daß das unterste 4 kByte Segment 00000H - 00FFFH des physikalischen Adressraums an die über das B Register bestimmte Position im CPU Adressraum geschaltet wird. Die Ausgabe von 00H würde das oberste physikalische Speicher Segment FF000H - FFFFFH ansprechen.

```
LD C,MMUPORT    ; C ADRESSIERT MMUPORT (06AH)
LD B,40         ; SEGMENT 4 VON 4000H - 4FFFH WIRD UMGESCHALTET
LD A,64         ; BANK 6, 7000H - 7FFFH WIRD EINGESCHALTET
CPL             ; COMPLEMENT DATA
OUT (C),A       ; PROGRAMMIEREN MMU
```


Echtzeit Uhr

Der Uhrenbaustein erlaubt die Angabe von Sekunde, Minute, Stunde, Tag, Wochentag und Monat. Ein externer Akku ermöglicht den Betrieb durch Versorgung über die VCMOS-Leitung auch bei ausgeschaltetem Gerät. Die Steuersignale des Uhrenbausteins werden über den 8255 I/O-Baustein (IC 22) beschaltet. Der Datenaustausch geschieht dabei seriell über die beiden Leitungen DATA IN und DATA OUT. Ein Time-Pulse der an die STI geführt wird erlaubt es ein Zeit-Raster per Interrupt über die System Software zu legen.

Der Abgleich des Uhrenoszillators geschieht über die beiden Kondensatoren C19 und C23. C23 ist gesondert einzubauen. C19 wird zum einfacheren Abgleich gesockelt (gedrehte Fassung) eingebaut.

STI Serial Timer Interrupt Controller

Der STI-Baustein stellt 8 parallele I/O-Kanäle, 1 Usart und 4 Timer-Zeitgeber zur Verfügung.

1. Pinbelegung parallel I/O-Lines I0 - I7:

Speicherverwaltung			
I0	Output	MMU, low active	Ein MMU bei L, aus bei H
Centronics			
I1	Input	Paper Out, high active	
I2	Input	Busy, high active	
I3	Input	Acknowledge, low active	
Echtzeit Uhr			
I4	Input	Data Out	
I5	Input	Time Pulse	Timer A Input fÜr Zeitraster
BOOT			
I6	Output	BOOT,high active	Ein Boot-Eprom bei H, aus bei L
Arithmetikprozessor			
I7	Input	END, low active	End of execution APU

2. Timer A-D:

TA	Pin 1	Output	NC
TB	Pin 2	Output	Baudrate for STI USART
TC	Pin 3	Output	Baudrate for DART Channel B
TD	Pin 4	Output	Baudrate for DART Channel A

Der STI Baustein ist derzeit nur in der A Version (4 MHz) erhältlich. Es ist deshalb bei einer Systemfrequenz von 6 MHz nicht möglich, die Vektor Interrupt Fähigkeiten des Bausteins auszunutzen.

PIO 8255 Parallel Interface Baustein

Die I/O Leitungen des 8255 Bausteins sind wie folgt beschaltet:

Centronics

PA 0	Pin 4	I/O	Data 0	
'		'	'	
PA 7	Pin 37	I/O	Data 7	
PB 3	Pin 21	Output	Strobe	Low active
PB 4	Pin 22	Output	Direction Data	Low = Output

Echtzeit Uhr

PB 0	Pin 18	Output	C2
PB 1	Pin 19	Output	DATA IN
PB 2	Pin 20	Output	CLK
PB 6	Pin 24	Output	C0
PB 7	Pin 25	Output	C1

8 Bit Parallel Extension Port

PC 0	Pin 14	I/O	P 2
'		'	'
PC 7	Pin 10	I/O	P 9

Schnittstellen

1. RS-232

Es stehen zwei serielle asynchrone vollduplex Kanäle zur Verfügung. Alternativ ist durch Austausch der Z80-DART durch eine Z80-SIO-0 auch synchrone Datenübertragung möglich. Die Baudrate ist softwaremäßig getrennt für beide Kanäle einstellbar. Channel A der DART wird von Timer D der STI und Channel B von Timer C mit dem Baudrate Takt versorgt.

Die Signale Receive und Transmit Data, Clear to Send und Request to Send sind bei beiden Kanälen auf RS-232 Pegel gepuffert. Die Signale Data Carrier Detect, Data Terminal Ready und Ring Indicator sind direkt auf den 26 poligen Stecker geführt. Die Belegung dieses Steckers entspricht der RS-232 Norm. Jeweils 13 Leitungen können in Schneid-Klemm-Technik direkt auf einen 25 poligen D-Stecker geführt werden. Es ist dabei jedoch besonders auf die beiden Signale Data Carrier Datect und Ring Indicator zu achten. Eine Beschaltung mit RS-232 12 Volt Pegel würde zur Zerstörung der Z80-DART führen.

Belegung RS 232:

Channel A:	14	Receive Data	RS-232	I
	16	Transmit Data	RS-232	O
	18	Request to Send	RS-232	O
	19	Ring Indicator	5 Volt TTL	I
	20	Clear to Send	RS-232	I
	23	Data Terminal Ready	5 Volt TTL	O
	24	GND		
	26	Data Carrier Detect	5 Volt TTL	I
Channel B:	1	Receive Data	RS-232	I
	3	Transmit Data	RS-232	O
	5	Request to Send	RS-232	O
	6	Ring Indicator	5 Volt TTL	I
	7	Clear to Send	RS-232	I
	10	Data Terminal Ready	5 Volt TTL	O
	11	GND		
	13	Data Carrier Detect	5 Volt TTL	I

2. Centronics

Der 8-Bit Datenstrom wird über einen bidirektionalen Buffer (LS 245) gepuffert. Der Direction-Eingang des Puffers ist softwaremäßig beschaltbar. Es ist somit möglich die 8 Parallelsignale für andere Anwendungen auch als Eingänge zu beschalten. Der Strobe Ausgang ist über einen LS 367 gepuffert. Die drei Eingänge Acknowledge, Busy und Paper Empty sind an die STI geführt und können somit Interrupts auslösen. Sämtliche Signale sind auf einen 26 poligen Stecker nach Centronics Norm geführt.

Belegung:		
	1	Strobe
	3-17	D0 - D7
	19	Acknowledge
	21	Busy
	23	Paper Empty

3. Seriell TTL

Eine weitere asynchrone Schittstelle stellt der STI-Baustein zur Verfügung. Sie wird auf einem 10-poligen Stecker ungepuffert bereitgestellt. Die Baudrate ist auch hier softwaremäßig einstellbar (STI Timer B). Die Schnittstelle dient zum Anschluß einer Tastatur oder ähnlicher TTL-Schnittstellen.

Belegung:				
	2	Vcc	3	GND
	4	NMI	5	GND
	6	GND	7	GND
	8	Seriell in	9	GND
	10	Seriell out		

4. 8-Bit parallel

Der Port C der PIO ist auf einen 10 poligen Stecker geführt. Dieser Port kann in 2x4 Leitungen aufgespalten werden, jeweils vier Signale können als Ein- oder Ausgang programmiert werden.

Belegung:		
	1	GND
	2	D0
	:	:
	8	D6
	9	D7
	10	Vcc

Steckbrücken

Steckbrücke CLK

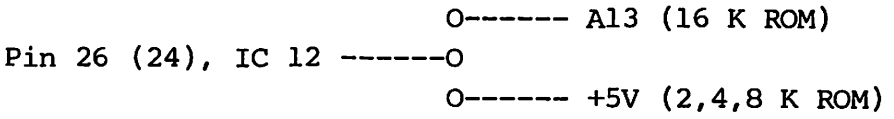
Der Systemtakt läßt sich wahlweise auch extern zuführen. Ist die CLK-Verbindung geöffnet, so ist der auf der CPU-Baugruppe befindliche Taktgenerator deaktiviert und der Systemtakt kann extern zugeführt werden.

Steckbrücke BUSAK

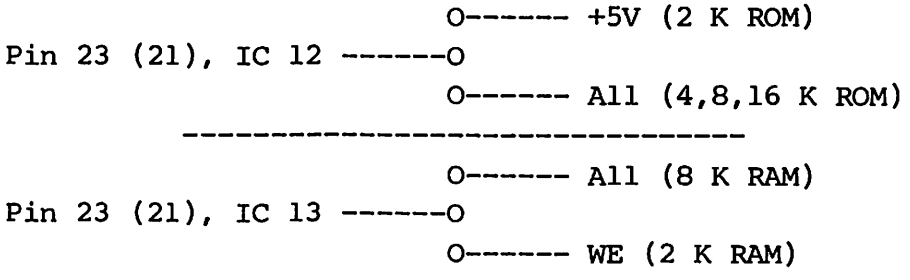
Das BUSAK-Signal der CPU ist kein Tri-State-Signal. Um die CPU Baugruppe zum Beispiel als I/O Karte einsetzen zu können, ist es möglich, das Signal BUSAK vom System-BUS abzutrennen (Brücke nicht gesteckt).

Byte Wide Steckbrücken

Um die beiden 28 poligen Byte-Wide Fassungen (IC 12, 13) sind 9 Steckkontakte angeordnet, die je nach Art der verwendeten Speicher beschaltet werden müssen. Auf den drei gedrehten Fassungsstiften zwischen IC 12 und IC 14 läßt sich Pin 26 des ICs 12 wahlweise mit +5V oder A13 beschalten. Pin 26 (24) ist nur bei 16 K-Byte EPROMs mit A13 zu verbinden.



Auf den 6 Pfosten am Kartenrand neben IC 13 läßt sich Pin 23 (21) der beiden Speicher wahlweise mit Write, A11 oder +5V beschalten.



Einsatzgebiete:

Dank der bidirektionalen BUS-Pufferung und der Steuerung durch einen PROM läßt sich die CPU Baugruppe in einer Vielzahl von Anwendungen einsetzen.

Einsatz als I/O und RAM/ROM Karte

Mit Ausnahme der Speicher Verwaltungs Einheit sind alle auf der CPU-Karte befindlichen Baugruppen auch von einer externen CPU aus ansprechbar. D. h. die CPU-Baugruppe kann auch in einem bestehenden System als I/O-Karte eingesetzt werden. Dies ermöglicht die einfache Entwicklung von Systemsoftware für Spezialsysteme mit der CPU-Karte. Während der Testphase wird die CPU-Baugruppe einfach in ein bestehendes System eingesteckt und als I/O Karte angesprochen. Unter zu Hilfenahme von komfortablen Softwarewerkzeugen (Pascal, Assembler, Debugger, usw.) kann der Test, ein sonst sehr kritischer Vorgang, unter der vollen Kontrolle des Hauptsystems sicher durchgeführt werden.

Beim Einsatz als I/O Karte sind folgende Punkte zu beachten:

1, Der Systemtakt ist extern zuzuführen. Dh. der interne Taktgenerator muß durch Entfernen der Steckbrücke deaktiviert werden.
2, Der BUSAK-Ausgang auf den BUS ist durch Entfernen der entsprechenden Steckbrücke zu unterbrechen.
3, Der CPU-Baustein muß aus entnommen und durch eine Brücke von GND (Pin 29) nach BUSAK (Pin 23) ersetzt werden.
4, Das PWRCLR-Signal auf dem Systembus muß als Open-Collector ausgeführt sein.
5, Der zweifache Systemtakt 2xCLK (BUS 16a) darf nicht im Hauptsystem erzeugt werden, oder muß auf der CPU-Karte durchtrennt werden.

Einsatz als Slave Prozessor

Ebenfalls kann die CPU-Karte als Slave in einem Multiprozessor oder Muli-User-System eingesetzt werden. Die Karte arbeitet dabei parallel zu anderen Prozessoren in dem System ohne die Aktivitäten des Masters auf dem Systembus zu stören. Der Datenaustausch

zwischen Slave und Master kann dabei mit der vollen Geschwindigkeit des Systembuses erfolgen. Der Datenaustausch zwischen Slave und Master erfolgt über einen RAM-Baustein auf der CPU-Karte (28-poliger JEDEC-Sockel).

Folgende Änderungen sind hierbei vorzunehmen:

1, Pin 1 des Memory-PROMs wird mit BUSAK beschaltet.
2, Ein spezieller Memory-PROM ist einzusetzen.
3, Ein spezieller Bus-PROM ist einzusetzen.
4, Die Leitungen IEI, IEO, 2xCLK, CLK , DESLCT, WAIT, INT, NMI sind nicht auf den MASTER-Bus zu schalten.
5, Die PWRCLR-Leitung muß als Open-Collector ausgeführt sein.
6, BUSREQ und BUSAK sind ebenfalls nicht auf den BUS zu legen, sondern über zwei parallele I/O-Leitungen mit dem Master zu verbinden:
Die BUSREQ Leitung dient dann als Slave-Request-Leitung, die BUSAK-Leitung als Slave-Acknowledge-Signal. Eine I/O-Leitung auf der CPU-Karte kann mit einer I/O-Leitung des Masters verbunden werden, die das Master-Request-Signal führt.

Es wird empfohlen bei mehreren Slave Baugruppen im System eine spezielle Rückwandverdrahtung anzufertigen.

Funktionsweise als Slave-Baugruppe

Nach einem Systemreset sind sämtliche Bus-Leitungstreiber in beiden Richtungen im Tri-State-Zustand. Master und Slave beginnen simultan autonom zu arbeiten. Nun kann z.B. der Slave über die Master-Request-Leitung, die einen Interrupt im Master-System auslöst, eine Bearbeitung beantragen. Der Master muß das Slave-Request-Signal, das an BUSREQ der Slave-CPU herangeführt ist, aktivieren. Die Slave-CPU antwortet mit BUSAK (Slave Acknowledge) das gleichzeitig an den Memory-PROM herangeführt wird. Dies hat zur Folge, daß der auf der Slave-Karte befindliche Speicher in eine Adreßlage gebracht werden kann, die sich nicht mit dem Speicher des Masters überlagert. Ebenfalls werden sämtliche Treiberbausteine aktiviert. Der Master kann nun ungehindert auf den Speicher der Slave Karte zugreifen. Nach Zurücknehmen des Slave-Request-Signals und des daraus resultierenden inaktiv werden des Slave-Acknowledge-Signals können sowohl Master als auch Slave wieder autonom arbeiten.

Pin Description

B/$\overline{A}$. *Channel A Or B Select* (input, High selects Channel B). This input defines which channel is accessed during a data transfer between the CPU and the Z-80 DART.

C/$\overline{D}$. *Control Or Data Select* (input, High selects Control). This input specifies the type of information (control or data) transferred on the data bus between the CPU and the Z-80 DART.

$\overline{CE}$. *Chip Enable* (input, active Low). A Low at this input enables the Z-80 DART to accept command or data input from the CPU during a write cycle, or to transmit data to the CPU during a read cycle.

CLK. *System Clock* (input). The Z-80 DART uses the standard Z-80 single-phase system clock to synchronize internal signals.

$\overline{CTSA}$, $\overline{CTSB}$. *Clear To Send* (inputs, active Low). When programmed as Auto Enables, a Low on these inputs enables the respective transmitter. If not programmed as Auto Enables, these inputs may be programmed as general-purpose inputs. Both inputs are Schmitt-trigger buffered to accommodate slow-risetime signals.

D_0-D_7. *System Data Bus* (bidirectional, 3-state) transfers data and commands between the CPU and the Z-80 DART.

$\overline{DCDA}$, $\overline{DCDB}$. *Data Carrier Detect* (inputs, active Low). These pins function as receiver enables if the Z-80 DART is programmed for Auto Enables; otherwise they may be used as general-purpose input pins. Both pins are Schmitt-trigger buffered.

$\overline{DTRA}$, $\overline{DTRB}$. *Data Terminal Ready* (outputs, active Low). These outputs follow the state programmed into the DTR bit. They can also be programmed as general-purpose outputs.

IEI. *Interrupt Enable In* (input, active High) is used with IEO to form a priority daisy chain when there is more than one interrupt-driven device. A High on this line indicates that no other device of higher priority is being serviced by a CPU interrupt service routine.

IEO. *Interrupt Enable Out* (output, active High). IEO is High only if IEI is High and the CPU is not servicing an interrupt from this Z-80 DART. Thus, this signal blocks lower priority devices from interrupting while a higher priority device is being serviced by its CPU interrupt service routine.

$\overline{INT}$. *Interrupt Request* (output, open drain, active Low). When the Z-80 DART is requesting an interrupt, it pulls $\overline{INT}$ Low.

$\overline{M1}$. *Machine Cycle One* (input from Z-80 CPU, active Low). When $\overline{M1}$ and $\overline{RD}$ are both active, the Z-80 CPU is fetching an instruction from memory; when $\overline{M1}$ is active while $\overline{IORQ}$ is active, the Z-80 DART accepts $\overline{M1}$ and $\overline{IORQ}$ as an interrupt acknowledge if the Z-80 DART is the highest priority device that has interrupted the Z-80 CPU.

$\overline{IORQ}$. *Input/Output Request* (input from CPU, active Low). $\overline{IORQ}$ is used in conjunction with B/$\overline{A}$, C/$\overline{D}$, $\overline{CE}$ and $\overline{RD}$ to transfer commands and data between the CPU and the Z-80 DART. When $\overline{CE}$, $\overline{RD}$ and $\overline{IORQ}$ are all active, the channel selected by B/$\overline{A}$ transfers data to the CPU (a read operation). When $\overline{CE}$ and $\overline{IORQ}$ are active, but $\overline{RD}$ is inactive, the channel selected by B/$\overline{A}$ is written to by the CPU with either data or control information as specified by C/$\overline{D}$.

$\overline{RxCA}$, $\overline{RxCB}$. *Receiver Clocks* (inputs). Receive data is sampled on the rising edge of $\overline{RxC}$. The Receive Clocks may be 1, 16, 32 or 64 times the data rate.

$\overline{RD}$. *Read Cycle Status.* (input from CPU, active Low). If $\overline{RD}$ is active, a memory or I/O read operation is in progress.

RxDA, RxDB. *Receive Data* (inputs, active High).

$\overline{RESET}$. *Reset* (input, active Low). Disables both receivers and transmitters, forces TxDA and TxDB marking, forces the modem controls High and disables all interrupts.

$\overline{RIA}$, $\overline{RIB}$. *Ring Indicator* (inputs, Active Low). These inputs are similar to $\overline{CTS}$ and $\overline{DCD}$. The Z-80 DART detects both logic level transitions and interrupts the CPU. When not used in switched-line applications, these inputs can be used as general-purpose inputs.

$\overline{RTSA}$, $\overline{RTSB}$. *Request to Send* (outputs, active Low). When the RTS bit is set, the $\overline{RTS}$ output goes Low. When the RTS bit is reset, the output goes High after the transmitter empties.

$\overline{TxCA}$, $\overline{TxCB}$. *Transmitter Clocks* (inputs). TxD changes on the falling edge of $\overline{TxC}$. The Transmitter Clocks may be 1, 16, 32 or 64 times the data rate; however, the clock multiplier for the transmitter and the receiver must be the same. The Transmit Clock inputs are Schmitt-trigger buffered. Both the Receiver and Transmitter Clocks may be driven by the Z-80 CTC Counter Time Circuit for programmable baud rate generation.

TxDA, TxDB. *Transmit Data* (outputs, active High).

$\overline{W/RDYA}$, $\overline{W/RDYB}$. *Wait/Ready* (outputs, open drain when programmed for Wait function, driven High and Low when programmed for Ready function). These dual-purpose outputs may be programmed as Ready lines for a DMA controller or as Wait lines that synchronize the CPU to the Z-80 DART data rate. The reset state is open drain.

Functional Description

The functional capabilities of the Z-80 DART can be described from two different points of view: as a data communications device, it transmits and receives serial data, and meets the requirements of asynchronous data communications protocols; as a Z-80 family peripheral, it interacts with the Z-80 CPU and other Z-80 peripheral circuits, and shares the data, address and control buses, as well as being a part of the Z-80 interrupt structure. As a peripheral to other microprocessors, the Z-80 DART offers valuable features such as non-vectored interrupts, polling and simple handshake capability.

The first part of the following functional description introduces Z-80 DART data communications capabilities; the second part describes the interaction between the CPU and the Z-80 DART.

The Z-80 DART offers RS-232 serial communications support by providing device signals for external modem control. In addition to dual-channel Request To Send, Clear To Send, and Data Carrier Detect ports, the Z-80 DART also features a dual channel Ring Indicator (RIA, RIB) input to facilitate local/remote or station-to-station communication capability.

Communications Capabilities. The Z-80 DART provides two independent full-duplex channels for use as an asynchronous receiver/transmitter. The following is a short description of receiver/transmitter capabilities. For more details, refer to the Asynchronous Mode section of the *Z-80 SIO Technical Manual*. The Z-80 DART offers transmission and reception of five to eight bits per character, plus optional even or odd parity. The transmitter can supply one, one and a half or two stop bits per character and can provide a break output at any time. The receiver break detection logic interrupts the CPU both at the start and end of a received break. Reception is protected from spikes by a transient spike rejection mechanism that checks the signal one-half a bit time after a Low level is detected on the Receive Data input. If the Low does not persist—as in the case of a transient—the character assembly process is not started.

Framing errors and overrun errors are detected and buffered together with the character on which they occurred. Vectored interrupts allow fast servicing of interrupting conditions using dedicated routines. Furthermore, a built-in checking process avoids interpreting a framing error as a new start bit: a framing error results in the addition of one-half a bit time to the point at which the search for the next start bit is begun.

The Z-80 DART does not require symmetric Transmit and Receive Clock signals—a feature that allows it to be used with a Z-80 CTC or any other clock source. The transmitter and receiver can handle data at a rate of 1, 1/16, 1/32 or 1/64 of the clock rate supplied to the Receive and Transmit Clock inputs. When using Channel B, the bit rates for transmit and receive operations must be the same because $\overline{RxC}$ and $\overline{TxC}$ are bonded together ($\overline{RxTxCB}$).

I/O Interface Capabilities. The Z-80 DART offers the choice of Polling, Interrupt (vectored or non-vectored) and Block Transfer modes to transfer data, status and control information to and from the CPU. The Block Transfer mode can be implemented under CPU or DMA control.

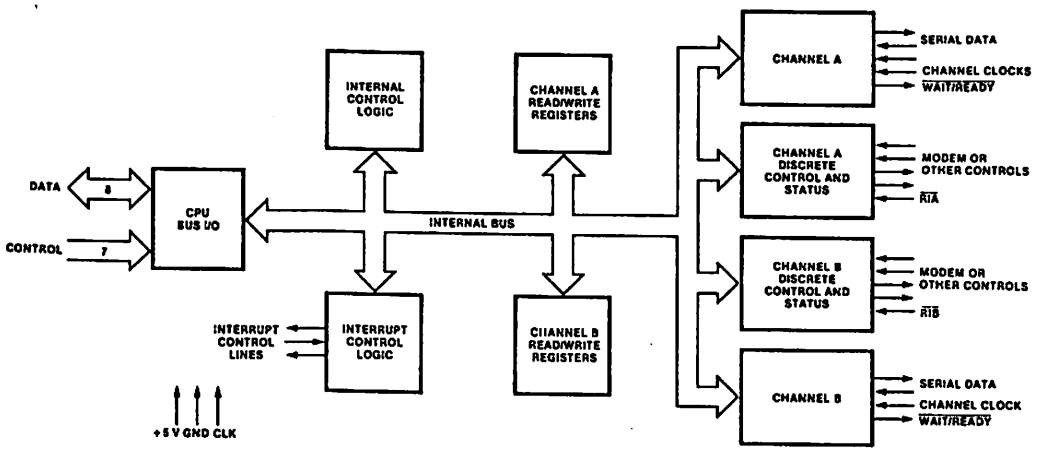


Figure 3. Block Diagram

Functional Description (Continued)

POLLING. There are no interrupts in the Polled mode. Status registers RR0 and RR1 are updated at appropriate times for each function being performed. All the interrupt modes of the Z-80 DART must be disabled to operate the device in a polled environment.

While in its Polling sequence, the CPU examines the status contained in RR0 for each channel; the RR0 status bits serve as an acknowledge to the Poll inquiry. The two RR0 status bits D_0 and D_2 indicate that a data transfer is needed. The status also indicates Error or other special status conditions (see "Z-80 DART Programming"). The Special Receive Condition status contained in RR1 does not have to be read in a Polling sequence because the status bits in RR1 are accompanied by a Receive Character Available status in RR0.

INTERRUPTS. The Z-80 DART offers an elaborate interrupt scheme that provides fast interrupt response in real-time applications. As a member of the Z-80 family, the Z-80 DART can be daisy-chained along with other Z-80 peripherals for peripheral interrupt-priority resolution. In addition, the internal interrupts of the Z-80 DART are nested to prioritize the various interrupts generated by Channels A and B. Channel B registers WR2 and RR2 contain the interrupt vector that points to an interrupt service routine in the memory. To eliminate the necessity of writing a status analysis routine, the Z-80 DART can modify the interrupt vector in RR2 so it points directly to one of eight interrupt service routines. This is done under program control by setting a program bit (WR1, D_2) in Channel B called "Status Affects Vector." When this bit is set, the interrupt vector in RR2 is modified according to the assigned priority of the various interrupting conditions.

Transmit interrupts, Receive interrupts and External/Status interrupts are the main sources of interrupts. Each interrupt source is enabled under program control with Channel A having a higher priority than Channel B, and with Receiver, Transmit and External/Status interrupts prioritized in that order within each channel. When the Transmit interrupt is enabled, the CPU is interrupted by the transmit buffer *becoming* empty. (This implies that the transmitter must have had a data character written into it so it can become empty.) When enabled, the receiver can interrupt the CPU in one of three ways:

- Interrupt on the first received character
- Interrupt on all received characters
- Interrupt on a Special Receive condition

Interrupt On First Character is typically used with the Block Transfer mode. Interrupt On All Receive Characters can optionally modify the interrupt vector in the event of a parity error. The Special Receive Condition interrupt can occur on a character basis. The Special Receive condition can cause an interrupt only if the Interrupt On First Receive Character or Interrupt On All Receive Characters mode is selected. In Interrupt On First Receive Character, an interrupt can occur from Special Receive conditions (except Parity Error) after the first receive character interrupt (example: Receive Overrun interrupt).

The main function of the External/Status interrupt is to monitor the signal transitions of the $\overline{CTS}$, $\overline{DCD}$ and $\overline{RI}$ pins; however, an External/Status interrupt is also caused by the detection of a Break sequence in the data stream. The interrupt caused by the Break sequence has a special feature that allows the Z-80 DART to interrupt when the Break sequence is detected or terminated. This feature facilitates the proper termination of the current message, correct initialization of the next message, and the accurate timing of the Break condition.

CPU/DMA BLOCK TRANSFER. The Z-80 DART provides a Block Transfer mode to accommodate CPU block transfer functions and DMA block transfers (Z-80 DMA or other designs). The Block Transfer mode uses the $\overline{W/RDY}$ output in conjunction with the Wait/Ready bits of Write Register 1. The $\overline{W/RDY}$ output can be defined under software control as a Wait line in the CPU Block Transfer mode or as a Ready line in the DMA Block Transfer mode.

To a DMA controller, the Z-80 DART Ready output indicates that the Z-80 DART is ready to transfer data to or from memory. To the CPU, the Wait output indicates that the Z-80 DART is not ready to transfer data, thereby requesting the CPU to extend the I/O cycle.

Internal Architecture

The device internal structure includes a Z-80 CPU interface, internal control and interrupt logic, and two full-duplex channels. Each channel contains read and write registers, and discrete control and status logic that provides the interface to modems or other external devices.

The read and write register group includes five 8-bit control registers and two status registers. The interrupt vector is written into an additional 8-bit register (Write Register 2) in Channel B that may be read through Read Register 2 in Channel B. The registers for both channels are designated as follows:

WR0-WR5 — Write Registers 0 through 5
RR0-RR2 — Read Registers 0 through 2

The bit assignment and functional grouping of each register is configured to simplify and organize the programming process.

The logic for both channels provides formats, bit synchronization and validation for data transferred to and from the channel interface. The modem control inputs Clear to Send ($\overline{CTS}$), Data Carrier Detect ($\overline{DCD}$) and Ring Indicator ($\overline{RI}$) are monitored by the control logic under program control. All the modem control signals are general purpose in nature and can be used for functions other than modem control.

For automatic interrupt vectoring, the interrupt control logic determines which channel and which device within the channel has the highest priority. Priority is fixed with Channel A assigned a higher priority than Channel B; Receive, Transmit and External/Status interrupts are prioritized in that order within each channel.

Data Path. The transmit and receive data path illustrated for Channel A in Figure 4 is identical for both channels. The receiver has three 8-bit buffer registers in a FIFO arrangement in addition to the 8-bit receive shift register. This scheme creates additional time for the CPU to service a Receive Character Available interrupt in a high-speed data transfer.

The transmitter has an 8-bit transmit data register that is loaded from the internal data bus, and a 9-bit transmit shift register that is loaded from the transmit data register.

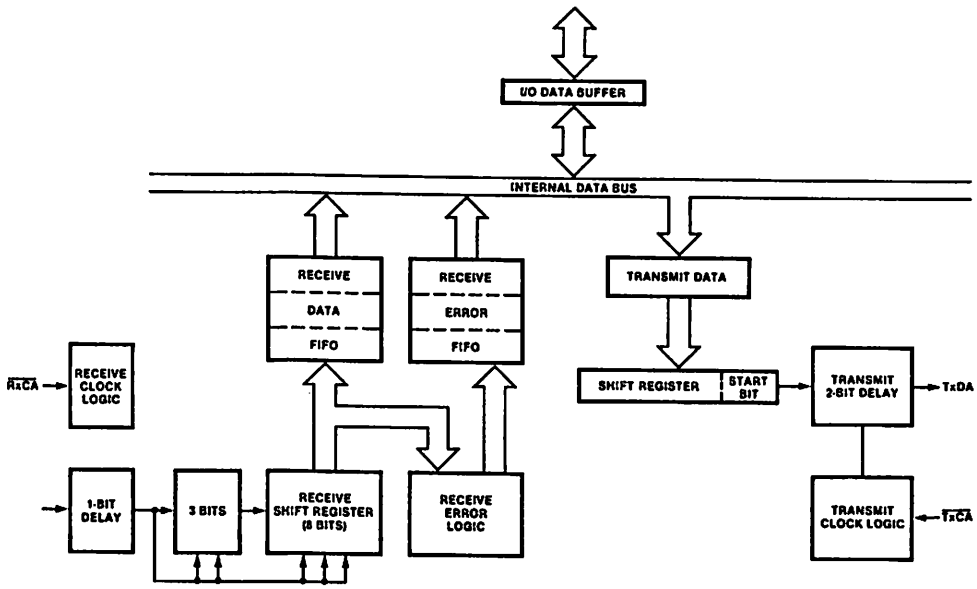


Figure 4. Data Path

Z-80 DART Programming

To program the Z-80 DART, the system program first issues a series of commands that initialize the basic mode and then other commands that qualify conditions within the selected mode. For example, the character length, clock rate, number of stop bits, even or odd parity are first set, then the Interrupt mode and, finally, receiver or transmitter enable.

Both channels contain command registers that must be programmed via the system program prior to operation. The Channel Select input (B/$\overline{A}$) and the Control/Data input (C/$\overline{D}$) are the command structure addressing controls, and are normally controlled by the CPU address bus.

Write Registers. The Z-80 DART contains six registers (WR0-WR5) in each channel that are programmed separately by the system program to configure the functional personality of the channels (Figure 4). With the exception of WR0, programming the write registers requires two bytes. The first byte contains three bits (D_0-D_2) that point to the selected register; the second byte is the actual control word that is written into the register to configure the Z-80 DART.

WR0 is a special case in that all the basic commands (CMD_0-CMD_2) can be accessed with a single byte. Reset (internal or external) initializes the pointer bits D_0-D_2 to point to WR0. This means that a register cannot be pointed to in the same operation as a channel reset.

Write Register Functions	
WR0	Register pointers, initialization commands for the various modes, etc.
WR1	Transmit/Receive interrupt and data transfer mode definition.
WR2	Interrupt vector (Channel B only)
WR3	Receive parameters and control
WR4	Transmit/Receive miscellaneous parameters and modes
WR5	Transmit parameters and controls

Read Registers. The Z-80 DART contains three registers (RR0-RR2) that can be read to obtain the status information for each channel (except for RR2, which applies to Channel B only). The status information includes error conditions, interrupt vector and standard communications-interface signals.

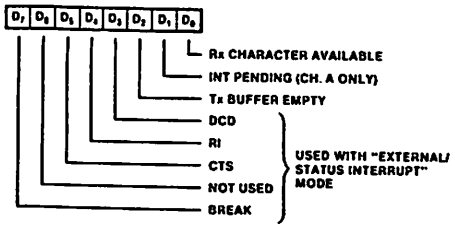
To read the contents of a selected read register other than RR0, the system program must first write the pointer byte to WR0 in exactly the same way as a write register operation. Then, by executing an input instruction, the contents of the addressed read register can be read by the CPU.

The status bits of RR0 and RR1 are carefully grouped to simplify status monitoring. For example, when the interrupt vector indicates that a Special Receive Condition interrupt has occurred, all the appropriate error bits can be read from a single register (RR1).

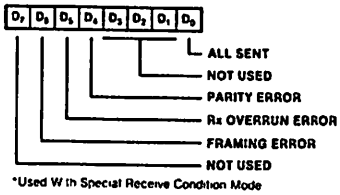
Read Register Functions	
RR0	Transmit/Receive buffer status, interrupt status and external status
RR1	Special Receive Condition status
RR2	Modified interrupt vector (Channel B only)

Z-80 DART Read and Write Registers

READ REGISTER 0

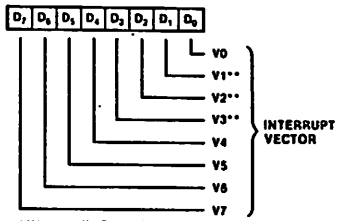


READ REGISTER 1*



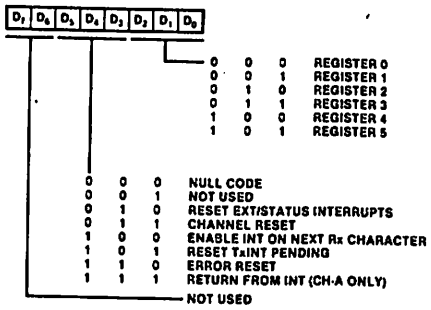
*Used With Special Receive Condition Mode

READ REGISTER 2

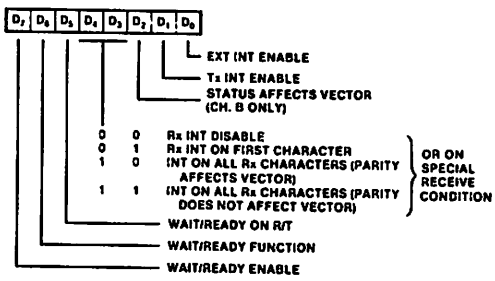


**Variable If Status Affects Vector Is Programmed

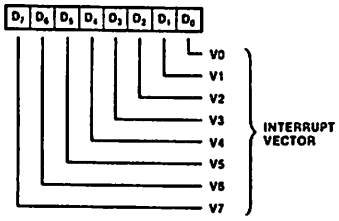
WRITE REGISTER 0



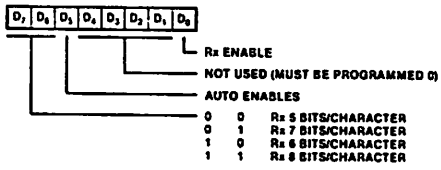
WRITE REGISTER 1



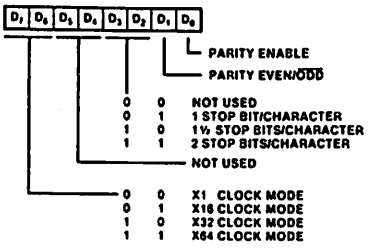
WRITE REGISTER 2 (CHANNEL B ONLY)



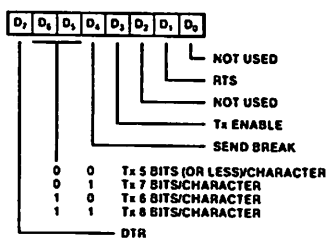
WRITE REGISTER 3



WRITE REGISTER 4



WRITE REGISTER 5



SIGNAL NAME	DESCRIPTION
V_{SS}	Ground
V_{CC}	+5 volts (± 5 percent)
$\overline{CE}$	Chip Enable (Input, active low)
$\overline{RD}$	Read Enable (Input, active low)
$\overline{WR}$	Write Enable (Input, active low)
A_0-A_3	Address Inputs. Used to address one of the internal registers during a read or write operation
D_0-D_7	Data Bus (bi-directional)
$\overline{RESET}$	Device Reset (negative true). When activated, all internal registers (except for Timer or USART Data registers) will be cleared, all timers stopped, USART turned off, all interrupts disabled and all pending interrupts cleared, and all I/O lines placed in tri-state input mode.
I_0-I_7	General purpose I/O and interrupt lines
$\overline{INT}$	Interrupt Request (Output, active low, open drain)
$\overline{IORQ}$	Input/Output Request from Z80-CPU (input, active low). The $\overline{IORQ}$ signal is used in conjunction with $\overline{M1}$ to signal the MK3801 that the CPU is acknowledging its interrupt.
IEI	Interrupt Enable In, active High
IEO	Interrupt Enable Out, active High
SO	Serial Output
SI	Serial Input
RC	Receiver Clock Input
TC	Transmit Clock Input
TAO-TDO	Timer Outputs
TCLK	Timer Clock Input
$\overline{M1}$	Z80 Machine Cycle One (negative true)

PIN DESCRIPTION

Figure 1 illustrates the pinout of the MK3801. The functions of these individual pins are described above.

INTERNAL ORGANIZATION

Figure 2 illustrates the MK3801 internal organization, which supports the full set of timing, communications, parallel I/O, and interrupt processing functions available in the device.

CPU BUS I/O

The CPU BUS I/O provides the means of communications between the system and the MK3801. Data, Status, and Control Registers in the MK3801 are accessed by the bus in order to establish device parameters, assert control, and transfer status and data between the system and the MK3801.

Each register in the MK3801 is addressed over the address bus in conjunction with Chip Enable ($\overline{CE}$), while data is transferred over the eight bit Data bus under control of Read ($\overline{RD}$) and Write ($\overline{WR}$) signals.

REGISTER ACCESSES

All register accesses are independent of any system clock. To read a register, both $\overline{CE}$ and $\overline{RD}$ must be active. The internal read control signal is essentially the combination of both $\overline{CE}$ and $\overline{RD}$ active; thus the read operation will begin when the later of these two signals goes active and will end when the first signal goes inactive. The address bus must be stable prior to the start of the operation and must remain stable until the end of the operation. Unless a read operation or an interrupt acknowledge cycle is in progress, the data bus (D_0-D_7) will remain in the tri-state condition.

To write a register, both $\overline{CE}$ and $\overline{WR}$ must be active. The address must be stable prior to the start of the operation and must remain stable until the end of the operation. The data must be stable prior to the end of the operation and must remain stable until the end of the operation. The data presented on the bus will be latched into the register shortly after either $\overline{WR}$ or $\overline{CE}$ goes inactive.

INTERNAL REGISTERS

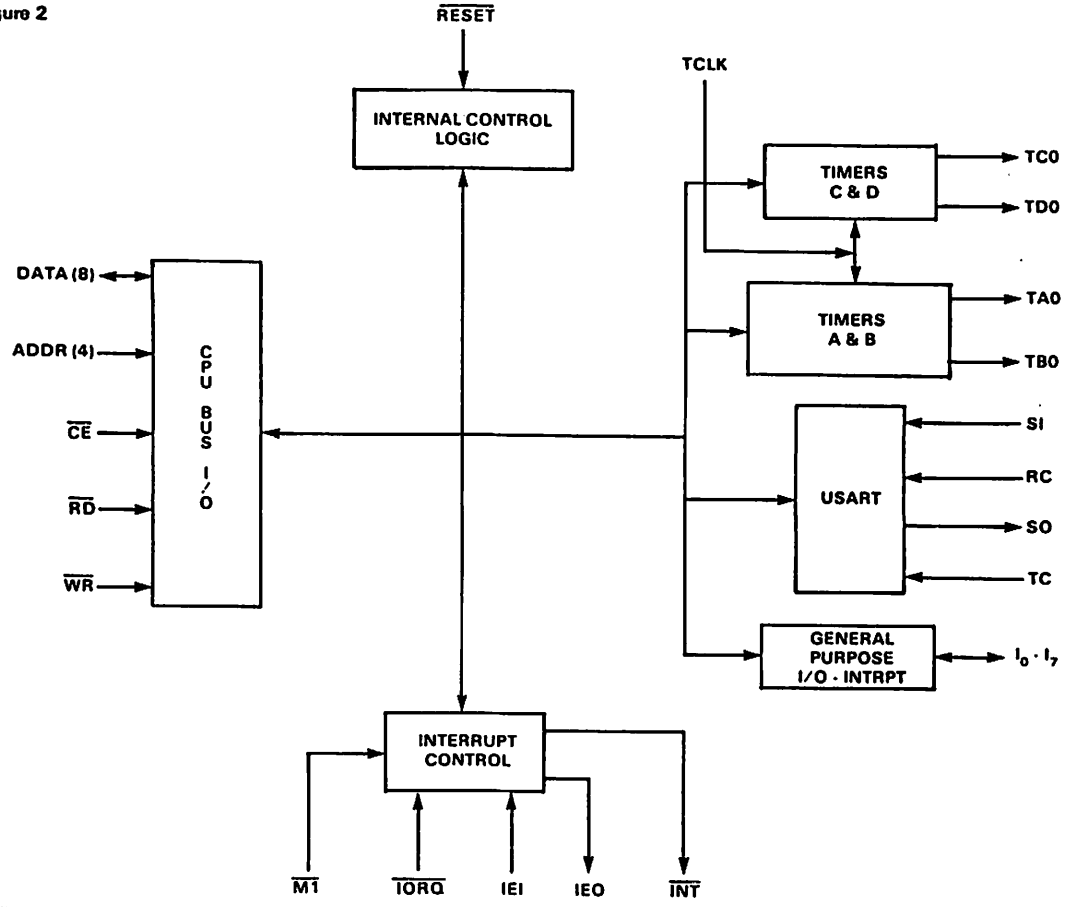
There are 24 internal registers used to control the operation of the STI. Sixteen of these registers are directly addressable and accessible. Eight registers are indirectly addressable via the Pointer/Vector Register and accessible via the Indirect Data Register.

DIRECTLY ADDRESSABLE REGISTERS

The Directly Addressable Registers are accessed by placing the address of the desired register on the address lines (A_0-A_3) during a write or read cycle. Figure 3 lists the Directly Addressable Registers.

INTERNAL ORGANIZATION

Figure 2



DIRECTLY ACCESSIBLE REGISTERS

Figure 3

ADDRESS	ABBREVIATION	REGISTER NAME
0	IDR	Indirect Data Register
1	GPIP	General Purpose I/O-Interrupt
2	IPRB	Interrupt Pending Register B
3	IPRA	Interrupt Pending Register A
4	ISRB	Interrupt in-Service Register B
5	ISRA	Interrupt in-Service Register A
6	IMRB	Interrupt Mask Register B
7	IMRA	Interrupt Mask Register A
8	PVR	Pointer/Vector Register
9	TABCR	Timers A and B Control Register

DIRECTLY ACCESSIBLE REGISTERS (Continued)
Figure 3

ADDRESS	ABBREVIATION	REGISTER NAME
A	TBDR	Timer B Data Register
B	TADR	Timer A Data Register
C	UCR	USART Control Register
D	RSR	Receiver Status Register
E	TSR	Transmitter Status Register
F	UDR	USART Data Register

INDIRECTLY ADDRESSABLE REGISTERS
Figure 4

INDIRECT ADDRESS	ABBREVIATION	REGISTER NAME
0	SCR	Sync Character Register
1	TDDR	Timer D Data Register
2	TCDR	Timer C Data Register
3	AER	Active Edge Register
4	IERB	Interrupt Enable Register B
5	IERA	Interrupt Enable Register A
6	DDR	Data Direction Register
7	TCDCR	Timers C and D Control Register

INDIRECTLY ADDRESSABLE REGISTERS

Indirectly Addressable Registers are addressed by placing the indirect address in bits IA0-IA2 of the Pointer/Vector Register, as defined in Figure 5. Data may be written to or read from the register indicated by these Indirect Register Address bits by a write or read access of the Indirect Data Register (selected when A_0-A_3 are all zero). The indirect address bits of the Pointer/Vector Register will remain unchanged after an indirect access. Repeated accesses of the Indirect Data Register will access the same indirect register as long as the indirect address in the Pointer/Vector Register remains unchanged. The Indirectly Addressable Registers are listed in Figure 4.

INTERRUPT VECTOR DEFINITION

Each individual function in the MK3801 is provided with a unique interrupt vector that is presented to the system during the interrupt acknowledge cycle. The interrupt vector returned during interrupt acknowledge is formed as shown in Figure 6. There are 16 vector addresses generated internally by the MK3801, one for each of the 16 interrupt channels. The three most significant bits of these vector addresses correspond to the three most significant bits of the Pointer/Vector Register shown in Figure 5. The least significant bit of each vector address is always 0, thus producing even vector addresses. The remaining 4 bits (IV_1 through IV_4) identify each of the 16 interrupt channels individually. The lowest priority channel responds with 0000 for IV_4-IV_1 respectively. The next higher priority channel responds with 0001, and so on in binary order, with the highest priority channel responding with 1111. Figure 7 lists each of the 16 interrupt channels in order of descending priority.

INTERRUPT CONTROL REGISTERS

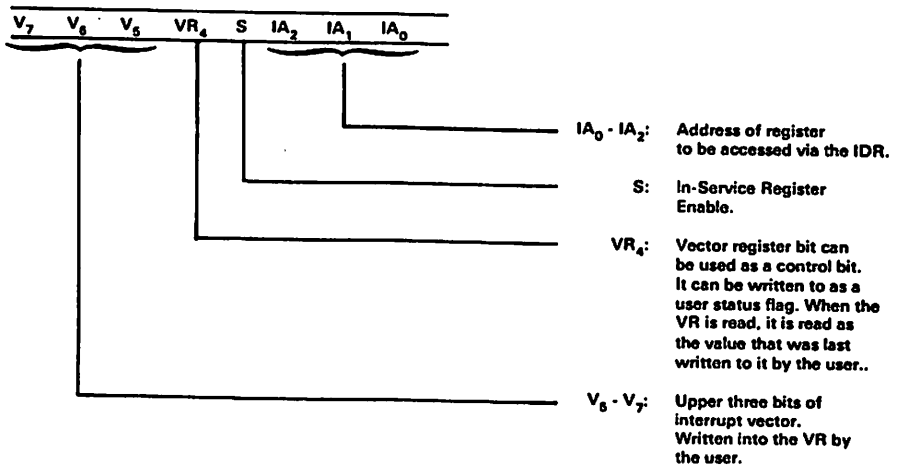
The Interrupt Control Registers provide control of interrupt processing for all I/O facilities of the MK3801. These registers allow the programmer to enable or disable any or all of the 16 interrupts, provide masking for any interrupts, and access to the pending or in-service status of the interrupts. Optional End-of-Interrupt modes are available under software control. The format of each of the Interrupt Control Registers is presented in Figure 8.

INTERRUPT OPERATION

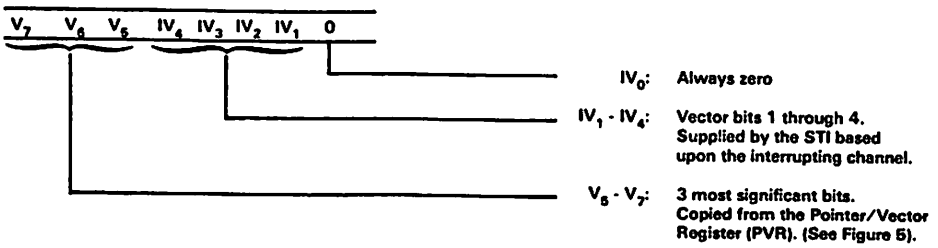
The Interrupt Enable Registers enable or disable the setting of an interrupt in the Interrupt Pending Registers. A 0 in a bit of the Interrupt Enable Registers disables the interrupt for the associated channel while a 1 enables the interrupt.

Once an interrupt is enabled, the occurrence of an interrupting condition on that channel will cause the

POINTER/VECTOR REGISTER (PVR) Port 08
Figure 5



INTERRUPT VECTOR
Figure 6



corresponding bit in the Interrupt Pending Register to be set. This indicates that an interrupt is pending in the MK3801.

Pending interrupts are presented to the Z80 CPU in order of priority (see Figure 1) unless they have been masked off. This is done by clearing the bit in the Interrupt Mask Register corresponding to the channel whose interrupt is to be masked. The channel's interrupt will remain pending until the mask bit for that channel is set, at which time the interrupt for that channel will be processsed in order of priority.

When an interrupt vector is generated for a pending interrupt and passed to the Z80 CPU, the bit in the Interrupt Pending Register, associated with the channel generating the interrupt, will be cleared. At this time, no history of the interrupt remains in the MK3801.

In order to retain historical evidence of an interrupt being serviced by the Z80, the In-Service Register may be enabled by setting the S-bit in the Pointer/Vector Register (see Figure 5). If the In-Service Register is enabled, the bit of the In-Service Register corresponding to the interrupting channel will be set when the interrupt vector is passed to the Z80. At the same time, the Interrupt Pending bit will be cleared since the interrupt is now in service. The In-Service bit will be cleared on execution of a Return-from-Interrupt (H'ED4D') instruction. The In-Service Registers are directly addressable, and the In-Service bit for any interrupt may be cleared by writing to the In-Service Register if the Return-from-Interrupt instruction is not used.

INTERRUPT CONTROL REGISTER DEFINITIONS
Figure 7

There are sixteen interrupt channels on the STI arranged in the following priority:

PRIORITY	CHANNEL	DESCRIPTION	ALTERNATE USAGE
HIGHEST	1111	General Purpose Interrupt 7 (I_7)	
	1110	General Purpose Interrupt 6 (I_6)	
	1101	Timer A	
	1100	Receive Buffer Full	
	1011	Receive Error	
	1010	Transmit Buffer Empty	
	1001	Transmit Error	
	1000	Timer B	
	0111	General Purpose Interrupt 5 (I_5)	
	0110	General Purpose Interrupt 4 (I_4)	TA (PW-Event)
	0101	Timer C	
	0100	Timer D	
	0011	General Purpose Interrupt 3 (I_3)	TB (PW-Event)
	0010	General Purpose Interrupt 2 (I_2)	
	0001	General Purpose Interrupt 1 (I_1)	DMA (TR)TX
LOWEST	0000	General Purpose Interrupt 0 (I_0)	DMA (RR)REC

INTERRUPT CONTROL REGISTERS
Figure 8

INTERRUPT ENABLE REGISTERS

ADDRESS		7	6	5	4	3	2	1	0
Indirect Port 5	A (IERA)	GPIP 7	GPIP 6	TIMER A	RCV Buffer Full	RCV Error	XMIT Buffer Empty	XMIT Error	TIMER B
Indirect Port 4	B (IERB)	GPIP 5	GPIP 4	TIMER C	TIMER D	GPIP 3	GPIP 2	GPIP 1	GPIP 0

INTERRUPT MASK REGISTERS

		7	6	5	4	3	2	1	0
Port 7	A (IMRA)	GPIP 7	GPIP 6	TIMER A	RCV Buffer Full	RCV Error	XMIT Buffer Empty	XMIT Error	TIMER B
Port 6	B (IMRB)	GPIP 5	GPIP 4	TIMER C	TIMER D	GPIP 3	GPIP 2	GPIP 1	GPIP 0

1 = UNMASKED, 0 = MASKED

INTERRUPT PENDING REGISTERS

		7	6	5	4	3	2	1	0
Port 3	A (IPRA)	GPIP 7	GPIP 6	TIMER A	RCV Buffer Full	RCV Error	XMIT Buffer Empty	XMIT Error	TIMER B
Port 2	B (IPRB)	GPIP 5	GPIP 4	TIMER C	TIMER D	GPIP 3	GPIP 2	GPIP 1	GPIP 0

WRITING 0 = CLEAR
WRITING 1 = UNCHANGED

INTERRUPT CONTROL REGISTERS (Continued)

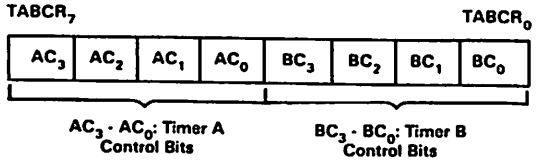
Figure 8

INTERRUPT SERVICE REGISTERS

ADDRESS		7	6	5	4	3	2	1	0
Port 5	A (ISRA)	GPIP 7	GPIP 6	TIMER A	RCV Buffer Full	RCV Error	XMIT Buffer Empty	XMIT Error	TIMER B
Port 4	B (ISRB)	GPIP 5	GPIP 4	TIMER C	TIMER D	GPIP 3	GPIP 2	GPIP 1	GPIP 0

TIMER A and B CONTROL REGISTER (TABCR) Port 9

Figure 9



The four control bits are used to select the timer mode and prescale value, as follows:

CONTROL BIT DEFINITION

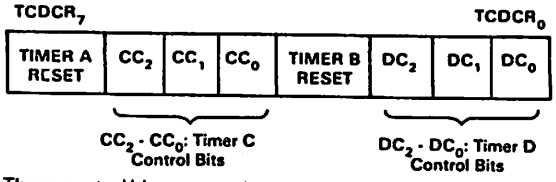
C_3	C_2	C_1	C_0	
0	0	0	0	Timer Stopped
0	0	0	1	Delay Mode, ÷4 Prescale
0	0	1	0	Delay Mode, ÷10 Prescale
0	0	1	1	Delay Mode, ÷16 Prescale
0	1	0	0	Delay Mode, ÷50 Prescale
0	1	0	1	Delay Mode, ÷64 Prescale
0	1	1	0	Delay Mode, ÷100 Prescale
0	1	1	1	Delay Mode, ÷200 Prescale
1	0	0	0	Event Count Mode
1	0	0	1	Pulse Width Mode, ÷4 Prescale
1	0	1	0	Pulse Width Mode, ÷10 Prescale
1	0	1	1	Pulse Width Mode, ÷16 Prescale
1	1	0	0	Pulse Width Mode, ÷50 Prescale
1	1	0	1	Pulse Width Mode, ÷64 Prescale
1	1	1	0	Pulse Width Mode, ÷100 Prescale
1	1	1	1	Pulse Width Mode, ÷200 Prescale

TIMER A DATA REGISTER AND TIMER B DATA REGISTER (TADR, TBDR) Port B & Port A

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

TIMER C and D CONTROL REGISTER (TCDCR) Indirect Port 7

Figure 10



Three control bits are used to control each timer, as defined below:

CONTROL BIT DEFINITION

C_2	C_1	C_0	
0	0	0	Timer Stopped
0	0	1	Delay Mode, ÷4 Prescale
0	1	0	Delay Mode, ÷10 Prescale
0	1	1	Delay Mode, ÷16 Prescale
1	0	0	Delay Mode, ÷50 Prescale
1	0	1	Delay Mode, ÷64 Prescale
1	1	0	Delay Mode, ÷100 Prescale
1	1	1	Delay Mode, ÷200 Prescale

TIMER C DATA REGISTER and TIMER D DATA REGISTER (TCDR, TDDR) Indirect, Port 2 and Indirect Port 1

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

TIMERS

Four timers are available on the MK3801. Two provide full service features including delay timer operation, event counter operation, pulse width measurement operation, and pulse generation. The two other timers provide delay timer features only, and may be used for baud rate generators for use with the USART.

All timers are prescaler/counter timers, with a common independent clock input, and are not required to be operated from the system clock. In addition, all timers have a time-out output function that toggles each time the timer times out.

TIMER CONTROL REGISTERS

The 4 timers (A,B,C, and D) are programmed via 2 control registers and 4 timer data registers. Timers A and B are controlled by a single register (TABCR) and two timer data registers (TADR,TBDR). Timers C and D are controlled by a second control register (TCDCR) and two timer data

registers (TCDR, TDDR). Bits in the control registers allow the selection of operational mode, prescale, and control, while the data registers are used to read the timer or write the time constant register. General Purpose I/O Interrupt pins 3 (TB) and 4 (TA) are used for timer B and A inputs in event and pulse width modes. Figure 9 illustrates the Control and Data Register for timers A and B, while Figure 10 illustrates the Control and Data registers for timers C and D.

USART

Serial Communication is provided by the USART, which is capable of either asynchronous or synchronous operation. Variable word width and start/stop bit configurations are available under software control for asynchronous operation. For synchronous operation, a Sync Word is provided to establish synchronization during receive operations. The Sync Word will also be repeatedly transmitted when no other data is available for transmission. Operational modes exist to allow stripping of all Sync Words received in synchronous operation, and to allow the operation of DMA control handshake lines by the USART through General Purpose I/O Port lines 0 and 1. Separate receive and transmit clocks are available, and separate receive and transmit status and data bytes allow independent operation of the transmit and receive sections.

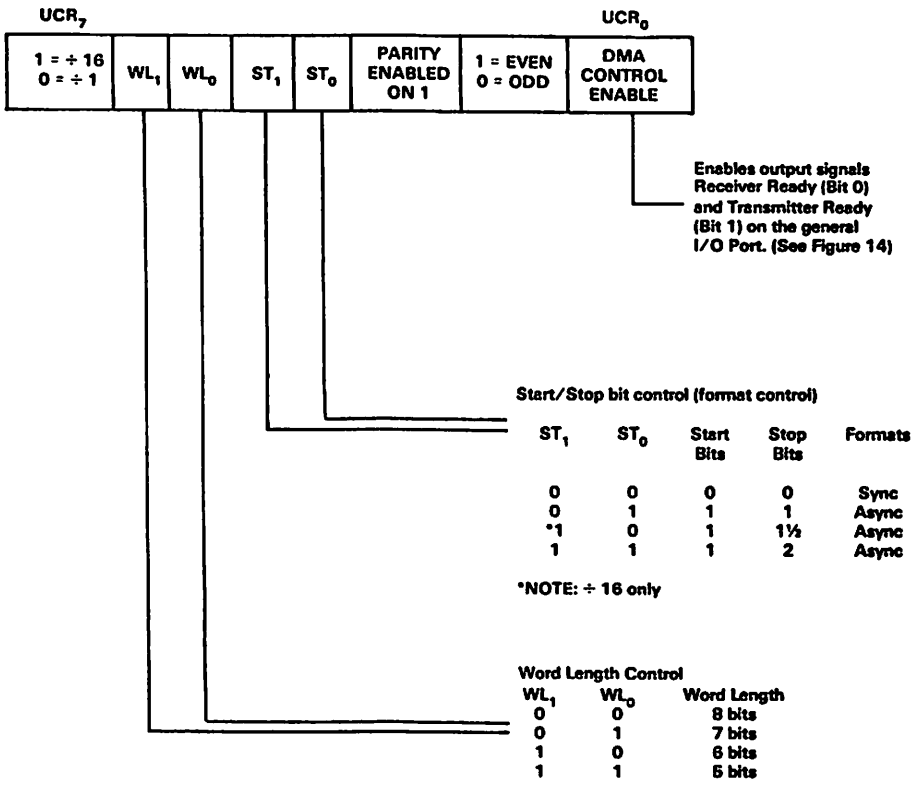
USART CONTROL REGISTERS

The USART is provided with 3 control/status registers and a data register. The programmer may specify operational parameters for the USART via the Control Register, as shown in Figure 11. Status of both the Receiver and Transmitter sections is accessed by means of the 2 Status Registers, as shown in Figure 12. Data written to the Data Register is passed to the transmitter, while reading the data register will access data received by the USART. The USART Data Register form is illustrated in Figure 13.

ERROR CONDITIONS

Error conditions in the USART are determined by monitoring the Receive Status Register (Port D) and the Transmitter Status Register (Port E). These error conditions are only valid for each word boundary and are not latched. When executing block transfers of data, it is necessary to save any errors so that they can be checked at the end of a block. In order to save error conditions during data transfer, the STI interrupt controller may be used by enabling error

USART CONTROL REGISTER (UCR) Port C
Figure 11



RECEIVER STATUS REGISTER (RSR) Port D

Figure 12

RSR_7							RSR_0
BUFFER FULL	OVERRUN ERROR	PARITY ERROR	FRAME ERROR	FOUND/SEARCH OR BREAK DETECT	MATCH/CHARACTER IN PROGRESS	SYNC STRIP ENABLE	RECEIVER ENABLE

TRANSMITTER STATUS REGISTER (TSR) Port E

TSR_7							TSR_0
BUFFER EMPTY	UNDERRUN ERROR	AUTO TURNAROUND	END OF TRANSMISSION	BREAK	HIGH	LOW	TRANSMITTER ENABLE

H	L	Serial Output State
0	0	Hi-Z
0	1	Low ("0")
1	0	High
1	1	Loop*

*Connects transmitter output to receiver input. In loopback mode, transmitter goes high when disabled. Also connects clocks with TC given priority.

USART DATA REGISTER (UDR) Port F

Figure 13

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

GENERAL PURPOSE I/O CONTROL REGISTERS

Figure 14

ACTIVE EDGE CONTROL REGISTER (AER) Indirect Port 3

1 = RISING
0 = FALLING

GPIP 7	GPIP 6	GPIP 5	GPIP 4	GPIP 3	GPIP 2	GPIP 1	GPIP 0

DATA DIRECTION REGISTER (DDR) Indirect Port 6

1 = OUTPUT
0 = INPUT

GPIP 7	GPIP 6	GPIP 5	GPIP 4	GPIP 3	GPIP 2	GPIP 1	GPIP 0

GENERAL PURPOSE I/O DATA REGISTER (GPIP) Port 1

GPIP 7	GPIP 6	GPIP 5	GPIP 4	GPIP 3	GPIP 2	GPIP 1 (TR)	GPIP 0 (RR)
			TIMER A INPUT	TIMER B INPUT			

interrupts (Port 5, Indirect) for the desired channel (Receive error or Transmit error) and by masking these bits off (Port 7). Once the transfer is complete, the Interrupt Pending Register (Port 3) can be polled to determine the presence of a pending error interrupt, and therefore an error.

GENERAL PURPOSE I/O - INTERRUPT PORT

The General Purpose I/O - Interrupt Port provides eight I/O lines that may be operated either as inputs or outputs under software control. In addition, each line may generate an interrupt on either a positive going edge or a negative going edge of the input signal.

Two of the lines in this port provide auxiliary input functions for the timers in the pulse width measurement mode and the event counter mode. Two others serve as auxiliary output lines for the USART, one indicating the Receive Buffer Full condition (RR) and the other indicating the Transmitter Buffer Empty condition (TR). These may be used as handshake signals for a DMA controller or other external control circuitry.

GENERAL PURPOSE I/O CONTROL REGISTERS

The General Purpose I/O and Interrupt Port has 2 control registers. One allows the programmer to specify the Active Edge for each bit that will trigger the interrupt associated with that bit. The other register specifies the Data Direction (input and output) associated with each bit. The third register is the actual data I/O register used to input or output data to the port. When the USART is programmed to use DMA signals, this overrides the GPIP data and the DDR. The General Purpose I/O Control and Data Registers are illustrated in Figure 14.

1-6. Command

REGISTER CONTROL MODE GROUP [Group 0]

C2 = 0

MODE & COMMAND	CODE C2 C1 C0	FUNCTION	DATA OUT
REGISTER HOLD	0 0 0	Data of 40 Bit S/R are not shifted, and the data is held. The TEST MODE is released by this command.	1 Hz
REGISTER SHIFT	0 0 1	Data of the 40 Bit S/R can be shifted to Data OUT terminal and the external data can be shifted to the 40 Bit S/R, each synchronizing with a clock input at the CLK terminal.	LSB = 0 or 1
TIME SET	0 1 0	Data of the 40 Bit S/R are not shifted. The T/C is kept to transfer data from the 40 Bit S/R and the 11 – 15 stage remains reset. Set a command (ie. 000, 001, 011) besides this command to start T/C.	LSB = 0 or 1
TIME READ	0 1 1	Data of 40 Bit S/R are not shifted. Data of the T/C is kept to transfer to the 40 Bit S/R.	0.5 Hz

note S/R: Shift Register
T/C: Time Counter

Only to change TIME READ mode to another mode, the time to be changed another mode has to be 40 μs MAX. (=t_{d2}) from the rising edge of STB pulse for setting a new command.

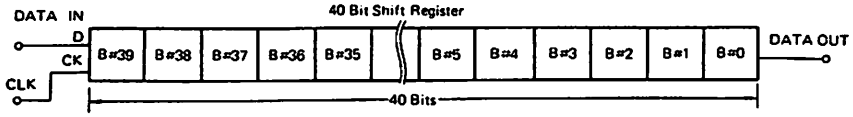
Don't set Register Hold (000) command after Time Read (011) command, because basic time is delayied.

TP CONTROL MODE AND TEST MODE GROUP [Group 1]

C2 = 1

MODE & COMMAND	CODE C2 C1 C0	FUNCTION
TP = 64 Hz	1 0 0	TP terminal is the 64 Hz output.
TP = 256 Hz	1 0 1	TP terminal is the 256 Hz output.
TP = 2 048 Hz	1 1 0	TP terminal is the 2 048 Hz output.
TEST MODE	1 1 1	In this mode, TP terminal is the 32 Hz output. The TEST MODE is released by TP MODE (100, 101, 110) or REGISTER HOLD MODE (000). When is released by MODE (000), TP terminal is 64 Hz output.

1-7. The correspondence of the data between the 40 Bit Shift Register and the Time Counter.

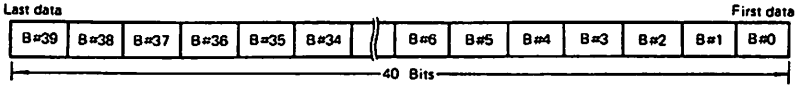


40 Bit Shift Register	Time Counter	Data of Time Counter
B#0	B#0 (LSB) of Seconds/Units	0 through 9 (BCD)
B#1	B#1 "	
B#2	B#2 "	
B#3	B#3 (MSB) "	
B#4	B#0 (LSB) of Seconds/Tens	0 through 5 (BCD)
B#5	B#1 "	
B#6	B#2 (MSB) "	
B#7	0	
B#8	B#0 (LSB) of Minutes/Units	0 through 9 (BCD)
B#9	B#1 "	
B#10	B#2 "	
B#11	B#3 (MSB) "	
B#12	B#0 (LSB) of Minutes/Tens	0 through 5 (BCD)
B#13	B#1 "	
B#14	B#2 (MSB) "	
B#15	0	
B#16	B#0 (LSB) of Hours/Units	0 through 9 (BCD)
B#17	B#1 "	
B#18	B#2 "	
B#19	B#3 (MSB) "	
B#20	B#0 (LSB) of Hours/Tens	0 through 2 (BCD)
B#21	B#1 (MSB) "	
B#22	0	
B#23	0	
B#24	B#0 (LSB) of Date/Units	0 through 9 (BCD)
B#25	B#1 "	
B#26	B#2 "	
B#27	B#3 (MSB) "	
B#28	B#0 (LSB) of Date/Tens	0 through 3 (BCD)
B#29	B#1 (MSB) "	
B#30	0	
B#31	0	
B#32	B#0 (LSB) of Day of Week	0 through 6 (BCD)
B#33	B#1 "	
B#34	B#2 (MSB) "	
B#35	0	
B#36	B#0 (LSB) of Month	1 through 0CH (Hexa Decimal)
B#37	B#1 "	
B#38	B#2 "	
B#39	B#3 (MSB) "	

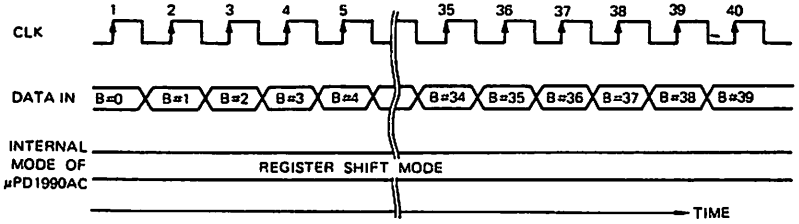
1-8. The form of loading and reading the data

● Loading of data to 40 Bit Shift Register in μPD1990AC.

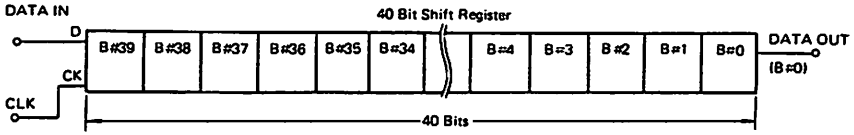
1. The external serial data loading to 40 Bit Shift Register of the μPD1990AC is defined as follows.



2. Loading operation. (The μPD1990AC is in the Register Shift mode.)

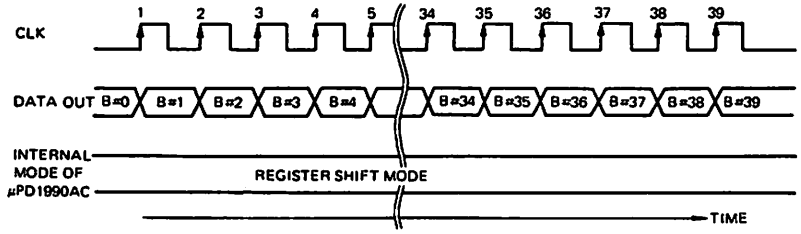


3. Data of 40 Bit Shift Register in the μPD1990AC, after loading operation



● Reading the data of 40 Bit Shift Register in the μPD1990AC.

1. Reading operation. (In the Register Shift mode.)



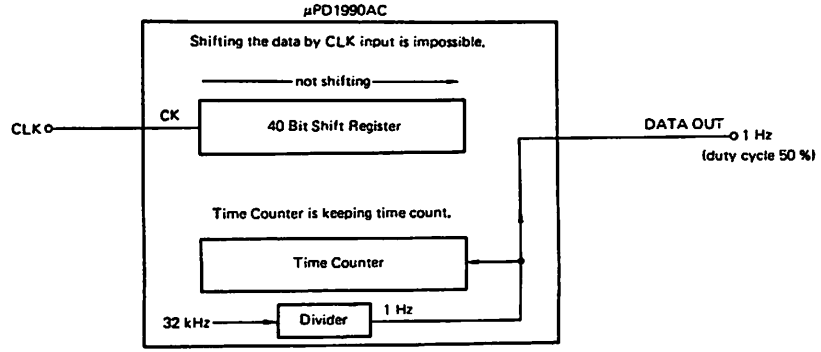
2. COMMAND DESCRIPTION

2-1. Register control modes. [Group "0"]

These mode are set by Register Control command which does not affect with TP (Timing Pulse) control mode. [Group "1"].

★ **Register Hold Mode (Set by Register Hold Command (000))**

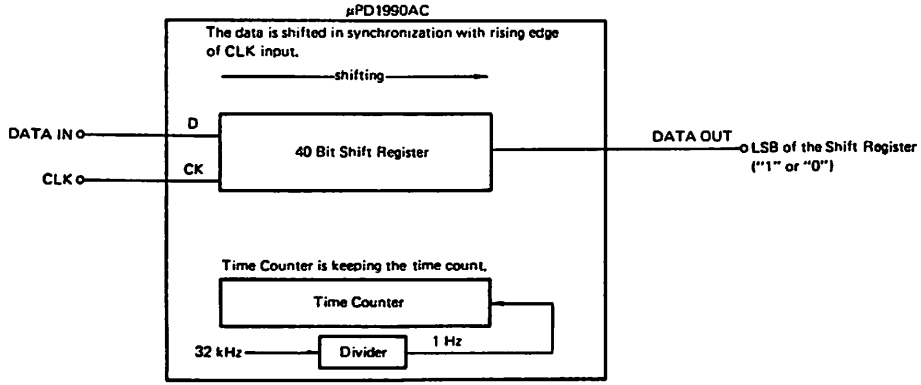
It is impossible for the data of the 40 Bit Shift Register to shift during this mode. And the 1 Hz signal is provided at DATA OUT terminal.



note When μPD1990AC is in TEST MODE (Group "1") by setting this mode the μPD1990AC is released from TEST MODE and TP = 64 Hz mode is set.

★ **Register Shift Mode (Set by Register Shift Command (001))**

Only in this mode, it is possible for the data of 40 Bit Shift Register to shift in synchronization with rising edge of CLK clock input.



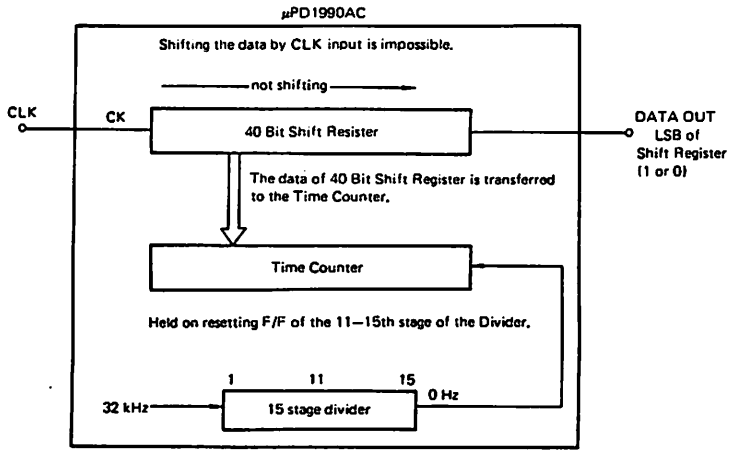
note When μPD1990AC the Register Shift command "001" is set, input level of the CLK terminal has to be low level.

The logic level present at the DATA IN input is transferred into the first register stage and shifted over one stage at each positive-going clock transition of CLK terminal. And data of the last register stage (=LSB) is provided at DATA OUT terminal.

★ **Time Set mode. (Set by Time Set command (010))**

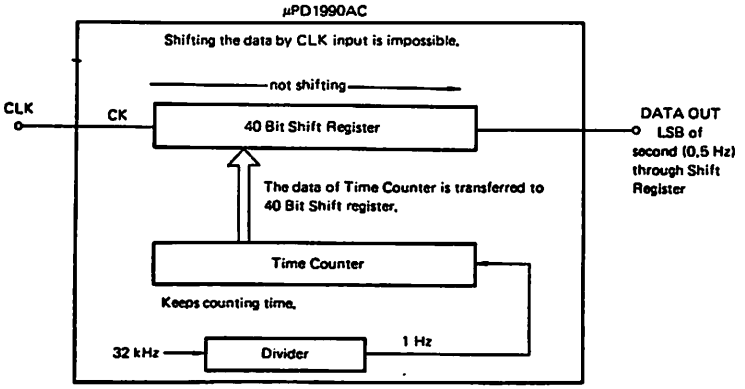
The data of 40 Bit Shift Register is transferred to Time Counter, and the flip-flop of the 11th – 15th stage in the 15 Stage Binary Divider is reset and the Time Counter is held.

As soon as the μPD1990AC is released from the Time Set mode, by setting another Register Control command, such as Register Hold, Register Shift or Time Read commands, the Time Counter starts to keep time.



★ **Time Read mode. (Set by Time Read command (011))**

The data of the Time Counter is transferred to the 40 Bit Shift Register. LSB of the units of seconds (0.5 Hz) output is provided at DATA OUT terminal through the 40 Bit Shift Register.



note: When μPD1990AC is in the Time Read mode, another new mode becomes to enable after 40 μs (MAX.) from the rising edge of STB pulse for setting the new command, because the new mode is delayed to synchronize with the highest frequency (ie. 32 768 kHz) in the μPD1990AC.

2-2. TP (Timing Pulse) Control mode and TEST mode

These modes are controlled only by TP control commands and TEST MODE set command, which does not affect Register Control modes. [Group "0"]

★ **TP=64 Hz mode. (Set by TP=64 Hz command (100))**
The TP terminal is 64 Hz output.

★ **TP=256 Hz mode. (Set by TP=256 Hz command (101))**
The TP terminal is 256 Hz output.

★ **TP=2 048 Hz mode. (Set by TP=2 048 Hz command (110))**
The TP terminals is 2 048 Hz output.

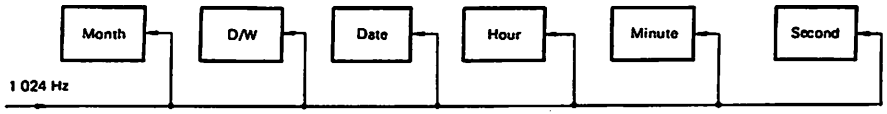
★ **TEST MODE. (Set by TEST mode of command (111))**
This mode set the μPD1990AC to LSI TEST mode. (not to be used for ordinary operation)

TEST MODE (Set by command of TEST mode (111))

In this mode, DATA OUT terminal is enabled regardless of OUT ENBL input.
There are 2 type TEST MODE to be selected by OUT ENBL terminal.
While μPD1990AC is in Register Hold mode, it is impossible to change Test mode by setting TEST MODE COMMAND(111)

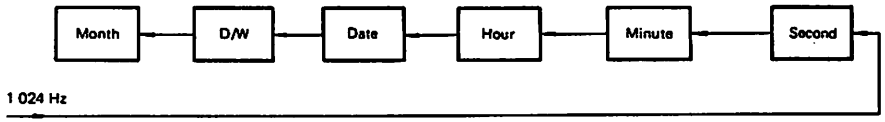
TEST MODE 1 ; OUT ENBL=0

In this mode, every Counter ("Month", "Day of Week", "Date", "Hour", "Minute", "Second") is advanced at a 1 024 Hz rate in parallel. In this case the overflow carry of each counter does not affect to the next counter.



TEST MODE 2 ; OUT ENBL = 1

In this mode TIME COUNTER is advanced at a 1 024 Hz rate instead of 1 Hz for Second counter input.

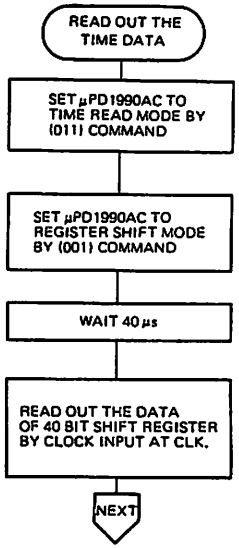


The following table shows the signals to be provided at DATA OUT and TP terminals during the μPD1990AC is in the TEST MODE.

MODE	CODE C2 C1 C0	DATA OUT	TP		
REGISTER HOLD	0 0 0	1 Hz	64 Hz	By this command, TEST MODE is released.	
REGISTER SHIFT	0 0 1	LSB of 40 Bit S/R "0" or "1"	32 Hz	TEST MODE is re-mained.	T/C is advanced at 1 024 Hz in parallel or serial.
TIME SET	0 1 0	LSB of 40 Bit S/R "0" or "1"	32 Hz		
TIME READ	0 1 1	512 Hz	32 Hz		T/C is advanced at 1 024 Hz in parallel or serial.

note. While μPD1990AC is in TEST mode, by setting the Register Hold mode (Group "0" mode), Test Mode changes to TP=64 Hz mode in group "1" and as a result Register Hold mode is set. There is a 1 Hz output at DATA OUT terminal and a 64 Hz output at TP terminal.

3-1. How to read out the Time Data from the μPD1990AC.

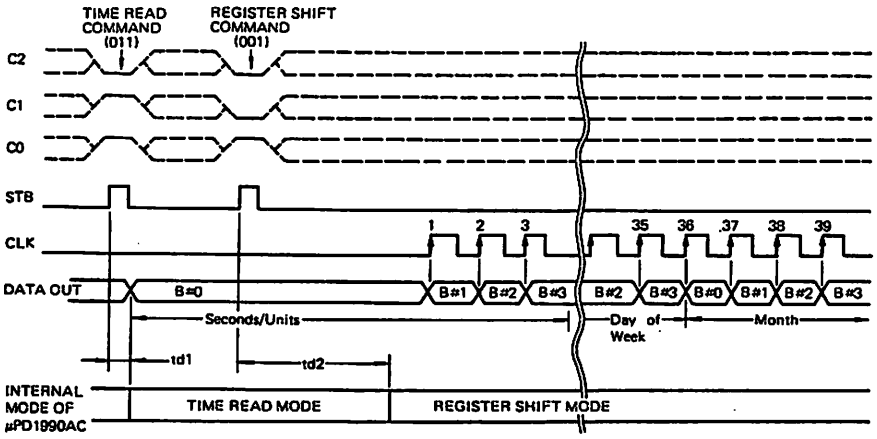


The data of the Time Counter (=Time Data) is transferred to the 40 Bit Shift Register.
But the data of 40 Bit Shift Register cannot be shifted by the clock input at CLK terminal.

The data of 40 Bit Shift Register (=Time Data) can be shifted by the clock input at CLK terminal.

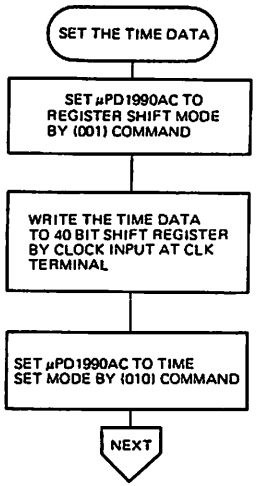
Wait 40 μs until the Register Shift mode becomes valid.

The data of the 40 Bit Shift Register (=Time Data) is shifted to DATA OUT terminal in synchronization with the rising edge of the clock input at CLK terminal.



Note: td1=4 μs (Delay time of μPD1990AC being in the ordinaly mode)
td2=40 μs (Delay time of μPD1990AC being in the Time Read mode)

3-2. How to set the Time Data to the μPD1990AC



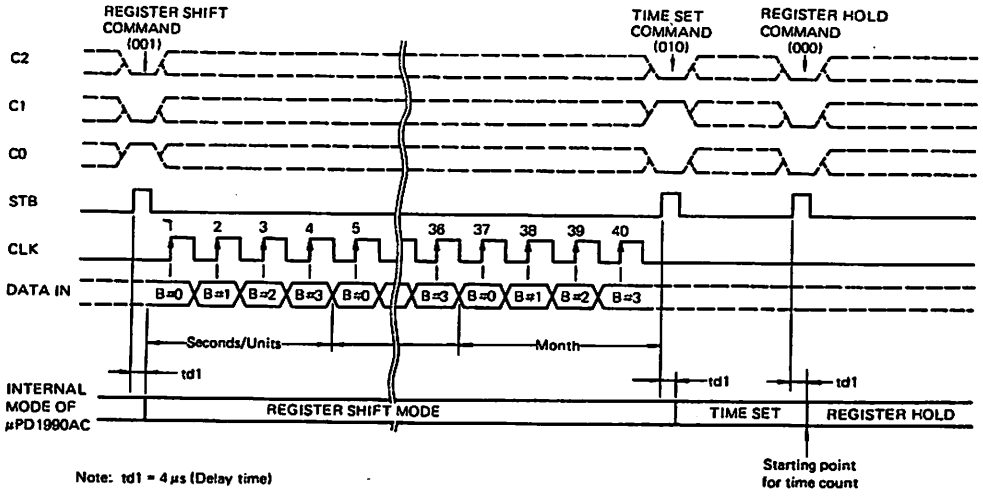
The data of 40 Bit Shift Register can be shifted by the clock input at CLK terminal.

. Note: When Register Shift mode command (001) is set, input level of CLK has to be low level.

Time Data is loaded to 40 Bit Shift Register by setting serial data at DATA IN terminal in synchronous with rising edge of clock input at CLK terminal.

The data of 40 Bit Shift Register (Time Data) is transferred to Time Counter. And Time Counter is kept on being held, during Time Set mode.

Note: As soon as Time Set mode is released by setting Register Hold (000), Register Shift (001) or Time Read (011) command, the Time Counter will start to count time.



Note: td1 = 4 μs (Delay time)

Programm Beispiel:

Die im Folgendem aufgeführten Programmbeispiele sollen die prinzipielle Programmierung des CPU-Moduls verdeutlichen. Besondere Beachtung sind dabei den Initialisierungsroutinen, die die prinzipielle Programmierung der I/O-Bausteine zeigt, zu schenken:

```
CP/M MACRO ASSEM 2.0    £001    CPU-Modul utility subroutines

                MACLIB  Z80

0100            ORG 100H

0011 =          XON     EQU     11H


                ; PORT ADRESS EQUATES

0050 =          P$STI   EQU     50H
0060 =          P$SIO   EQU     60H
0064 =          P$8255  EQU     64H
006A =          P$MMU   EQU     6AH


                ; STI CHANNELS

0050 =          P$IDR   EQU     P$STI +0        ; INDIRECT DATA REGISTER
0051 =          P$GPIP  EQU     P$STI +1        ; GENERAL PURPOSE I/O LINES
0058 =          P$PVR   EQU     P$STI +8        ; POINTER VECTOR REGISTER
005D =          P$RSR   EQU     P$STI +13       ; RECEIVER STATUS
005E =          P$TSR   EQU     P$STI +14       ; TRANSMITTER STATUS
005F =          P$UDR   EQU     P$STI +15       ; USART DATA REGISTER


                ; DART CHANNELS

0060 =          P$SIOA$DAT      EQU     P$SIO +0
0061 =          P$SIOA$CTR      EQU     P$SIO +1
0062 =          P$SIOB$DAT      EQU     P$SIO +2
0063 =          P$SIOB$CTR      EQU     P$SIO +3


                ; PIO 8255 CHANNELS

0064 =          P$CENTR EQU     P$8255 +0       ; PORT A
0065 =          P$TIME  EQU     P$8255 +1       ; PORT B
0067 =          P$MODE  EQU     P$8255 +3       ; CONTROL PORT
```



```
                ;********************** INIT DART ************************
                ;*********************************************************

                INITDART:

                 ; FIRST CHANNEL A
0100 0E61        MVI C,P$SIOA$CTR
0102 0607211001  MVI B,7 ! LXI H,V24ATAB        ; 7 OUT-COMMANDS
                 OUTIR
0107+EDB3        DB      0EDH,0B3H

                 ; CHANNEL B
0109 0E63        MVI C,P$SIOB$CTR
010B 0607        MVI B,7                        ; 7 OUT-COMMANDS
                 OUTIR
010D+EDB3        DB      0EDH,0B3H
010F C9          RET

                V24ATAB:
0110 18          DB      18H             ; CHANNEL RESET
0111 03C1        DB      3,0C1H          ; WR3: RXENABLE, 8 BITS
0113 0444        DB      4,044H          ; WR4: X16 CLOCK MODE, 1 STOP, NO PARITY
0115 05EA        DB      5,0EAH          ; WR5: DTR, 8 BITS, TXENABLE, RTS

                V24BTAB:
0117 18          DB      18H
0118 0341        DB      3,041H          ; WR3: RXENABLE, 7 BITS
011A 044F        DB      4,04FH          ; WR4: X16 CLOCK MODE, 2 STOP, PARITY EVEN
011C 052A        DB      5,02AH          ; WR5: RTS, 7 BITS, TXENABLE



                ;******************* RS 232 A *********************
                ;INPUT/OUTPUT CHARACTERS VIA DART CHANNEL A
                ;**************************************************

                V24A$OUST:

011E DB61        IN P$SIOA$CTR                  ; TEST OUTPUT STATE
0120 E604C8F6FF  ANI 04H ! RZ ! ORI -1 ! RET    ; RET Z=1 IF NOT READY

                V24A$INST:

0126 DB61        IN P$SIOA$CTR                  ; TEST INPUT STATE
0128 E601C8F6FF  ANI 1 ! RZ ! ORI -1 ! RET      ; RET Z=1 IF NO CHAR AVIABLE


                V24A$IN:                        ; INPUT A CHARACTER FROM CHANNEL A

012E CD2601      CALL V24A$INST                 ; TEST INPUT STATE
                 JRZ V24A$IN                    ; LOPP BACK IF NOT READY
0131+28FB        DB      28H,V24A$IN-$-1
0133 DB60C9      IN P$SIOA$DAT ! RET            ; RETURN WITH CHARACTER IN <A>


                V24A$OUT:                       ; OUTPUT CHARACTER IN <C> TO CH. A

0136 CD1E01      CALL V24A$OUST                 ; TEST OUTPUT STATE
                 JRZ V24A$OUT                   ; LOOP BACK IF BUSY
0139+28FB        DB      28H,V24A$OUT-$-1
013B 79D360C9    MOV A,C ! OUT P$SIOA$DAT ! RET ; PUT CHARACTER TO CHANNEL A
```



```
                ;*********** INIT SERIAL TIMER INTERRUPT STI **************
                ;***********************************************************

                STIINIT:

013F 215E01     LXI H,STITAB                    ; POINTER TO INIT-TABLE
0142 0E51060F   MVI C,P$GPIP ! MVI B,15         ; FIRST OF 15 DIRECT REGISTER

                STIDIR:

0146 7E         MOV A,M ! OUTP A                ; OUTPUT CONSTANT
0147+ED79       DB      0EDH,A*8+41H
0149 230C       INX H ! INR C ! DJNZ STIDIR     ; NEXT CONSTANT, NEXT REGISTER
014B+10F9       DB      10H,STIDIR-$-1
014D 0608       MVI B,8                         ; 8 INDIRECT REGISTERS

                STINIDR:

014F DB58E6F8   IN P$PVR ! ANI 0F8H             ; GET STATE OF POINTER VECTOR REG.
0153 05B004     DCR B ! ORA B ! INR B           ; CHOOSE INDIRECT REG., SAVE <B>
0156 D358       OUT P$PVR                       ; SELECT INDIRECT REG. TO PVR
0158 7ED350     MOV A,M ! OUT P$IDR             ; CONSTANT TO INDIRECT-REG.
015B 23         INX H ! DJNZ STIINDR            ; NEXT CONSTANT, NEXT INDIRECT
015C+10F1       DB      10H,STIINDR-$-1



                ;**************** INIT TABLE STI ******************
                ;**************************************************

                STITAB:

                ; 15 DIRECT REGISTERS: START WITH REGISTER 1 GPIP

015E 89         DB      1000$1001B      ; GPIB-MMU AUS,BOOT AUS
015F 0000       DB      0,0             ; CLEAR PENDING INTERRUPTS IPRB, IPRA
0161 0000       DB      0,0             ; CLEAR ISRB, ISRA -> NO INT UNDER SERVICE
0163 0000       DB      0,0             ; CLEAR IMRB, IMRA -> MASK INTERRUPTS
0165 08         DB      08              ; PVR, SOFTWARE END OF INTERRUPT MODE
0166 81         DB      1000$0001B      ; TABCR-TA EVENT COUNT, TB DELAY PRESC./4
0167 02         DB      2               ; TBDR 9600 BD.->TC=2.4576MHZ/(2X4X16X9600)
0168 18         DB      24              ; TADR -> INPUT = 64 HZ -> 4 HZ INTERRUPT
0169 88         DB      1000$1000B      ; UCR /16; 8 BIT; 1STOP; NO PARITY; NONDMA
016A 01         DB      01              ; RCR RECEIVER ENABLE
016B 05         DB      0101B           ; TSR TRANSMITTER ENABLE
016C 11         DB      011H            ; UDR -> XON

                ; 8 INDIRECT REGISTERS: START WITH REGISTER 7 TCDCR

016D 11         DB      0001$0001B      ; TCDCR DELAY MODE,PRESCALE /4
016E 41         DB      0100$0001B      ; DDR BIT0,6=OUTPUT
016F A0         DB      1010$0000B      ; IERA TIMER A, APU END
0170 0E         DB      0000$1110B      ; IERB ENABLE CENTRONICS
0171 26         DB      0010$0110B      ; AER-ACTIVE EDGE REGISTER
0172 10         DB      16              ; TCDR 1200BAUD->TC=2.4576MHZ/(2X16X4X1200)
0173 02         DB      02              ; TDDR 9600BAUD->TC=2.4576MHZ/(2X16X4X9600)
0174 AA         DB      0AAH            ; SCR-SYNCH CHARACTER
```



```
                ;********** KEYBOARD ************
                ; INPUT/OUTPUT CHARACTERS TO KEY
                ; PERFORM REVERSE XON-PROTOCOLL
                ;*******************************

                KEYINP:                                 ; INPUT A CHARACTER FORM KEYBOARD

0175 DB5DE6F0    IN P$RSR ! ANI 0F0H                    ; RECEIVER STAT. CHARACTER AVIABLE ?
                 JRZ KEYINP                             ; NO -> LOOP BACK
0179+28FA        DB      28H,KEYINP-$-1
                 BIT 7,A                                ; BUFFER FULL ?
017B+CB7F        DB      0CBH,7*8+A+40H
                 JRNZ KEYREC                            ; SKIP IF SO
017D+200B        DB      20H,KEYREC-$-1
017F 3E01D35D    MVI A,01H ! OUT P$RSR                  ; ELSE ERRORS, RESET IT
0183 0E11CD9C01  MVI C,XON ! CALL KEYOUTP               ; OUTPUT XON TO KEYBOARD
                 JR KEYINP                              ; NEW RETRY
0188+18EB        DB      18H,KEYINP-$-1

                KEYREC:

018A DB5F47      IN P$UDR ! MOV B,A                     ; SAVE CHARACTER
018D 0E11CD9C01  MVI C,XON ! CALL KEYOUTP               ; SEND XON TO KEY
0192 78C9        MOV A,B ! RET                          ; RET WITH INPUT CHARACTER IN <A>

                KEY$STAT:
                                                        ; TEST RECEIVER STATE
0194 DB5DE680C8  IN P$RSR ! ANI 80H ! RZ                ; RET WITH Z=1 IF NO CHAR. AVIABLE
0199 F6FFC9      ORI -1 ! RET                           ; ELSE RET Z=0

                KEY$OUTP:
                                                        ; OUTPUT CHARACTER IN <C> TO KEY
019C DB5E        IN P$TSR ! BIT 7,A                     ; TRANSMITTER EMPTY ?
019E+CB7F        DB      0CBH,7*8+A+40H
                 JRZ KEY$OUTP                           ; NOT EMPTY ? -> LOOP BACK
01A0+28FA        DB      28H,KEY$OUTP-$-1
01A2 79D35FC9    MOV A,C ! OUT P$UDR ! RET              ; EMPTY, OUTPUT CHARACTER


                ;*********** PIO 8255 INIT - TIME INIT   *************
                ;*****************************************************

                TIMEINIT:

01A6 3E80D367    MVI A,80H ! OUT P$MODE                 ; MODE CONTROL WORD 8255
01AA 3E08D365    MVI A,08H ! OUT P$TIME                 ; RESET STROBE, SET DIR 245 CENTRONICS
01AE 0601CDEE01  MVI B,1 ! CALL CMDTIME                 ; TP = 64 HZ

01B3 C9          RET
```



```
                ;*************** CENTRONICS *****************
                ; OUTPUT CHARACTER IN A TO CENTRONICS PRINTER
                ;********************************************

                CENTOUT:

01B4 CDD401      CALL CENT$STAT ! JRZ CENT$OUT    ; CENTRONICS READY ?
01B7+28FB        DB      28H,CENT$OUT-$-1
01B9 79D364      MOV A,C ! OUT P$CENTR            ; SEND CHARACTER
01BC DB65        IN P$TIME ! RES 3,A              ; PREPARE STROBE
01BE+CB9F        DB      0CBH,3*8+A+80H
01C0 CDCD01      CALL CENT$DELAY                  ; DELAY
01C3 D365        OUT P$TIME ! SETB 3,A            ; GIVE STROBE
01C5+CBDF        DB      0CBH,3*8+A+0C0H
01C7 CDCD01D365  CALL CENTDELAY! OUT P$TIME! RET  ; RESET STROBE AFTER DELAY

                CENT$DELAY:

01CD C5060A      PUSH B! MVI B,10
                 DJNZ CENTDELAY+3                 ; DELAY SAVING <BC>
01D0+10FE        DB      10H,CENTDELAY+3-$-1
01D2 C1C9        POP B! RET

                CENT$STAT:

01D4 DB51E606    IN P$GPIP ! ANI 0000$0110B       ; BUSY, PAPER EMPTY ?
                 JRNZ CENT$NOT$RDY
01D8+2003        DB      20H,CENT$NOT$RDY-$-1
01DA F6FFC9      ORI -1 ! RET                     ; READY -> RET WITH Z=0

                CENT$NOT$RDY:
01DD AFC9        XRA A ! RET                      ; NOT READY -> RET WITH Z = 1



                ;******************* TIME *********************
                ; UTILITY SUBROUTINES
                ; 1. SEND COMMAND TO REAL TIME CLOCK UPD 1990
                ; 2. GIVE PULS TO 1990 CLK INPUT
                ;**********************************************

                CLKTIME:

01DF DB65        IN P$TIME ! SETB 2,A             ; READ PORT B, CLOCK = 1
01E1+CBD7        DB      0CBH,2*8+A+0C0H
01E3 D365        OUT P$TIME                       ; GIVE PULS
01E5 0602        MVI B,2

                CLKTIM1:

                 DJNZ CLKTIM1                     ; DELAY
01E7+10FE        DB      10H,CLKTIM1-$-1
                 RES 2,A
01E9+CB97        DB      0CBH,2*8+A+80H
01EB D365C9      OUT P$TIME ! RET                 ; CLK = L


                CMDTIME:                          ; COMMAND-WORD IN REG <B>

01EE DB65E618    IN P$TIME ! ANI 00011000B        ; MASK BIT 3,4
01F2 B0E6D9      ORA B ! ANI 11011001B            ; SELECT COMMAND, RESET STB,CLK,DIN
01F5 D365        OUT P$TIME                       ; PUT COMMAND OUT
01F7 0603        MVI B,3
```



```
                CMDTIM1:
                 DJNZ CMDTIM1                    ; DELAY 2 US
01F9+10FE        DB      10H,CMDTIM1-$-1
                 SETB 5,A                        ; STROBE = HIGH
01FB+CBEF        DB      0CBH,5*8+A+0C0H
01FD D3650603    OUT P$TIME ! MVI B,3

                CMDTIM2:
                 DJNZ CMDTIM2                    ; DELAY 2 US
0201+10FE        DB      10H,CMDTIM2-$-1
                 RES 5,A
0203+CBAF        DB      0CBH,5*8+A+80H
0205 D365C9      OUT P$TIME ! RET                ; STROBE = LOW


                ;***************** INIT MMU *********************
                ; SELECT FOR ALL SEGMENTS BANK 0 , SWITCH ON MMU
                ;************************************************

                INITMMU:

0208 AF          XRA A          ; REG <A> ADRESS FOR MMU, START WITH SEGM.0
0209 0E6A        MVI C,P$MMU    ; REG <C> CONTAINS MMU-PORTADRESS
020B 16FF        MVI D,-1       ; REG <D> CONTAINS COMPLEMENTED SEGMENT-DATA

                MMULOOP:
020D 47          MOV B,A                    ; SEGMENT-ADRESS IN REG B
                 OUTP D                     ; PROGRAMM MMU-REGISTER
020E+ED51        DB      0EDH,D*8+41H
0210 15          DCR D                      ; NEXT SEGMENT COMPLEMENTED
0211 C610        ADI 10H                    ; NEXT ADRESS IN <A>
                 JRNZ MMULOOP               ; 16 ADRESSES REACHED ?
0213+20F8        DB      20H,MMULOOP-$-1
0215 DB51        IN P$GPIP ! RES 0,A        ; SWITCH ON MMU
0217+CB87        DB      0CBH,0*8+A+80H
0219 D351C9      OUT P$GPIP ! RET


                ;********************** SELECT BANK **************************
                ; REG A CONTAINS BANKADRESS TO SWITCH FROM 0 - F
                ; SWITCH 15 MMU-REGISTERS, LAST REGISTER UNCHANGED
                ; LAST SEGMENT (0F000H - 0FFFFH) IS ALWAYS BANK 0 IN CONTEXT
                ;*************************************************************
F000             ORG 0F000H                 ; SELBANK MAY NOT BE SWITCHED
                SELBANK:

                 REPT 4
                 RLC                        ; BANK ADRESS * 16 -> A16 - A19
                 ENDM
F000+070707      RLC ! RLC ! RLC ! RLC      ; BANK ADRESS * 16 -> A16 - A19
F004 2F57        CMA ! MOV D,A              ; COMPLEMENTED D = DATA MMU
F006 0E6AAF      MVI C,P$MMU ! XRA A        ; C=PORT, A=ADRESSCOUNTER, STARTPAGE = 0

                BANKLOOP:
F009 47          MOV B,A                    ; SEGMENT-ADRESS IN REG B
                 OUTP D                     ; PROGRAMM MMU-REGISTER
F00A+ED51        DB      0EDH,D*8+41H
F00C 15C610      DCR D ! ADI 10H            ; NEXT SEGMENT COMPL. IN D, NEXT ADR. IN A
F00F FEF0        CPI 0F0H                   ; DO NOT SWITCH HIGHSEGM. FROM 0F000H-FFFFH
                 JRC BANK$LOOP
F011+38F6        DB      38H,BANK$LOOP-$-1
F013 C9          RET
```


CPU1 - Bestückungsliste

IC's

IC 00,01	74 LS 245
IC 02	TBP 28 SA 42
IC 03,04	74 LS 245
IC 05	TBP 24 SA 10
IC 06	74 LS 245
IC 07	74 LS 194
IC 08	74 LS 14
IC 09,10	74 S 189
IC 11	AMD 9511
IC 12,13	28 pol.
IC 14	Z80 B CPU
IC 15	TBP 24 S 10
IC 16	74 LS 393
IC 17	74 LS 367
IC 18	74 LS 00
IC 19	74 LS 04
IC 20	upD 1990
IC 21	74 LS 245
IC 22	upD 8255
IC 23	Z80 A STI
IC 24	Z80 B DART
IC 25	ICL 7660
IC 26	MC 1488
IC 27	MC 1489

Dioden

D 00-02	AA 143
D 03	1N 4148
D 04	AA 143
D 05	Z 2,7
D 06	1N 4148

Transistoren

T 00	2 N 4209 (2N 3546)
T 01	2 N 3646

Widerstände

R 00,01	680	R 05	470	R 09	22	R 13	470
R 02	4,7 K	R 06	47 K	R 10	120	R 14	2,2 K
R 03	470	R 07	4,7 K	R 11	390	R 15,16	33 K
R 04	680	R 08	22	R 12	470		
R 17-25	4,7 K
---	---
R 26,27	220
R 28	220
R 29	1 nF

Netzwerke

SIL 0	7 x 680
SIL 1	5 x 4,7 K
SIL 2	7 x 4,7 K

Kondensatoren

C 00-03	4,7 uF, 10V
C 04	100 nF
C 05	4,7 uF, 10V
C 06	4,7 uF, 16V
C 07,08	4,7 uF, 10V
C 09	100 nF
C 10	10 uF, 6,3V
C 11	220 Ohm
C 12	18 pF
C 13	100 nF
C 14-16	10 uF, 16V
C 17,18	4,7 uF, 10V
C 19	15/18/22 pF
C 20	100 nF
C 21	18 pF
C 22	1 nF
C 23	33 pF

Quarze

QU 0	9,8304 MHz
QU 1	32,768 KHz
QU 2	12,000 MHz