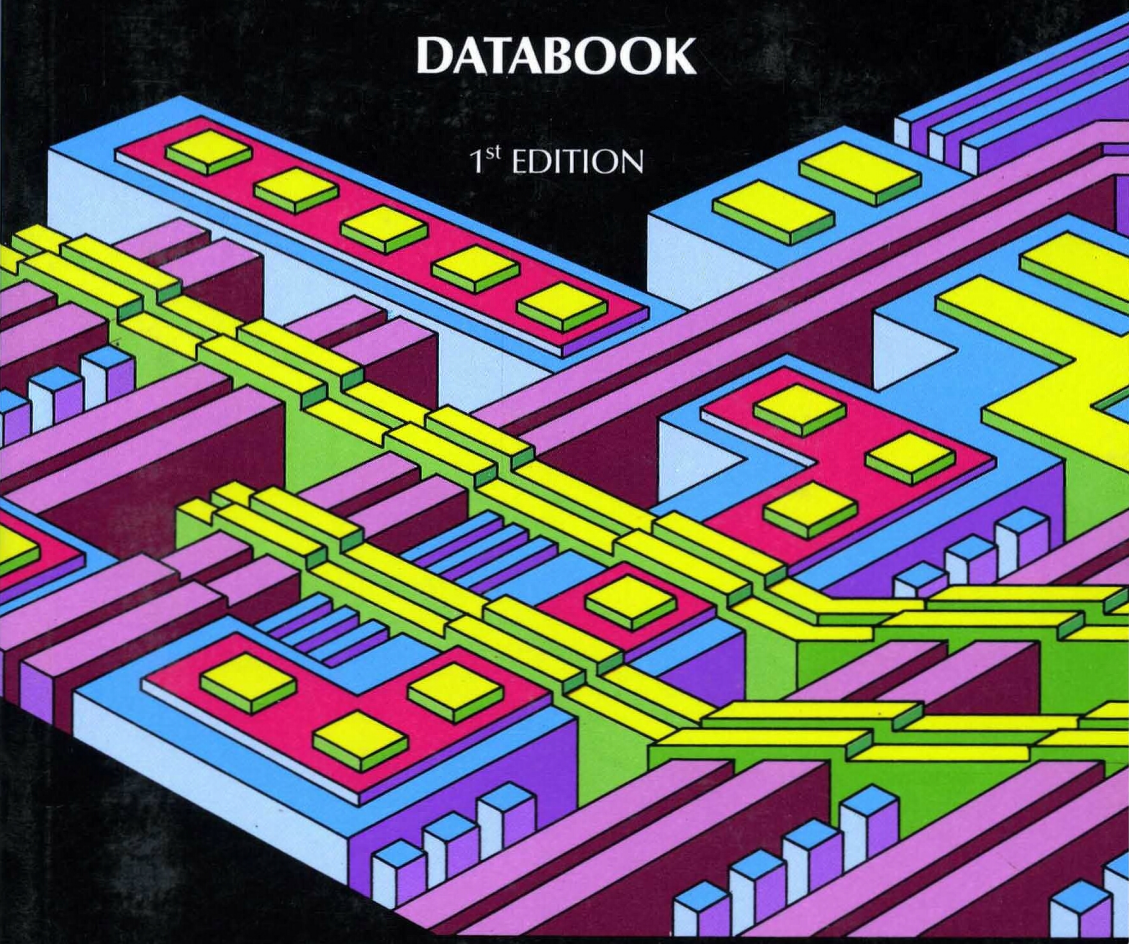
Z80 MICROPROCESSOR FAMILY

DATABOOK

1st EDITION



SGS-THOMSON MICROELECTRONICS

Z80 MICROPROCESSOR FAMILY

1th EDITION

JANUARY 1990

USE IN LIFE SUPPORT DEVICES FOR SYSTEMS MUST BE EXPRESSLY AUTHORIZED

SGS-THOMSON PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF SGS-THOMSON Microelectronics. As used herein:

1. Life support devices or systems are those which (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided with the product, can be reasonably expected to result in significant injury to the user.

2. A critical component is any component of a life support device or system whose failure to perform can reasonably be expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

TABLE OF CONTENTS

ALPHANUMERICAL INDEX

PRODUCT GUIDE

Estabilished as the industry standard, the Z80 CPU offers many features not found on comparable microprocessors: on chip refresh for dynamic memories, comprehensive bit test, reset instructions, block transfer and search instructions, two 16 bit index registers, powerful vectored interrupts and a dual register bank for fast context switching. Complementing the power of the CPU is a complete family of versatile peripheral components.

And today, with the introduction of low power CMOS versions, the Z80 is more than ever the first choice for 8-bit applications.

Central Processor Unit

The Z80 CPU features 158 instruction, software compatible whit all the 78 of the 8080A with added powerfu bit, word and string operations. Three modes of high speed interrupt including a unique vectored interrupt. Dual register sets for context switching plus two 16-bit index registers for memory reference.

Direct Memory Access

The Z80 DMA performs data transfers and searches in a wide variety of 8-bit CPU environments. It is unique among DMA's in that it takes full control of the systems address, data and control buses - and is therefore a special purpose processor - when enabled by the CPU to do so. The DMA also provides complete interfacing to the system bus.

Parallel Input/Output

The Z80 PIO is a programmable two port device which provides a TTL compatible interface between peripheral devices and the Z80 CPU. The programmer can configure the Z80 PIO to interface with a wide range of peripheral devices with no other external logic required.

Counter/Timer Circuit

The Z80 CTC is a programmable component with four independent channel that provide counting and timing functions for microcomputer system based on the Z80 CPU.
The programmer can configure the CTC to operate under various modes to interface with a wide range of devices.

Serial Input/Output

The Z80 SIO is a dual channel multi-function peripheral component designed to satisfy a wide variety of serial data communications requirements in microcomputer systems.
Its basic function is a serial-to-parallel, parallel-to-serial converter/controller but within that role its personality is software configurable so that it can be optimized for a given serial data communications application.

Dual Asynchronous Receiver Transmitter

The Z80 DART is a dual channel multi-function peripheral component that satisfies a wide variety of serial data communications requirements in microcomputer systems. The Z80 DART is used as a serial-to-parallel, parallel-to-serial converter/controller in asynchronous applications.
In addition it also provides modem controls for both channels.

4 Clock Speeds and 4 Package Types

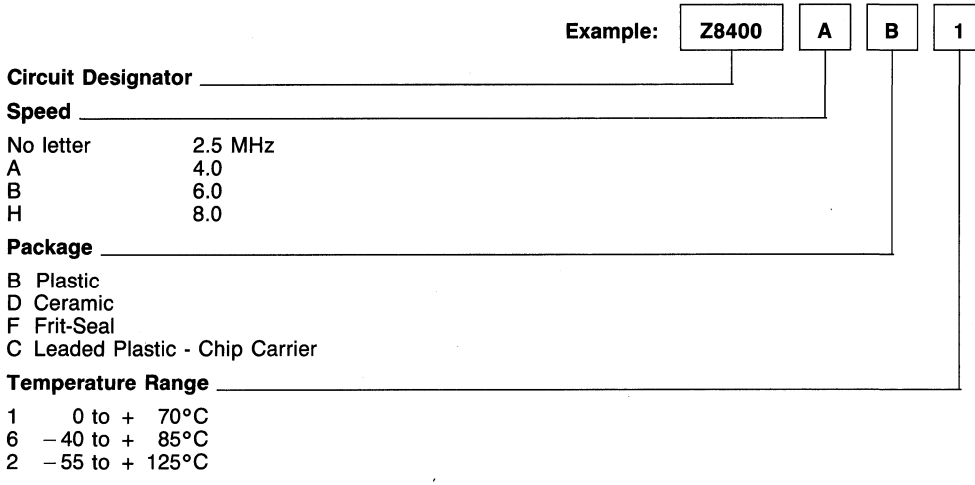
The NMOS & CMOS family Z80 offers a wide selection of speeds with 2.5, 4, 6 and 8 MHz versions and come in plastic and ceramic dual-in-line packages as well as leaded plastic quad-in-line packages.

	SUFFIX DESCRIPTION												
											TEMP. SUFFIX		
	BASE PART NUMBER	SPEED SUFFIX (MHz) 2.5	4.0	6.0	8.0	PACKAGE SUFFIX PL	FR	CE	PL	LC	0/+70°C	−40/+85°C	−55/+125°C
SGS-THOMSON	**Z84XX**	*	**A**	**B**	**H**	**B**	**F**	**D**	**C**	**na**	**1**	**6**	**2**
ZILOG	Z084XX	na	04	06	08	P	D	C	V	L	S	E	M
SGS-THOMSON	**Z84CXX**	**na**	**A**	**B**	**H**	**B**	**na**	**D**	**C**	**na**	**na**	**6**	**2**
TOSHIBA	TMPZ84XX	na	*	−6	−8	P	na	na	T	na	na	*	na
ZILOG	Z084CXX	na	04	06	08	P	D	C	V	L	S	E	M
SHARP	LH508X	na	*	na	na	*	na	na	na	na	na	*	na
NEC	μPD70008AXX	na	−4	−6	−8	*	na	na	na	na	na	*	na

	DEVICE TYPE					
	CPU	**DMA**	**PIO**	**CTC**	**SIO**	**DART**
SGS-THOMSON	**Z8400**	**Z8410**	**Z8420**	**Z8430**	**Z8440/1/2**	**Z8470**
ZILOG	Z08400	Z08410	Z08420	Z08430	Z08440/1/2	na
SGS-THOMSON	**Z84C00**	**Z84C10**	**Z84C20**	**Z84C30**	**Z84C40/1/2**	**na**
TOSHIBA	TMPZ84C00	TMPZ84C10	TMPZ84C20	TMPZ84C30	TMPZ84C40/1/2	na
ZILOG	Z084C00	Z084C10	Z084C20	Z084C30	Z084C40/1/2	na
SHARP	LH5080	LH5083	LH5081	LH5082	na	na
NEC	μPD70008A	na	na	na	na	na

Notes: * Standard Version - No suffix required
na: Not Available

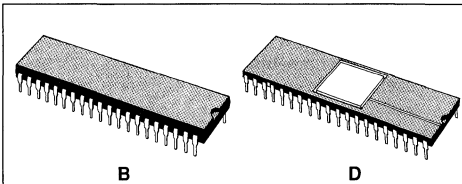
PART NUMBER IDENTIFICATION



CMOS FAMILY DATASHEETS

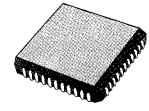
Z80C CPU CMOS VERSION

- THE INSTRUCTION SET CONTAINS 158 INSTRUCTIONS. THE 78 INSTRUCTIONS OF THE 8080A ARE INCLUDED AS A SUBSET ; 8080A AND Z80 SOFTWARE COMPATIBILITY IS MAINTAINED
- 8 MHz, 6 MHz AND 4 MHz CLOCKS FOR THE Z80CH, Z80CB AND Z80CA, THE Z80C CPU, RESULT IN RAPID INSTRUCTION EXECUTION WITH CONSEQUENT HIGH DATA THROUGHPUT
- THE EXTENSIVE INSTRUCTION SET INCLUDES STRING, BIT, BYTE, AND WORD OPERATIONS. BLOCK SEARCHES AND BLOCK TRANSFERS TOGETHER WITH INDEXED AND RELATIVE ADDRESSING RESULT IN THE MOST POWERFUL DATA HANDLING CAPABILITIES IN THE MICROCOMPUTER INDUSTRY
- THE Z80C MICROPROCESSORS AND ASSOCIATED FAMILY OF PERIPHERAL CONTROLLERS ARE LINKED BY A VECTORED INTERRUPT SYSTEM. THIS SYSTEM MAY BE DAISY-CHAINED TO ALLOW IMPLEMENTATION OF A PRIORITY INTERRUPT SCHEME. LITTLE, IF ANY, ADDITIONAL LOGIC IS REQUIRED FOR DAISY-CHAINING
- DUPLICATE SETS OF BOTH GENERAL-PURPOSE AND FLAG REGISTERS ARE PROVIDED, EASING THE DESIGN AND OPERATION OF SYSTEM SOFTWARE THROUGH SINGLE-CONTEXT SWITCHING, BACKGROUND-FOREGROUND PROGRAMMING, AND SINGLE-LEVEL INTERRUPT PROCESSING. IN ADDITION, TWO 16-BIT INDEX REGISTERS FACILITATE PROGRAM PROCESSING OF TABLES AND ARRAYS
- THERE ARE THREE MODES OF HIGH SPEED INTERRUPT PROCESSING : 8080 SIMILAR, NON-Z80 PERIPHERAL DEVICE, AND Z80 FAMILY PERIPHERAL WITH OR WITHOUT DAISY CHAIN
- ON-CHIP DYNAMIC MEMORY REFRESH COUNTER
- SINGLE 5 V ± 10 % POWER SUPPLY
- LOW POWER CONSUMPTION :
  - 9 mA TYP. AT 4 MHz
  - 15 mA TYP. AT 6 MHz
  - 20 mA TYP. AT 8 MHz
  - LESS THAN 10 µA IN POWER DOWN MODE
- EXTENDED OPERATING TEMPERATURE
  - 40 °C TO + 85 °C



B
DIP-40
(Plastic)

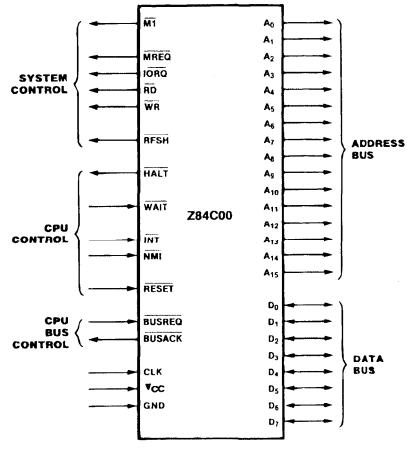
D
DIP-40
(Ceramic)



C
PLCC44
(Plastic)

(Ordering Information at the end of the datasheet)

LOGIC FUNCTIONS



DESCRIPTION

Z80 CMOS Family is fabricated using SGS-THOMSON' CMOS Silicon Gate Technology, which provides low power operation and high performance.

The Z80C CPU is third-generation single-chip microprocessors with exceptional computational power. They offer higher system throughput and more efficient memory utilization than comparable second-and third-generation microprocessors. The internal registers contain 208 bits of read/write memory that are accessible to the programmer. These registers include two sets of six general-purpose registers which may be used individually as either 8-bit registers or as 16-bit register pairs. In addition, there are two sets of accumulator and flag registers. A group of "Exchange" instructions makes either set of main or alternate registers accessible to the programmer. The alternate set allows operation in foreground-background mode or it may be reserved for very fast interrupt response.

The Z80C also contains a Stack Pointer, Program Counter, two index registers, a Refresh register (counter), and an Interrupt register.

The CPU is easy to incorporate into a system since it requires only a single + 5 V power source, all output signals are fully decoded and timed to control standard memory or peripheral circuits, and is supported by an extensive family of peripheral controllers. The internal block diagram (figure 3) shows the primary functions of the Z80C processors. Subsequent text provides more detail on the Z80C I/O controller family, registers, instruction set, interrupts and daisy chaining, and CPU timing.

Figure 1 : Dual in Line Pin Configuration.

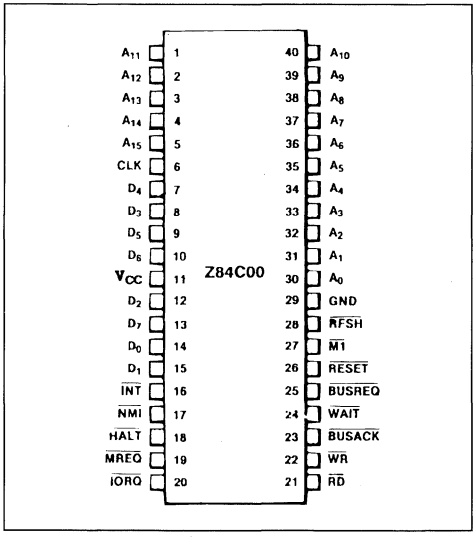


Figure 2 : Chip Carrier Pin Configuration.

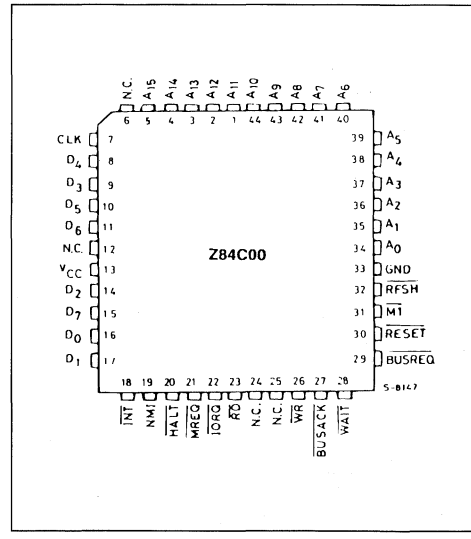
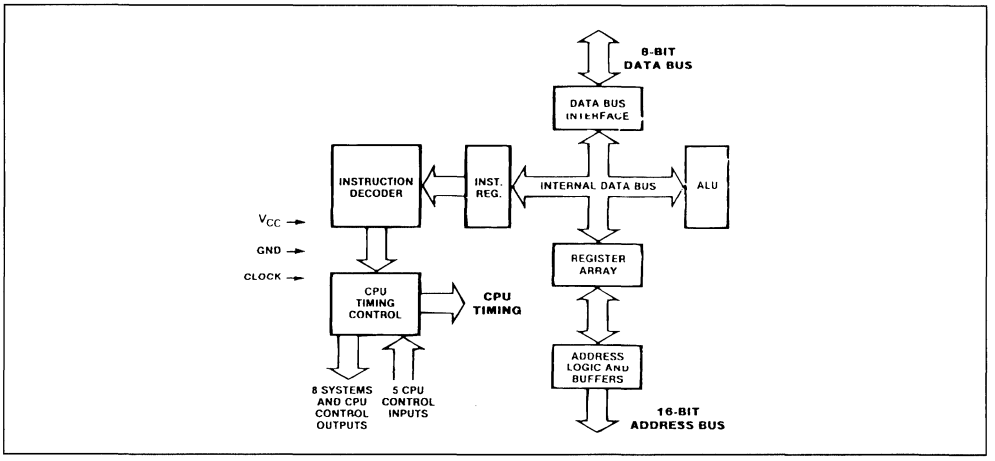


Figure 3 : CPU Block Diagram.



CPU REGISTERS

Figure 4 shows three groups of registers within the CPU. The first group consists of duplicate sets of 8-bit registers : a principal set and an alternate set (designated by '[prime], e.g., A'). Both sets consist

Figure 4 : CPU Registers.

Main Register Set		Alternate Register Set	
A Accumulator	F Flag Register	A' Accumulator	F' Flag Register
B General Purpose	C General Purpose	B' General Purpose	C' General Purpose
D General Purpose	E General Purpose	D' General Purpose	E' General Purpose
H General Purpose	L General Purpose	H' General Purpose	L' General Purpose

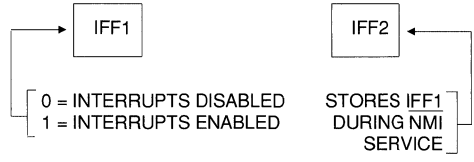
← 8 Bits →

← 16 Bits →

IX Index Register
IY Index Register
SP Stack Pointer
PC Program Counter
I Interrupt Vector / R Memory Refresh

← 8 Bits →

INTERRUPT FLIP-FLOPS STATUS



INTERRUPT MODE FLIP-FLOPS

IMF_a	IMF_b	
0	0	INTERRUPT MODE 0
0	1	NOT USED
1	0	INTERRUPT MODE 1
1	1	INTERRUPT MODE 2

of the Accumulator Register, the Flag Register, and six general-purpose registers. Transfer of data between these duplicate sets of registers is accomplished by use of "Exchange" instructions. The result is faster response to interrupts and easy, efficient implementation of such versatile programming techniques as background-foreground data processing. The second set of registers consists of six registers with assigned functions. These are the I (Interrupt Register), the R (Refresh Register), the IX and IY (Index Registers), the SP (Stack Pointer), and the PC (Program Counter). The third group consists of two interrupt status flip-flops, plus and additional pair of flip-flops which assists in identifying the interrupt mode at any particular time. Table 1 provides further information on these registers.

INTERRUPTS : GENERAL OPERATION

The CPU accepts two interrupt input signals : $\overline{NMI}$ and $\overline{INT}$. The $\overline{NMI}$ is a non-maskable interrupt and has the highest priority. $\overline{INT}$ is a lower priority interrupt and it requires that interrupts be enabled in software in order to operate. $\overline{INT}$ can be connected to multiple peripheral devices in a wired-OR configuration.

The Z80C has a single response mode for interrupt service for the non-maskable interrupt. The maskable interrupt, $\overline{INT}$, has three programmable response modes available.

These are :

- Mode 0 - compatible with the 8080 microprocessor.
- Mode 1 - Peripheral Interrupt service, for use with non-8080/Z80 systems.
- Mode 2 - a vectored interrupt scheme, usually daisy-chained, for use with Z80 Family and compatible peripheral devices.

The CPU services interrupts by sampling the $\overline{NMI}$ and $\overline{INT}$ signals at the rising edge of the last clock of an instruction. Further interrupt service processing depends upon the type of interrupt that was detected. Details on interrupt responses are shown in the CPU Timing Section.

NON-MASKABLE INTERRUPT ($\overline{NMI}$)

The non-maskable interrupt cannot be disabled by program control and therefore will be accepted at all times by the CPU. $\overline{NMI}$ is usually reserved for servicing only the highest priority type interrupts, such as that for orderly shut-down after power failure has been detected.

Table 1. CPU Registers

Register		Size (Bits)	Remarks
A, A'	Accumulator	8	Stores an Operand or the Results of an Operation
F, F'	Flags	8	See Instruction Set.
B, B'	General Purpose	8	Can be used separately or as a 16-bit register with C.
C, C'	General Purpose	8	See B, above.
D, D'	General Purpose	8	Can be used separately or as a 16-bit register with E.
E, E'	General Purpose	8	See D, above.
H, H'	General Purpose	8	Can be used separately or as a 16-bit register with L.
L, L'	General Purpose	8	See H, above. Note : The (B, C), (D, E), and (H, L) sets are combined as follows : B–High Byte C–Low Byte D–High Byte E–Low Byte H–High Byte L–Low Byte
I	Interrupt Register	8	Stores upper eight bits of memory address for vectored interrupt processing.
R	Refresh Register	8	Provides user-transparent dynamic memory refresh. Lower seven bits are automatically incremented and all eight are placed on the address bus during each instruction fetch cycle refresh time.
IX	Index Register	16	Used for indexed addressing.
IY	Index Register	16	Same as IX, above.
SP	Stack Pointer	16	Holds address of the top of the stack. See Push or Pop in Instruction Set.
PC	Program Counter	16	Holds address of next instruction.
IFF_1–IFF_2	Interrupt Enable	Flip-Flops	Set or reset to indicate interrupt status (see figure 4).
IMFa–IMFb	Interrupt Mode	Flip-Flops	Reflect Interrupt Mode (see figure 4).

After recognition of the NMI signal (providing BUSREQ is not active), the CPU jumps to restart location 0066H. Normally, software starting at this address contains the interrupt service routine.

MASKABLE INTERRUPT (INT)

Regardless of the interrupt mode set by the user, the Z80C response to a maskable interrupt input follows a common timing cycle. After the interrupt has been detected by the CPU (provided that interrupts are enabled and BUSREQ is not active) a special interrupt processing cycle begins. This is a special fetch (MI) cycle in Which IORQ becomes active rather than MREQ, as in normal M1 cycle.

In addition, this special M1 cycle is automatically extended by two WAIT states, to allow for the time required to acknowledge the interrupt request.

MODE 0 INTERRUPT OPERATION

This mode is similar with the 8080 microprocessor interrupt service procedures. The interrupting device places an instruction on the data bus.

This is normally a Restart Instruction, which will initiate a call to the selected one of eight restart locations in page zero of memory. Unlike the 8080, the Z80 CPU responds to the Call instruction with only one interrupt acknowledge cycle followed by two memory read cycles.

MODE 1 INTERRUPT OPERATION

Mode 1 operation is very similar to that for the NMI. The principal difference is that the Mode 1 interrupt has a restart location of 0038H only.

MODE 2 INTERRUPT OPERATION

This interrupt mode has been designed to utilize most effectively the capabilities of the Z80C microprocessor and its associated peripheral family. The interrupting peripheral device selects the starting address of the interrupt service routine. It does this by placing an 8-bit vector on the data bus during the interrupt acknowledge cycle. The CPU forms a pointer using this byte as the lower 8-bits and the contents of the I register as the upper 8-bits. This points to an entry in a table of addresses for interrupt service routines. The CPU then jumps to the routine at that address. This flexibility in selecting the interrupt service routine address allows the peripheral device to use several different types of service routines.

These routines may be located at any available location in memory. Since the interrupting device supplies the low-order byte of the 2-byte vector, bit 0 (A_0) must be a zero.

INTERRUPT PRIORITY (Daisy Chaining and Nested Interrupts).

The interrupt priority of each peripheral device is determined by its physical location within a daisy-chain configuration. Each device in the chain has an interrupt enable input line (IEI) and an interrupt enable output line (IEO), which is fed to the next lower priority device. The first device in the daisy chain has its IEI input hardwired to a High level. The first device has highest priority, while each succeding device has a corresponding lower priority. This arrangement permits the CPU to select the highest priority interrupt from several simultaneously interrupting peripherals.

The interrupting device disables its IEO line to the next lower priority peripheral until it has been serviced. After servicing, its IEO line is raised, allowing lower priority peripherals to demand interrupt servicing.

The Z80C CPU will nest (queue) any pending interrupts or interrupts received while a selected peripheral is being serviced.

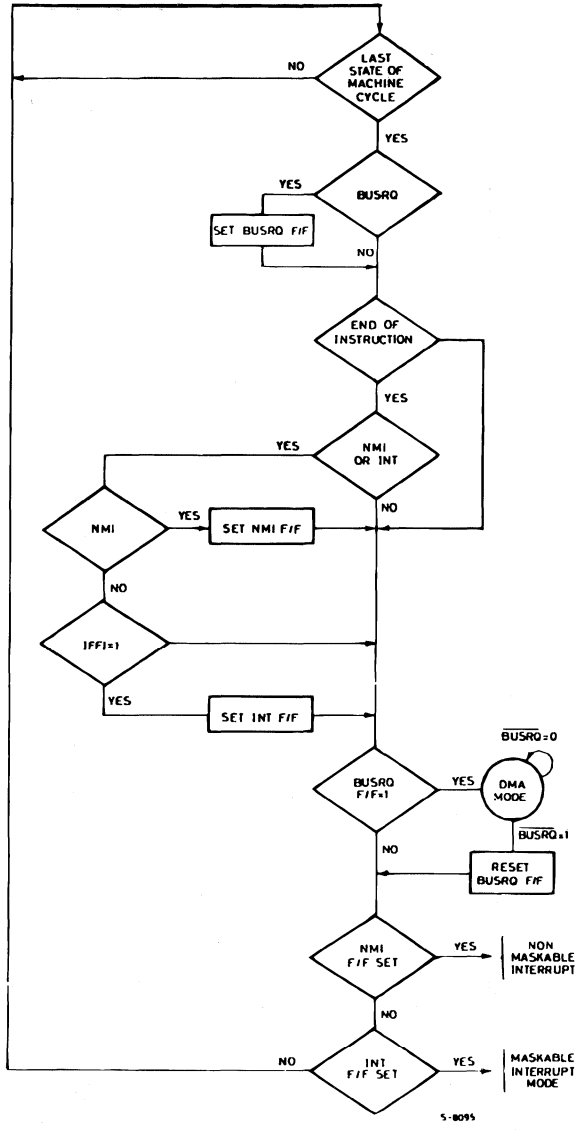
INTERRUPT ENABLE/DISABLE OPERATION

Two flip-flops, IFF_1 and IFF_2, referred to in the register description are used to signal the CPU interrupt status. Operation of the two flip-flops is described in table 2. For more details, refer to the *Z80 CPU Technical Manual.*

Table 2. State of Flip-Flops

Action	IFF_2	IFF_2	Comments
CPU Reset	0	0	Maskable Interrupt INT Disabled
DI Instruction Execution	0	0	Maskable Interrupt INT Disabled
EI Instruction Execution	1	1	Maskable Interrupt INT Enabled
LD A, I Instruction Execution	•	•	$IFF_2 \rightarrow$ Parity Flag
LD A, R Instruction Execution	•	•	$IFF_2 \rightarrow$ Parity Flag
Accept NMI	0	IFF_1	$IFF_1 \rightarrow IFF_2$ (maskable interrupt INT disabled)
RETN Instruction Execution	IFF_2	•	$IFF_2 \rightarrow IFF_1$ at Completion of an NMI Service Routine.

CPU INTERRUPT SEQUENCE



Notes : 1. $\overline{INT}$ and $\overline{NMI}$ are always acted on at the end of an instruction.
2. $\overline{BUSRQ}$ is acted on at the end of a machine cycle.
3. While the CPU is in the DMA MODE, it will not respond to active inputs on $\overline{INT}$ or $\overline{NMI}$.
4. These three inputs are acted on in the following order of priority.
   1) $\overline{BUSRQ}$ -- highest
   2) $\overline{NMI}$
   3) $\overline{INT}$ -- lowest.

INSTRUCTION SET

The Z80C microprocessor has one of the most powerful and versatile instruction sets available in any 8-bit microprocessor. It includes such unique operations as a block move for fast, efficient data transfers within memory or between memory and I/O. It also allows operations on any bit in any location in memory.

The following is a summary of the Z80C instruction set and shows the assembly language mnemonic, the operation, the flag status, and gives comments on each instruction. The *Z80 CPU Technical Manual and Z80 CPU Programming Manual* contain significantly more details for programming use.

The instructions are divided into the following categories :

- 8-BIT LOADS
- 16-BIT LOADS
- EXCHANGES, BLOCK TRANSFERS, AND SEARCHES
- 8-BIT ARITHMETIC AND LOGIC OPERATIONS
- GENERAL-PURPOSE ARITHMETIC AND CPU CONTROL
- 16-BIT ARITHMETIC OPERATIONS
- ROTATES AND SHIFT
- BIT SET, RESET, AND TEST OPERATIONS
- JUMPS
- CALLS, RETURNS, AND RESTARTS
- INPUT AND OUTPUT OPERATIONS

A variety of addressing modes are implemented to permit efficient and fast data transfer between various registers, memory locations, and input/output devices. These addressing modes include :

- Immediate
- Immediate extended
- Modified page zero
- Relative
- Extended
- Indexed
- Register
- Register indirect
- Implied
- Bit

16-BIT ARITHMETIC GROUP

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210					
ADD HL, ss	HL ← HL + ss	•	•	X	X	X	•	0	↕	00 ss1 001		1	3	11	ss Reg.
ADC HL, ss	HL ← HL + ss + CY	↕	↕	X	X	X	V	0	↕	11 101 101	ED	2	4	15	00 BC
										01 ss1 010					01 DE
SBC HL, ss	HL ← HL + ss – CY	↕	↕	X	X	X	V	1	↕	11 101 101	ED	2	4	15	10 IX
										01 ss0 010					11 SP
ADD IX, pp	IX ← + pp	•	•	X	X	X	•	0	↕	11 011 101	DD	2	4	15	pp Reg.
										01 pp1 001					00 BC
ADD IY, rr	IY ← IY + rr	•	•	X	X	X	•	0	↕	11 111 101	FD	2	4	15	01 DE
										00 rr1 001					10 IX
INC ss	ss ← ss + 1	•	•	X	•	X	•	•	•	00 ss0 011		1	1	6	11 SP
INC IX	IX ← IX + 1	•	•	X	•	X	•	•	•	11 011 101	DD	2	2	10	rr Reg.
										00 100 011	23				00 BC
INC IY	IY ← IY + 1	•	•	X	•	X	•	•	•	11 111 101	FD	2	2	10	01 DE
										00 100 011	23				10 IY
DEC ss	ss ↔ ss - 1	•	•	X	•	X	•	•	•	00 ss1 011		1	1	6	11 SP
DEC IX	IX ← IX – 1	•	•	X	•	X	•	•	•	11 011 101	DD	2	2	10	
										00 101 011	2B				
DEC IY	IY ← IY – 1	•	•	X	•	X	•	•	•	11 111 101	FD	2	2	10	
										00 101 011	2B				

Notes : ss is any of the register pairs BC, DE, HL, SP.
pp is any of the register pairs BC, DE, IX, SP.
rr is any of the register pairs BC, DE, IY, SP.

INSTRUCTION SET (continued)

8-BIT LOAD GROUP

Symbol	Symbolic Operation	Flags S	Z		H		P/V	N	C	Opcode 76 543 210	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
LD r, r'	r ← r'	•	•	X	•	X	•	•	•	01 r r'		1	1	4	r, r' Reg.
LD r, n	r ← n	•	•	X	•	X	•	•	•	00 r 110 ← n →		2	2	7	000 B 001 C 010 D 011 E 100 H 101 L 111 A
LD r, (HL)	r ← (HL)	•	•	X	•	X	•	•	•	01 r 110	DD	1	2	7	
LD r, (IX+d)	r ← (IX + d)	•	•	X	•	X	•	•	•	11 011 101 01 r 101 ← d →		3	5	19	
LD r, (IY+d)	r ← (IY + d)	•	•	X	•	X	•	•	•	11 111 101 01 r 110 ← d →	FD	3	5	19	
LD(HL), r	(HL) ← r	•	•	X	•	X	•	•	•	01 110 r		1	2	7	
LD(IX+d), r	(IX + d) ← r	•	•	X	•	X	•	•	•	11 011 101 01 110 r ← d →	DD	3	5	19	
LD(IY+d), r	(IY + d) ← r	•	•	X	•	X	•	•	•	11 111 101 01 110 r ← d →	FD	3	5	19	
LD(HL), n	(HL) ← n	•	•	X	•	X	•	•	•	00 110 110 ← n →	36	2	3	10	
LD(IX+d), n	(IX + d) ← n	•	•	X	•	X	•	•	•	11 011 101 00 110 110 ← d → ← n →	DD 36	4	5	19	
LD(IY+d), n	(IY + d) ← n	•	•	X	•	X	•	•	•	11 111 101 00 110 110 ← d → ← n →	FD 36	4	5	19	
LD A, (BC)	A ← (BC)	•	•	X	•	X	•	•	•	00 001 010	0A	1	2	7	
LD A, (DE)	A ← (DE)	•	•	X	•	X	•	•	•	00 011 010	1A	1	2	7	
LD A, (nn)	A ← (nn)	•	•	X	•	X	•	•	•	00 111 010 ← n → ← n →	3A	3	4	13	
LD(BC), A	(BC) ← A	•	•	X	•	X	•	•	•	00 000 010	02	1	2	7	
LD(DE), A	(DE ← A	•	•	X	•	X	•	•	•	00 010 010	12	1	2	7	
LD(nn), A	(nn) ← A	•	•	X	•	X	•	•	•	00 110 010 ← n → ← n →	32	3	4	13	
LD A, I	A ← I	↕	↕	X	0	X	IFF	0	•	11 101 101 01 010 111	ED 57	2	2	9	
LD A, R	A ← R	↕	↕	X	0	X	IF	0	•	11 101 101 01 011 111	ED 5F	2	2	9	
LD I, A	I ← A	•	•	X	•	X	•	•	•	11 101 101 01 000 111	ED 47	2	2	9	
LD R, A	R ← A	•	•	X	•	X	•	•	•	11 101 101 01 001 111	ED 4F	2	2	9	

Notes : r, r' means any of the registers A, B, C, D, E, H, L.
IFF the content of the interrupt enable flip-flop, (IFF) is copied into the P/V flag.
For an explanation of flag notation and symbols for mnemonic tables, see Symbolic Notation section following tables.

INSTRUCTION SET (continued)

16-BIT LOAD GROUP

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210					
LD dd, nn	dd ← nn	•	•	X	•	X	•	•	•	00 dd0 001 ← n → ← n →		3	3	10	dd Pair 00 BC 01 DE 10 HL 11 SP
LD IX, nn	IX ← nn	•	•	X	•	X	•	•	•	11 011 101 00 100 001 ← n → ← n →	DD 21	4	4	14	
LD IY, nn	IY ← nn	•	•	X	•	X	•	•	•	11 111 101 00 100 001 ← n → ← n →	FD 21	4	4	14	
LD HL, (nn)	H ← (nn + 1) L ← (nn)	•	•	X	•	X	•	•	•	00 101 010 ← n → ← n →	2A	3	5	16	
LD dd, (nn)	dd_H ← (nn + 1) dd_L ← (nn)	•	•	X	•	X	•	•	•	11 101 101 01 dd1 011 ← n → ← n →	ED	4	6	20	
LD IX, (nn)	IX_H ← (nn + 1) IX_L ← (nn)	•	•	X	•	X	•	•	•	11 011 101 01 101 010 ← n → ← n →	DD 2A	4	6	20	
LD IY, (nn)	IY_H ← (nn + 1) IY_L ← (nn)	•	•	X	•	X	•	•	•	11 111 101 00 101 010 ← n → ← n →	FD 2A	4	6	20	
LD (nn), HL	(nn + 1) ← H (nn) ← L	•	•	X	•	X	•	•	•	00 100 010 ← n → ← n →	22	3	5	16	
LD (nn), dd	(nn + 1) ← dd_H (nn) ← dd_L	•	•	X	•	X	•	•	•	11 101 101 01 dd0 011 ← n → ← n →	ED	4	6	20	
LD (nn), IX	(nn + 1) ← IX_H (nn) ← IX_L	•	•	X	•	X	•	•	•	11 011 101 00 100 010 ← n → ← n →	DD 22	4	6	20	
LD (nn), IY	(nn + 1) ← IY_H (nn) ← IY_L	•	•	X	•	X	•	•	•	11 111 101 00 100 010 ← n → ← n →	FD 22	4	6	20	

Notes : dd is any of the register pairs BC, DE, HL, SP.
qq is any of the registers pairs AF, BC, DE, HL.
$(PAIR)_H$, $(PAIR)_L$ refer to high order and low order eight bits of the register pair respectively,
e.g., BC_L = C, AF_H = A.

INSTRUCTION SET (continued)

16-BIT LOAD GROUP (continued)

Symbol	Symbolic Operation	Flags S	Z		H		P/V	N	C	Opcode 76 543 210	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
LD SP, HL	$SP \leftarrow HL$	•	•	X	•	X	•	•	•	11 111 001	F9	1	1	6	
LD SP, IX	$SP \leftarrow IX$	•	•	X	•	X	•	•	•	11 011 101 11 111 001	DD F9	2	2	10	
LD SP, IY	$SP \leftarrow IY$	•	•	X	•	X	•	•	•	11 111 101 11 111 001	FD F9	2	2	10	
PUSH qq	$(SP-2) \leftarrow qq_L$ $(SP-1) \leftarrow qq_H$ $SP \rightarrow SP-2$	•	•	X	•	X	•	•	•	11 qq0 101		1	3	11	qq Pair 00 BC 01 DE 10 HL 11 AF
PUSH IX	$(SP-2) \leftarrow IX_L$ $(SP-1) \leftarrow IX_H$ $SP \rightarrow SP-2$	•	•	X	•	X	•	•	•	11 011 101 11 100 101	DD E5	2	4	15	
PUSH IY	$(SP-2) \leftarrow IY_L$ $(SP-1) \leftarrow IY_H$ $SP \rightarrow SP-2$	•	•	X	•	X	•	•	•	11 111 101 11 100 101	FD E5	2	4	15	
POP qq	$qq_H \leftarrow (SP+1)$ $qq_L \leftarrow (SP)$ $SP \rightarrow SP+2$	•	•	X	•	X	•	•	•	11 qq0 001		1	3	10	
POP IX	$IX_H \leftarrow (SP+1)$ $IX_L \leftarrow (SP)$ $SP \rightarrow SP+2$	•	•	X	•	X	•	•	•	11 011 101 11 100 001	DD E1	2	4	14	
POP IY	$IY_H \leftarrow (SP+1)$ $IY_L \leftarrow (SP)$ $SP \rightarrow SP+2$	•	•	X	•	X	•	•	•	11 111 101 11 100 001	FD E1	2	4	14	

Notes : dd is any of the register pairs BC, DE, HL, SP.
qq is any of the registers pairs AF, BC, DE, HL.
$(PAIR)_H$, $(PAIR)_L$ refer to high order and low order eight bits of the register pair respectively,
e.g., $BC_L = C$, $AF_H = A$.

EXCHANGE, BLOCK TRANSFER, BLOCK SEARCH GROUPS

Symbol	Symbolic Operation	Flags S	Z		H		P/V	N	C	Opcode 76 543 210	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
EX DE, HL, EX AF, AF' EXX	DE ↔ HL, AF ↔ AF' BC ↔ BC' DE ↔ DE' HL ↔ HL'	• •	• •	X X	• •	X X	• •	• •	• •	11 101 011 00 001 000 11 011 001	EB 08 D9	1 1 1	1 1 1	4 4 4	Register Bank and Auxiliary Register Bank Exchange
EX (SP), HL	H ↔ (SP + 1) L ↔ (SP)	•	•	X	•	X	•	•	•	11 100 011	E3	1	5	19	
EX (SP), IX	$IX_H \leftrightarrow (SP+1)$ $IX_L \leftrightarrow (SP)$	•	•	X	•	X	•	•	•	11 011 101 11 100 011	DD E3	2	6	23	
EX (SP), IY	$IY_H \leftrightarrow (SP+1)$ $IY_L \leftrightarrow (SP)$	•	•	X	•	X	•	•	•	11 111 101 11 100 011	FD E3	2	6	23	

Notes : 1. If the result of B – 1 is zero the Z flag is set, otherwise it is reset.
2. Z flag is set upon instruction completion only.

INSTRUCTION SET (continued)

EXCHANGE, BLOCK TRANSFER, BLOCK SEARCH GROUPS (continued)

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210					
LDI	(DE) ← (HL) DE ← DE + 1 HL ← HL + 1 BC ← BC − 1	•	•	X	0	X	① ↕	0	•	11 101 101 10 100 000	ED A0	2	4	16	Load (HL) into (DE), increment the pointers and decrement the byte counter (BC)
LDIR	(DE) ← (HL) DE ← DE + 1 HL ← HL + 1 BC ← BC − 1 Repeat Until BC = 0	•	•	X	0	X	① 0	0	•	11 101 101 10 110 000	ED B0	2 2	5 4	21 16	If BC ≠ 0 If BC = 0
LDD	(DE) ← (HL) DE ← DE + 1 HL ← HL + 1 BC ← BC − 1	•	•	X	0	X	① ↕	0	•	11 101 101 10 101 000	ED A8	2	4	16	
LDDR	(DE) ← (HL) DE ← DE − 1 HL ← HL − 1 BC ← BC − 1 Repeat Until BC = 0	•	•	X	0	X	② 0	0	•	11 101 101 10 111 000	ED B8	2 2	5 4	21 16	If BC ≠ 0 If BC = 0
CPI	A ← (HL) HL ← HL + 1 BC ← BC − 1	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 100 001	ED A1	2	4	16	
CPIR	A − (HL) HL ← HL + 1 BC ← BC − 1 Repeat Until A = (HL) or BC = 0	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 110 001	ED B1	2 2	5 4	21 16	If BC ≠ 0 and A ≠ (HL) If BC = 0 or A = (HL)
CPD	A − (HL) HL ← HL + 1 BC ← BC − 1	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 101 001	ED A9	2	4	16	
CPDR	A − (HL) HL ← HL + 1 BC ← BC − 1 Repeat Until A = (HL) or BC = 0	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 111 001	ED B9	2 2	5 4	21 16	If BC ≠ 0 and A ≠ (HL) If BC = 0 or A = (HL)

Notes : ① If the result of B − 1 is zero the Z flag is set, otherwise it is reset.
② Z flag is set upon instruction completion only.

INSTRUCTION SET (continued)

8–BIT ARITHMETIC AND LOGICAL GROUP

Symbol	Symbolic Operation	Flags								Opcode			Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76	543	210					
ADD A, r	A ← A + r	↕	↕	X	↕	X	V	0	↕	10	[000]	r		1	1	4	r Reg.
ADD A, n	A ← A + n	↕	↕	X	↕	X	V	0	↕	11	[000]	110		2	2	7	000 B
										←	n	→					001 C
ADD A, (HL)	A ← A + (HL)	↕	↕	X	↕	X	V	0	↕	10	[000]	110		1	2	7	010 D
ADD A, (IX+d)	A ← A + (IX + d)	↕	↕	X	↕	X	V	0	↕	11	011	101	DD	3	5	19	011 E
										10	[000]	110					100 H
										←	d	→					101 L
ADD A, (IY+d)	A ← A + (IY + d)	↕	↕	X	↕	X	V	0	↕	11	111	101	FD	3	5	19	111 A
										10	[000]	110					
										←	d	→					
ADC A, s	A ← A + s + CY	↕	↕	X	↕	X	V	0	↕		[001]						s is any of r, n, (HL), (IX+d), (IY+d) as shown for ADD instruction. The indicated bits replace the [000] in the ADD set above.
SUB s	A ← A − s	↕	↕	X	↕	X	V	1	↕		[010]						
SBC A, s	A ← A − s − CY	↕	↕	X	↕	X	V	1	↕		[011]						
AND s	A ← A ∧ s	↕	↕	X	1	X	P	0	0		[100]						
OR s	A ← A ∨ s	↕	↕	X	0	X	P	0	0		[110]						
XOR s	A ← A ⊕ s	↕	↕	X	0	X	P	0	0		[101]						
CP s	A ← s	↕	↕	X	↕	X	V	1	↕		[111]						
INC r	r ← r + 1	↕	↕	X	↕	X	V	0	•	00	r	[100]		1	1	4	
INC (HL)	(HL) ← (HL) + 1	↕	↕	X	↕	X	V	0	•	00	110	[100]		1	3	11	
INC (IX + d)	(IX + d) ←	↕	↕	X	↕	X	V	0	•	11	011	101	DD	3	6	23	
	(IX + d) + 1									00	110	[100]					
										←	d	→					
INC (IY+d)	(IY + d) ←	↕	↕	X	↕	X	V	0	•	11	111	101	FD	3	6	23	
	(IY + d) +1									00	110	[100]					
										←	d	→					
DEC m	m ← m − 1	↕	↕	X	↕	X	V	1	•			[101]					m is any of r, (HL), (IX+d), (IY+d) as shown for INC. DEC same format and states as INC. Replace [100] with [101] in opcode.

INSTRUCTION SET (continued)

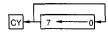
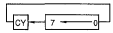
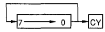
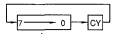
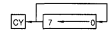
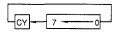
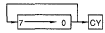
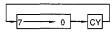
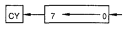
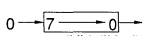
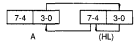
GENERAL-PURPOSE ARITHMETIC AND CPU CONTROL GROUPS

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210					
DAA	Converters acc ; content into packed BCD following add or subtract with packed BCD operands	↕	↕	X	↕	X	P	•	↕	00 100 111	27	1	1	4	Decimal Adjust Accumulator.
CPL	$A \leftarrow \overline{A}$	•	•	X	1	X	•	1	•	00 101 111	2F	1	1	4	Complement Accumulator (one's complement)
NEG	$A \leftarrow 0 - A$	↕	↕	X	↕	X	V	1	↕	11 101 101 01 000 100	ED 44	2	2	8	Negate Acc. (two's complement).
CCF	$CY \leftarrow \overline{CY}$	•	•	X	X	X	•	0	↕	00 111 111	3F	1	1	4	Complement Carry Flag.
SCF	$CY \leftarrow 1$	•	•	X	0	X	•	0	1	00 110 111	37	1	1	4	Set Carry Flag.
NOP	No Operation	•	•	X	0	X	•	0	1	00 000 000	00	1	1	4	
HALT	CPU Halted	•	•	X	•	X	•	•	•	01 110 110	76	1	1	4	
DI*	IFF ← 0	•	•	X	•	X	•	•	•	11 110 011	F3	1	1	4	
EI*	IFF ← 1	•	•	X	•	X	•	•	•	11 111 011	FB	1	1	4	
IM 0	Set Interrupt Mode 0	•	•	X	•	X	•	•	•	11 101 101 01 000 110	ED 46	2	2	8	
IM 1	Set Interrupt Mode 1	•	•	X	•	X	•	•	•	11 101 101 01 010 110	ED 56	2	2	8	
IM2	Set Interrupt Mode 2	•	•	X	•	X	•	•	•	11 101 101 01 011 110	ED 5E	2	2	8	

Notes : IFF indicates the interrupt enable flip-flop. CY indicates the carry flip-flop. * indicates interrupts are not sampled at the end of EI or DI.

INSTRUCTION SET (continued)

ROTATE AND SHIFT GROUP

Mnemonic	Symbolic Operation	S	Z	H	P/V	N	C	Opcode 76 543 210	Hex	No.of Bytes	No.of Cycles	No.of States	Comments
RLCA		•	•	X	0	X	•	0	↕	00 000 111	07	1	1	4	Rotate left circular accumulator.
RLA		•	•	X	0	X	•	0	↕	00 010 111	17	1	1	4	Rotate left accumulator.
RRCA		•	•	X	0	X	•	0	↕	00 001 111	0F	1	1	4	Rotate right circular accumulator.
RRA		•	•	X	0	X	•	0	↕	00 011 111	1F	1	1	4	Rotate right accumulator.
RLC r		↕	↕	X	0	X	P	0	↕	11 001 011 00 [000] r	CB	2	2	8	Rotate left circular register r.
RLC (HL)		↕	↕	X	0	X	P	0	↕	11 001 011 00 [000] 110	CB	2	4	15	r Reg 000 B
RLC (IX + d)		↕	↕	X	0	X	P	0	↕	11 011 101 11 001 011 ← d → 00 [000] 110	DD CB	4	6	23	001 C 010 D 011 E 100 H
RLC (IY + d)	r, (HL), (IX+d), (IY+d)	↕	↕	X	0	X	P	0	↕	11 111 101 11 001 011 ← d → 00 [000] 110	FD CB	4	6	23	101 L 111 A Instruction format
RL m	 m ≈ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	[010]					and states are as shown for RLC's. To form new opcode replace [000] or RLC's with shown code.
RRC m	 m ≈ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	[001]					
RR m	 m ≈ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	[011]					
SLA m	 m ≈ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	[100]					
SRA m	m ≈ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	[101]					
SRL m	 m ≈ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	[111]					
RLD		↕	↕	X	0	X	P	0	•	11 101 101 01 101 111	ED 6F	2	5	18	Rotate digit left and right between the accumulator and location (HL). The content of the upper half of the accumulator is unaffected.
RRD		↕	↕	X	0	X	P	0	•	11 101 101 01 100 111	ED 67	2	5	18	

INSTRUCTION SET (continued)

BIT SET, RESET AND TEST GROUP

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210					
BIT b, r	$Z \leftarrow r_b$	X	↕	X	1	X	X	0	•	11 001 011 01 b r	CB	2	2	8	r, Reg. 000 B 001 C 010 D 011 E 100 H 101 L 111 A
BIT b, (HL)	$Z \leftarrow (\overline{HL})_b$	X	↕	X	1	X	X	0	•	11 001 011 01 b 110	CB	2	3	12	
BIT b, (IX+d)$_b$	$Z \leftarrow (\overline{IX+d})_b$	X	↕	X	1	X	X	0	•	11 011 101 11 001 011 ← d → 01 b 110	DD CB	4	5	20	
BIT b, (IY+d)$_b$	$Z \leftarrow (\overline{IY+d})_b$	X	↕	X	1	X	X	0	•	11 111 101 11 001 011 ← d → 01 b 110	FD CB	4	5	20	b Bit Tested 000 0 001 1 010 2 011 3 100 4 101 5 110 6 111 7
SET b, r	$r_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 001 011 [11] b r	CB	2	2	8	
SET b, (HL)	$(HL)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 001 011 [11] b 110	CB	2	4	15	
SET b, (IX+d)	$(IX+d)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 011 101 11 001 011 ← d → [11] b 110	DD CB	4	6	23	
SET b, (IY+d)	$(IY+d)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 111 101 11 001 011 ← d → [11] b 110	FD CB	4	6	23	
RES b, m	$m_b \leftarrow 0$ m ≡ r, (HL), (IX + d), (IY + d)	•	•	X	•	X	•	•	•	[10]					To form new opcode replace [11] of SET b, s with [10]. Flags and time states for SET instruction.

Notes : The notation m_b indicates bit b (0 to 7) of location m.

INSTRUCTION SET (continued)

JUMP GROUP

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543210					
JP nn	PC ← nn	•	•	X	•	X	•	•	•	11 000 011 ← n → ← n →	C3	3	3	10	
JP cc, nn	If condition cc is true PC ← nn, otherwise continue	•	•	X	•	X	•	•	•	11 cc 010 ← n → ← n →		3	3	10	cc Condition 000 NZ non-zero 001 Z zero 010 NC non-carry 011 C carry 100 PO parity odd 101 PE parity even 110 P sign positive 111 M sign negative
JR e	PC ← PC + e	•	•	X	•	X	•	•	•	00 011 000 ← e–2 →	18	2	3	12	
JR C, e	If C = 0, continue If C = 1, PC ← PC+e	•	•	X	•	X	•	•	•	00 111 000 ← e–2 →	38	2 2	2 3	7 12	If condition not met. If condition is met.
JR NC, e	If C = 1, continue If C = 0, PC ← PC+e	•	•	X	•	X	•	•	•	00 110 000 ← e–2 →	30	2 2	2 3	7 12	If condition not met. If condition is met.
JP Z, e	If Z = 0 continue If Z = 1, PC ← PC+e	•	•	X	•	X	•	•	•	00 101 000 ← e–2 →	28	2 2	2 3	7 12	If condition not met. If condition is met.
JR NZ, e	If Z = 1, continue If Z = 0, PC ← PC+e	•	•	X	•	X	•	•	•	00 100 000 ← e–2 →	20	2 2	2 3	7 12	If condition not met. If condition is met.
JP (HL)	PC ← HL	•	•	X	•	X	•	•	•	11 101 001	E9	1	1	4	
JP (IX)	PC ← IX	•	•	X	•	X	•	•	•	11 011 101 11 101 001	DD E9	2	2	8	
JP (IY)	PC ← IY	•	•	X	•	X	•	•	•	11 111 101 11 101 001	FD E9	2	2	8	
DJNZ, e	B ← B – 1 If B = 0, continue If B ≠ 0, PC ← PC+e	•	•	X	•	X	•	•	•	00 010 000 ← e–2 →	10	2 2	2 3	8 13	If B = 0. If B ≠ 0.

Notes : e represents the extension in the relative addressing mode. e is signed two's complement number in the range < – 126, 129 >. e – 2 in the opcode provides an effective address of pc + e as PC is incremented by 2 prior to the addition of e.

INSTRUCTION SET (continued)

CALL AND RETURN GROUP

Symbol	Symbolic Operation	Flags S	Z		H		P/V	N	C	Opcode 76 543 210	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
CALL nn	(SP – 1) ← PC_H (SP – 2) ← PC_L PC ← nn	•	•	X	•	X	•	•	•	11 001 101 ← n → ← n →	CD	3	5	17	
CALL cc, nn	If condition cc is false continue, otherwise same as CALL nn	•	•	X	•	X	•	•	•	11 cc 100 ← n → ← n →		3 3	3 5	10 17	If cc is false. If cc is true.
RET	PC_L ← (SP) PC_H ← (SP + 1)	•	•	X	•	X	•	•	•	11 001 001	C9	1	3	10	
RET cc	If condition cc is false continue, otherwise same as RET	•	•	X	•	X	•	•	•	11 cc 000		1 1	1 3	5 11	If cc is false. If cc is true
RETI	Return from interrupt	•	•	X	•	X	•	•	•	11 101 101 01 001 101	ED 4D	2	4	14	
RETN [1]	Return from non-maskable interrupt	•	•	X	•	X	•	•	•	11 101 101 01 000 101	ED 45	2	4	14	
RSTp	(SP – 1) ← PC_H (SP – 2) ← PC_L PC_H ← 0 PC_L ← p	•	•	X	•	X	•	•	•	11 t 111		1	3	11	

cc Condition
000 NZ Non-zero
001 Z Zero
010 NC Non-carry
011 C Carry
100 PO Parity Odd
101 PE Parity Even
110 P Sign Positive
111 M Sign Negative

t p
000 0H
001 08H
010 10H
011 18H
100 20H
101 28H
110 30H
111 38H

Note : RETN loads IFF_2 – IFF_1.

INSTRUCTION SET (continued)

INPUT AND OUTPUT GRUP

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210					
IN A, (n)	A ← (n)	•	•	X	•	X	•	•	•	11 011 011 ← n →	DB	2	3	11	n to A_0 ~ A_7 Acc. to A_8 ~ A_{15}
IN r, (C)	r ← (C) If r = 110 only the flags will be affected	↕	↕	X	↕	X	P	0	•	11 101 101 01 r 000	ED	2	3	12	C to A_0 ~ A_7 B to A_8 ~ A_{15}
INI	(HL) ← (C) B ← B – 1 HL ← HL + 1	X	① ↕	X	X	X	X	1	X	11 101 101 10 100 010	ED A2	2	4	16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
INIR	(HL) ← (C) B ← B – 1 HL ← HL + 1 Repeat unil B = 0	X	1	X	X	X	X	1	X	11 101 101 10 110 010	ED B2	2 2	5 (if B≠0) 4 (if B=0)	21 16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
IND	(HL) ← (C) B ← B – 1 HL ← HL – 1	X	① ↕	X	X	X	X	1	X	11 101 101 10 101 010	ED AA	2	4	16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
INDR	(HL) ← (C) B ← B – 1 HL ← HL – 1 Repeat until B = 0	X	1	X	X	X	X	1	X	11 101 101 10 111 010	ED BA	2 2	5 (if B≠0) 4 (if B=0)	21 16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
OUT (n), A	(n) ← A	•	•	X	•	X	•	•	•	11 010 011 ← n →	D3	2	3	11	n to A_0 ~ A_7 Acc.to A_8~A_{15}
OUT (C), r	(C) ← r	•	•	X	•	X	•	•	•	11 101 101 01 r 001	ED	2	3	12	C to A_0 ~ A_7 B to A_8 ~ A_{15}
OUTI	(C) ← (HL) B ← B – 1 HL ← HL + 1	X	① ↕	X	X	X	X	1	X	11 101 101 10 100 011	ED A3	2	4	16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
OTIR	(C) ← (HL) B ← B – 1 HL ← HL + 1 Repeat until B = 0	X	1	X	X	X	X	1	X	11 101 101 10 110 011	ED B3	2 2	5 (if B≠0) 4 (if B=0)	21 16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
OUTD	(C) ← (HL) B ← B – 1 HL ← HL – 1	X	① ↕	X	X	X	X	1	X	11 101 101 10 101 011	ED AB	2	4	16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
OTDR	(C) ← (HL) B ← B – 1 HL ← HL + 1 Repeat until B = 0	X	① 1	X	X	X	X	1	X	11 101 101 10 111 011	ED	2 2	5 (if B≠0) 4 (if B=0)	21 16	C to A_0 ~ A_7 B to A_8 ~ A_{15}

Note : 1. If the result of B – 1 is zero the Z flag is set, otherwise it is reset.

SUMMARY OF FLAG OPERATION

Symbol	Operation
S	Sign Flag. S = 1 if the MSB of the result is 1.
Z	Zero Flag. Z = 1 if the result of the operation is 0.
P/V	Parity or Overflow Flag. Parity (P) and overflow (V) share the same flag. Logical operations affect this flag with the parity of the result while arithmetic operations affect this flag with the overflow of the result. If P/V holds parity, P/V = 1 if the result of the operation is even, P/V = 0 if result is odd. If P/V holds overflow, P/V = 1 if the result of the operation produced an overflow.
H	Half-carry Flag. H = 1 if the add or subtract operation produced a carry into or borrow from bit 4 of the accumulator.
N	Add/Subtract Flag. N = 1 if the previous operation was a subtract.
H & N	H and N flags are used in conjunction with the decimal adjust instruction (DAA) to properly correct the result into packed BCD format following addition or subtraction using operands with packed BCD format.
C	Carry/Link Flag. C = 1 if the operation produced a carry from the MSB of the operand or result.
↕	The flag is affected according to the result of the operation.
•	The flag is unchanged by the operation.
0	The flag is reset by the operation.
1	The flag is set by the operation.
X	The flag is a "don't care".
V	P/V flag affected according to the overflow result of the operation.
P	P/V flag affected according to the parity result of the operation.
r	Any one of the CPU Registers A, B, C, D, E, H, L.
s	Any 8-bit location for all the addressing modes allowed for the particular instruction.
ss	Any 16-bit location for all the addressing modes allowed for that instruction.
ii	Any one of the two Index registers IX or IY.
R	Refresh Counter
n	8-bit Value in Range < 0.255 >
nn	16-bit Value in Range < 0.65535 >

SYMBOLIC NOTATION

Instruction	D_7 S	Z		H		P/V	N	D_0 C	Comments
ADD A, s ; ADC A, s	↕	↕	X	↕	X	V	0	↕	8-bit Add or Add with Carry.
SUB s; SBC A, s ; CP s ; NEG	↕	↕	X	↕	X	V	1	↕	8-Bit subtract, subtract with carry, compare and negate accumulator.
AND s OR s, XOR s	↕ ↕	↕ ↕	X X	1 0	X X	P P	0 0	0 0	Logical Operations
INC s	↕	↕	X	↕	X	V	0	•	8-bit Increment
DEC s	↕	↕	X	↕	X	V	1	•	8-bit Decrement
ADD DD, ss	•	•	X	X	X	•	0	↕	16-bit Add
ADC HL, ss	↕	↕	X	X	X	V	0	↕	16-bit Add with Carry
SBC HL, ss	↕	↕	X	X	X	V	1	↕	16-bit Subtract with Carry.
RLA, RLCA, RRA ; RRCA	•	•	X	0	X	•	0	↕	Rotate Accumulator.
RL m ; RLC m ; RR m ; RRC m ; SLA m SRA m ; SRL m	↕	↕	X	0	X	P	0	↕	Rotate and Shift Locations.
RLD ; RRD	↕	↕	X	0	X	P	0	•	Rotate Digit Left and Right
DAA	↕	↕	X	↕	X	P	•	↕	Decimal Adjust Accumulator.
CPL	•	•	X	1	X	•	1	•	Complement Accumulator
SCF	•	•	X	0	X	•	0	1	Set Carry
CCF	•	•	X	X	X	•	0	↕	Complement Carry
IN r (C)	↕	↕	X	0	X	P	0	•	Input Register Indirect
INI, IND, OUTI ; OUTD INIR ; INDR ; OTIR ; OTDR	X X	↕ 1	X X	X X	X X	X X	1 1	• •	Block Input and Output. Z = 0 if B ≠ otherwise Z = 0
LDI ; LDD LDIR ; LDDR	X X	X X	X X	0 0	X X	↕ 0	0 0	• •	Block Transfer Instructions. P/V = 1 if BC ≠ 0, otherwise P/V = 0
CPI ; CPIR ; CPD ; CPDR	X	↕	X	X	X	↕	1	•	Block Search Instructions. Z = 1 if A = (HL),otherwise Z = 0. P/V = 1 if BC ≠ 0, otherwise P/V = 0.
LD A, I ; LD A, R	↕	↕	X	0	X	IFF	0	•	The content of the interrupt enable flip-flop (IFF) is copied into P/V flag.
BIT b, s	X	↕	X	1	X	X	0	•	The state of bit b of location is copied into the Z flag.

PIN DESCRIPTIONS

A_0-A_{15}. *Address Bus* (Output, Active High, 3-state). A_0-A_{15} form a 16-bit address bus. The Address Bus provides the address for memory data bus exchanges (up to 64 K bytes) and for I/O device exchanges.

$\overline{BUSACK}$. *Bus Acknowledge* (Output, Active Low). Bus Acknowledge indicates to the requesting device that the CPU address bus, data bus, and control signals MREQ, IORQ, RD, and WR have entered their high-impedance states. The external circuitry can now control these lines.

$\overline{BUSREQ}$. *Bus Request* (Input, Active Low). Bus Request has a higher priority than NMI and is always recognized at the end of the current machine cycle. $\overline{BUSREQ}$ forces the CPU address bus, data bus, and control signals MREQ, IORQ, RD, and WR to go to a high-impedance state so that other devices can control these lines. $\overline{BUSREQ}$ is normally wire-ORed and requires an external pullup for these applications. Extended $\overline{BUSREQ}$ periods due to extensive DMA operations can prevent the CPU from properly refreshing dynamic RAMs.

D_0-D_7. *Data Bus* (Input/Output), active High, 3-state). D_0-D_7 constitute an 8-bit bidirectional data bus, used for data exchanges with memory and I/O.

$\overline{HALT}$. *Halt State* (Output, Active Low). $\overline{HALT}$ indicates that the CPU has executed a Halt instruction and is awaiting either a non-maskable or a maskable interrupt (with the mask enabled) before operation can be resumed.

While halted, the CPU executes NOPs to maintain memory refresh.

$\overline{INT}$. *Interrupt Request* (Input, Active Low). Interrupt Request is generated by I/O devices. The CPU honors a request at the end of the current instruction if the internal software-controlled interrupt enable flip-flop (IFF) is enabled. $\overline{INT}$ is normally wire-ORed and requires an external pullup for these applications.

$\overline{IORQ}$. *Input/Output Request* (Output, Active Low, 3-state). IORQ indicates that the lower half of the address bus holds a valid I/O address for an I/O read or write operation.

$\overline{IORQ}$ is also generated concurrently with $\overline{M1}$ during an interrupt acknowledge cycle to indicate that an interrupt response vector can be placed on the data bus.

$\overline{M1}$. *Machine Cycle One* (Output, Active Low). $\overline{M1}$, together with MREQ, indicates that the current machine cycle is the opcode fetch cycle of an instruction execution. $\overline{M1}$, together with $\overline{IORQ}$, indicates an interrupt acknowledge cycle.

$\overline{MREQ}$. Memory Request (Output, Active Low, 3-state). $\overline{MREQ}$ indicates that the address bus holds a valid address for a memory read or memory write operation.

$\overline{NMI}$. *Non-Maskable Interrupt* (Input, Negative edge-triggered). $\overline{NMI}$ has a higher priority than $\overline{INT}$. $\overline{NMI}$ is always recognized at the end of the current instruction, independent of the status of the interrupt enable flip-flop, and automatically forces the CPU to restart at location 0066H.

$\overline{RD}$. *Read* (Output, Active Low, 3-state). $\overline{RD}$ indicates that the CPU wants to read data from memory or an I/O device. The addressed I/O device or memory should use this signal to gate data onto the CPU data bus.

$\overline{RESET}$. *Reset* (Input, Active Low). $\overline{RESET}$ initializes the CPU as follows : it resets the interrupt enable flip-flop, clears the PC and Registers I and R, and sets the interrupt status to Mode 0. During reset time, the address and data bus go to a high-impedance state, and all control output signals go to the inactive state.

Note that $\overline{RESET}$ must be active for a minimum of three full clock cycles before the reset operation is complete.

$\overline{RFSH}$. *Refresh* (Output, Active Low). $\overline{RFSH}$, together with MREQ, indicates that the lower seven bits of the system's address bus can be used as a refresh address to the system's dynamic memories.

$\overline{WAIT}$. *Wait* (Input, Active Low). $\overline{WAIT}$ indicates to the CPU that the addressed memory or I/O devices are not ready for a data transfer. The CPU continues to enter a Wait state as long as this signal is active. Extended $\overline{WAIT}$ periods can prevent the CPU from refreshing dynamic memory properly.

$\overline{WR}$. *Write* (Output, Active Low, (3-state). $\overline{WR}$ indicates that the CPU data bus holds valid data to be stored at the addressed memory or I/O location.

CPU TIMING

The Z80C CPU executes instructions by proceeding through a specific sequence of operations :

- Memory read or write
- I/O device read or write
- Interrupt acknowledge

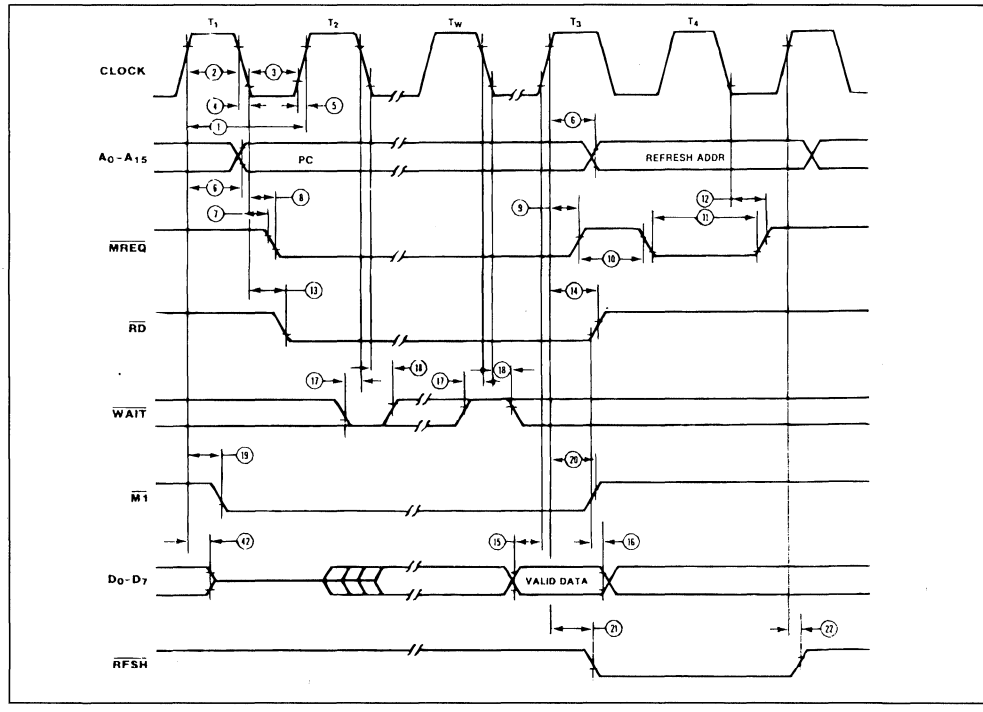
The basic clock period is referred to as a T time or cycle, and three or more T cycles make up a machine cycle (M1, M2 or M3 for instance). Machine cycles can be extended either by the CPU automatically inserting one or more Wait states or by the insertion of one or more Wait states by the user.

INSTRUCTION OPCODE FETCH

The CPU places the contents of the Program Counter (PC) on the address bus at the start of the cycle (figure 5). Approximately one-half clock cycle later, MREQ goes active. When active, RD indicates that the memory data can be enabled onto the CPU data bus.

The CPU samples the WAIT input with the falling edge of clock state T_2. During clock states T_3 and T_4 of an M1 cycle dynamic RAM refresh can occur while the CPU starts decoding and executing the instruction. When the Refresh Control signal becomes active, refreshing of dynamic memory can take place.

Figure 5 : Instruction Opcode Fetch.



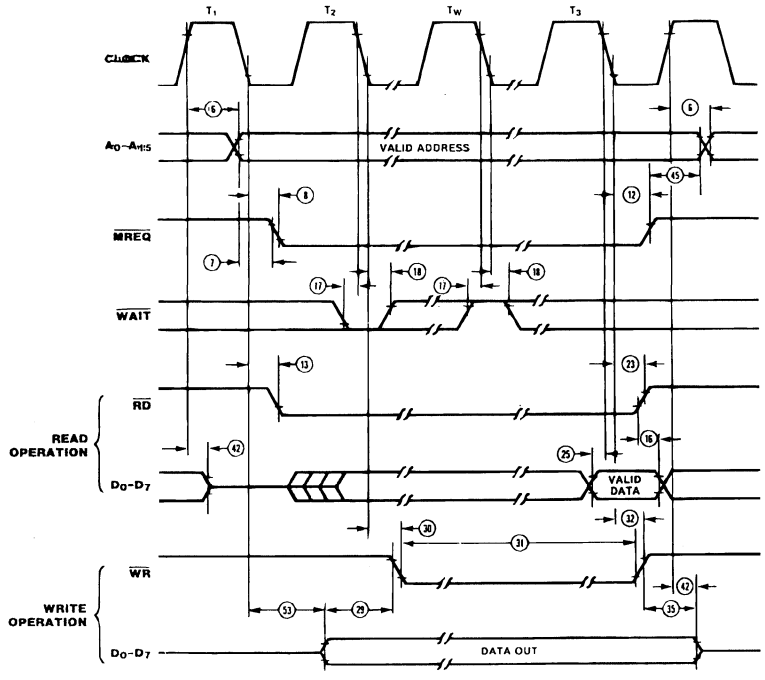
Note : T_W-Wait cycle added when necessary for slow ancilliary devices.

MEMORY READ OR WRITE CYCLES

Figure 6 shows the timing of memory read or write cycles other than an opcode fetch (M1) cycle. The MREQ and RD signals function exactly as in the fetch cycle. In a memory write cycle, MREQ also becomes active when the address bus is stable.

The WR line is active when the data bus is stable, so that it can be used directly as an R/W pulse to most semiconductor memories.

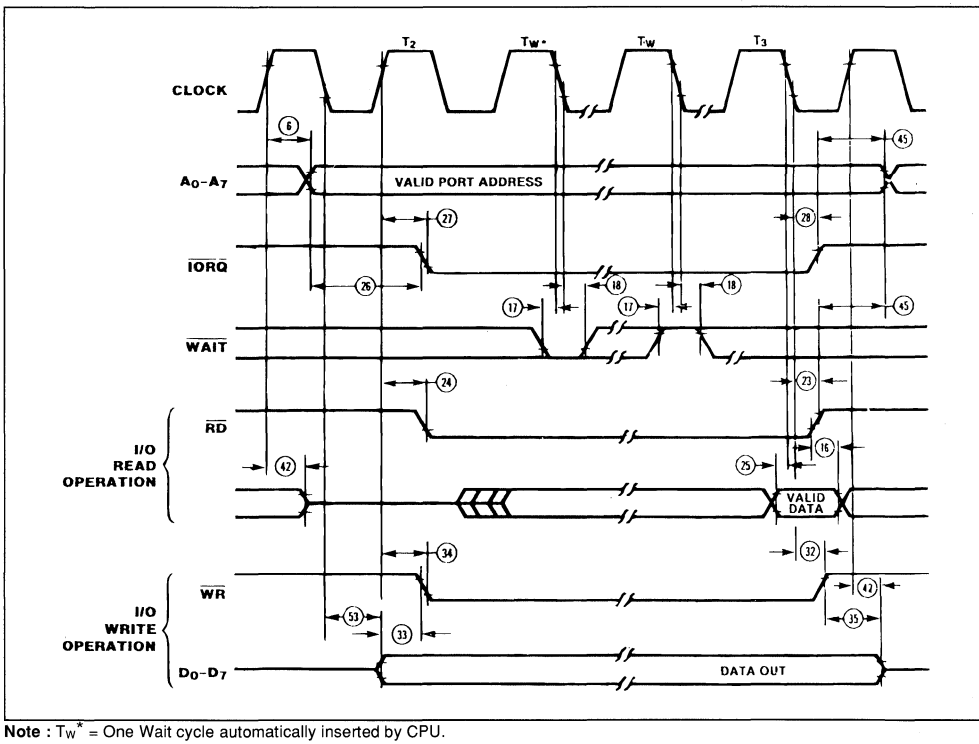
Figure 6 : Memory Read or Write Cycles.



INPUT OR OUTPUT CYCLES

Figure 7 shows the timing for an I/O read or I/O write operation. During I/O operations, the CPU automatically inserts a single Wait state (T_W). This extra Wait state allows sufficient time for an I/O port to decode the address from the port address lines.

Figure 7 : Input or Output Cycles.



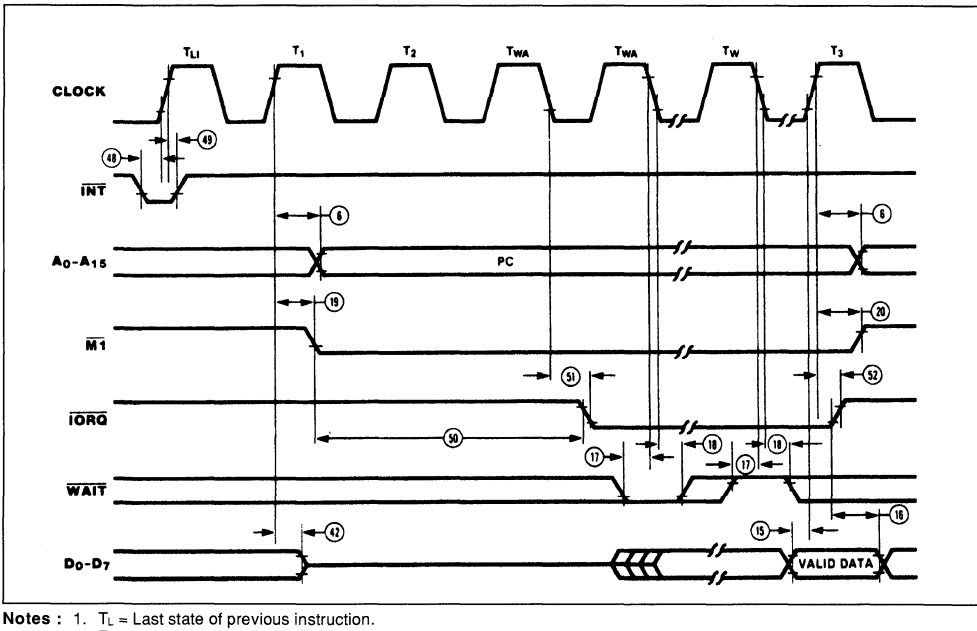
Note : T_W^* = One Wait cycle automatically inserted by CPU.

INTERRUPT REQUEST/ACKNOWLEDGE CYCLE

The CPU samples the interrupt signal with the rising edge of the last clock cycle at the end of any instruction (figure 8). When an interrupt is accepted, a special $\overline{M1}$ cycle is generated.

During this $\overline{M1}$ cycle, $\overline{IORQ}$ becomes active (instead of $\overline{MREQ}$) to indicate that the interrupting device can place an 8-bit vector on the data bus. The CPU automatically adds two Wait states to this cycle.

Figure 8 : Interrupt Request/Acknowledge Cycle.

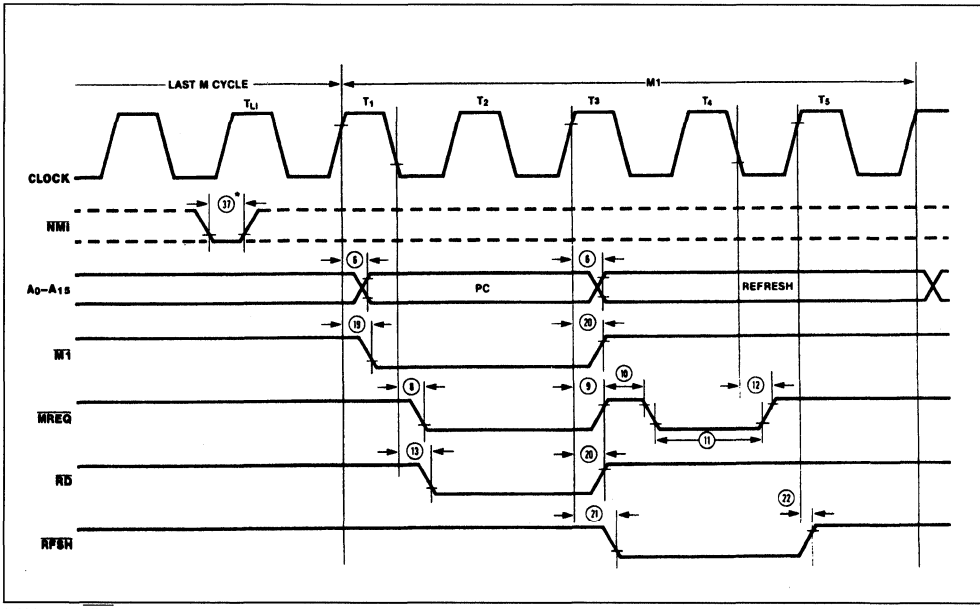


Notes : 1. T_L = Last state of previous instruction.
2. Two Wait cycles automatically inserted by CPU (*).

NON-MASKABLE INTERRUPT REQUEST CYCLE

$\overline{NMI}$ is sampled at the same time as the maskable interrupt input $\overline{INT}$ bus has higher priority and cannot be disabled under software control. The subsequent timing is similar to that of a normal instruction fetch except that data put on the bus by the memory is ignored. The CPU instead executes a restart (RST) operation and jumps to the $\overline{NMI}$ service routine located at address 0066H (figure 9).

Figure 9 : Non-maskable Interrupt Request Operation.

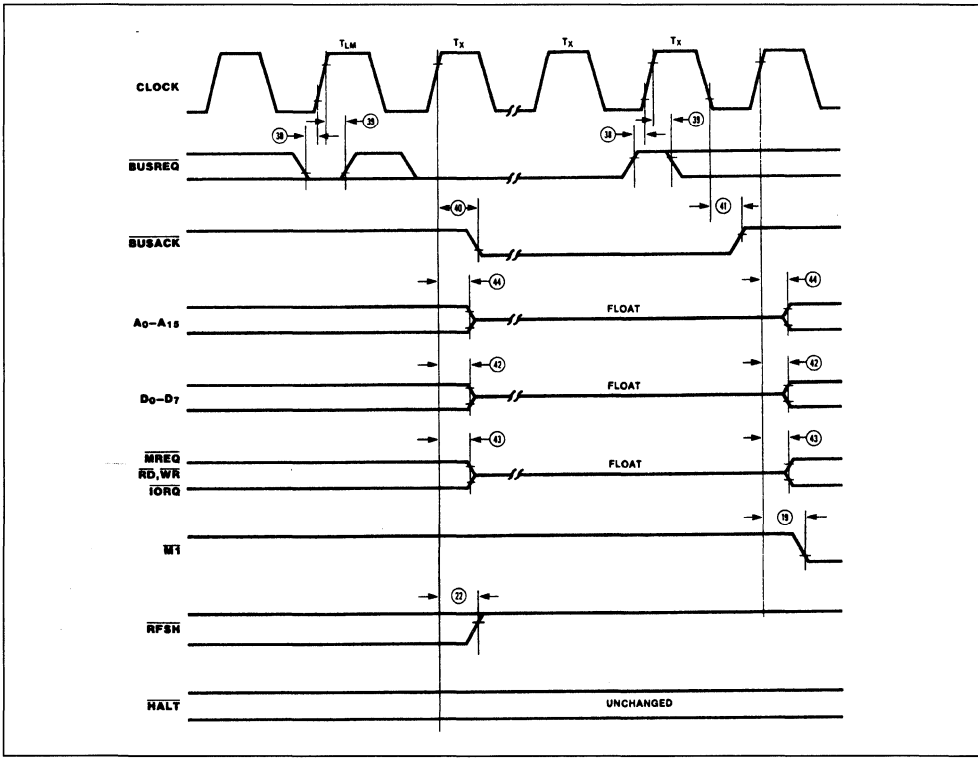


* Although $\overline{NMI}$ is an asynchronous input, to guarantee its being recognized on the following machine cycle, NMI's falling edge must occur no later than the rising edge of the clock cycle preceding T_{LAST}.

BUS REQUEST/ACKNOWLEDGE CYCLE

The CPU samples $\overline{BUSREQ}$ with the rising edge of the last clock period of any machine cycle (figure 10). If $\overline{BUSREQ}$ is active, the CPU sets its address, data, and MREQ, IORQ, RD, and WR lines to a high-impedance state with the rising edge of the next clock pulse. At that time, any external device can take control of these lines, usually to transfer data between memory and I/O devices.

Figure 10 : Z-Bus Request/Acknowledge Cycle.



Notes : 1. T_L = Last state of any M cycle.
2. T_X = An arbitrary clock cycle used by requesting device.

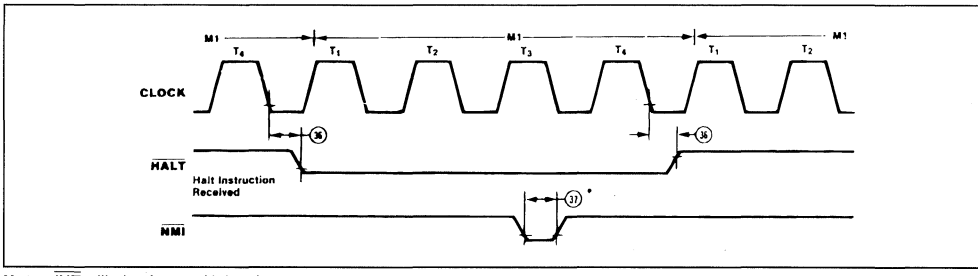
HALT ACKNOWLEDGE CYCLE

When the CPU receives an Halt instruction, it executes NOP states until either an INT or NMI input is received. When in the Halt state, the HALT output is active and remains so until an interrupt is processed (figure 11).

RESET CYCLE

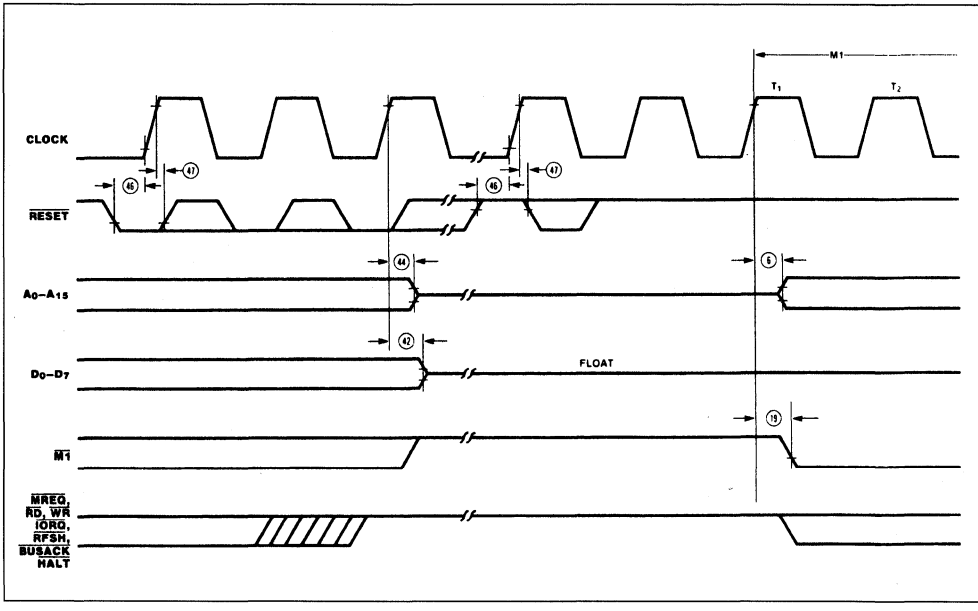
RESET must be active for at least three clock cycles for the CPU to properly accept it. As long a RESET remains active, the address and data buses float, and the control outputs are inactive. Once RESET goes inactive, three internal T cycles are consumed before the CPU resumes normal processing operation. RESET clears the PC register, so the first opcode fetch will be to location 0000 (figure 12).

Figure 11 : Halt Acknowledge Cycle.



Note : INT will also force a Halt exit.
* See note, Figure 9.

Figure 12 : Reset Cycle.



POWER DOWN

When the CPU system clock is stopped at either a high or low level, the CPU stops its operation and maintains registers and control signals.

However I_{CC2} Stand-by Supply Current is guaranteed only when the supplied system clock is stopped at a low level during T4 state of the following machine cycle (actually that is M1 cycle and executes NOP instruction) next to OPcode fetch cycle of HALT instruction. The timing diagram when POWER DOWN function is implemented by HALT instruction is shown in figure 13.

This function can be easily realized when a clock generator controller is connected with the CPU.

RELEASE FROM POWER DOWN STATE

The system clock must be supplied to the CPU to release power down state.

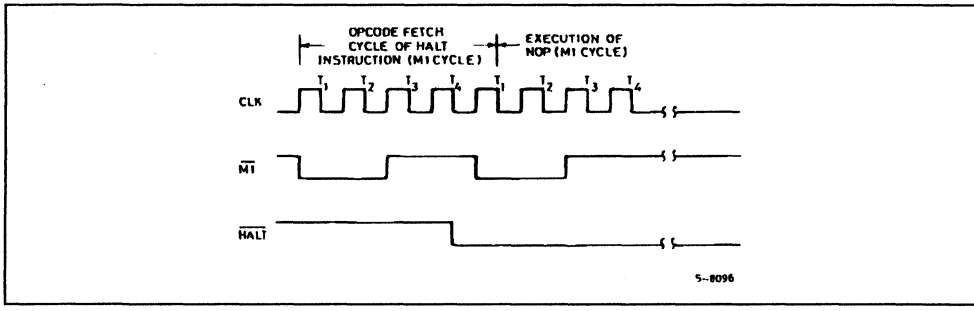
When the system clock is supplied to the CPU CLK terminal, CPU restarts operation continuously from the state when power down function has been implemented.

Note the followings when release from power down state.

(1) When external oscillator has been stopped to enter power down state, some warming-up time may be required to obtain precious and stable system clock for release from power down state.

(2) When HALT instruction is executed to enter power down state, the CPU will enter HALT state. An interrupt signal (NMI or INT) or RESET signal must be generated after the system clock is supplied to release power down state. Otherwise the CPU is still in HALT state even if the system clock is supplied.

Figure 13 : Timing Diagram of Power Down Function by Halt Instruction.



AC CHARACTERISTICS

N°	Symbol	Parameter	Z84C00A		Z84C00B		Z84C00H	
			Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
1	TcC	Clock Cycle Time	250	DC	165	DC	125	DC
2	TwCh	Clock Pulse Width (high)	110	DC	65	DC	55	DC
3	TwCl	Clock Pulse Width (low)	110	DC	65	DC	55	DC
4	TfC	Clock Fall Time		30		20		10
5	TrC	Clock Rise Time		30		20		10
6	TdCr(A)	Clock ↑ to Address Valid Delay		110		90		80
7	TdA(MREQf)	Address Valid to $\overline{MREQ}$ ↓ Delay	65		35		20	
8	TdCf(MREQf)	Clock ↓ to $\overline{MREQ}$ ↓ Delay		85		70		60
9	TdCr(MREQr)	Clock ↑ to $\overline{MREQ}$ ↑ Delay		85		70		60
10	TwMREQh	$\overline{MREQ}$ Pulse Width (high)	110		65		45	
11	TwMREQl	$\overline{MREQ}$ Pulse Width (low)	220		135		100	
12	TdCf(MREQr)	Clock ↓ to $\overline{MREQ}$ ↑ Delay		85		70		60
13	TdCf(RDf)	Clock ↓ to $\overline{RD}$ ↓ Delay		95		80		70
14	TdCr(RDr)	Clock ↑ to $\overline{RD}$ ↑ Delay		85		70		60
15	TsD(Cr)	Data Setup Time to Clock ↑	35		30		30	
16	ThD(RDr)	Data Hold Time to $\overline{RD}$ ↑	0		0		0	
17	TsWAIT(Cf)	$\overline{WAIT}$ Setup Time to Clock ↓	70		60		50	
18*	ThWAIT(Cf)	$\overline{WAIT}$ Hold Time after Clock ↓	10		10		10	
19	TdCr(Mlf)	Clock ↑ to $\overline{MI}$ ↓ Delay		100		80		70
20	TdCr(Mlr)	Clock ↑ to $\overline{MI}$ ↑ Delay		100		80		70
21	TdCr(RFSHf)	Clock ↑ to $\overline{RFSH}$ ↓ Delay		130		110		95
22	TdCr(RFSHr)	Clock ↑ to $\overline{RFSH}$ ↑ Delay		120		100		85
23	TdCf(RDr)	Clock ↓ to $\overline{RD}$ ↑ Delay		85		70		60
24	TdCr(RDf)	Clock ↑ to $\overline{RD}$ ↓ Delay		85		70		60
25	TsD(Cf)	Data Setup to Clock ↓ during M_2, M_3, M_4 or M_5 Cycles	50		40		30	
26	TdA(IORQf)	Address Stable Prior to $\overline{IORQ}$ ↓	180		110		75	
27	TdCr(IORQf)	Clock ↑ to $\overline{IORQ}$ ↓ Delay		75		65		55
28	TdCf(IORQr)	Clock ↓ to $\overline{IORQ}$ ↑ Delay		85		70		60
29	TdCf(WRf)	Data Stable Prior to $\overline{WR}$ ↓	80		25		5	
30	TdDf(WRf)	Clock ↓ to $\overline{WR}$ ↓ Delay		80		70		60
31	TwWR	$\overline{WR}$ Pulse Width	220		135		100	
32	TdCf(WRr)	Clock ↓ to $\overline{WR}$ ↑ Delay		80		70		60
33	TdD(WRf)	Data Stable Prior to $\overline{WR}$ ↓	− 10		− 55		− 55	
34	TdCr(WRf)	Clock ↑ to $\overline{WR}$ ↓ Delay		65		60		55
35	TdWRr(D)	Data Stable from $\overline{WR}$ ↑	60		30		15	
36	TdCf(HALT)	Clock ↓ to $\overline{HALT}$ ↑ or ↓		300		260		225

Note : * Not compatible with NMOS Specifications.

AC CHARACTERISTICS (continued)

N°	Symbol	Parameter	Z84C00A		Z84C00B		Z84C00H	
			Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
37	TwNMI	$\overline{NMI}$ Pulse Width		80		70		60
38	TsBUSREQ(Cr)	$\overline{BUSREQ}$ Setup Time to Clock ↑	50		50		40	
39*	TcBUSUREQ(Cr)	$\overline{BUSREQ}$ Hold Time after Clock ↑	10		10		10	
40	TdCr(BUSACKf)	Clock ↑ to $\overline{BUSACK}$ ↓ Delay		100		90		80
41	TdCf(BUSACKr)	Clock ↓ to $\overline{BUSACK}$ ↑ Delay		100		90		80
42	TdCr(Tz)	Clock ↑ to Data Float Delay		90		80		70
43	TdCr(CTz)	Clock ↑ to Control Outputs Float Delay ($\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$, and $\overline{WR}$)		80		70		60
44	TdCr(Az)	Clock ↑ to Address Float Delay		90		80		70
45	TdCTr(A)	$\overline{MREQ}$ ↑, $\overline{IORQ}$ ↑, $\overline{RD}$ ↑, and $\overline{WR}$ ↑ to Address Hold Time	80		35		20	
46	TsRESET(Cr)	$\overline{RESET}$ to Clock ↑ Setup Time	60		60		45	
47*	ThRESET(Cr)	$\overline{RESET}$ to Clock ↑ Hold Time	10		10		10	
48	TsINTf(Cr)	$\overline{INT}$ to Clock ↑ Setup Time	80		70		55	
49*	ThINTr(Cr)	$\overline{INT}$ to Clock ↑ Hold Time	10		10		10	
50	TdMIf(IORQf)	$\overline{MI}$ ↓ to $\overline{IORQ}$ ↓ Delay	565		365		270	
51	TdCf(IORQf)	Clock ↓ to $\overline{IORQ}$ ↓ Delay		85		70		60
52	TdCf(IORQr)	Clock ↑ to $\overline{IORQ}$ ↑ Delay		85		70		60
53	TdCf(D)	Clock ↓ to Data Valid Delay		150		130		115

Note : * Not compatible with NMOS Specification.

ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_{CC}	V_{CC} Supply Voltage with Respect to V_{SS}	– 0.5 to 7	V
V_{IN}	Input Voltage	– 0.5 to V_{CC} + 0.5	V
P_D	Power Dissipation (T_A = 85 °C)	250	mW
T_{SOLDER}	Soldering Temperature (soldering time 10 sec)	260	°C
T_{stg}	Storage Temperature	– 65 to 150	°C
T_{opr}	Operating Temperature	– 40 to 85	°C

DC CHARACTERISTICS

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_{ILC}	Clock Input Low Voltage		− 0.3	−	0.6	V
V_{IHC}	Clock Input High Voltage		V_{CC} − 0.6	−	V_{CC} + 0.3	V
V_{IL}	Input Low Voltage (except CLK)		− 0.5	−	0.8	V
V_{IH}	Input High Voltage (except CLK)		2.2	−	V_{CC}	V
V_{OL}	Output Low Voltage	I_{OL} = 2.0 mA	−	−	0.4	V
V_{OH1}	Output High Voltage (1)	I_{OH} = − 1.6 mA	2.4	−	−	V
V_{OH2}	Output High Voltage (2)	I_{OH} = − 250 µA	V_{CC} − 0.8	−	−	V
I_{LI}	Input Leakage Current	$V_{SS} \le V_{IN} \le V_{CC}$	−	−	10	µA
I_{LO}	3-State Output Leakage Current in Float	V_{SS} + 0.4 ≤ $V_{OUT} \le V_{CC}$	− 10	−	10	µA
I_{CC1}	Operating Supply Current 4 MHz 6 MHz 8 MHz	 V_{CC} = 5 V, V_{IL} = 0.2 V V_{IH} = V_{CC} − 0.2 V	 − − −	 9 15 20	 15 22 25	 mA mA mA
$I_{CC2(1)}$	Stand-by Supply Current	V_{CC} = 5 V CLK = (1) V_{IL} = V_{CC} − 0.2 V V_{IH} = 0.2 V	−	0.5	10	µA

Note : 1. I_{CC2} Stand-by Current is guaranteed only when the supplied clock is stopped at a low level during T4 state of the following machine cycle (M1) next to OPcode fetch cycle of HALT instruction.

TEST CONDITIONS

T_A = - 40 °C to + 85 °C

V_{CC} = 5 V ± 10 %

V_{SS} = 0 V

AC test conditions

- Inputs except CLK (clock) are driven at 2.4 V for a logic "1" and 0.4 V for a logic "0". Clock input is driven at V_{CC} − 0.6 V for a logic "1" and 0.6 V for a logic "0".
- Timing measurements are made at 2.2 V for a logic "1" and 0.8 V for a logic "0".

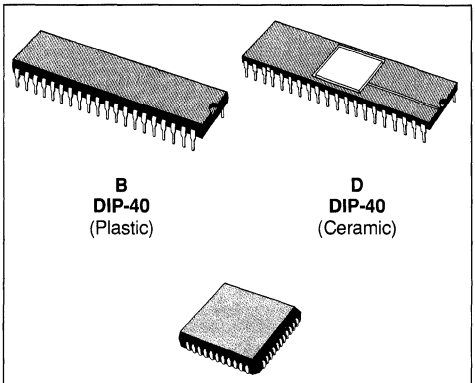
All AC parameters assume a load capacitance of 100 pF.

ORDERING INFORMATION

Type	Package	Temp.	Clock	Description
Z84C00AB6	DIP-40 (plastic)	− 40/+ 85°C	4 MHz	Z80C Central Processing Unit CMOS Version
Z84C00AD6	DIP-40 (ceramic)	− 40/+ 85°C		
Z84C00AD2	DIP-40 (ceramic)	− 55/+ 125°C		
Z84C00AC6	PLCC44 (plastic chip-carrier)	− 40/+ 85°C		
Z84C00BB6	DIP-40 (plastic)	− 40/+ 85°C	6 MHz	
Z84C00BD6	DIP-40 (ceramic)	− 40/+ 85°C		
Z84C00BD2	DIP-40 (ceramic)	− 55/+ 125°C		
Z84C00BC6	PLCC44 (plastic chip-carrier)	− 40/+ 85°C		
Z84C00HB6	DIP-40 (plastic)	− 40/+ 85°C	8 MHz	
Z84C00HD6	DIP-40 (ceramic)	− 40/+ 85°C		
Z84C00HC6	PLCC44 (plastic chip-carrier)	− 40/+ 85°C		

Z80 DMA CMOS DIRECT MEMORY ACCESS CONTROL

- TRANSFERS, SEARCHES AND SEARCH/TRANSFERS IN BYTE-AT-A-TIME, BURST OR CONTINUOUS MODES. CYCLE LENGTH AND EDGE TIMING CAN BE PROGRAMMED TO MATCH THE SPEED OF ANY PORT
- DUAL PORT ADDRESSES (sources and destination) GENERATED FOR MEMORY-TO-I/O, MEMORY-TO-MEMORY, OR I/O-TO-I/O OPERATIONS. ADDRESSES MAY BE FIXED OR AUTOMATICALLY INCREMENTED/DECREMENTED
- NEXT-OPERATION LOADING WITHOUT DISTURBING CURRENT OPERATIONS VIA BUFFERED STARTING ADDRESS REGISTERS. AN ENTIRE PREVIOUS SEQUENCE CAN BE REPEATED AUTOMATICALLY
- EXTENSIVE PROGRAMMABILITY OF FUNCTIONS. CPU CAN READ COMPLETE CHANNEL STATUS
- STANDARD Z80 FAMILY BUS-REQUEST AND PRIORITIZED INTERRUPT-REQUEST DAISY CHAINS IMPLEMENTED WITHOUT EXTERNAL LOGIC. SOPHISTICATED, INTERNALLY MODIFIABLE INTERRUPT VECTORING
- DIRECT INTERFACING TO SYSTEM BUSES WITHOUT EXTERNAL LOGIC
- SINGLE 5 V ± 10 % POWER SUPPLY
- LOW POWER CONSUMPTION :
  - 5 mA TYP. AT 4 MHz
  - 6 mA TYP. AT 6 MHz
  - LESS THAN 10 µA IN POWER DOWN MODE
- EXTENDED OPERATING TEMPERATURE
  - 40 °C TO + 85 °C



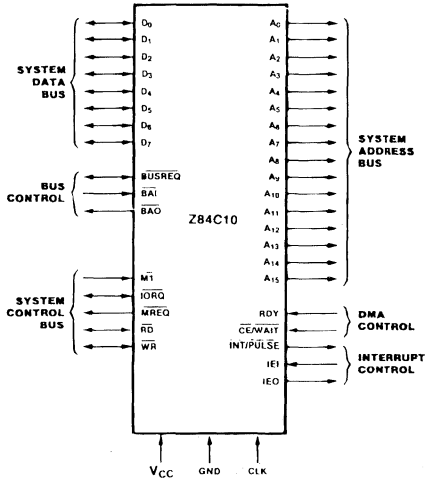
B
DIP-40
(Plastic)

D
DIP-40
(Ceramic)

C
PLCC44
(Plastic)

(Ordering Information at the end of the datasheet)

LOGIC FUNCTIONS



DESCRIPTION

The Z80C DMA (Direct Memory Access) is a powerful and versatile device for controlling and processing transfers of data. Its basic function of managing CPU-independent transfers between two ports is augmented by an array of features that optimize transfer speed and control with little or no external logic in systems using an 8- or 16-bit data bus and a 16-bit address bus.

Transfers can be done between any two ports (source and destination), including memory-to I/O,

Figure 1 : Dual in Line Pin Configuration.

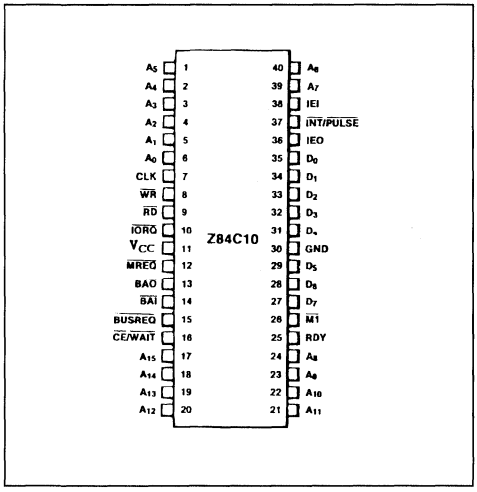
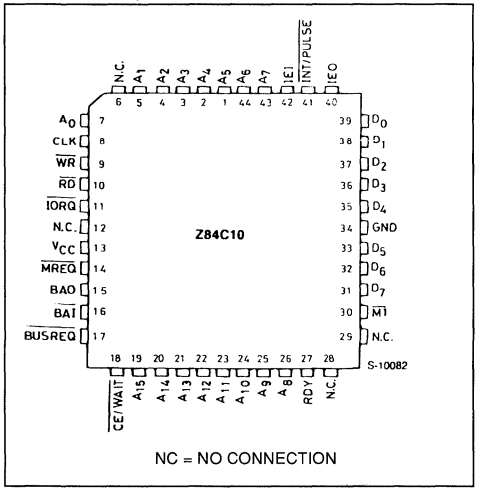


Figure 2 : Chip Carrier Pin Configuration.



NC = NO CONNECTION

memory-to-memory, and I/O-to-I/O. Dual port addresses are automatically generated for each transaction and may be either fixed or incrementing/decrementing. In addition, bit-maskable byte searches can be performed either concurrently with transfers or as an operation in itself.

The Z80C DMA contains direct interfacing to and independent control of system buses, as well as sophisticated bus and interrupt controls. Many programmable features, including variable cycle timing and autorestart, minimize CPU software overhead. They are especially useful in adapting this special-purpose transfer processor to a broad variety of memory, I/O and CPU environments.

The Z80C DMA is an n-channel silicon-gate depletion-load device and uses a single + 5 V power supply and the standard Z80C Family single-phase clock.

FUNCTIONAL DESCRIPTION

Classes of Operation. The Z80C DMA has three basic classes of operation :

- Transfers of data between two ports (memory or I/O peripheral)
- Searches for a particular 8-bit maskable byte at a single port in memory or an I/O peripheral
- Combined transfers with simultaneous search between two ports

Figure 4 illustrates the basic functions served by these classes of operation.

During a transfer, the DMA assumes control of the system address and data buses. Data is read from one addressable port and written to the other addressable port, byte by byte. The ports may be programmed to be either system main memory or peripheral I/O devices. Thus, a block of data may be written from one peripheral to another, from one area of main memory to another, or from a peripheral to main memory and vice versa.

During a search-only operation, data is read from the source port and compared byte by byte with a DMA-internal register containing a programmable match byte. This match byte may optionally be masked so that only certain bits within the match byte are compared.

Search rates up to 2 M bytes per second can be obtained with the 4 MHz Z80C DMA or 3 M bytes per second with the 6 MHz Z80C DMA.

In combined searches and transfers, data is transferred between two ports while simultaneously searching for a bit-maskable byte match.

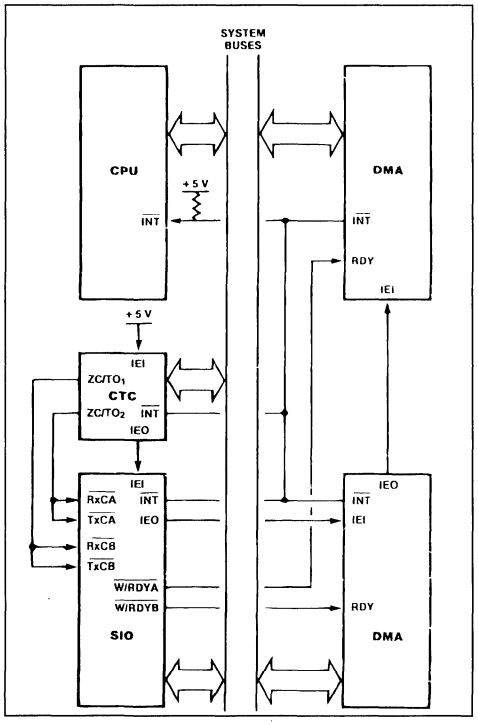
Data transfers or searches can be programmed to stop or interrupt under various conditions. In addition, CPU-readable status bits can be programmed to reflect the condition.

MODES OF OPERATION

The DMA can be programmed to operate in one of three transfer and/or search modes :

- *Byte-at-a-Time* : data operations are performed one byte at a time. Between each byte operation

Figure 3 : Typical Z80C Environment.



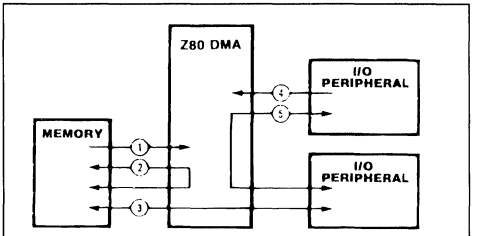
the system buses are released to the CPU. The buses are requested again for each succeeding byte operation.

- *Burst* : data operations continue until a port's Ready line to the DMA goes inactive. The DMA then stops and releases the system buses after completing its current byte operation.
- *Continuous* : data operations continue until the end of the programmed block of data is reached before the system buses are released. If a port's Ready line goes inactive before this occurs, the DMA simply pauses until the Ready line comes active gain.

In all modes, once a byte of data is read into the DMA, the operation on the byte will be completed in an orderly fashion, regardless of the state of other signals (including a port's Ready line).

Due to the DMA's high-speed buffered method of reading data, operations on one byte are not completed until the next byte is read in. This means that total transfer or search block lengths must be two or more bytes, and that block lengths programmed into the DMA must be one byte less than the desired

Figure 4 : Function of the Z80 DMA.



1. Search memory
2. Transfer memory-to-memory (optional search)
3. Transfer memory-to-I/O (optional search)
4. Search I/O
5. Transfer I/O-to-I/O (optional search)

block length (cont is N-1 where N is the block length).

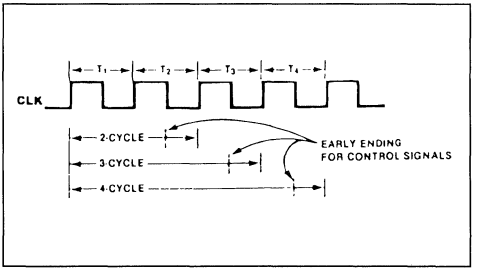
COMMANDS AND STATUS

The Z80C DMA has several writable control registers and readable status registers available to the CPU. Control bytes can be written to the DMA whenever the DMA is not controlling the system buses, but the act of writing a control byte to the DMA disables the DMA until it is again enabled by a specific command. Status bytes can also be read at any such time, but writting the Read Status Byte command or the Initial Read Sequence command disables the DMA.

Control bytes to the DMA include those which effect immediate command actions such as enable, disable, reset, load starting-address buffers, continue, clear counters, clear status bits and the like. In addition, many mode-setting control bytes can be written, including mode and class of operation, port configuration, starting addresses, block length, address counting rule, match and match-mask byte, interrupt conditions, interrupt vector, status-affects-vector condition, pulse counting, auto restart, Ready-line and Wait-line rules, and read mask.

Readable status registers include a general status

Figure 5 : Variable Cycle Lenght.



byte reflecting Ready-line, end-of-block, byte-match and interrupt conditions, as well as 2-byte registers for the current byte count, Port A address and Port B address.

VARIABLE CYCLE

The DMA has the unique feature of programmable operation-cycle length. This is valuable in tailoring the DMA to the particular requirements of other system components (fast or slow) and maximizes the data-transfer rate. It also eliminates external logic for signal conditioning.

There are two aspects to the variable cycle feature. First, the entire read and write cycles (periods) associated with the source and destination ports can be independently programmed as 2, 3 or 4 T-cycles long (more if Wait cycles are used), thereby increasing or decreasing the speed with which all DMA signals change (figure 5).

Second, the four signals in each port specifically associated with transfers of data (I/O Request, Memory Request, Read, and Write) can each have its active trailing edge terminated one-half T-cycle early. This adds a further dimension of flexibility and speed, allowing such things as shorter-than-normal Read or Write signals that go inactive before data starts to change.

ADDRESS GENERATION

Two 16-bit addresses are generated by the Z80C DMA for every transfer operation, one address for the source port and another for the destination port. Each address can be either variable or fixed. Variable addresses can increment or decrement from the programmed starting address. The fixed-address capability eliminates the need for separate enabling wires to I/O ports.

Port addresses are multiplexed onto the system address bus, depending on whether the DMA is reading the source port or writing to the destination port. Two readable address counters (2 bytes each) keep the current address of each port.

AUTO RESTART

The starting addresses of either port can be reloaded automatically at the end of a block. This option is selected by the Auto Restart control bit. The byte counter is cleared when the addresses are reloaded.

The Auto Restart feature relieves the CPU of software overhead for repetitive operations such as CRT refresh and many others. Moreover, when the CPU has access to the buses during byte-at-a-time or burst transfers, different starting addresses can be written into buffer registers during transfers, causing the Auto Restart to begin at a new location.

INTERRUPTS

The Z80C DMA can be programmed to interrupt the CPU on three conditions :

- Interrupt on Ready (before requesting bus)
- Interrupt on Match
- Interrupt on End of Block

Any of these interrupts cause an interrupt-pending status bit to be set, and each of them can optionally alter the DMA's interrupt vector. Due to the buffered constraint mentioned under "Modes of Operation", interrupts on Match at End of Block are caused by matches to the byte just prior to the last byte in the block.

The DMA shares the Z80 Family's elaborate interrupt scheme, which provides fast interrupt service in real-time applications. In a Z80C CPU environment, the DMA passes its internally modifiable 8-bit interrupt vector to the CPU, which adds an additional eight bits to form the memory address of the interrupt routine table. This table contains the address of the beginning of the interrupt routine itself. In this process ; CPU control is transferred directly to the interrupt routine, so that the next instruction executed after an interrupt acknowledge is the first instruction of the interrupt routine itself.

PULSE GENERATION

External devices can keep track of how many bytes have been transferred by using the DMA's pulse output, which provides a signal at 256-byte intervals. The interval sequence may be offset at the beginning by 1 to 255 bytes.

The Interrupt line outputs the pulse signal in a manner that prevents misinterpretation by the CPU as an interrupt request, since it only appears when the Bus Request and Bus Acknowledge lines are both active.

PIN DESCRIPTIONS

A_0-A_{15} .*System Address Bus* (output, 3-state). Addresses generated by the DMA are sent to both source and destination ports (main memory or I/O peripherals) on these lines.

$\overline{BAI}$.*Bus Acknowledge In* (input, active Low). Signals that the system buses have been released for DMA control. In multiple-DMA configurations, the $\overline{BAI}$ pin of the highest priority DMA is normally connected to the Bus Acknowledge pin of the CPU. Lower-priority DMAs have their $\overline{BAI}$ connected to the $\overline{BAO}$ of a higher-priority DMA.

$\overline{BAO}$.*Bus Acknowledge Out* (output, active Low). In a multiple-DMA configuration, this pin signals that no other higher-priority DMA has requested the system buses. BAI and BAO from daisy chain for multiple-DMA priority resolution over bus control.

$\overline{BUSREQ}$.*Bus Request* (Bidirectional, active Low, open drain). As an output, it sends requests for control of the system address bus, data bus and control bus to the CPU. As an input, when multiple DMAs are strung together in a priority daisy chain via BAI and BAO, it senses when another DMA has requested the buses and causes this DMA to refrain from bus requesting until the other DMA is finished. Because it is a bidirectional pin, there cannot be any buffers between this DMA and any other DMA. It can, however, have a buffer between it and the CPU because it is unidirectional into the CPU. A pull-up resistor is connected to this pin

$\overline{CE}$/$\overline{WAIT}$.*Chip Enable and Wait* input, active Low). Normally this functions only as a $\overline{CE}$ line, but it can also be programmed to serve a $\overline{WAIT}$ function. As a $\overline{CE}$ line from the CPU, it becomes active when $\overline{WR}$ and $\overline{IORQ}$ are active and the I/O port address on the system address bus is the DMA's address, thereby allowing a transfer of control or command bytes from the CPU to the DMA. As a $\overline{WAIT}$ line from memory or I/O devices, after the DMA has received a bus-request acknowledge from the CPU, it causes wait states to be inserted in the DMA's operation cycles thereby slowing the DMA to a speed that matches the memory or I/O device.

$\overline{CLK}$.*System Clock* (input). Standard Z80 single-phase clock at 4.0 MHz (Z80CA DMA) or 6.0 MHz (Z80CB DMA). For slower system clocks, a TTL gate whith a pullup resistor may be adequate to meet the timing and voltage level specification. For higher-speed systems, use a clock driver with an active pullup to meet the V_{IH} specification and risetime requirements. In all cases there should be a resistive pullup to the power supply of 10 K ohms (max) to ensure proper power when the DMA is reset.

D_0-D_7 .*System Data Bus* (bidirectional, 3-state). Commands from the CPU, DMA status, and data from memory or I/O peripherals are transferred on these lines.

IEI .*Interrupt Enable In* (input, active High). This 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 DMA. Thus, this signal block lower-priority devices from interrupting while a higher-priority device is being serviced by its CPU interrupt service routine.

INT/PULSE .*Interrupt Request* (output, active Low, open drain). This requests a CPU interrupt. The CPU acknowledges the interrupt by pulling its IORQ output Low during an M1 cycle. It is typically connected to the $\overline{INT}$ pin of the CPU with a pullup resistor and tied to all other $\overline{INT}$ pins in the system. This pin can also be used to generate periodic pulses to an external device. It can be used this way only when the DMA is bus master (i.e., the CPU's $\overline{BUSREQ}$ and BUSACK lines are both Low and the CPU cannot see interrupts).

$\overline{IORQ}$.*Input/Output Request* (bidirectional ; active Low, 3-state). As an input, this indicates that the lower half of the address bus holds a valid I/O port address for transfer of control or status bytes from or to the CPU, respectively ; this DMA is the addressed port if its $\overline{CE}$ pin and its $\overline{WR}$ or $\overline{RD}$ pins are simultaneously active. As an output, after the DMA has taken control of the system buses, it indicates that the 8-bit or 16-bit address bus holds a valid port address for another I/O device involved in a DMA transfer of data. When $\overline{IORQ}$ and M1 are both active simultaneously, an interrupt acknowledge is indicated.

$\overline{M1}$.*Machine Cycle One* (input, active Low). Indicates that the current CPU machine cycle is an instruction fetch. It is used by the DMA to decode the return-from-interrupt instruction (RETI) (ED-4D) sent by the CPU. During two-byte instruction fetches, M1 is active as each opcode byte is fetched. An interrupt acknowledge is indicated when both M1 and $\overline{IORQ}$ are active.

$\overline{MREQ}$.*Memory Request* (output, active Low, 3-state). This indicates that the address bus holds a valid address for a memory read or write operation. After the DMA has taken control of the system buses, it indicates a DMA transfer request from or to memory.

$\overline{RD}$. *Read* (bidirectional, active Low, 3-state). As an input, this indicates that the CPU wants to read status bytes from the DMA's read registers. As an output, after the DMA has taken control of the system buses, it indicates a DMA-controlled read from a memory or I/O port address.

RDY. *Ready* (input, programmable active Low or High). This is monitored by the DMA to determine when a peripheral device associated with a DMA port is ready for a read or write operation. Depending on the mode of DMA operation (Byte, Burst or Continuous), the RDY line indirectly controls DMA

activity by causing the $\overline{\text{BUSREQ}}$ line to go Low or High.

$\overline{\text{WR}}$ *Write* (bidirectional, active Low, 3-state). As and input, this indicates that the CPU wants to write control or command bytes to the DMA write registers. As an output, after the DMA has taken control of the system buses, it indicates a DMA-controlled write to a memory or I/O port address.

INTERNAL STRUCTURE

The internal structure of the Z80C DMA includes driver and receiver circuitry for interfacing with an 8-bit system data bus, a 16-bit system address bus, and system control lines (figure 6). In a Z80C CPU environment, the DMA can be tied directly to the analogous pins on the CPU (figure 7) with no additional buffering, except for the CE/$\overline{\text{WAIT}}$ line.

The DMA's internal data bus interfaces with the system data bus and services all internal logic and registers. Addresses generated from this logic for Ports A and B (source and destination) of the DMA's single transfer channel are multiplexed onto the system address bus.

Specialized logic circuits in the DMA are dedicated to the various functions of external bus interfacing, internal bus control, byte matching, byte counting, periodic pulse generation, CPU interrupts, bus request, and address generation. A set of twenty-one writable control registers and seven readable status registers provides the means by which the CPU governs and monitors the activities of these logic circuits. All registers are eight bits wide, with double-byte information stored in adjacent registers. The two address counters (two bytes each) for Ports A and B are buffered by the two starting addresses.

The 21 writable control register are organized into seven base-register groups, most of which have multiple registers. The base registers in each writable group contain both control/command bits and pointer bits that can be set to address other registers within the group. The seven readable status registers have no analogous second-level registers.

The registers are designated as follows, according to their base-register groups :

WR0-WR6 - Write Register groups 0 through 6 (7 base registers plus 14 associated registers)

RR0-RR6 - Read Registers 0 through 6

Writing to a register within a write-register group involves first writing to the base register, with the appropriate pointer bits set, then writing to one or more of the other register within the group. All seven of the readable status registers are accessed sequentially according to a programmable mask contained in one of the writable registers. The section entitled "Programming" explains this in more detail.

A pipelining scheme is used for reading data in. The programmed block length is the number of bytes compared to the byte counter, which increments at the end of each cycle. In searches, data byte comparisons with the match byte are made during the

Figure 6 : Block Diagram.

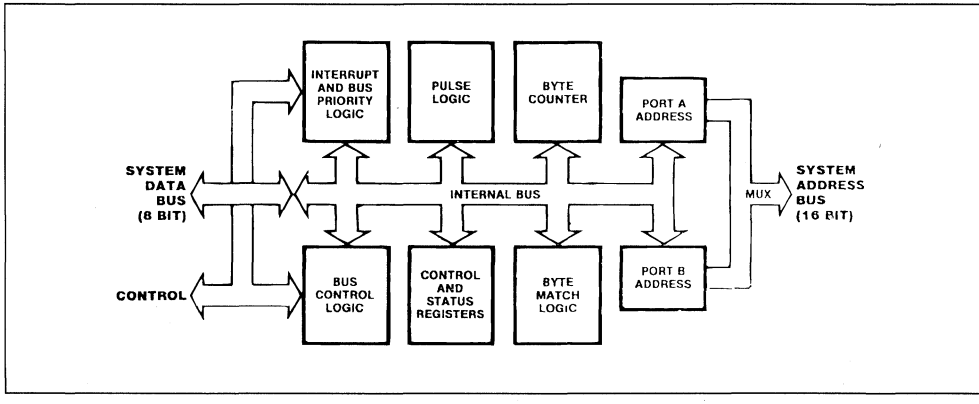
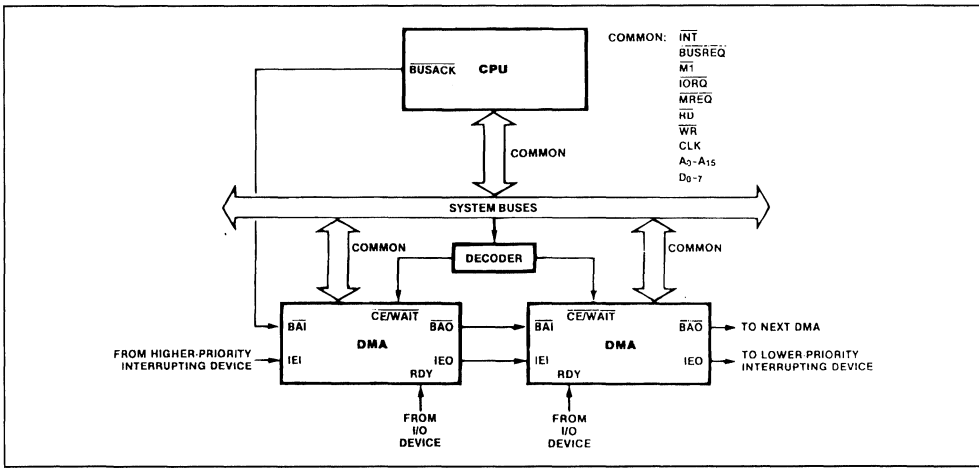


Figure 7 : Multiple-DMA Interconnection to the Z80 CPU.



Write Registers	
WR0	Base Register Byte
	Port A starting Address (low byte)
	Port A starting Address (high byte)
	Block Length (low byte)
	Block Length (high byte)
WR1	Base Register Byte
	Port A Variable-timing Byte
WR2	Base Register Byte
	Port B Variable-timing Byte
WR3	Base Register Byte
	Mask Byte
	Match Byte
WR4	Base Register Byte
	Port B starting Address (low byte)
	Port B starting Address (high byte)
	Interrupt Control Byte
	Pulse Control Byte
	Interrupt Vector
WR5	Base Register Byte
WR6	Base Register Byte
	Read Mask

Read Registers	
RR0	Status Byte
RR1	Byte Counter (low byte)
RR2	Byte Counter (high byte)
RR3	Port A Address Counter (low byte)
RR4	Port A Address Counter (high byte)
RR5	Port B Address Counter (low byte)
RR6	Port B Address Counter (high byte)

read cycle of the next byte. Matches are, therefore, discovered only after the next byte is read in.

In multiple-DMA configurations, interrupt request daisy chains are priorized by the order in which their IEI and IEO lines are connected. The system bus, however, may not be pre-empted.

Any DMA that gains access to the system bus keeps the bus until it is finished.

PROGRAMMING

The Z80C DMA has two programmable fundamental states : (1) an enabled state, in which it can gain control of the system buses and direct the transfer of data between ports, and (2) a disabled state, in which it can initiate neither bus requests nor data transfers. When the DMA is powered up or reset by any means, it is automatically placed into the disabled state. Program commands can be written to it by the CPU in either state, but this automatically puts the DMA in the disabled state, which is maintained until an enable command is issued by the CPU. The CPU must program the DMA in advance of any data search or transfer by addressing it as an I/O port and sending a sequence of control bytes using an Output instruction (such as OTIR for the Z80C CPU).

WRITING

Control or command bytes are written into one or more of the Write Register groups (WR0-WR6) by first writing to the base register byte in that group.

ted registers in a group are sequentially accessed by first writing a byte to the base register containing register-group identification and ponter bits (1's) to one or more of that base register's associated registers.

This is illustrated in figure 8b. In this figure, the sequence in which associated registers within a group can be written to is shown by the vertical position of the associated registers. For example, if a byte written to the DMA contains the bits that identify WR0 (bits D0, D1 and D7), and also contains 1's in the bit positions that point to the associated "Port A Starting Address (low byte)" and "Port A Starting Address (high byte)", then the next two byte written to the DMA will be stored in these two registers, in that order.

READING

The Read Registers (RR0-RR6) are read by the CPU by addressing the DMA as an I/O port using an Input instruction (such as INIR for the Z80C CPU). The readable bytes contain DMA status, byte-counter values, and port addresses since the last DMA reset. The registers are always read in a fixed sequence beginning with RR0 and ending with RR6. However, the register read in this sequence is determined by programming the Read Mask in WR6. The sequence of reading is initialized by writing an Initiate Read Sequence or Set Read Status command to WR6. After a Reset DMA, the sequence must be initialized with the Initiate Read Sequence command or a Read Status command. The sequence of reading all registers that are not excluded by the Read Mask register must be completed before a new Initiate Read Sequence or Read Status command.

FIXED-ADDRESS PROGRAMMING

A special circumstance arises when programming a destination port to have a fixed address. The load command in WR6 only loads a fixed address to a port selected as the source, not to a port selected as the destination.

Therefore, a fixed destination address must be loaded by temporarily declaring it a fixed-source address and subsequently declaring the true source as such, thereby implicitly making the other a destination.

The following example illustrates the steps in this procedure, assuming that transfers are to occur from a variable-address source (Port A) to a fixed-address destination (Port B) :

1. Temporarily declare Port B as source in WR0.
2. Load Port B address in WR6.
3. Declare Port A as source in WR0.
4. Load Port A address in WR6.
5. Enable DMA in WR6.

Figure 9 illustrates a program to transfer data from memory (Port A) to a peripheral device (Port B). In this example, the Port A memory starting address is 1050_H and the Port B peripheral fixed address is 05_H. Note that the data flow is 1001_H bytes - one more than specified by the clock length. The table of DMA commands may be stored in consecutive memory locations and transferred to the DMA with an output instruction such as the Z80 CPU's OTIR instruction.

Figure 8a : Read Registers.

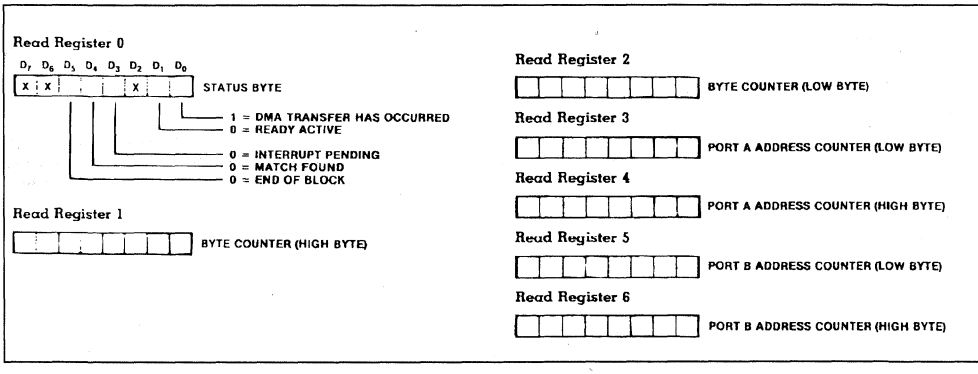


Figure 8b : Write Registers.

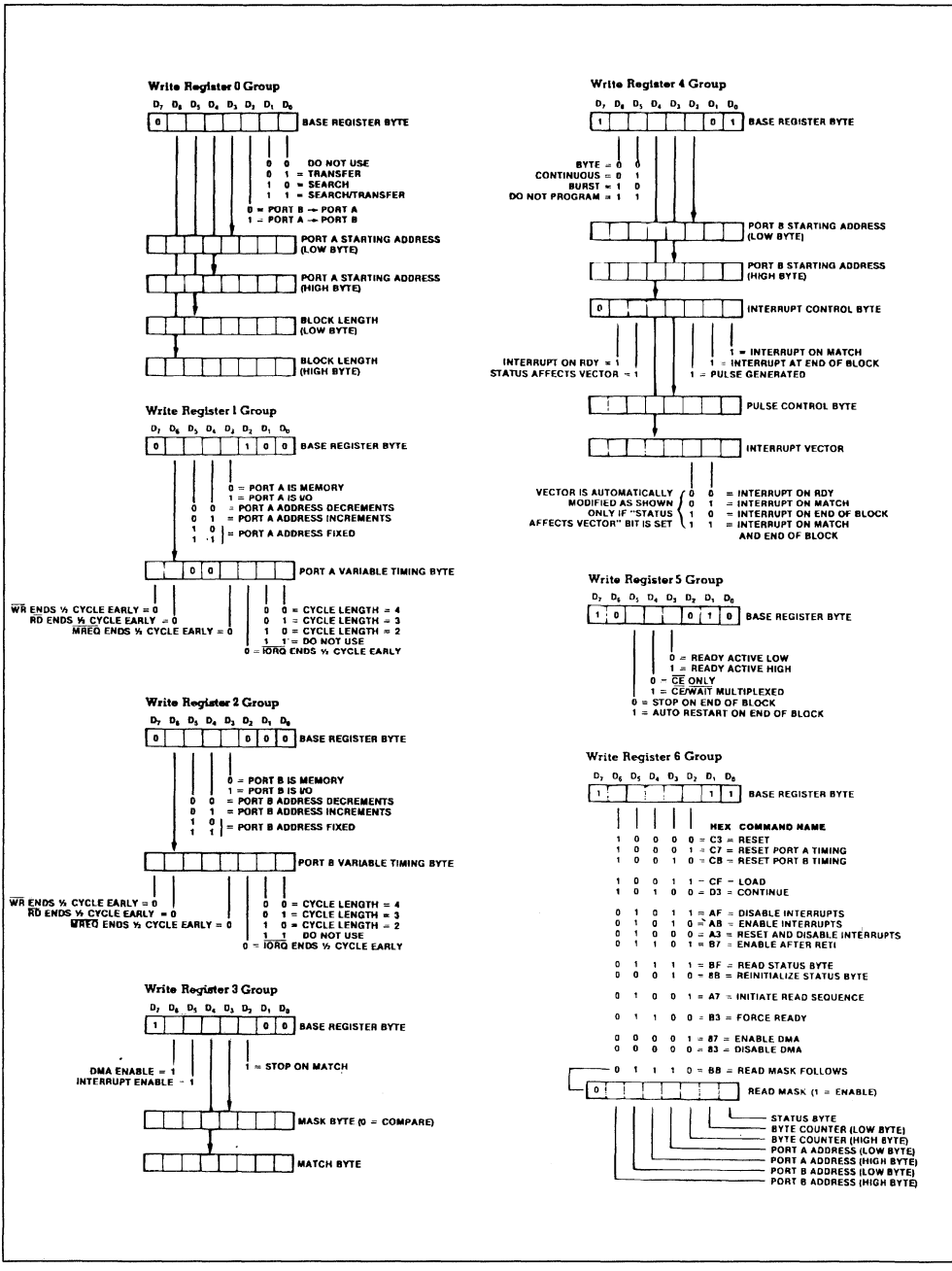


Figure 9 : Sample DMA Program.

Comments	D_7	D_6	D_5	D_4	D_3
WR0 sets DMA to receive block length. Port A starting address and temporarily sets Port B as source.	0	1 Block Length Upper Follows	1 Block Length Lower Follows	1 Port A Upper Address Follows	1 Port A Lower Address Follows
Port A Address (lower)	0	1	0	1	0
Port A Address (upper)	0	0	0	1	0
Block Length (lower)	0	0	0	0	0
Block Length (upper)	0	0	0	1	0
WR1 defines Port A as memory with fixed incrementing address.	0	0 No Timing Follows	0 Address Changes	1 Address Increments	0 Port is Memory
WR2 defines Port B as peripheral with fixed address.	0	0 No Timing Follows	1 Fixed Address	0	1 Port is I/O
WR4 sets mode to Burst, sets DMA to expect Port B address.	1	1	0	0 No Interrupt Control Byte Follows	0 No Upper Address
		Burst Mode			
Port B Address (lower)	0	0	0	0	0
WR5 Sets Ready Active High.	1	0	0 No Auto Restart	0 No Wait States	1 RDY Active High
WR6 Loads Port B Address and Resets Block Counter. *	1	1	0	0	1
WR0 Sets Port A as Source. *	0	0	0	0	0
		No Address or Block Length Bytes			
WR6 Loads Port A Address and Resets Block Counter.	1	1	0	0	1
WR6 Enables DMA to start operation.	1	0	0	0	

Note : The actual number of bytes transferred is one more than specified by the block length.
* These entries are necessary only in the case of a fixed destination address.

Figure 9 : Sample DMA Program (continued).

Comments	D_2	D_1	D_0	HEX
WR0 sets DMA to receive block length. Port A starting address and temporarily sets Port B as source.	0 B → A Temporary for Loading B Address *	0	1	79
		Transfer, No Search		
Port A Address (lower)	0	0	0	50
Port A Address (upper)	0	0	0	10
Block Length (lower)	0	0	0	00
Block Length (upper)	0	0	0	10
WR1 defines Port A as memory with fixed incrementing address.	1	0	0	14
WR2 defines Port B as peripheral with fixed address.	0	1	0	28
WR4 sets mode to Burst, sets DMA to expect Port B address.	1 Port B Lower Address Follows	0	1	C5
Port B Address (lower)	1	0	1	05
WR5 Sets Ready Active High.	0	1	0	8A
WR6 Loads Port B Address and Resets Block Counter. *	1	1	1	CF
WR0 Sets Port A as Source. *	1	0	1	05
	A → B	Transfer, No Search		
WR6 Loads Port A Address and Resets Block Counter.	1	1	1	CF
WR6 Enables DMA to start operation.	1	1	1	87

Note : The actual number of bytes transferred is one more than specified by the block length.
* These entries are necessary only in the case of a fixed destination address.

INACTIVE STATE TIMING (DMA as CPU Peripheral).

In its disabled or inactive state, the DMA is addressed by the CPU as an I/O peripheral for write and read (control and status) operations. Write timing is illustrated in figure 10.

Reading of the DMA's status byte, byte counter or port address counters is illustrated in figure 11. These operations require less than three T-cycles. The CE, IORQ and RD lines are made active over two rising edges of CLK, and data appears on the bus approximately one T-cycle after they become active.

ACTIVE STATE TIME (DMA as Bus Control-

DEFAULT READ AND WRITE CYCLES

By default, and after reset, the DMA's timing of read and write operations is exactly the same as the Z80C CPU's timing of read and write cycles for memory and I/O peripherals, with one exception : during a read cycle, data is latched on the falling edge of T_3 and held on the data bus across the boundary between read and write cycles, through the end of the following write cycle.

Figure 12 illustrates the timing for memory-to-I/O port transfers and figure 13 illustrates I/O-to-memory transfers. Memory-to-memory and I/O-to-I/O transfers timings are simply permutations of these diagrams.

The default timing uses three T-cycles for memory transactions and four T-cycles for I/O transactions, which include one automatically inserted wait cycle between T_2 and T_3. If the CE/WAIT line is programmed to act a WAIT line during the DMA's active state, it is sampled on the falling edge of T_2 for memory transactions and the falling edge of T_W for I/O transactions. If CE/WAIT is Low during this time an-

Figure 10 : CPU-to-DMA Wrtite Cycle.

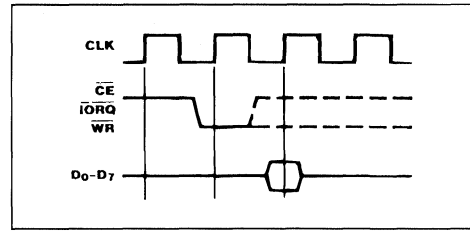


Figure 11 : CPU-to-DMA Read Cycle.

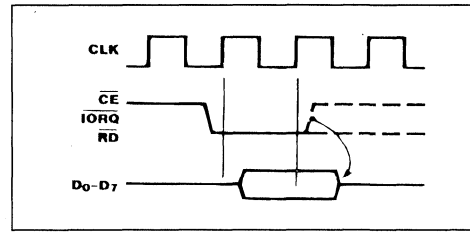
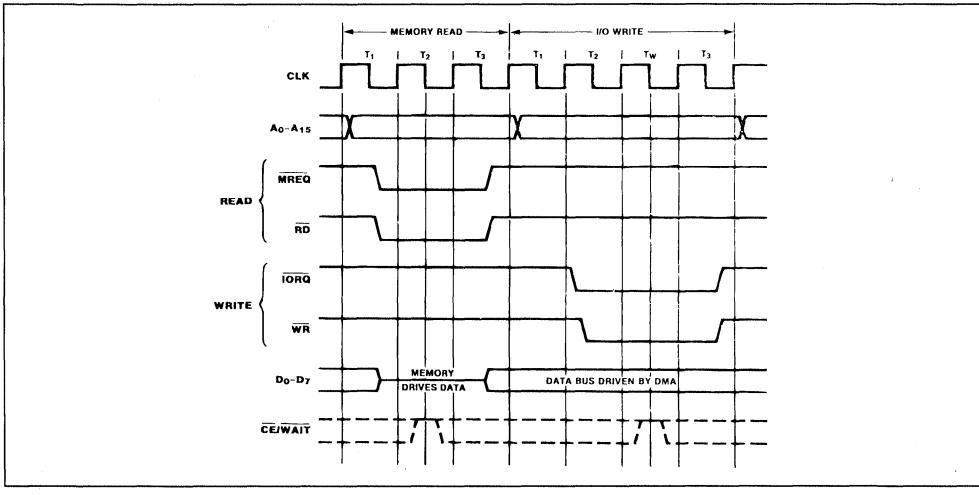


Figure 12 : Memory-to-I/O Transfer.



other T-cycle is added, during which the CE/WAIT line will again be sampled. The duration of transactions can thus be indefinitely extended.

VARIABLE CYCLE AN EDGE TIMING

The Z80C DMA's default operation-cycle length for the source (read) port and destination (write) port can be independently programmed. This variable-cycle feature allows read or write cycles consisting of two, three of four-T-cycle (more if Wait cycles are inserted), thereby increasing or decreasing the speed of all signals generated by the DMA. In addition, the trailing edges of the IORQ, MREQ, RD an WR signals can be independently terminated one-half cycle early. Figure 14 illustrates this.

In the variable-cycle mode, unlike default timing, IORQ comes active one-half cycle before MREQ, RD and WR. CE/WAIT can be used to extend only the 3 or 4 T-cycle variable memory cycles and only the 4-cycle variable I/O cycle. The CE/WAIT line is sampled at the falling edge of T_2 for 3- or 4-cycle memory cycles, and at the falling edge of T_3 for 4-cycle I/O cycles.

During transfers, data is latched on the clock edge causing the rising edge of RD and held through the end of the write cycle.

BUS REQUESTS

Figure 15 illustrates the bus request and acceptance timing. The RDY line, which may be programmed active High or Low, is sampled on every rising edge of CLK. If it is found to be active, and if the bus is not in use by any other device, the following rising edge of CLK drives BUSREQ low. After receiving BUSREQ the CPU acknowledges on the BAI input either directly or through a multiple-DMA daisy chain. When a Low is detect on BAI for two consecutive rising edges of CLK, the DMA will begin transferring data on the next rising edge of CLK.

BUS RELEASE BYTE-AT-A-TIME

In Byte-at-a-Time mode, BUSREQ is brought High on the rising edge of CLK prior to the end of each read cycle (search-only) or write cycle (transfer and transfer/search) as illustrated in figure 16. This is done regardless of the state of RDY. There is no possibility of confusion when a Z80C CPU is used since the CPU cannot begin an operation until the following T-cycle. Most other CPUs are not bothered by this either, although note should be taken of it. The next bus request for the next byte will come after both BUSREQ and BAI have returned High.

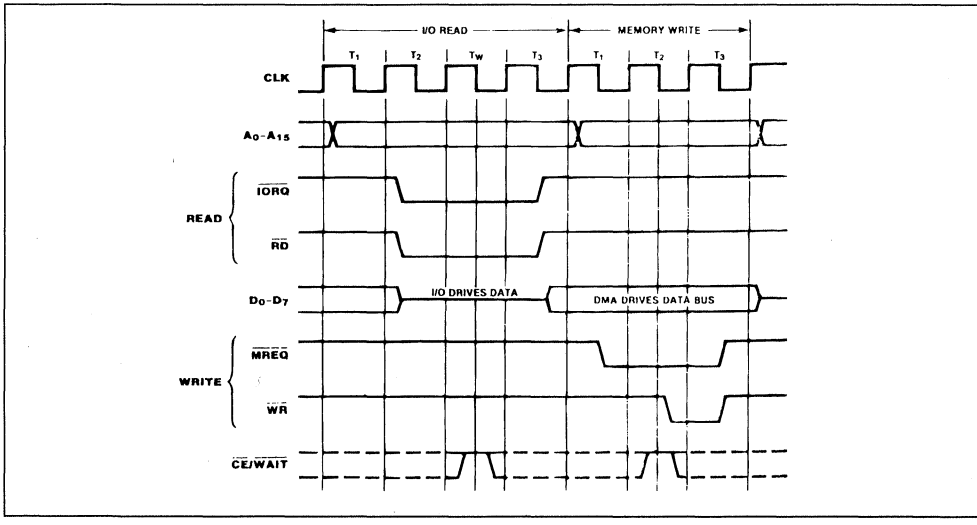
BUS RELEASE AT END OF BLOCK

In Burst and Continuos modes, an end of block causes BUSREQ to go High usually on the same rising edge of CLK in which the DMA completes the transfer of the data block (figure 17). The last byte in the block is transferred even if RDY goes inactive before completion of the last byte transfer.

BUS RELEASE AND NOT READY

In Burst mode, when RDY goes inactive it causes BUSREQ to go High on the next rising edge of CLK after the completion of its current byte operation

Figure 13 : I/O -to-Memory Transfer.



(figure 18). The action on $\overline{BUSREQ}$ is thus somewhat delayed from action on the RDY line. The DMA always completes its current byte operation in an orderly fashion before releasing the bus.

By contrast, $\overline{BUSREQ}$ is not released in Continuos mode when RDY goes inactive. Instead, the DMA idles after completing the current byte operation, awaiting an active RDY again.

BUS RELEASE ON MATCH

If the DMA is programmed to stop on match in Burst or Continuous modes, a match causes $\overline{BUSREQ}$ to go inactive on the next DMA operation, i.e., at the end of the next read in a search or at the end of the following write in a transfer (figure 19). Due to the pipelining sheme, matches are determined while the next DMA read or write is being performed.
The RDY line can go inactive after the matching operation begins without affecting this bus-release timing.

INTERRUPTS

Timings for interrupt acknowledge and return from interrupt are the same as timings for these in other Z80 peripherals (See Applicatilonn Note "Z80 Family Interrupt Structure).

Interrupt on RDY (interrupt before requesting bus) does not directly affect the $\overline{BUSREQ}$ line. Instead, the interrupt service routine must handle this by issuing the following commands to WR6 :

1. Enable after Return From Interrupt (RETI) Command-Hex B7
2. Enable DMA-Hex 87
3. An RETI instruction that reset the Interrupt Under Service latch in the Z80 DMA.

Figure 14 : Variable-Cycle and Edge Timing.

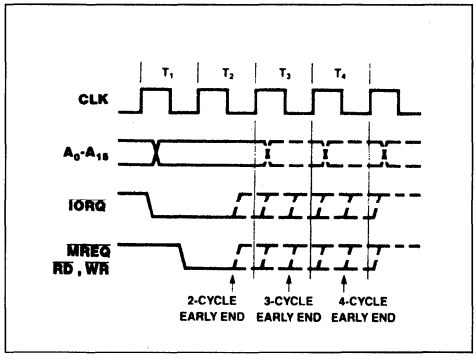


Figure 15 : Bus Request and Acceptance.

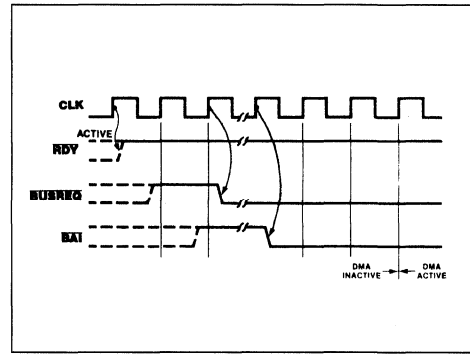


Figure 16 : Bus Release (Byte-at-a-Time-Mode).

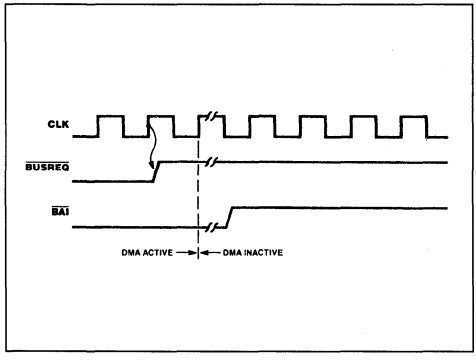


Figure 17 : Bus Release at End of Block (Burst and Continuous Modes).

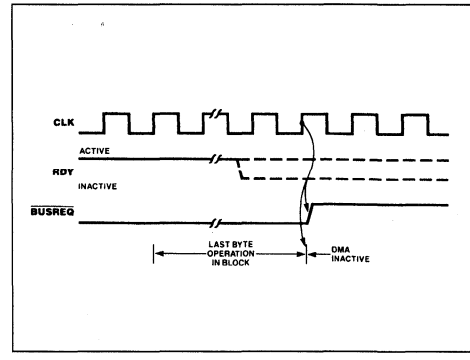


Figure 18 : Bus Release When Not Ready (Burst Mode).

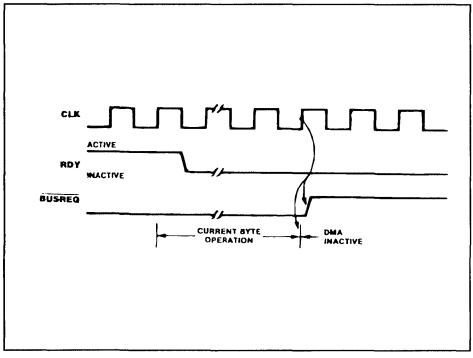
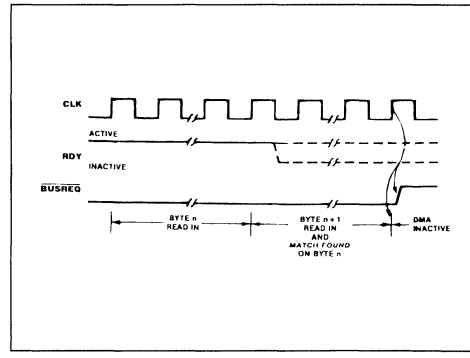


Figure 19 : Bus Release on Match (Burst and Continuous Modes).



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_{CC}	V_{CC} Supply Voltage with Respect to V_{SS}	– 0.5 to 7	V
V_{IN}	Input Voltage	– 0.5 to V_{CC} + 0.5	V
P_D	Power Dissipation (T_A = 85 °C)	250	mW
T_{SOLDER}	Soldering Temperature (soldering time 10 sec)	260	°C
T_{stg}	Storage Temperature	– 65 to 150	°C
T_{opr}	Operating Temperature	– 40 to 85	°C

DC CHARACTERISTICS

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_{ILC}	Clock Input Low Voltage		– 0.3		0.6	V
V_{IHC}	Clock Input High Voltage		V_{CC} – 0.6		V_{CC} + 0.3	V
V_{IL}	Input Low Voltage (except CLK)		– 0.5		0.8	V
V_{IH}	Input High Voltage (except CLK)		2.2		V_{CC}	V
V_{OL}	Output Low Voltage	I_{OL} = 3.2 mA for $\overline{BUSREQ}$ I_{OL} = 2.0 mA or all others			0.4	V
V_{OH1}	Output High Voltage (1)	I_{OH} = – 1.6 mA	2.4			V
V_{OH2}	Output High Voltage (2)	I_{OH} = – 250 µA	V_{CC} – 0.8			V
I_{LI}	Input Leakage Current	$V_{SS} \le V_{IN} \le V_{CC}$			± 10	µA
I_{LO}	3-State Output Leakage Current in Float	V_{SS} + 0.4 ≤ $V_{IN} \le V_{CC}$			± 10	µA
I_{CC1}	Operating Supply Current : 4 MHz 6 MHz	 V_{CC} = 5 V, fCKL = 1/T_CC(min) $V_{IH} = V_{CC}$ – 0.2 V, V_{IL} = 0.2 V		 5 6	 7 10	 mA mA
I_{CC2}	Standby Supply Current	V_{CC} = 5 V, $V_{IH} = V_{CC}$ – 0.2 V V_{IL} = 0.2 V			10	µA

TEST CONDITIONS

$T_A = -40$ °C to + 85 °C

$V_{CC} = 5$ V ± 10 %

$V_{SS} = 0$ V

AC TEST CONDITIONS

Inputs except CLK (clock) are driven at 2.4 V for a logic "1" and 0.4 V for a logic "0". Clock input is driven at $V_{CC} - 0.6$ V for a logic "1" and 0.6 V for a logic "0".

Timing measurements are made at 2.2 V for a logic "1" and 0.8 V for a logic "0".

All AC parameters assume a load capacitance of 100 pF.

CAPACITANCE

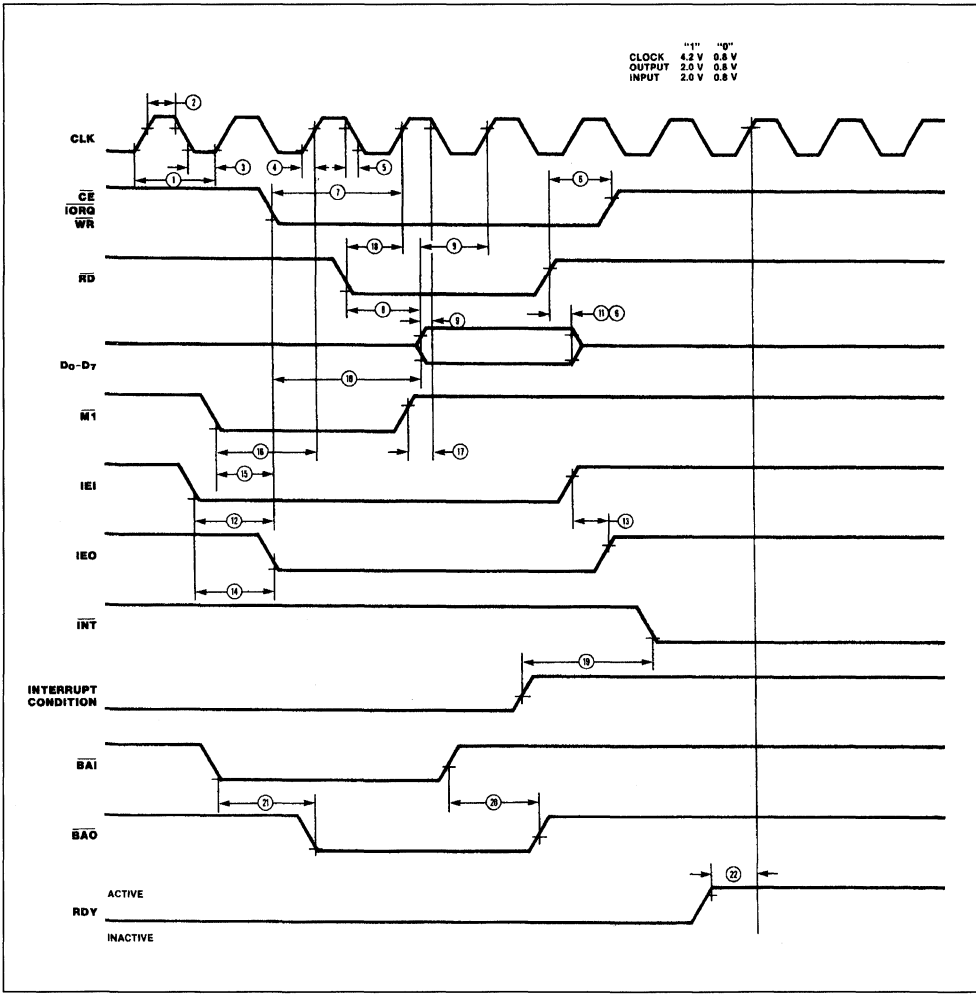
Symbol	Parameter	Test Conditions	Min.	Max.	Unit
C	Clock Capacitance	Unmeasured Pins		5	pF
C_{IN}	Input Capacitance	Returned to Ground		5	pF
C_{OUT}	Output Capacitance			5	pF

INACTIVE STATE AC CHARACTERISTICS

N°	Symbol	Parameter	Z84C10A		Z84C10B	
			Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
1	TcC	Clock Cycle Time	250	DC	165	DC
2	TwCh	Clock Width (high)	110	DC	65	DC
3	TwCl	Clock Width (low)	110	DC	65	DC
4	TrC	Clock Rise Time		30		20
5	TfC	Clock Fall Time		30		20
6	Th	Hold Time for any Specified Setup Time	0		0	
7	TsC(Cr)	$\overline{IORQ}$, $\overline{WR}$, $\overline{CE}$ ↓ to Clock ↑ Setup	145		60	
8	TdDO(RDf)	$\overline{RD}$ ↓ to Data Output Delay		380		300
9	TsWM(Cr)	Data in to Clock ↑ Setup ($\overline{WR}$ or $\overline{MI}$)	50		30	
10	TdCf(DO)	$\overline{IORQ}$ ↓ to Data Out Delay (INTA cycle)		160		100
11	TdRD(Dz)	$\overline{RD}$ ↑ to Data Float Delay (output buffer disable)		110		70
12	TsIEI(IORQ)	IEI ↓ to $\overline{IORQ}$ ↓ Setup (INTA cycle)	140		100	
13	TdIEOr(IEIr)	IEI ↑ to IEO ↑ Delay		160		70
14	TdIEOf(IEIf)	IEI ↓ to IEO ↓ Delay		130		70
15	TdMI(IEO)	$\overline{MI}$ ↓ to IEO ↓ Delay (interrupt just prior to $\overline{MI}$ ↓)		190		100
16	TsMIf(Cr)	$\overline{MI}$ ↓ to Clock ↑ Setup	90		70	
17	TsMIr(Cf)	$\overline{MI}$ ↑ to Clock ↓ Setup	− 10		− 10	
18	TsRD(Cr)	$\overline{RD}$ ↓ to Clock ↑ Setup ($\overline{MI}$ cycle)	115		60	
19	TdI(INT)	Interrupt Cause to $\overline{INT}$ ↓ Delay ($\overline{INT}$ generated only when DMA is inactive)		500		450
20	TdBAIr(BAOr)	$\overline{BAI}$ ↑ to $\overline{BAO}$ ↑ Delay		150		100
21	TdBAIf(BAOf)	$\overline{BAI}$ ↓ to $\overline{BAO}$ ↓ Delay		150		100
22	TsRDY(Cr)	RDY Active to Clock ↑ Setup	100		50	

Note : 1. Negative minimum setup values mean that the first mentioned event can come after the second mentioned event.

INACTIVE STATE AC CHARACTERISTICS (continued).



ACTIVE STATE AC CHARACTERISTICS

N°	Symbol	Parameter	Z84C10A		Z84C10B	
			Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
1	TcC	Clock Cycle Time	250	DC	165	DC
2	TwCh	Clock Width (high)	110	DC	65	DC
3	TwCl	Clock Width (low)	110	DC	65	DC
4	TrC	Clock Rise Time		30		20
5	TfC	Clock Fall Time		110		90
6	TdA	Address Output Delay		110		90
7	TdC(Az)	Clock ↑ to Address Float Delay		90		80
8	TsA(MREQ)	Address to $\overline{MREQ}$ ↓ Setup (memory cycle)	65		35	
9	TsA(IRW)	Address Stable to $\overline{IORQ}$, $\overline{RD}$, $\overline{WR}$ ↓ Setup (I/O cycle)	180		110	
*10	TdRW(A)	$\overline{RD}$, $\overline{WR}$ ↑ to Addr. Stable Delay	90		35	
*11	TdRW(Az)	$\overline{RD}$, $\overline{WR}$ ↑ to Addr. Float	95		65	
12	TdCf(DO)	Clock ↓ to Data Out Delay		150		130
*13	TdCr(Dz)	Clock ↑ to Data Float Delay (write cycle)		90		70
14	TsDI(Cr)	Data in to Clock ↑ Setup (read cycle when rising edge ends read)	35		30	
15	TsDI(Cf)	Data in to Clock ↓ Setup (read cycle when falling edge ends read)	50		40	
*16	TsDO(WfM)	Data out to $\overline{WR}$ ↓ Setup (memory cycle)	80		25	
17	TsDO(WfI)	Data Out to $\overline{WR}$ ↓ Setup (I/O cycle)	100		55	
*18	TdWr(DO)	$\overline{WR}$ ↑ to Data Out Delay	70		30	
19	Th	Hold Time for Any Specified Setup Time	0		0	
20	TdCr(Mf)	Clock ↑ to $\overline{MREQ}$ ↓ Delay		85		70
21	TdCf(Mf)	Clock ↓ to $\overline{MREQ}$ ↓ Delay		85		70
22	TdCr(Mr)	Clock ↑ to $\overline{MREQ}$ ↑ Delay		85		70
23	TdCf(Mr)	Clock ↓ to $\overline{MREQ}$ ↑ Delay		85		70
24	TwMl	$\overline{MREQ}$ Low Pulse Width	220		135	
*25	TwMh	$\overline{MREQ}$ High Pulse Width	120		65	
26	TdCf(If)	Clock ↓ to $\overline{IORQ}$ ↓ Delay		85		70
27	TdCr(If)	Clock ↑ to $\overline{IORQ}$ ↓ Delay		75		65
28	TdCr(Ir)	Clock ↑ to $\overline{IORQ}$ ↑ Delay		85		70
*29	TdCf(Ir)	Clock ↓ to $\overline{IORQ}$ ↑ Delay		85		70

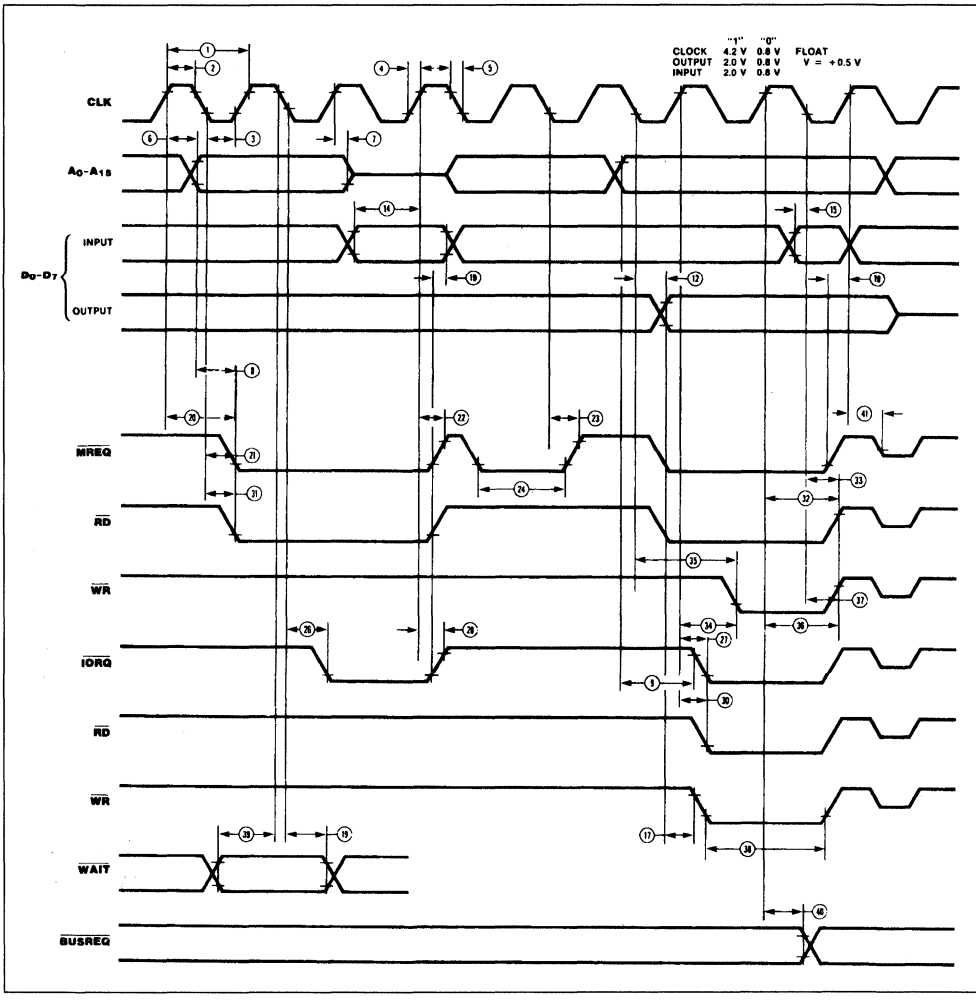
Notes : 1. Numbers in parentheses are other parameter-numbers in this table ; their values should be substituted in equations.
2. All equations imply DMA default (standart) timing.
3. Data must be enabled onto data bus when $\overline{RD}$ is active.
4. Asterisk (*) before parameter number means the parameter is not illustrated in the AC Timing Diagrams.

ACTIVE STATE AC CHARACTERISTICS (continued)

N°	Symbol	Parameter	Z84C10A		Z84C10B	
			Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
30	TdCr(Rf)	Clock ↑ to $\overline{RD}$ ↓ Delay		85		70
31	TdCf(Rf)	Clock ↓ to $\overline{RD}$ ↓ Delay		95		80
32	TdCr(Rr)	Clock ↑ to $\overline{RD}$ ↓ Delay		85		70
33	TdCf(Rr)	Clock ↓ to $\overline{RD}$ ↑ Delay		85		70
34	TdCr(Wf)	Clock ↑ to $\overline{WR}$ ↓ Delay		65		60
35	TdCf(Wf)	Clock ↓ to WR ↓ Delay		80		60
36	TdCr(Wr)	Clock ↑ to $\overline{WR}$ ↑ Delay		80		70
37	TdCf(Wr)	Clock ↓ to $\overline{WR}$ ↑ Delay		80		70
38	TwWl	$\overline{WR}$ Low Pulse Width	220		135	
39	TsWA(Cf)	$\overline{WAIT}$ to Clock ↓ Setup	70		60	
40	TdCr(B)	Clock ↑ to $\overline{BUSREQ}$ Delay		100		90
41	TdCr(Iz)	Clock ↑ to $\overline{IORQ}$, $\overline{MREQ}$, $\overline{RD}$, $\overline{WR}$ Float Delay		80		70

Notes : 1. Numbers in parentheses are other parameter-numbers in this table ; their values should be substituted in equations.
2. All equations imply DMA default (standart) timing.
3. Data must be enabled onto data bus when $\overline{RD}$ is active.
4. Asterisk (*) before parameter number means the parameter is not illustrated in the AC tTiming Diagrams.

ACTIVE STATE AC CHARACTERISTICS (continued).

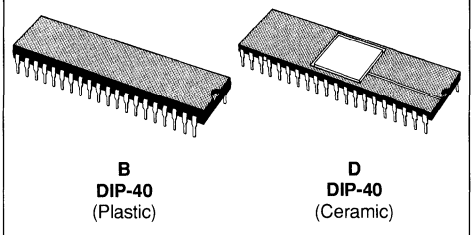


ORDERING INFORMATION

Type	Package	Temp.	Clock	Description
Z84C10AB6	DIP–40 (plastic)	– 40/ + 85 °C	4 MHz	Z80C Direct Memory Access Unit
Z84C10AC6	PLCC44 (plastic chip-carrier)	– 40/ + 85 °C		
Z84C10BB6	DIP–40 (plastic)	– 40/ + 85 °C	6 MHz	
Z84C10BC6	PLCC44 (plastic chip-carrier)	– 40/ + 85 °C		

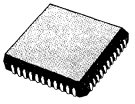
Z80C PIO CMOS VERSION

- PROVIDES A DIRECT INTERFACE BETWEEN Z80 MICROCOMPUTER SYSTEMS AND PERIPHERAL DEVICES
- BOTH PORTS HAVE INTERRUPT-DRIVEN HANDSHAKE FOR FAST RESPONSE
- FOUR PROGRAMMABLE OPERATING MODES : BYTE INPUT, BYTE OUTPUT, BYTE INPUT/OUTPUT (port A only), AND BIT INPUT/OUTPUT
- PROGRAMMABLE INTERRUPTS ON PERIPHERAL STATUS CONDITIONS
- STANDARD Z80 FAMILY BUS-REQUEST AND PRIORITIZED INTERRUPT-REQUEST DAISY CHAINS IMPLEMENTED WITHOUT EXTERNAL LOGIC
- THE EIGHT PORT B OUTPUTS CAN DRIVE DARLINGTON TRANSISTORS (1.5 mA at 1.5 V)
- SINGLE 5 V ± 10 % POWER SUPPLY
- LOW POWER CONSUMPTION :
  - 2 mA typ. at 4 MHz
  - 3 mA typ. at 6 MHz
  - less than 10 µA in Power Down mode
- EXTENDED OPERATING TEMPERATURE :
  - 40 °C TO + 85 ° C



B
DIP-40
(Plastic)

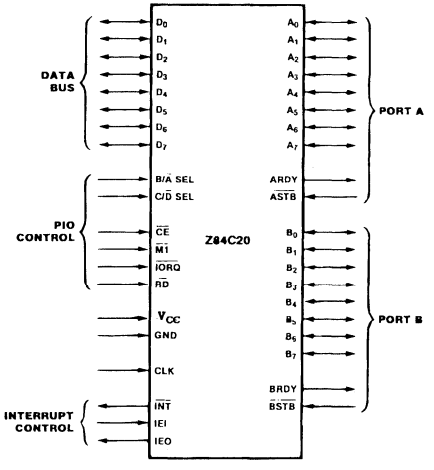
D
DIP-40
(Ceramic)



C
PLCC44
(Plastic)

(Ordering Information at the end of the datasheet)

LOGIC FUNCTIONS



DESCRIPTION

The Z80C PIO Parallel I/O Circuit is a programmable, dual-port device that provides a TTL-compatible interface between peripheral devices and the CPU. The CPU configures the PIO to interface with a wide range of peripheral devices with no other external logic. Typical peripheral devices that are compatible with the PIO include most keyboards, paper tape readers and punches, printers, PROM programmers, etc.

One characteristic of the Z80 peripheral controllers that separates them from other interface controllers is that all data transfer between the peripheral device and the CPU is accomplished under interrupt control.

Thus, the interrupt logic of the PIO permits full use of the efficient interrupt capabilities of the CPU during I/O transfers. All logic necessary to implement a fully nested interrupt structure is included in the PIO.

Figure 1 : Dual in Line Pin Configuration.

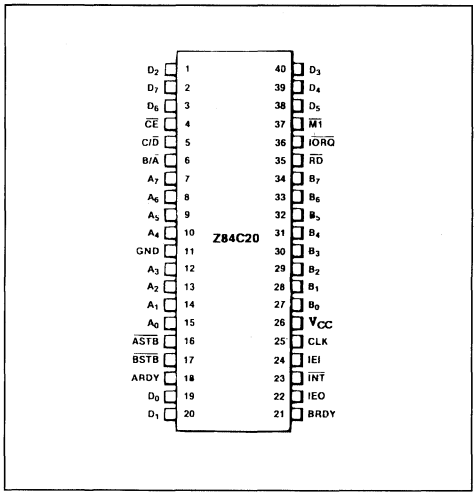
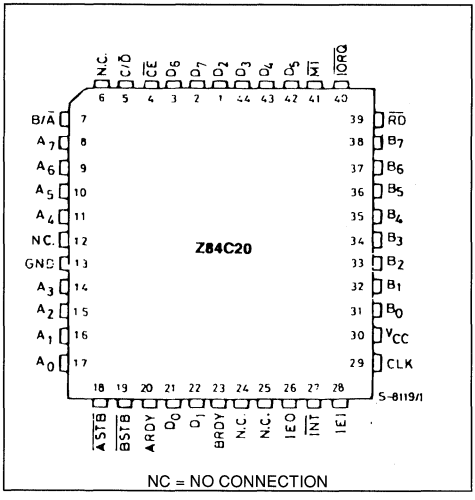


Figure 2 : Chip Carrier Pin Configuration.



NC = NO CONNECTION

Another feature of the PIO is the ability to interrupt the CPU upon occurrence of specified status conditions in the peripheral device. For example, the PIO can be programmed to interrupt if any specified peripheral alarm conditions should occur. This interrupt capability reduces the time the processor must spend in polling peripheral status.

The Z80C PIO interfaces to peripherals via two independent general-purpose I/O ports, designated Port A and Port B. Each port has eight data bits and two handshake signals, Ready and Strobe, which control data transfer. The Ready output indicates to the peripheral that the port is ready for a data transfer. Strobe is an input from the peripheral that indicates when a data transfer has occured.

OPERATING MODES

The PIO ports can be programmed to operate in four modes : byte output (Mode 0), byte input (Mode 1), byte input/output (Mode 2) and bit input/output (Mode 3).

In Mode 0, either Port A or Port B can be programmed to output data. Both ports have output registers that are individually addressed by the CPU ; data can be written to either port at any time. When data is written to a port, an active Ready output indicates to the external device that data is available at the associated port and is ready for transfer to the external device. After the data transfer, the external device responds with an active Strobe input, which generates an interrupt, if enabled.

In Mode 1, either Port A or Port B can be configured in the input mode. Each port has an input register addressed by the CPU. When the CPU reads data from a port, the PIO sets the Ready signal, which is detected by the external device. The external device then places data on the I/O lines and strobes the I/O port, which latches the data into the Port Input Register, resets Ready, and triggers the Interrupt Request, if enabled. The CPU can read the input data at any time, which again sets Ready.

Mode 2 is bidirectional and uses Port A, plus the interrupts and handshake signals from both ports. Port B must be set to Mode 3 and masked off. In operation, Port A is used for both data input and output. Output operation is similar to Mode 0 except that data is allowed out onto the Port A bus only when ASTB is Low. For input, operation is similar to Mode 1, except that the data input uses the Port B handshake signals and the Port B interrupt (if enabled).

Both ports can be used in Mode 3. In this mode, the individual bits are defined as either input or output bits. This provides up to eight separate, individually defined bits for each port. During operation, Ready and Strobe are not used. Instead, an interrupt is generated if the condition of one input changes, or if all inputs change. The requirements for generating an interrupt are defined during the programming operation ; the active level is specified as either High or Low, and the logic condition is specified as either one input active (OR) or all inputs active (AND). For example, if the port is programmed for active Low inputs and the logic function is AND, then all inputs at the specified port must go Low to generate an interrupt.

Data outputs are controlled by the CPU and can be written or changed at any time.

- Individual bits can be masked off
- The handshake signals are not used in Mode 3, Ready is held Low, and Strobe is disabled
- When using PIO interrupts, the CPU interrupt mode must be set to Mode 2.

INTERNAL STRUCTURE

The internal structure of the Z80C PIO consists of a CPU bus interface, internal control logic, Port A I/O logic, Port B I/O logic, and interrupt control logic (figure 3). The CPU bus interface logic allows the PIO to interface directly to the CPU with no other external logic. The internal control logic synchronizes the CPU data bus to the peripheral device interfaces (Port A and Port B). The two I/O ports (A and B) are virtually identical and are used to interface directly to peripheral devices.

PORT LOGIC

Each port contains separate input and output registers, handshake control logic, and the control registers shown in figure 4. All data transfers between the peripheral unit and the CPU use the data input and output registers. The handshake logic associated with each port controls the data transfers through the input and the output registers. The mode control register (two bits) selects one of the four programmable operating modes.

The control mode (Mode 3) uses the remaining registers. The input/output control register specifies which of the eight data bits in the port are to be outputs and enables these bits ; the remaining bits are inputs. The mask register and the mask control register control Mode 3 interrupt conditions. The mask register specifies which of the bits in the port are active and which are masked or inactive.

The mask control register specifies two conditions : first, whether the active state of the input bits is High or Low, and second, whether an interrupt is generated when any one unmasked input bit is active (OR condition) or if the interrupt is generated when all unmasked input bits are active (AND condition).

INTERRUPT CONTROL LOGIC

The interrupt control logic section handles all CPU interrupt protocol for nested-priority interrupt structures. Any device's physical location in a daisy-chain configuration determines its priority. Two lines (IEI and IEO) are provided in each PIO to form this daisy chain. The device closest to the CPU has the highest priority. Within a PIO, Port A interrupts have higher priority than those of Port B. In the byte input, byte output, or bidirectional modes, an interrupt can be generated whenever the peripheral requests a new byte transfer. In the bit control mode, an interrupt can be generated when the peripheral status matches a programmed value. The PIO provides for complete control of nested interrupts. That is, lower priority devices may not interrupt higher priority devices that have not had their interrupt service routines completed by the CPU. Higher priority devices may interrupt the servicing of lower priority devices.

If the CPU (in interrupt Mode 2) accepts an interrupt, the interrupting device must provide an 8-bit interrupt vector for the CPU. This vector forms a pointer to a location in memory where the address of the interrupt service routine is located. The 8-bit vector from the interrupting device forms the least significant eight bits of the indirect pointer while the I Register in the CPU provides the most significant eight

Figure 3 : Block Diagram.

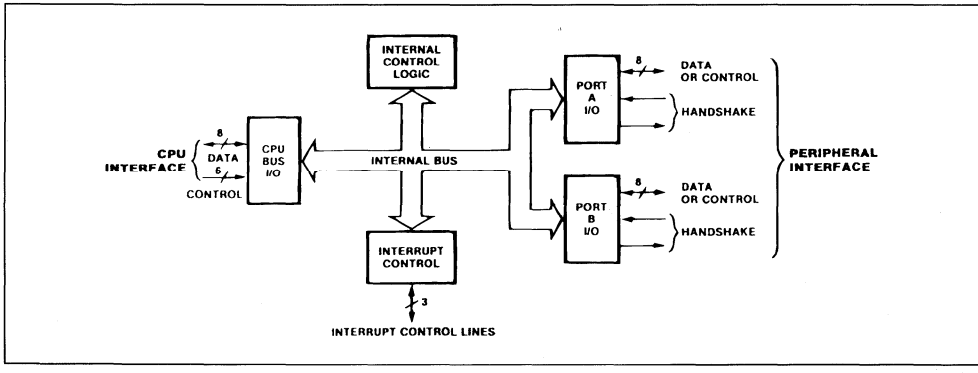
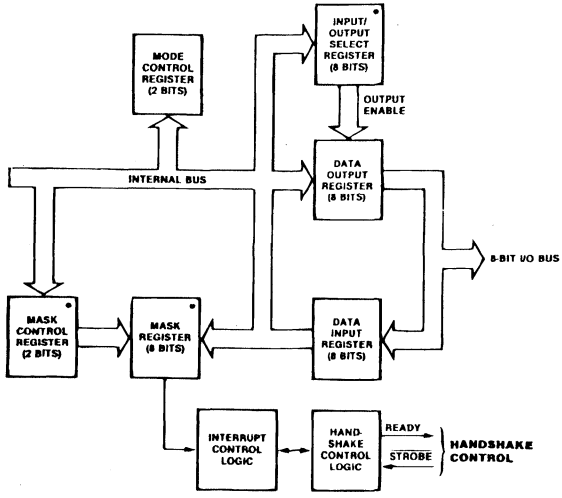


Figure 4 : Typical Port I/O Block Diagram.



* Used in the bit mode only to allow generation of an interrupt if the peripheral I/O pins go to the specified state.

bits of the pointer. Each port (A and B) has an independent interrupt vector. The least significant bit of the vector is automatically set to 0 within the PIO because the pointer must point to two adjacent memory locations for a complete 16-bit address.

Unlike the other Z80C peripherals, the PIO does not enable interrupts immediately after programming. It waits until $\overline{M1}$ goes Low (e.g., during an opcode fetch). This condition is unimportant in the Z80C environment but might not be if another type of CPU is used.

The PIO decodes the RETI (Return From Interrupt) instruction directly from the CPU data bus so that each PIO in the system knows at all times whether it is being serviced by the CPU interrupt service routine. No other communication with the CPU is required.

CPU BUS I/O LOGIC

The CPU bus interface logic interfaces the PIO directly to the CPU so no external logic is necessary. For large systems, however, address decoders and/or buffers may be necessary.

INTERNAL CONTROL LOGIC

This logic receives the control words for each port during programming and, in turn, controls the operating functions of the PIO. The control logic synchronizes the port operations, controls the port mode, port addressing, selects the read/write function, and issues appropriate commands to the ports and the interrupt logic. The PIO does not receive a write input from the CPU ; instead, the $\overline{RD}$, $\overline{CE}$, C/$\overline{D}$ and IORQ signals generate the write input internally.

PROGRAMMING

MODE 0, 1, OR 2.

(Byte Input, Output, or Bidirectional). Programming a port for Mode 0, 1, or 2 requires two words per port. These words are :

A MODE CONTROL WORD. Selects the port operating mode (figure 5). This word may be written any time.

AN INTERRUPT VECTOR. The Z80C PIO is designed for use with the Z80C CPU in interrupt Mode 2 (figure 6). When interrupts are enabled, the PIO must provide an interrupt vector.

MODE 3

(Bit Input/Output). Programming a port for Mode 3 operation requires a control word, a vector (if interrupts are enabled), and three additional words, described as follows :

I/O REGISTER CONTROL. When Mode 3 is selected, the mode control word must be followed by

Figure 5 : Mode Control Word.

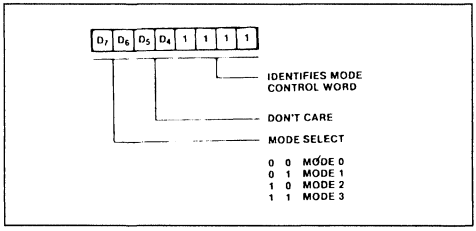


Figure 6 : Interrupt Vector Word.

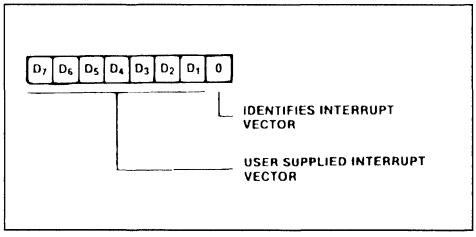


Figure 7 : I/O Register Control Word.

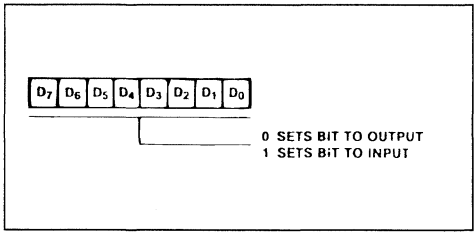
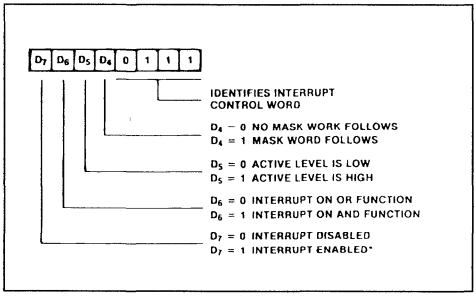


Figure 8 : Interrupt Control Word.



another control word that sets the I/O control register, which in turn defines which port lines are inputs and which are outputs (figure 7).

INTERRUPT CONTROL WORD. In Mode 3, handshake is not used. Interrupts are generated as a logic function of the input signal levels. The interrupt control word sets the logic conditions and the logic levels required for generating an interrupt. Two logic conditions or functions are available : AND (if all input bits change to the active level, an interrupt is triggered), and OR (if any one of the input bits changes to the active level, an interrupt is triggered). Bit D_6 sets the logic function, as shown in figure 8. The active level of the input bits can be set either High or Low. The active level is controlled by Bit D_5.

MASK CONTROL WORD. This word sets the mask control register, allowing any unused bits to be masked off. If any bits are to be masked, then D_4 must be set. When D_4 is set, the next word written to the port must be a mask control word (figure 9).

INTERRUPT DISABLE

There is one other control word which can be used to enable or disable a port interrupt. It can be used without changing the rest of the interrupt control word (figure 10).

Figure 9 : Mask Control Word.

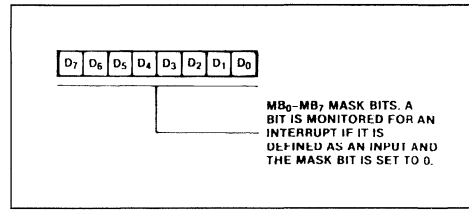
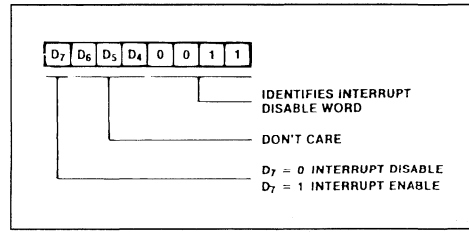


Figure 10 : Interrupt Disable Word.



PIN DESCRIPTIONS

A_0-A_7. *Port A Bus* (Bidirectional, 3-state). This 8-bit bus transfers data, status, or control information between Port A of the PIO and a peripheral device. A_0 is the least significant bit of the Port A data bus.

ARDY. *Register A Ready* (Output, Active High). The meaning of this signal depends on the mode of operation selected for Port A as follows :

OUTPUT MODE. This signal goes active to indicate that the Port A output register has been loaded and the peripheral data bus is stable and ready for transfer to the peripheral device.

INPUT MODE. This signal is active when the Port A input register is empty and ready to accept data from the peripheral device.

BIDIRECTIONAL MODE. This signal is active when data is available in the Port A output register for transfer to the peripheral device. In this mode, data is not placed on the Port A data bus, unless ASTB is active.

CONTROL MODE. This signal is disabled and forced to a Low state.

$\overline{\text{ASTB}}$. *Port A Strobe Pulse From Peripheral Device* (Input, Active Low). The meaning of this signal depends on the mode of operation selected for Port A as follows :

OUTPUT MODE. The positive edge of this strobe is issued by the peripheral to acknowledge the receipt of data made available by the PIO.

INPUT MODE. The strobe is issued by the peripheral to load data from the peripheral into the Port A input register. Data is loaded into the PIO when this signal is active.

BIDIRECTIONAL MODE. When this signal is active, data from the Port A output register is gated onto the Port A bidirectional data bus. The positive edge of the strobe acknowledges the receipt of the data.

CONTROL MODE. The strobe is inhibited internally.

B_0-B_7. *Port B Bus* (Bidirectional, 3-state). This 8-bit bus transfers data, status, or control information between Port B and a peripheral device. The Port B data bus can supply 1.5mA at 1.5V to drive Darlington transistors. B_0 is the least significant bit of the bus.

B/$\overline{\text{A}}$. *Port B Or A Select* (Input, High = B). This pin defines which port is accessed during a data transfer between the CPU and the PIO. A Low on this pin selects Port A ; a High selects Port B. Often address bit A_0 from the CPU is used for this selection function.

BRDY. *Register B Ready* (Output, Active High). This signal is similar to ARDY, except that in the Port A bidirectional mode this signal is High when the Port A input register is empty and ready to accept data from the peripheral device.

$\overline{\text{BSTB}}$. *Port B Strobe Pulse From Peripheral Device* (Input, Active Low). This signal is similar to ASTB, except that in the Port A bidirectional mode this signal strobes data from the peripheral device into the Port A input register.

C/$\overline{\text{D}}$. *Control Or Data Select* (Input, High = C). This pin defines the type of data transfer to be performed between the CPU and the PIO. A High on this pin during a CPU write to the PIO causes the data bus to be interpreted as a command for the port selected by the B/A Select line. A Low on this pin means that the data bus is being used to transfer data between the CPU and the PIO. Often address bit A_1 from the CPU is used for this function.

$\overline{\text{CE}}$. *Chip Enable* (Input, Active Low). A Low on this pin enables the PIO to accept command or data inputs from the CPU during a write cycle or to transmit data to the CPU during a read cycle. This signal is generally decoded from four I/O port numbers for Ports A and B, data, and control.

CLK. *System Clock* (Input). The Z80C PIO uses the standard single-phase Z80C system clock.

D_0-D_7. *CPU Data Bus* (Bidirectional, 3-state). This bus is used to transfer all data and commands between the CPU and the PIO. D_0 is the least significant bit.

IEI. *Interrupt Enable In* (Input, Active High). This signal is used to form a priority-interrupt daisy chain when more than one interrupt-driven device is being used. A High level on this pin indicates that no other devices of higher priority are being serviced by a CPU interrupt service routine.

IEO. *Interrupt Enable Out* (Output, Active High). The IEO signal is the other signal required to form a daisy chain priority scheme. It is High only if IEI is High and the CPU is not servicing an interrupt from this PIO. Thus this signal blocks lower priority devices from interrupting while a higher priority device is being serviced by its CPU interrupt service routine.

$\overline{\text{INT}}$. *Interrupt Request* (Output, Open Drain, Active Low). When $\overline{\text{INT}}$ is active the PIO is requesting an interrupt from the CPU.

$\overline{\text{IORQ}}$. *Input/Output Request* (Input from CPU, Active Low). $\overline{\text{IORQ}}$ is used in conjunction with B/$\overline{\text{A}}$, C/$\overline{\text{D}}$, $\overline{\text{CE}}$, and $\overline{\text{RD}}$ to transfer commands and data be-

tween the CPU and the PIO. When $\overline{CE}$, $\overline{RD}$, and IORQ are active, the port addressed by B/A transfers data to the CPU (a read operation). Conversely, when $\overline{CE}$ and IORQ are active but $\overline{RD}$ is not, the port addressed by B/A is written into from the CPU with either data or control information, as specified by C/D. Also, if IORQ and $\overline{M1}$ are active simultaneously, the CPU is acknowledging an interrupt ; the interrupting port automatically places its interrupt vector on the CPU data bus if it is the highest priority device requesting an interrupt.

$\overline{M1}$. *Machine Cycle* (Input from CPU, Active Low). This signal is used as a sync pulse to control several internal PIO operations. When both the $\overline{M1}$ and $\overline{RD}$ signals are active, the CPU is fetching an instruction from memory. Conversely, when both $\overline{M1}$ and IORQ are active, the CPU is acknowledging an interrupt. In addition, $\overline{M1}$ has two other functions within the PIO : it synchronizes the PIO interrupt logic ; when $\overline{M1}$ occurs without an active $\overline{RD}$ or IORQ signal, the PIO is reset.

$\overline{RD}$. *Read Cycle Status* (Input from CPU, Active Low). If $\overline{RD}$ is active, or an I/O operation is in progress, $\overline{RD}$ is used with B/A, C/D, $\overline{CE}$, and IORQ to transfer data from the PIO to the CPU.

TIMING

The following timing diagrams show typical timing in a Z80C CPU environment. For more precise specifications refer to the composite ac timing diagram.

WRITE CYCLE

Figure 11 illustrates the timing for programming the Z80C PIO or for writing data to one of its ports. No Wait states are allowed for writing to the PIO other than the automatically inserted T_{WA}. The PIO does not receive a specific write signal ; it internally generates its own from the lack of an active $\overline{RD}$ signal.

READ CYCLE

Figure 12 illustrates the timing for reading the data input from an external device to one of the PIO ports. No Wait states are allowed for reading the PIO other than the automatically inserted T_{WA}.

OUTPUT MODE (MODE 0)

An output cycle (figure 13) is always started by the execution of an output instruction by the CPU. The $\overline{WR}$* pulse from the CPU latches the data from the CPU data bus into the selected port's output register. The $\overline{WR}$* pulse sets the Ready flag after a Low-going edge of CLK, indicating data is available. Ready stays active until the positive edge of the strobe line is received, indicating that data was taken by the peripheral. The positive edge of the strobe pulse generates an $\overline{INT}$ if the interrupt enable flip-flop has been set and if this device has the highest priority.

INPUT MODE (MODE 1)

When $\overline{STROBE}$ goes Low, data is loaded into the selected port input register (figure 14). The next rising edge of strobe activates $\overline{INT}$, if Interrupt Enable is set and this is the highest-priority requesting device. The following falling edge of CLK resets Ready to an inactive state, indicating that the input register is full and cannot accept any more data until the CPU completes a read. When a read is complete, the positive edge of $\overline{RD}$ sets Ready at the next Low-going transition of CLK. At this time new data can be loaded into the PIO.

BIDIRECTIONAL MODE (MODE 2)

This is a combination of Modes 0 and 1 using all four handshake lines and the eight Port A I/O lines (figure 15). Port B must be set to the bit mode and its inputs must be masked. The Port A handshake lines are used for output control and the Port B lines are used for input control. If interrupts occur, Port A's vector will be used during port output and Port B's will be used during port input. Data is allowed out onto the Port A bus only when $\overline{ASTB}$ is Low. The rising edge of this strobe can be used to latch the data into the peripheral.

BIT MODE (MODE 3)

The bit mode does not utilize the handshake signals, and a normal port write or port read can be executed at any time. When writing, the data is latched into the output registers with the same timing as the output mode (figure 16).

When reading the PIO, the data returned to the CPU is composed of output register data from those port data lines assigned as outputs and input register data from those port data lines assigned as inputs. The input register contains data that was present immediately prior to the falling edge of $\overline{RD}$. An interrupt is generated if interrupts from the port are enabled and the data on the port data lines satisfy the logical equation defined by the 8-bit mask and 2-bit mask control registers. However, if Port A is programmed in bidirectional mode, Port B does not issue an interrupt in bit mode and must therefore be polled.

Figure 11 : Write Cycle Timing.

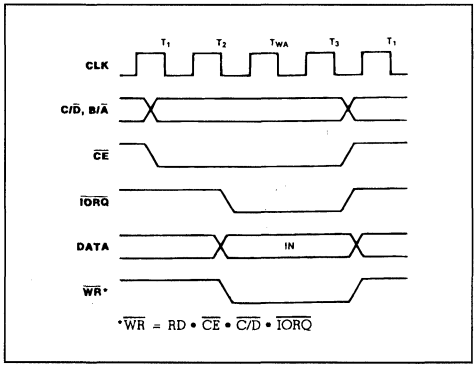


Figure 12 : Read Cycle Timing.

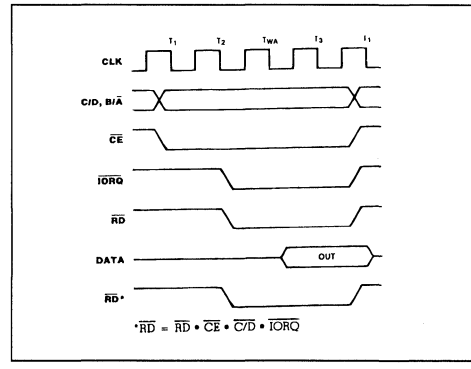
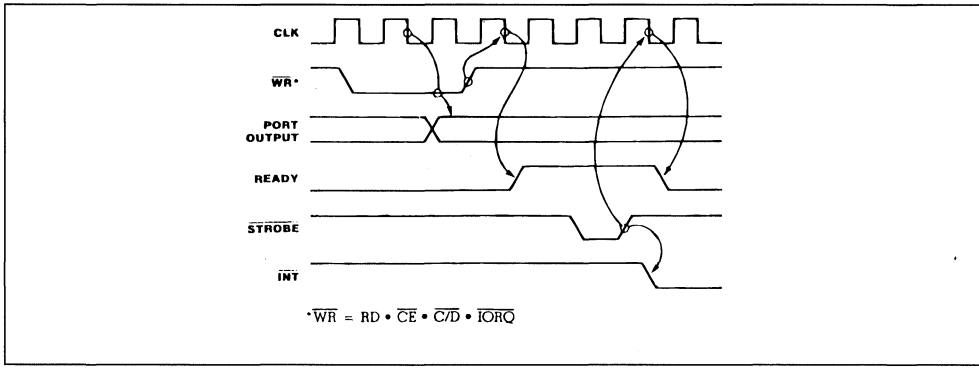


Figure 13 : Mode 0 Output Timing.



INTERRUPT ACKNOWLEDGE TIMING

During M1 time, peripheral controllers are inhibited from changing their interrupt enable status, permitting the Interrupt Enable signal to ripple through the daisy chain. The peripheral with IEI High and IEO Low during INTACK places a preprogrammed 8-bit interrupt vector on the data bus at this time (figure 17). IEO is held Low until a Return From Interrupt (RETI) instruction is executed by the CPU while IEI is High. The 2-byte RETI instruction is decoded internally by the PIO for this purpose.

RETURN FROM INTERRUPT CYCLE

If a Z80C peripheral has no interrupt pending and is not under service, then its IEO = IEI. If it has an interrupt under service (i.e., it has already interrupted and received an interrupt acknowledge) then its IEO is always Low, inhibiting lower priority devices from interrupting. If it has an interrupt pending which has not yet been acknowledged, IEO is Low unless an "ED" is decoded as the first byte of a 2-byte opcode (figure 18). In this case, IEO goes High until the next opcode byte is decoded, whereupon it goes Low again. If the second byte of the opcode was a "4D", then the opcode was an RETI instruction.

After an "ED" opcode is decoded, only the peripheral device which has interrupted and is currently under service has its IEI High and its IEO Low. This device is the highest-priority device in the daisy chain that has received an interrupt acknowledge. All other peripherals have IEI = IEO. If the next opcode byte decoded is "4D", this peripheral device resets its "interrupt under service" condition.

Figure 14 : Mode 1 Input Timing.

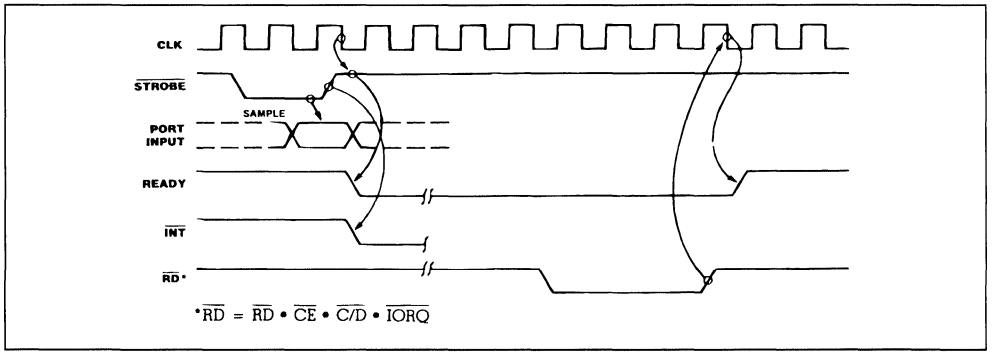


Figure 15 : Mode 2 Bidirectional Timing.

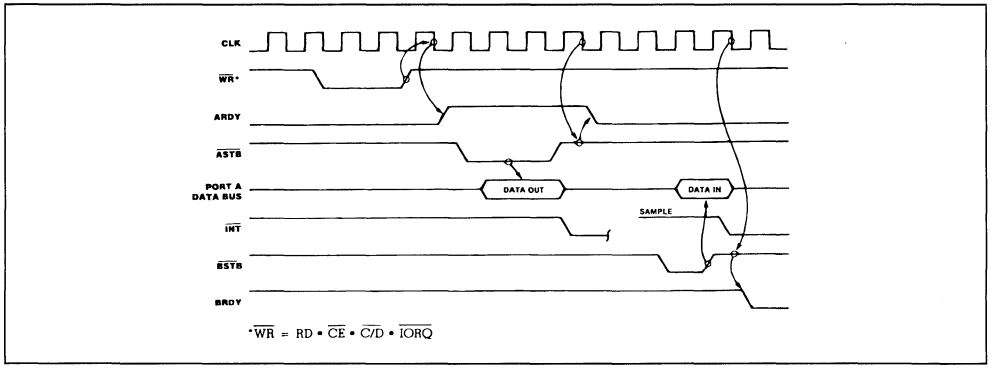
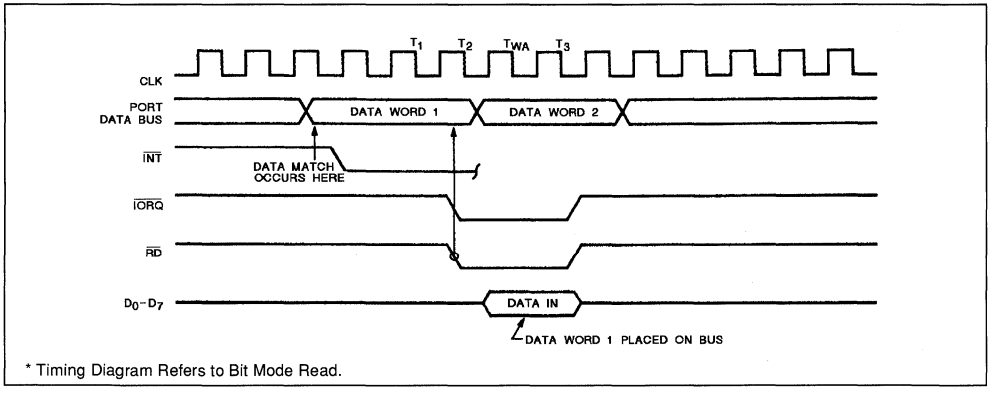


Figure 16 : Mode 3 Bit Mode Timing.



* Timing Diagram Refers to Bit Mode Read.

Figure 17 : Interrupt Acknowledge Timing.

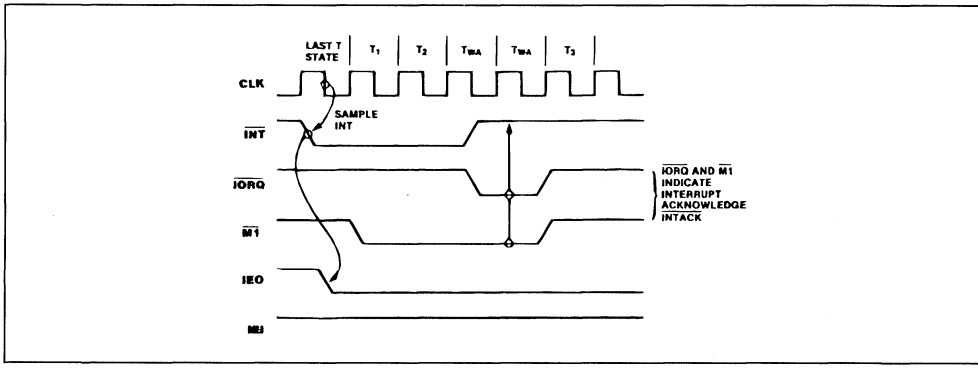
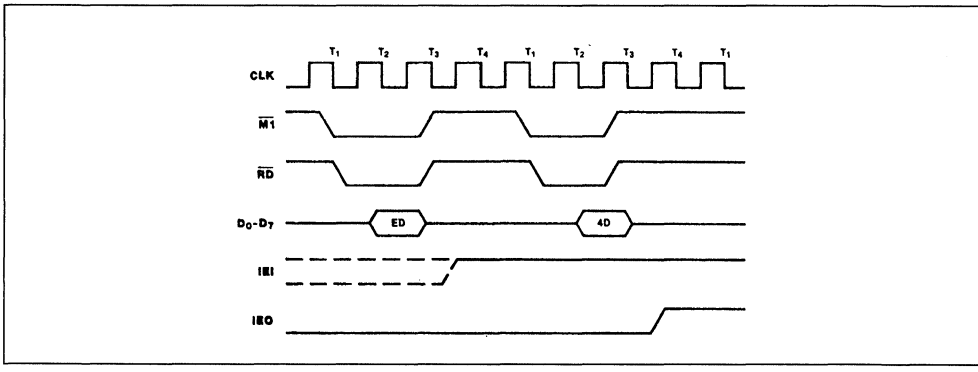
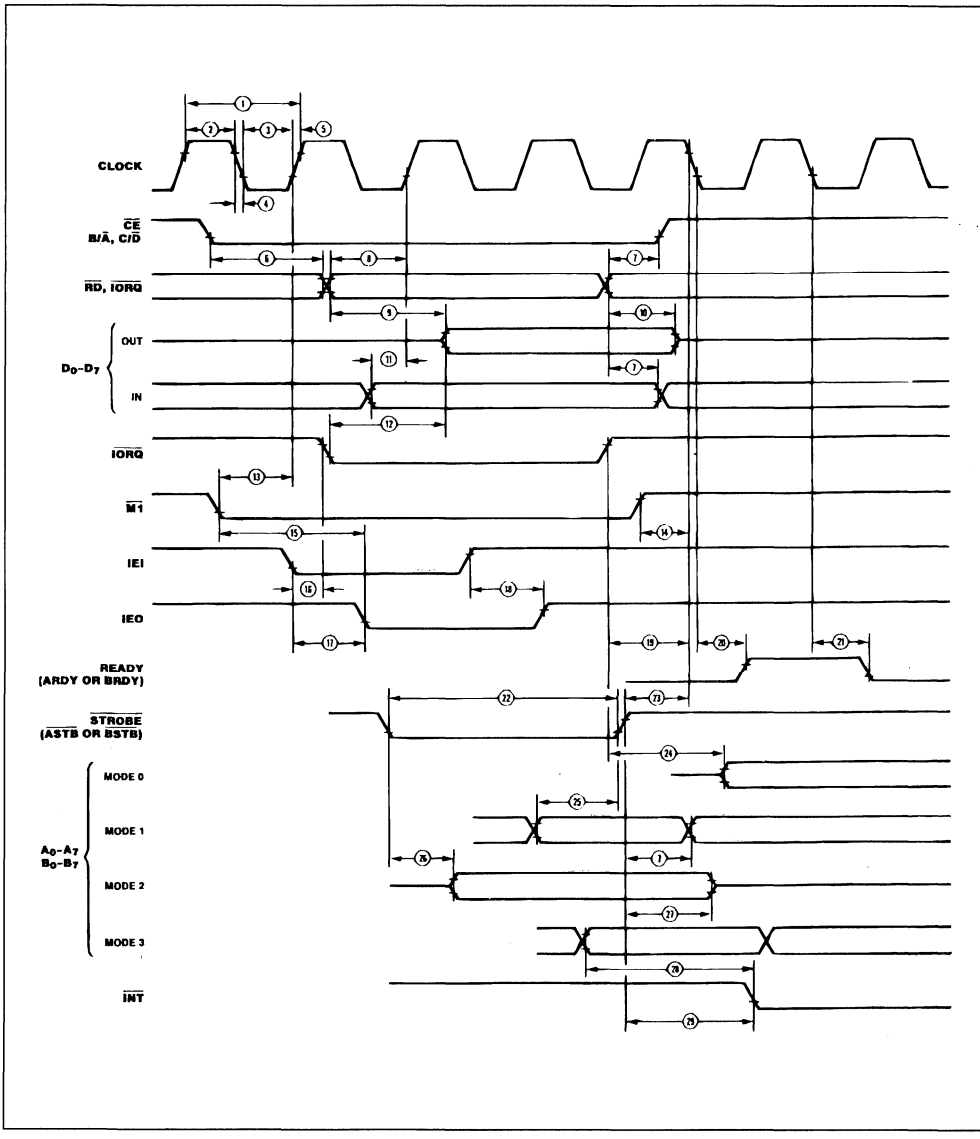


Figure 18 : Return From Interrupt.



AC CHARACTERISTICS



AC CHARACTERISTICS (continued)

N°	Symbol	Parameter	Z84C20A		Z84C20B	
			Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
1	TcC	Clock Cycle Time	250		165	
2	TwCh	Clock Width (high)	105		65	
3	TwCl	Clock Width (low)	105		65	
4	TfC	Clock Fall Time		30		20
5	TrC	Clock Rise Time		30		20
6	TsCS(RI)	$\overline{CE}$, B/$\overline{A}$, C/$\overline{D}$ to $\overline{RD}$, $\overline{IORQ}$ ↓ Setup Time	50		50	
7	Th	Any Hold Times for Specified Setup Time	40		40	
8	TsRI(C)	$\overline{RD}$, $\overline{IORQ}$ to Clock ↑ Setup Time	115		70	
9	TdRI(DO)	$\overline{RD}$, $\overline{IORQ}$ ↓ to Data Out Delay		380		300
10	TdRI(DOs)	$\overline{RD}$, $\overline{IORQ}$ ↑ to Data Out Float Delay		110		70
11	TsDI(C)	Data In to Clock ↑ Setup Time	50		40	
12	TdIO(DOI)	$\overline{IORQ}$ ↓ to Data Out Delay (INTACK Cycle)		160		120
13	TsMI(Cr)	$\overline{MI}$ ↓ to Clock ↑ Setup Time	90		70	
14	TsMI(Cf)	$\overline{MI}$ ↑ to Clock ↓ Setup Time ($\overline{MI}$ Cycle)	0		0	
15	TdMI(IEO)	$\overline{MI}$ ↓ to IEO ↓ Delay (interrupt immediately preceding $\overline{MI}$ ↓)		190		100
16	TsIEI(IO)	IEI ↓ to $\overline{IORQ}$ ↓ Setup Time (INTACK Cycle)	140		100	
17	TdIEI(IEOf)	IEI ↓ to IEO ↓ Delay		130		120
18	TdIEI(IEOr)	IEI ↑ to IEO ↑ Delay (after ED decode)		160		150
19	TcIO(C)	$\overline{IORQ}$ ↑ to Clock ↓ Setup Time (to activate READY on Next Clock Cycle)	200		170	
20	TdC(RDYr)	Clock ↓ to READY ↑ Delay		190		170
21	TdC(RDYf)	Clock ↓ to READY ↓ Delay		140		120
22	TwSTB	$\overline{STROBE}$ Pulse Width	150		120	
23	TsSTB(C)	$\overline{STROBE}$ ↑ to Clock ↓ Setup Time (to activate READY on Next Clock Cycle)	220		150	
24	TdIO(PD)	$\overline{IORQ}$ ↑ to PORT DATA Stable Delay (Mode 0)		180		160
25	TsPD(STB)	PORT DATA to $\overline{STROBE}$ ↑ Setup Time (Mode 1)	230		190	
26	TdSTB(PD)	$\overline{STROBE}$ ↓ to PORT DATA Stable (Mode 2)		210		180
27	TdSTB(PDr)	$\overline{STROBE}$ ↑ to PORT DATA Float Delay (Mode 2)		180		160
28	TdPD(INT)	PORT DATA Match to $\overline{INT}$ ↓ Delay (Mode 2)		490		430
29	TdSTB(INT)	$\overline{STROBE}$ ↑ to $\overline{INT}$ ↓ Delay		440		350

Note : * Not compatible with NMOS specifications.

ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_{CC}	V_{CC} Supply Voltage with Respect to V_{SS}	– 0.5 to 7	V
V_{IN}	Input Voltage	– 0.5 to V_{CC} + 0.5	V
P_D	Power Dissipation (T_A = 85 °C)	250	mW
T_{SOLDER}	Soldering Temperature (soldering time 10 sec)	260	°C
T_{STG}	Storage Temperature	– 65 to 150	°C
T_{OPR}	Operating Temperature	– 40 to 85	°C

DC CHARACTERISTICS (1)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_{ILC}	Clock Input Low Voltage		– 0.3	–	0.6	V
V_{IHC}	Clock Input High Voltage		V_{CC} – 0.6	–	V_{CC} + 0.3	V
V_{IL}	Input Low Voltage (except CLK)		– 0.5	–	0.8	V
V_{IH}	Input High Voltage (except CLK)		2.2	–	V_{CC}	V
V_{OL}	Output Low Voltage	I_{OL} = 2.0 mA	–	–	0.4	V
V_{OH1}	Output High Voltage (1)	I_{OH} = – 1.6 mA	2.4	–	–	V
V_{OH2}	Output High Voltage (2)	I_{OH} = – 250 µA	V_{CC} – 0.8	–	–	V
I_{LI}	Input Leakage Current	$V_{SS} \le V_{IN} \le V_{CC}$	–	–	± 10	µA
$I_{OL,}$	3-State Output Leakage Current in Float	V_{SS} + 0.4 ≤ $V_{OUT} \le V_{CC}$	–	–	± 10	µA
I_{CC1}	Operating Supply Current : 4 MHz 6 MHz	 V_{CC} = 5 V, CLK = 4 MHz $V_{IH} = V_{CC}$ – 0.2 V, V_{IL} = 0.2 V	 – –	 2 3	 5 8	 mA mA
I_{CC2}	Stand-by Supply Current	V_{CC} = 5 V, CLK = V_{CC} $V_{IH} = V_{CC}$ – 0.2 V V_{IL} = 0.2 V	–	0.5	10	µA
*I_{OHD}	Darlington Drive Current	V_{OH} = 1.5 V, R_{EXT} = 1.1 kΩ	– 1.5	–	– 5.0	mA

Notes : 1. * Applied to Port B only.
2. Typical value is specified at 25 °C.

TEST CONDITIONS

T_A = – 40 °C to + 85 °C
V_{CC} = 5V ± 10%
V_{SS} = 0V

AC TEST CONDITIONS

- Inputs except CLK (clock) are driven at 2.4V for a logic "1" and 0.4V for a logic "0". Clock input is driven at V_{CC} – 0.6V for a logic "1" and 0.6V for a logic "0".
- Timing measurements are made at 2.2V for a logic "1" and 0.8V for a logic "0".

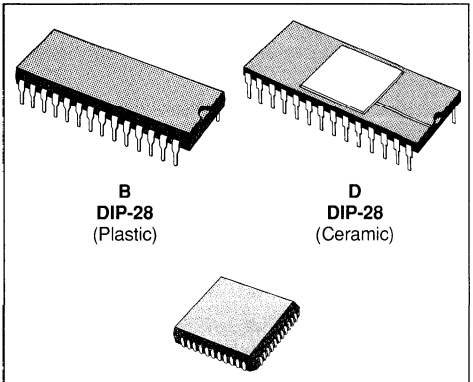
All AC parameters assume a load capacitance of 100pF.

ORDERING INFORMATION

Type	Package	Temp.	Clock	Description
Z84C20AB6	DIP-40 (plastic)	−40/+ 85°C	4 MHz	Z80C Parallel I/O Unit CMOS Version
Z84C20AD6	DIP-40 (ceramic)	−40/+ 85°C		
Z84C20AD2	DIP-40 (ceramic)	−55/+125°C		
Z84C20AC6	PLCC44 (plastic chip-carrier)	−40/+ 85°C		
Z84C20BB6	DIP-40 (plastic)	−40/+ 85°C	6 MHz	
Z84C20BD6	DIP-40 (ceramic)	−40/+ 85°C		
Z84C20BD2	DIP-40 (ceramic)	−55/+125°C		
Z84C20BC6	PLCC44 (plastic chip-carrier)	−40/+ 85°C		

Z80C CTC CMOS VERSION

- FOUR INDEPENDENTLY PROGRAMMABLE COUNTER/TIMER CHANNELS, EACH WITH A READABLE DOWNCOUNTER AND A SELECTABLE 16 OR 256 PRESCALER. DOWNCOUNTERS ARE RELOADED AUTOMATICALLY AT ZERO COUNT
- THREE CHANNELS HAVE ZERO COUNT/TIMEOUT OUTPUTS CAPABLE OF DRIVING DARLINGTON TRANSISTORS
- SELECTABLE POSITIVE OR NEGATIVE TRIGGER INITIATES TIMER OPERATION
- STANDARD Z80C FAMILY DAISY-CHAIN INTERRUPT STRUCTURE PROVIDES FULLY VECTORED, PRIORITIZED INTERRUPTS WITHOUT EXTERNAL LOGIC. THE CTC MAY ALSO BE USED AS AN INTERRUPT CONTROLLER
- INTERFACE DIRECTLY TO THE Z80C CPU OR-FOR BAUD RATE GENERATION - TO THE Z80C SIO
- SINGLE 5 V ± 10 % POWER SUPPLY
- LOW POWER CONSUMPTION :
  - 3 mA TYP. AT 4 MHz
  - 4 mA TYP. AT 6 MHz
  - LESS THAN 10 µA IN POWER DOWN MODE
- EXTENDED OPERATING TEMPERATURE :
  - 40 °C TO + 85 °C



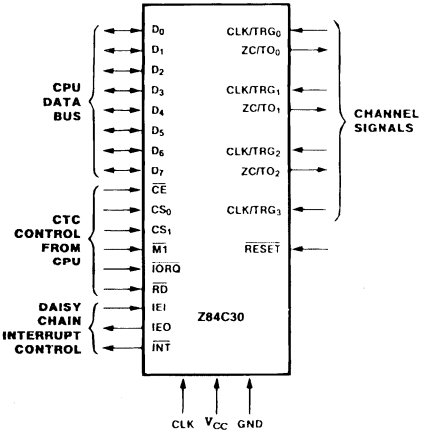
B
DIP-28
(Plastic)

D
DIP-28
(Ceramic)

C
PLCC44
(Plastic)

(Ordering Information at the end of the datasheet)

LOGIC FUNCTIONS



DESCRIPTION

The Z80C CTC four-channel counter/timer can be programmed by system software for a broad range of counting and timing applications. The four independently programmable channels of the CTC satisfy common microcomputer system requirements for event counting, interrupt and interval timing, and general clock rate generation.

System design is simplified because the CTC connects directly to both the CPU and the SIO with no additional logic. In larger systems, address decoders and buffers may be required.

Programming the CTC is straightforward : each channel is programmed with two bytes ; a third is necessary when interrupts are enabled. Once started, the CTC counts down, reloads its time constant automatically, and resumes counting. Software timing loops are completely eliminated.

Interrupt processing is simplified because only one vector need be specified ; the CTC internally generates a unique vector for each channel.

The Z80C CTC requires a single + 5 V power supply and the standard Z80C single-phase system clock. It is fabricated with n-channel silicon-gate depletion-load technology, and packaged in a 28-pin plastic or ceramic DIP.

Figure 1 : Dual in Line Pin Configuration.

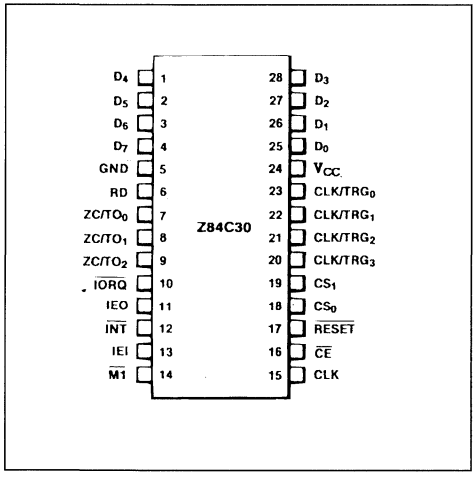
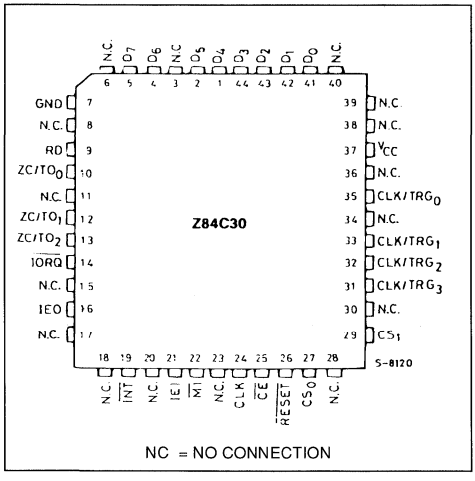


Figure 2 : Chip Carrier Pin Configuration.



NC = NO CONNECTION

FUNCTIONAL DESCRIPTION

The Z80C CTC has four independent counter/timer channels. Each channel is individually programmed with two words : a control word and a time-constant word. The control word selects the operating mode (counter or timer), enables or disables the channel interrupt, and selects certain other operating parameters. If the timing mode is selected, the control word also sets a prescaler, which divides the system clock by either 16 or 256. The time-constant word is a value from 1 to 256.

During operation, the individual counter channel counts down from the preset time constant value. In counter mode operation the counter decrements on each of the CLK/TRG input pulses until zero count is reached. Each decrement is synchronized by the system clock. For counts greater than 256, more than one counter can be cascaded. At zero count, the down-counter is automatically reset with the time constant value.

The timer mode determines time interval as small as 4 µs (Z80CA) without additional logic or software timing loops. Time intervals are generated by dividing the system clock with a prescaler that decrements a preset down-counter.

Thus, the time interval is an integral multiple of the clock period, the prescaler value (16 or 256) and the time constant that is preset in the down-counter. A timer is triggered automatically when its time constant value is programmed, or by an external CLK/TRG input.

Three channels have two outputs that occur at zero count. The first output is a zero-count/timeout pulse at the ZC/TO output. The fourth channel (Channel 3) does not have a ZC/TO output ; interrupt request is the only output available from Channel 3.

The second output is Interrupt Request ($\overline{INT}$), which occurs if the channel has its interrupt enabled during programming. When CPU acknowledges Interrupt Request, the CTC places an interrupt vector on the data bus.

The four channels of the CTC are fully prioritized and fit into four contiguous slots in a standard Z80C daisy-chain interrupt structure. Channel 0 is the highest priority and Channel 3 the lowest. Interrupts can be individually enabled (or disabled) for each of the four channels.

ARCHITECTURE

The CTC has four major elements, as shown in figure 3.

- CPU bus I/O
- Channel control logic
- Interrupt logic
- Counter/timer circuits

CPU BUS I/O

The CPU bus I/O circuit decodes the address inputs, and interfaces the CPU data and control signals to the CTC for distribution on the internal bus.

INTERNAL CONTROL LOGIC

The CTC internal control logic controls overall chip operating functions such as the chip enable, reset, and read/write logic.

INTERRUPT LOGIC

The interrupt control logic ensures that the CTC interrupts interface properly with the Z80C CPU interrupt system. The logic controls the interrupt priority of the CTC as a function of the IEI signal. If IEI is High, the CTC has priority. During interrupt processing, the interrupt logic holds IEO Low, which inhibits the interrupt operation on Lower priority devices. If the IEI input goes Low, priority is relinquished and the interrupt logic drives IEO Low.

If a channel is programmed to request an interrupt, the interrupt logic drives IEO Low at the zero count, and generates an INT signal to the CPU. When the CPU responds with interrupt acknowledge (M1 and IORQ), then the interrupt logic arbitrates the CTC internal priorities, and the interrupt control logic places a unique interrupt vector on the data bus.

If an interrupt is pending, the interrupt logic holds IEO Low. When the CPU issues a Return From Interrupt (RETI) instruction, each peripheral device decodes the first byte (ED_{16}). If the device has a pending interrupt, it raises IEO (High) for one M1 cycle. This ensures that all lower priority devices can decode the entire RETI instruction and reset properly.

COUNTER/TIMER CIRCUITS

The CTC has four independent counter/timer circuits, each containing the logic shown in figure 4.

CHANNEL CONTROL LOGIC

The channel control logic receives the 8-bit channel control word when the counter/timer channel is programmed. The channel control logic decodes the control word and sets the following operating conditions :

- Interrupt enable (or disable)
- Operating mode (timer or counter)
- Timer mode prescaler factor (16 or 256)
- Active slope for CLK/TRG input
- Timer mode trigger (automatic or CLK/TRG input)
- Time constant data word to follow
- Software reset

TIME CONSTANT REGISTER

When the counter/timer channel is programmed, the time constant register receives and stores an 8-bit time constant value, which can be anywhere from 1 to 256 (0 = 256). This constant is automatically

Figure 3 : Functional Block Diagram.

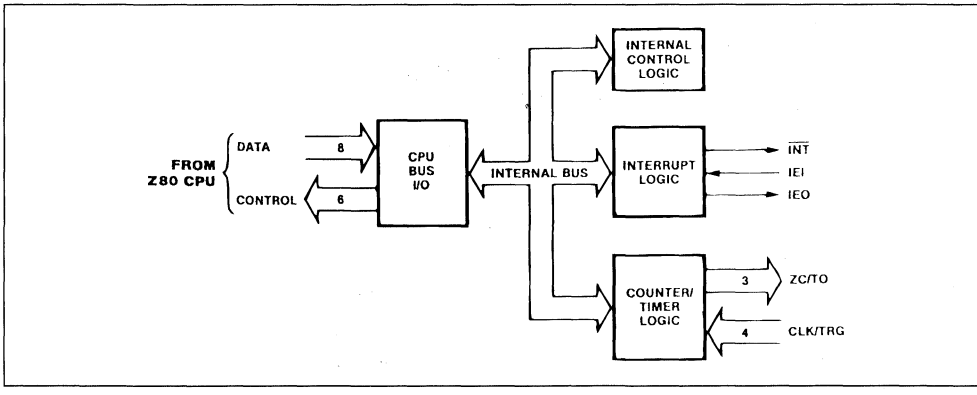
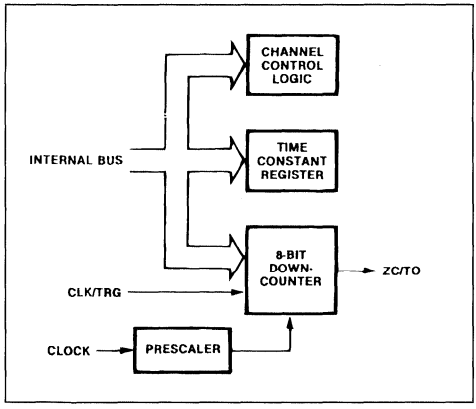


Figure 4 : Counter/ Timer Block Diagram.



loaded into the down-counter when the counter/time channel is initialized, and subsequently after each zero count.

PRESCALER

The prescaler, which is used only in timer mode, divides the system clock frequency by a factor of either 16 or 256. The prescaler output clocks the down-counter during timer operation. The effect of the prescaler on the down-counter is a multiplication of the system clock period by 16 or 256. The prescaler factor is programmed by bit 5 of the channel control word.

DOWN COUNTER

Prior to each count cycle, the down-counter is loaded with the time constant register contents. The counter is then decremented one of two ways, depending on operating mode :

- By the prescaler output (timer mode)
- By the trigger pulses into the CLK/TRG input (counter mode)

Without disturbing the down-count, the CPU can read the count remaining at any time by performing an I/O read operation at the port address assigned to the CTC channel. When the down-counter reaches the zero count, the ZC/TO output generates a positive-going pulse. When the interrupt is enabled, zero count also triggers an interrupt request signal (INT) from the interrupt logic.

PROGRAMMING

Each Z80C CTC channel must be programmed prior to operation.

Programming consists of writing two words to the I/O port that corresponds to the desired channel. The first word is a control word that selects the operating mode and other parameters ; the second word is a time constant, which is a binary data word with a value from 1 to 256. A time constant word must be preceded by a channel control word.

After initialization, channels may be reprogrammed at any time. If updated control and time constant words are written to a channel during the count operation, the count continues to zero before the news time constant is loaded into the counter.

If the interrupt on any CTC channel is enabled, the programming procedure should also include an interrupt vector. Only one vector is required for all four channels, because the interrupt logic automatically modifies the vector for the channel requesting service.

A control word is identified by a 1 in bit 0. A 1 in bit 2 indicates a time constant word is to follow. Interrupt vectors are always addressed to Channel 0, and identified by a 0 in bit 0.

ADDRESSING

During programming, channels are addressed with the channel select pins CS_1 and CS_2. A 2-bit binary code selects the appropriate channel as shown in the following table

Channel	CS_1	CS_0
0	0	0
1	0	1
2	1	0
3	1	1

RESET

The CTC has both hardware and software resets. The hardware reset terminates all down-counts and disables all CTC interrupts by resetting the interrupt bits in the control registers. In addition, the ZO/TO and Interrupt outputs go inactive, IEO reflects IEI, and D_0-D_7 go to the high-impedance state. All channels must be completely reprogrammed after a hardware reset.

The software reset is controlled by bit 1 in the channel control word. When a channel receives a soft-

ware reset, it stops counting. When a software reset is used, the other bits in the control word also change the contents of the channel control register. After a software reset a new time constant word must be written to the same channel.

If the channel control word has both bits D_1 and D_2 set to 1, the addressed channel stops operating, pending a new time constant word. The channel is ready to resume after the new constant is programmed. In timer mode, if $D_3 = 0$, operation is triggered automatically when the time constant word is loaded.

CHANNEL CONTROL WORD PROGRAMMING

The channel control word is shown in figure 5. It sets the modes and parameters described below.

Interrupt Enable. D_7 enables the interrupt, so that an interrupt output ($\overline{INT}$) is generated at zero count. Interrupts may be programmed in either mode and may be enabled or disabled at any time.

OPERATING MODE

D_6 selects either timer or counter mode.

Prescaler factor (Timer Mode Only). D_5 selects factor–either 16 or 256.

Trigger slope. D_4 selects the active edge or slope of the CLK/TRG input pulses. Note that reprogramming the CLK/TRG slope during operation is equivalent to issuing an active edge. If the trigger slope is changed by a control word update while a channel is pending operation in timer mode, the result is the same as a CLK/TRG pulse and the timer starts. Similarly, if the channel is in counter mode, the counter decrements.

Trigger mode (timer mode only). D_3 selects the trigger mode for timer operation. When D_3 is reset to 0, the timer is triggered automatically. The time constant word is programmed during an I/O write operation, which takes one machine cycle. At the end of the write operation there is a setup delay of one clock period. The timer starts automatically (decrements) on the rising edge of the second clock pulse (T_2) of the machine cycle following the write operation. Once started, the timer runs continuously. At zero count the timer reloads automatically and continues counting without interruption or delay, until stopped by a reset.

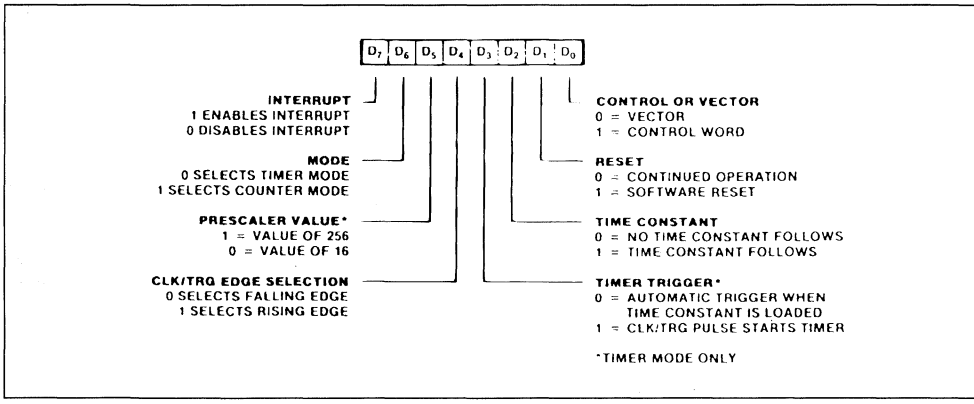
When D_3 is set to 1, the timer is triggered externally through the CLK/TRG input. The time constant word is programmed during an I/O write operation, which takes one machine cycle. The timer is ready for operation on the rising edge of the second clock pulse (T_2) of the following machine cycle. Note that the first timer decrement follows the active edge of the CLK/TRG pulse by a delay time of one clock cycle if a minimum setup time to the rising edge of clock is met. If this minimum is not met, the delay is extended by another clock period. Consequently, for immediate triggering, the CLK/TRG input must precede T_2 by one clock cycle plus its minimum setup time. If the minimum time is not met, the timer will start on the third clock cycle (T_3).

Once started the timer operates continuously, without interruption or delay, until stopped by a reset.

Time constant to follow. A 1 in D_2 indicates that the next word addressed to the selected channel is a time constant data word for the time constant register. The time constant word may be written at any time.

A 0 in D_2 indicates no time constant word is to follow. This is ordinarily used when the channel is already in operation and the new channel control word

Figure 5 : Channel Control Word..



is an update. A channel will not operate without a time constant value. The only way to write a time constant value is to write a control word with D_2 set.

Software reset. Setting D_1 to 1 causes a software reset, which is described in the Reset section.

Control word. Setting D_0 to 1 identifies the word as a control word.

TIME CONSTANT PROGRAMMING

Before a channel can start counting it must receive a time constant word from the CPU. During programming or reprogramming, a channel control word in which bit 2 is set must precede the time constant word to indicate that the next word is a time constant. The time constant word can be any value from 1 to 256 (figure 6). Note that 00_{16} is interpreted as 256.

In timer mode, the time interval is controlled by three factors :

- The system clock period (ϕ)
- The prescaler factor (P), which multiplies the interval by either 16 or 256
- The time constant (T), which is programmed into the time constant register.

Consequently, the time interval is the product of ϕ x P x T. The minimum timer resolution is 16 x ϕ (4 µs with a 4 MHz clock). The maximum timer interval is 256 x ϕ x 256 (16.4 ms with a 4 MHz clock). For longer intervals timers may be cascaded.

Figure 6 : Time Constant Word.

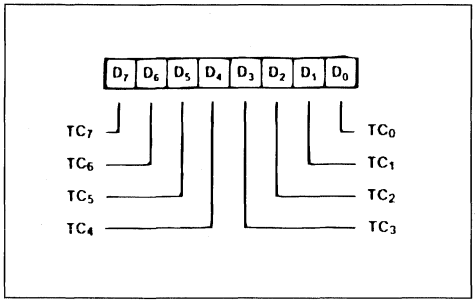
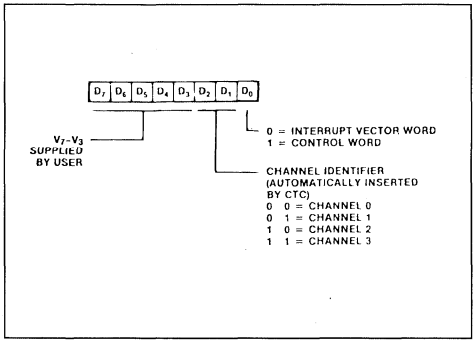


Figure 7 : Interrupt Vector Word.



INTERRUPT VECTOR PROGRAMMING

If the CTC has one or more interrupts enabled, it can supply interrupt vectors to the CPU. To do so, the CTC must be pre-programmed with the most-significant five bits of the interrupt vector. Programming consists of writing a vector word to the I/O port corresponding to the CTC Channel 0. Note that D_0 of the vector word is always zero, to distinguish the vector from a channel control word. D_1 and D_2 are not used in programming the vector word. These bits are supplied by the interrupt logic to identify the channel requesting interrupt service with a unique interrupt vector (figure 7). Channel 0 has the highest priority.

PIN DESCRIPTIONS

$\overline{CE}$. *Chip Enable* (input, active Low). When enabled the CTC accepts control words, interrupt vectors, or time constant data words from the data bus during an I/O write cycle ; or transmits the contents of the down-counter for the CPU during an I/O read cycle. In most applications this signal is decoded from the eight least significant bits of the address bus for any of the four I/Q port addresses that are mapped to the four counter-timer channels.

CLK. *System Clock* (input). Standard singlephase Z80C system clock.

CLK/TRG0-CLK/TRG3. *External Clock/Timer Trigger* (input, user-selectable active High or Low). Four pins corresponding to the four CTC channels. In counter mode, every active edge on this pin decrements the down-counter. In timer mode, an active edge starts the timer.

CS_0-CS_1. *Channel Select* (inputs active High). Two-bit binary address code selects one of the four CTC channels for an I/O write or read (usually connected to A_0 and A_1).

D_0-D_7. *System Data Bus* (bidirectional, 3-state). Transfers all data and commands between the CPU and the CTC.

IEI. *Interrupt Enable In* (input, active High). A high indicates that no other interrupting devices of higher priority in the daisy chain are being serviced by the CPU.

EO. *Interrupt Enable Out* (output, active High). High ınly if IEI is High and the CPU is not servicing an in- errupt from any CTC channel. IEO blocks lower ıriority devices from interrupting while a higher ıriority interrupting device is being serviced.

NT. *Interrupt Request* (output, open drain, active .ow). Low when any CTC channel that has been ırogrammed to enable interrupts has a zero-count :ondition in its down-counter.

ORQ. *Input/Output Request* (input from CPU, ac- ive Low). Used with CE and RD to transfer data and :hannel control words between the CPU and the :TC. During a write cycle IORQ and CE are active ınd RD inactive. The CTC does not receive a spe- :ific write signal ; rather, it internally generates its)wn from the inverse of an active RD signal. In a ead cycle, IORQ, CE and RD are active ; the con- ents of the down-counter are read by the CPU. If ORQ and M1 are both true, the CPU is acknow- edging an interrupt request, and the highest priority nterrupting channel places its interrupt vector on the lata bus.

M1. *Machine Cycle One* (input from CPU, active .ow). When M1 and IORQ are active, the CPU is ıcknowledging an interrupt. The CTC then places ın interrupt vector on the data bus if it has highest ıriority, and if a channel has requested an interrupt INT).

RD. *Read Cycle Status* (input, active Low). Used in :onjunction with IORQ and CE to transfer data and :hannel control words between the CPU and the :TC.

RESET. *Reset* (input active Low). Terminates all lown-counts and disables all interrupts by resetting he interrupt bits in all control registers ; the ZC/TO ınd the Interrupt outputs go inactive ; IEO reflects EI ; D0-D7 go to the high-impedance state.

ZC/TO$_0$-ZC/TO$_2$. *Zero Count/Timeout* (output, ac- ive High). Three ZC/TO pins corresponding to CTC :hannels 2 through 0 (Channel 3 has no ZC/TO pin). n both counter and timer modes the output is an ac- ive High pulse when the down-counter decrements o zero.

TIMING

READ CYCLE TIMING

Figure 9 shows read cycle timing. This cycle reads :he contents of a down-counter without disturbing :he count. During clock cycle T_2, the CPU initiates a ead cycle by driving the following inputs Low : RD, ORQ, and CE. A 2-bit binary code at inputs CS_1 ınd CS_0 selects the channel to be read. M1 must ɔe High to distinguish this cycle from an interrupt ac- ‹nowledge. No additional wait states are allowed.

WRITE CYCLE TIMING

Figure 10 shows write cycle timing for loading con- trol, time constant or vector words.

The CTC does not have a write signal input, so it generates one internally when the read (RD) input is High during T_1. During T_2 IORQ and CE inputs are Low. M1 must be High to distinguish a write cycle from an interrupt acknowledge. A 2-bit binary code at inputs CS_1 and CS_0 selects the channel to be addressed, and the word being written is placed on the data bus. The data word is latched into the appro- priate register with the rising edge of clock cycle T_3.

TIMER OPERATION

In the timer mode, a CLK/TRG pulse input starts the timer (figure 11) on the second succeeding rising

Figure 8 : A Typical Z80C Environment.

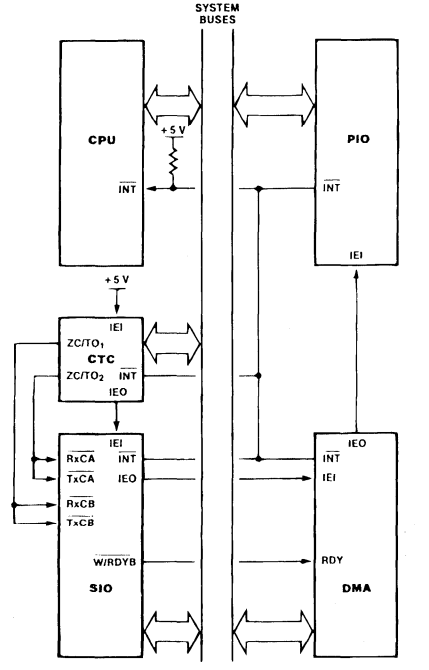


Figure 9 : Read Cycle Timing.

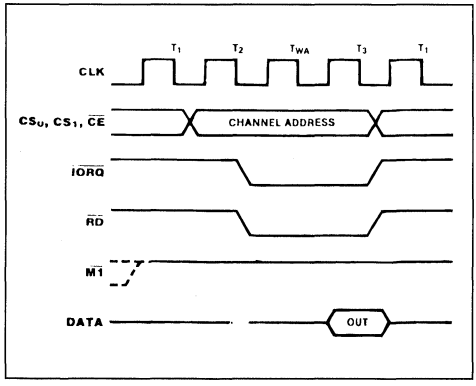
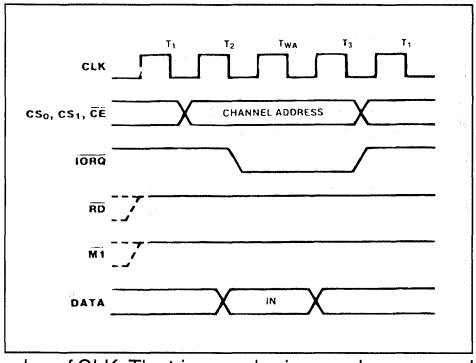


Figure 10 : Write Cycle Timing.



edge of CLK. The trigger pulse is asynchronous and it must have a minimum width. A minimum lead time (210 ns) is required between the active edge of the CLK/TRG and the next rising edge of CLK to enable the prescaler on the following clock edge. If the CLK/TRG edge occurs closer than this, the initiation of the timer function is delayed one clock cycle. This corresponds to the startup timing discussed in the programming section. The timer can also be started automatically if so programmed by the channel control word.

COUNTER OPERATION.

In the counter mode, the CLK/TRG pulse input decrements the downcounter. The trigger is asynchronous, but the count is synchronized with CLK. For the decrement to occur on the next rising edge of CLK, the trigger edge must precede CLK by a minimum lead time as shown in figure 12. If the lead time is less than specified, the count is delayed by one clock cycle. The trigger pulse must have a minimum width, and the trigger period must be at least twice the clock period.

The ZC/TO output occurs immediately after zero count, and follows the rising CLK edge.

Figure 11 : Timer Mode Timing.

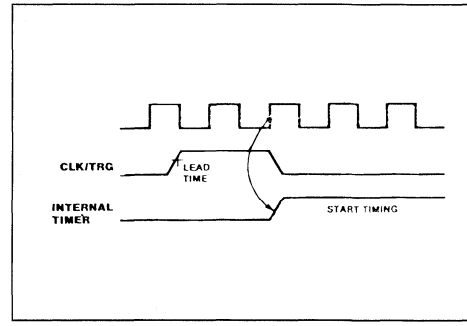
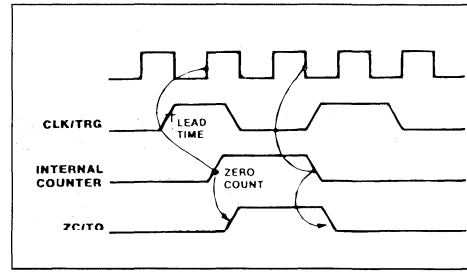


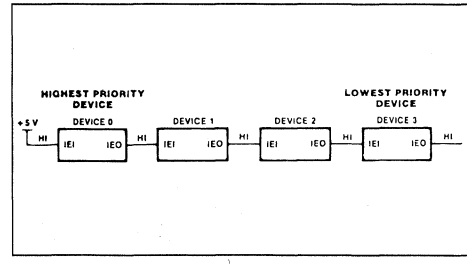
Figure 12 : Counter Mode Timing.



INTERRUPT OPERATION

The CTC follows the Z80C system interrupt protocol for nested priority interrupts and return from interrupt, wherein the interrupt priority of a peripheral is determined by its location in a daisy chain. Two lines–IEI and IEO–in the CTC connect it to the system daisy chain. The device closest to the + 5 V supply has the highest priority (figure 13). For additional information on the Z80C interrupt structure, refer *to the Z80 CPU Technical Manual.*

Figure 13 : Daisy-chain Interrupt Priorities.



INTERRUPT OPERATION

Within the CTC, interrupt priority is predetermined by channel number : Channel 0 has the highest priority, and Channel 3 the lowest. If a device or channel is being serviced with an interrupt routine, it cannot be interrupted by a device or channel with lower priority until service is complete. Higher priority devices or channels may interrupt the servicing of lower priority devices or channels.

A CTC channel may be programmed to request an interrupt every time its down-counter reaches zero. Note that the CPU must be programmed for interrupt mode 2. Some time after the interrupt request, the CPU sends an interrupt acknowledge. The CTC interrupt control logic determines the highest priority channel that is requesting an interrupt. Then, if the CTC IEI input is High (indicating that it has priority within the system daisy chain) it places an 8-bit interrupt vector on the system data bus. The high-order five bits of this vector were written to the CTC during the programming process ; the next two bits are provided by the CTC interrupt control logic as a binary code that identifies the highest priority channel requesting an interrupt ; the low-order bit is always zero.

INTERRUPT ACKNOWLEDGE TIMING

Figure 14 shows interrupt acknowledge timing. After an interrupt request, the CPU sends an interrupt acknowledge ($\overline{M1}$ and $\overline{IORQ}$). All channels are inhibited from changing their interrupt request status when $\overline{M1}$ is active–about two clock cycles earlier than $\overline{IORQ}$. $\overline{RD}$ is High to distinguish this cycle from an instruction fetch.

The CTC interrupt logic determines the highest priority channel requesting an interrupt. If the CTC interrupt enable input (IEI) is High, the highest priority interrupting channel within the CTC places its interrupt vector on the data bus when $\overline{IORQ}$ goes Low. Two wait states (T_{WA}) are automatically inserted at this time to allow the daisy chain to stabilize. Additional wait states may be added.

RETURN FROM INTERRUPT TIMING

At the end of an interrupt service routine the RETI (Return From Interrupt) instruction initializes the daisy chain enable lines for proper control of nested priority interrupt handling. The CTC decodes the 2-byte RETI code internally and determines whether it is intended for a channel being serviced. Figure 15 shows RETI timing.

If several Z80C peripherals are in the daisy chain, IEI settles active (High) on the chip currently being serviced when the opcode ED_{16} is decoded. If the following opcode is $4D_{16}$, the peripheral being serviced is released and its IEO becomes active. Additional wait states are allowed.

Figure 14 : Interrupt Acknowledge Timing.

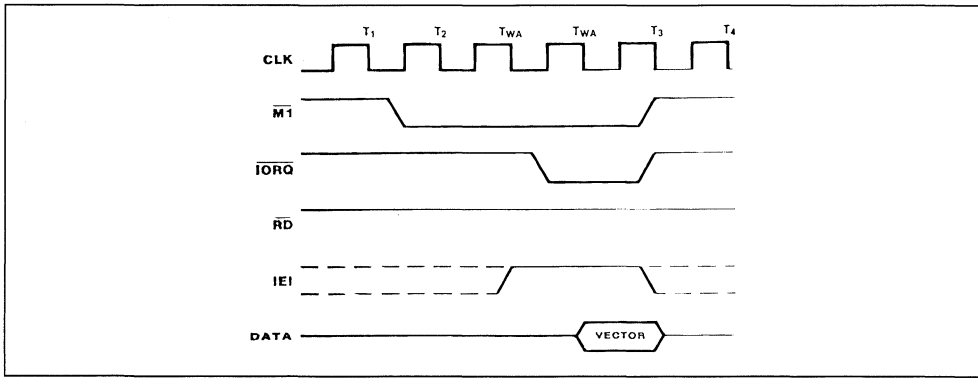
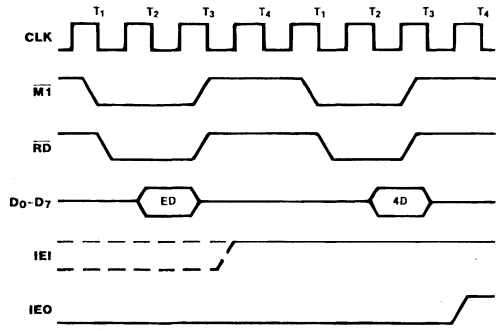
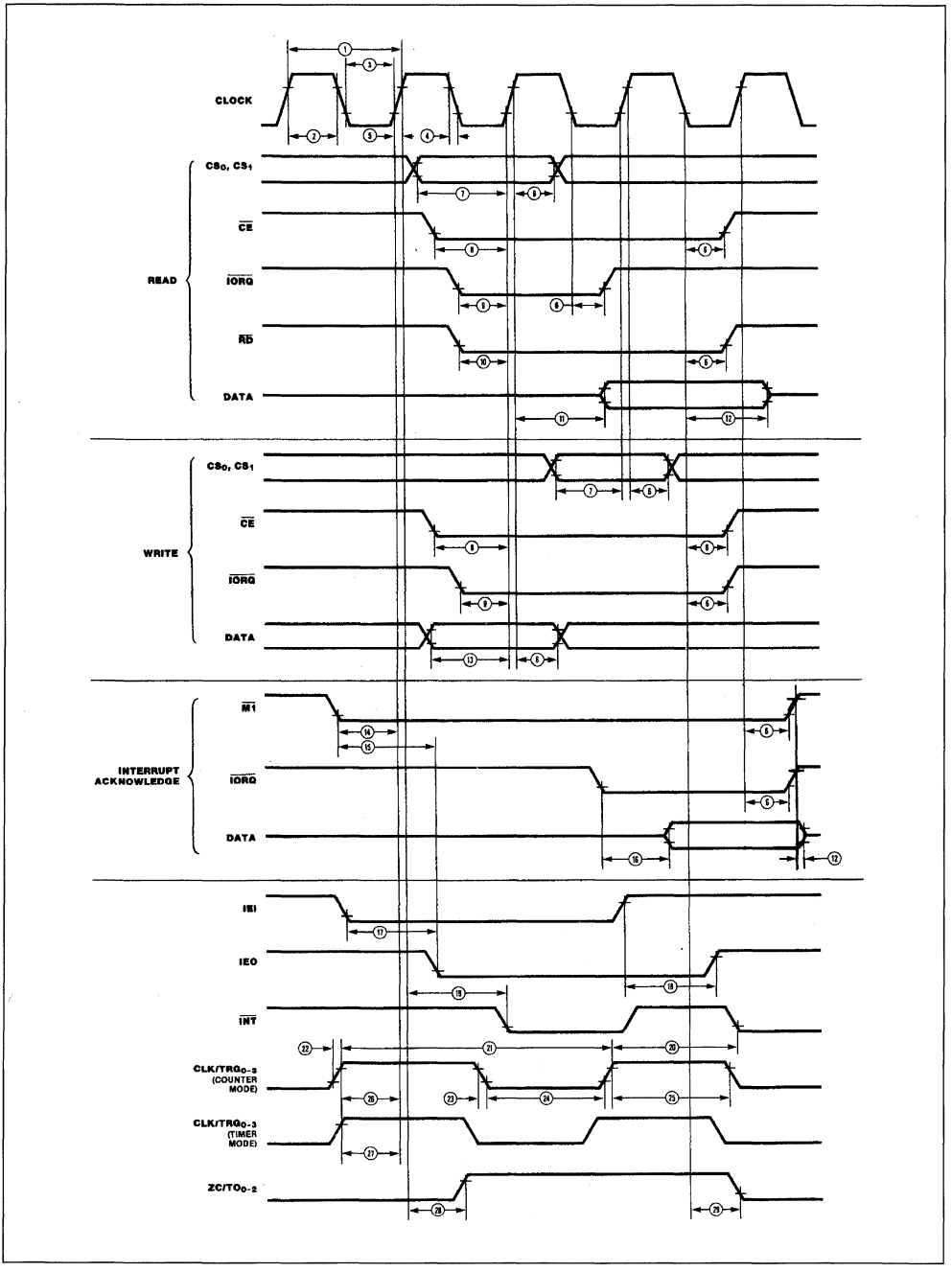


Figure 15 : Return From Interrupt Timing.



AC CHARACTERISTICS



AC CHARACTERISTICS (continued)

N°	Symbol	Parameter	Z84C30A		Z84C30B	
			Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
1	TcC	Clock Cycle Time	250		165	
2	TwCh	Clock Width (high)	105		65	
3	TwCl	Clock Width (low)	105		65	
4	TfC	Clock Fall Time		30		20
5	TrC	Clock Rise Time		30		20
6	Th	All Hold Times	0		0	
7	TsCS(C)	CS to Clock ↑ Setup Time	160		100	
8	TsCE(C)	$\overline{CE}$, to Clock ↑ Setup Time	150		100	
9	TsIO(C)	$\overline{IORQ}$ ↓ to Clock ↑ Setup Time	115		100	
10	TsRD(C)	$\overline{RD}$, ↓ to Clock ↑ Setup Time	115		70	
11	TdC(DO)	Clock ↑ to Data Out Delay		200		130
12	TdC(DOz)	$\overline{IORQ}$, $\overline{RD}$ ↑ to Data Float		110		90
13	TsDI(C)	Data in to Clock ↑ Setup Time	50		40	
14	TsMI(C)	$\overline{MI}$ to Clock ↑ Setup Time	90		70	
15	TdMI(IEO)	$\overline{MI}$ ↓ to IEO ↓ Delay (interrupt immediately preceding $\overline{MI}$)		190		130
16	TdIO(DOI)	$\overline{IORQ}$ ↓ to Data Out Delay (INTA cycle)		160		110
17	TdIEI(IEOf)	IEI ↓ to IEO ↓ Delay		130		100
18	TdIEI(IEOr)	IEI ↑ to IEO ↑ Delay (after ED decode)		160		110
19	TdC(INT)	Clock ↑ to $\overline{INT}$ ↓ Delay		(1)TcC+140		(1)TcC+120
20	TdCLK(INT)	CLK/TRG ↑ to $\overline{INT}$ ↓ tsCTR(C) satisfied tsCTR(C) not satisfied		(2) TcC+160 2TcC+370		(2) TcC+160 2TcC+370
21	TcCTR	CLK/TRG Cycle Time	(2) 2TcC			(2) 40
22	TrCTR	CLK/TRG Rise Time		50		40
23	TfCTR	CLK/TRG Fall Time		50		40
24	TwCTRl	CLK/TRG Width (low)	200		120	
25	TwCTRh	CLK/TRG Width (high)	200		120	
26	TsCTR(Cs)	CLK/TRG ↑ to Clock ↑ Setup Time for Immediate Count	(2) 210		(2) 150	
27	TsCTR(Cs)	CLK/TRG ↑ to Clock ↑ Setup Time for enabling of Prescaler on following Clock ↑	(1) 210		(1) 150	
28	TdC(ZC/TOr)	Clock ↑ to ZC/TO ↑ Delay		190		140
29	TdC(ZC/TOf)	Clock ↓ to ZC/TO ↓ Delay		190		140

Notes : 1. Timer mode.
2. Counter mode.

ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_{CC}	V_{CC} Supply Voltage with Respect to V_{SS}	– 0.5 to 7	V
V_{IN}	Input Voltage	– 0.5 to V_{CC} + 0.5	V
P_D	Power Dissipation (T_A = 85 °C)	250	mW
T_{SOLDER}	Soldering Temperature (soldering time 10 sec)	260	°C
T_{stg}	Storage Temperature	– 65 to 150	°C
T_{opr}	Operating Temperature	– 40 to 85	°C

DC CHARACTERISTICS (1)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_{ILC}	Clock Input Low Voltage		– 0.3	–	0.6	V
V_{IHC}	Clock Input High Voltage		V_{CC} – 0.6	–	V_{CC} + 0.3	V
V_{IL}	Input Low Voltage (except CLK)		– 0.5	–	0.8	V
V_{IH}	Input High Voltage (except CLK)		2.2	–	V_{CC}	V
V_{OL}	Output Low Voltage	I_{OL} = 2.0 mA	–	–	0.4	V
V_{OH1}	Output High Voltage (1)	I_{OH} = – 1.6 mA	2.4	–	–	V
V_{OH2}	Output High Voltage (2)	I_{OH} = – 250 µA	V_{CC} – 0.8	–	–	V
I_{LI}	Input Leakage Current	$V_{SS} \le V_{IN} \le V_{CC}$	–	–	± 10	µA
I_{LO}	3-State Output Leakage Current in Float	V_{SS} + 0.4 ≤ $V_{OUT} \le V_{CC}$	–	–	± 10	µA
I_{CC1}	Power Supply Current 4 MHz 6 MHz	 V_{CC} = 5 V, CLK = 4 MHz $V_{IH} = V_{CC}$ – 0.2 V, V_{IL} = 0.2 V	 – –	 2 4	 5 7	 mA mA
I_{CC2}	Stand-by Supply Current	V_{CC} = 5 V, CLK = V_{CC} $V_{IH} = V_{CC}$ – 0.2 V V_{IL} = 0.2 V	–	0.5	10	µA
I_{OHD}	Darlington Drive Current (1)	V_{OH} = 1.5 V, R_{EXT} = 1.1 kΩ	– 1.5	–	– 5.0	mA

Note : 1. Applied to $ZC/T0_0$, $ZC/T0_1$ and $ZC/T0_2$.

TEST CONDITIONS

T_A = – 40 °C to + 85 °C

V_{CC} = 5 V ± 10 %

V_{SS} = 0 V.

AC TEST CONDITIONS.

- Inputs except CLK (clock) are driven at 2.4 V for a logic "1" and 0.4 V for a logic "0". Clock input is driven at V_{CC}–0.6 V for a logic "1" and 0.6 V for a logic "0".
- Timing measurements are made at 2.2 V for a logic "1" and 0.8 V for a logic "0".

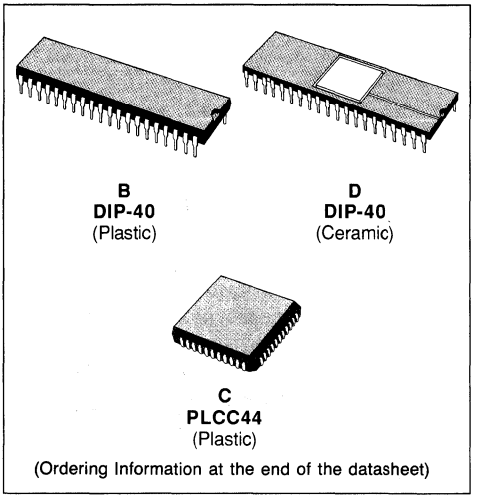
All AC parameters assume a load capacitance of 100 pF.

ORDERING INFORMATION

Type	Package	Temp.	Clock	Description
Z84C30AB6	DIP-28 (plastic)	−40/+ 85°C	4 MHz	Z80C Counter Timer Control CMOS Version
Z84C30AD6	DIP-28 (ceramic)	−40/+ 85°C		
Z84C30AD2	DIP-28 (ceramic)	−55/+125°C		
Z84C30AC6	PLCC44 (plastic chip-carrier)	−40/+ 85°C		
Z84C30BB6	DIP-28 (plastic)	−40/+ 85°C	6 MHz	
Z84C30BD6	DIP-28 (ceramic)	−40/+ 85°C		
Z84C30BD2	DIP-28 (ceramic)	−55/+125°C		
Z84C30BC6	PLCC44 (plastic chip-carrier)	−40/+ 85°C		

Z80C SIO CMOS VERSION

- TWO INDEPENDENT FULL-DUPLEX CHANNELS, WITH SEPARATE CONTROL AND STATUS LINES FOR MODEMS OR OTHER DEVICES
- DATA TRANSFER RATE UP TO 800K BIT/SECOND
- ASYNCHRONOUS PROTOCOLS : EVERYTHING NECESSARY FOR COMPLETE MESSAGES IN 5, 6, 7 OR 8 BITS/CHARACTER. INCLUDES VARIABLE STOP BITS AND SEVERAL CLOCK-RATE MULTIPLIERS ; BREAK GENERATION AND DETECTION ; PARITY ; OVERRUN AND FRAMING ERROR DETECTION
- SYNCHRONOUS PROTOCOLS : EVERYTHING NECESSARY FOR COMPLETE BIT- OR BYTE-ORIENTED MESSAGES IN 5, 6, 7 OR 8 BITS/CHARACTER, INCLUDING IBM BISYNC, SDLC, HDLC, CCITT-X.25 AND OTHERS. AUTOMATIC CRC GENERATION/CHECKING SYNC CHARACTER AND ZERO INSERTION/DELETION, ABORT GENERATION/DETECTION AND FLAG INSERTION
- RECEIVER DATA REGISTERS QUADRUPLY BUFFERED, TRANSMITTER REGISTERS DOUBLY BUFFERED
- HIGHLY SOPHISTICATED AND FLEXIBLE DAISY-CHAIN INTERRUPT VECTORING FOR INTERRUPTS WITHOUT EXTERNAL LOGIC
- SINGLE 5V ± 10% POWER SUPPLY
- LOW POWER CONSUPTION :
  - 2.5mA TYP. AT 4MHz
  - 4mA TYP. AT 6MHz
  - LESS THAN 10µA IN POWER DOWN MODE
- EXTENDED OPERATING TEMPERATURE – 40°C TO + 85°C



B
DIP-40
(Plastic)

D
DIP-40
(Ceramic)

C
PLCC44
(Plastic)

(Ordering Information at the end of the datasheet)

DESCRIPTION

The Z80C SIO Serial Input/Output Controller is a dual-channel data communication interface with extraordinary versatility and capability. Its basic functions as a serial-to-parallel, parallel-to-serial converter/controller can be programmed by a CPU for a broad range of serial communication applications.

The device supports all common asynchronous and synchronous protocols, byte- or bit-oriented and performs all of the functions traditionally done by UARTs, USARTs and synchronous communication controllers combined, plus additional functions traditionally performed by the CPU, Moreover, it does this on two fully-independent channels, with an exceptionally sophisticated interrupt structure that allows very fast transfers.

Full interfacing is provided for CPU or DMA control. In addition to data communication, the circuit can handle virtually all types of serial I/O with fast (or slow) peripheral devices.

While designed primarily as a member of Z80 family, its versatility makes it well suited to many other CPUs.

PIN DESCRIPTIONS

Figures 1 through 6 illustrate the three pin configurations (bonding options) available in the SIO. The constraints of a 40-pin package make it impossible to bring out the Receive Clock (RxC), Transmitt Clock (TxC), Data Terminal Ready (DTR) and Sync (SYNC) signals for both channels. Therefore, either Channel B lacks a signal or two signals are bonded together in the three bonding options offered :

- Z80C SIO-2 lacks $\overline{SYNCB}$
- Z80C SIO-1 lacks $\overline{DTRB}$
- Z80C SIO-0 as a four signal, but $\overline{TxCB}$ and $\overline{RxCB}$ are bonded together

The first bonding option above (SIO-2) is the preferred version for most applications. The Chip-Carrier package version, having a 44-pin facility, resume the three bonding option configurations. It is named Z84C44 (figure 7). The pin description are as follows :

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 SIO. Address bit A_0 from the CPU is often used for the selection function.

C/$\overline{D}$. *Control Or Data Select* (Input, High Selects Control). This input defines the type of information transfer performed between the CPU and the SIO. A High at this input during a CPU write to the SIO causes the information on the data bus to be interpreted as a command for the channel selected by B/$\overline{A}$. A Low at C/$\overline{D}$ means that the information on the data bus is data. Address bit A_1 is often used for this function.

$\overline{CE}$. *Chip Enable* (Input, Active Low). A Low level at this input enables the SIO 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 SIO uses the standard Z80 System Clock to synchronize internal signals. This is a single-phase clock.

$\overline{CTSA}$, **$\overline{CTSB}$**. *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. The SIO detects pulses on these inputs and interrupts the CPU on both logic level transitions. The Schmitt-trigger buffering does not guarantee a specified noise-level margin.

D_0-D_7. *System Data Bus* (Bidirectional, 3-state). The system data bus transfers data and commands between the CPU and the Z80C SIO. D_0 is the least significant bit.

$\overline{DCDA}$, **$\overline{DCDB}$**. *Data Carrier Detect* (Inputs, Active Low). These pins function as receiver enables if the SIO is programmed for Auto Enables ; otherwise they may be used as general-purpose input pins.

Figure 1 : Z80C SIO-2 Logic Functions.

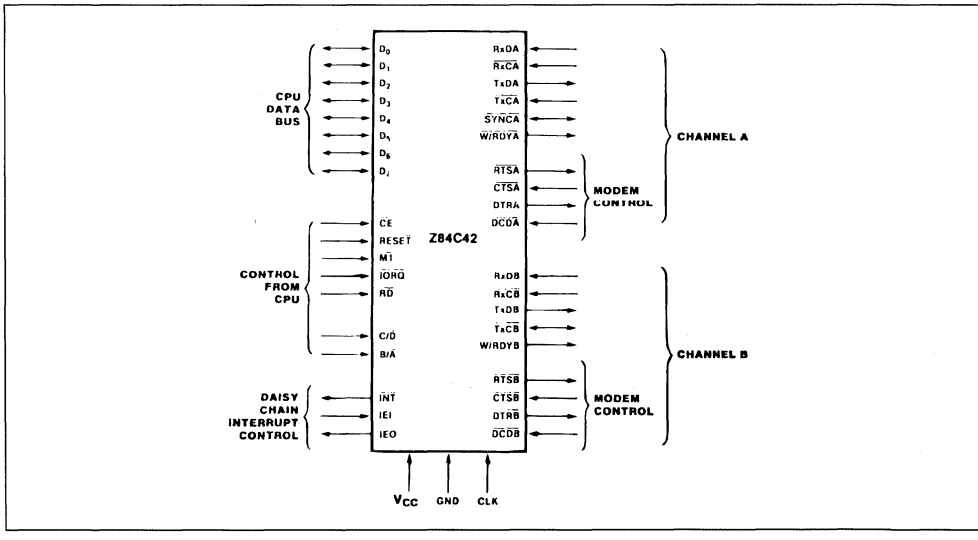
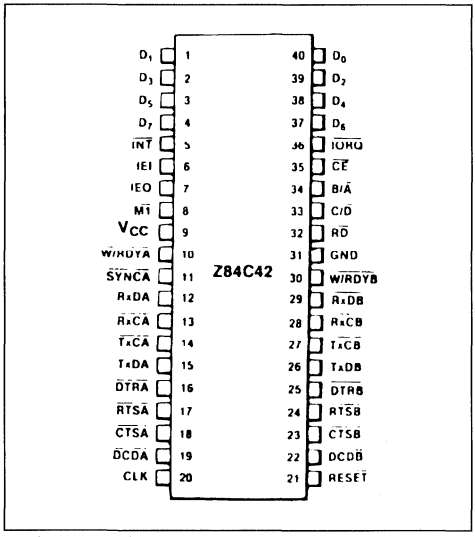


Figure 2 : Z80C SIO-2 Dual in Line Pin Configuration.



Both pins are Schmitt-trigger buffered to accommodate slow-risetime signals. The SIO detects pulses on these pins and interrupts the CPU on both logic level transitions. Schmitt-trigger buffering does not guarantee a specific noise-level margin.

DTRA, **DTRB**. *Data Terminal Ready* (Outputs, Active Low). These outputs follow the state programmed into Z80C SIO. They can also be programmed as general-purpose outputs.

In the Z80C SIO-1 bonding option, DTRB is omitted.

IEI. *Interrupt Enable In* (Input, Active High). This signal 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 SIO. Thus, this signal blocks lower priority devices form interrupting while a higher priority device is being serviced by its CPU interrupt service routine.

INT. *Interrupt Request* (Output, Open Drain, Active Low). When the SIO is requesting an interrupt, it pulls INT Low.

IORQ. *Input/Output Request* (Input from CPU, Active Low). IORQ is used in conjunction with B/A, C/D, CE and RD to transfer commands and data between the CPU and the SIO. When CE, RD and IORQ are all active, the channel selected by B/A transfers data to the CPU (a read operation). When CE and IORQ are active but RD is inactive, the channel selected by B/A is written to by the CPU with either data or

Figure 3 : Z80C SIO-1 Logic Functions.

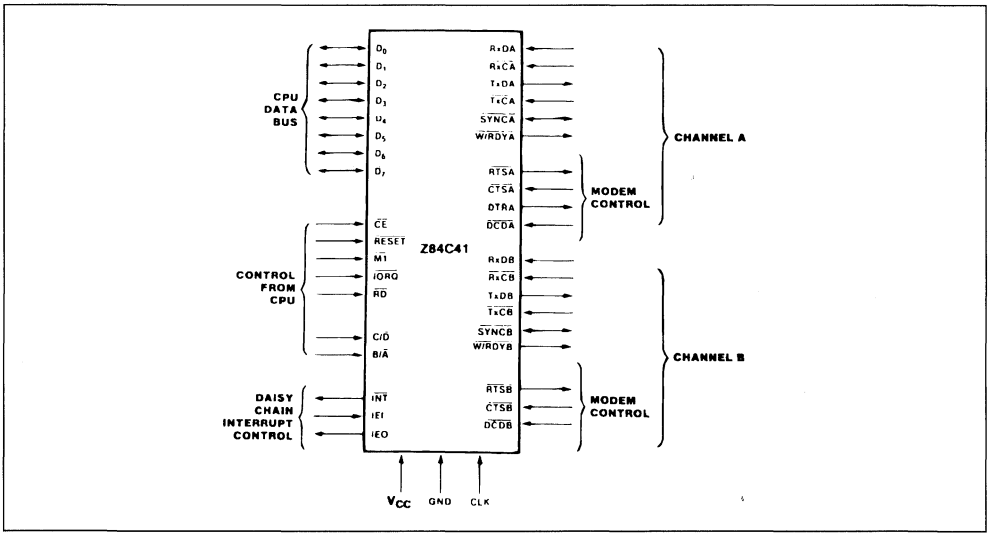
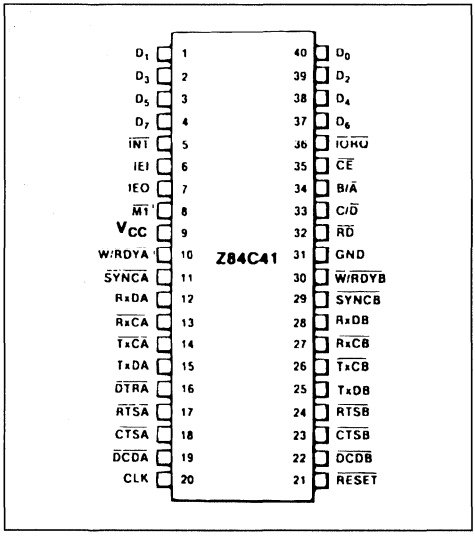


Figure 4 : Z80C SIO-1 Dual in Line Pin Configuration.



control information as specified by C/D. If IORQ and M1 are active simultaneously, the CPU is acknowledging an interrupt and the SIO automatically places its interrupt vector on the CPU data bus if it is the highest priority device requesting an interrupt.

M1. *Machine Cycle* (Input from Z80C CPU, Active Low). When M1 is active and RD is also active, the Z80C CPU is fetching an instruction from memory ; when M1 is active while IORQ is active, the SIO accepts M1 and IORQ as an interrupt acknowledge if the SIO is the highest priority device that has interrupted the Z80C CPU.

RxCA, **RxCB**. *Receiver Clocks* (Inputs). Receive data is sampled on the rising edge of RxC. The Receive Clocks may be 1, 16, 32 or 64 times the data rate in asynchronous modes. These clocks may be driven by the Z80C CTC Counter Timer Circuit for programmable baud rate generation. Both inputs are Schmitt-trigger buffered (no noise level margin is specified).

In the Z80C SIO-0 bonding option, RxCB is bonded together with TxCB.

RD. *Read Cycle Status* (Input from CPU, Active Low). If RD is active, a memory or I/O read operation is in progress. RD is used with B/A, CE and IORQ to transfer data from the SIO to the CPU.

RxDA, **RxDB**. *Receive Data* (Inputs, Active High). Serial data at TTL levels.

RESET. *Reset* (Input, Active Low). A Low RESET disables both receivers and transmitters, forces TxDA and TxDB marking, forces the modem con-

Figure 5 : ZCC SIO-0 Logic Functions.

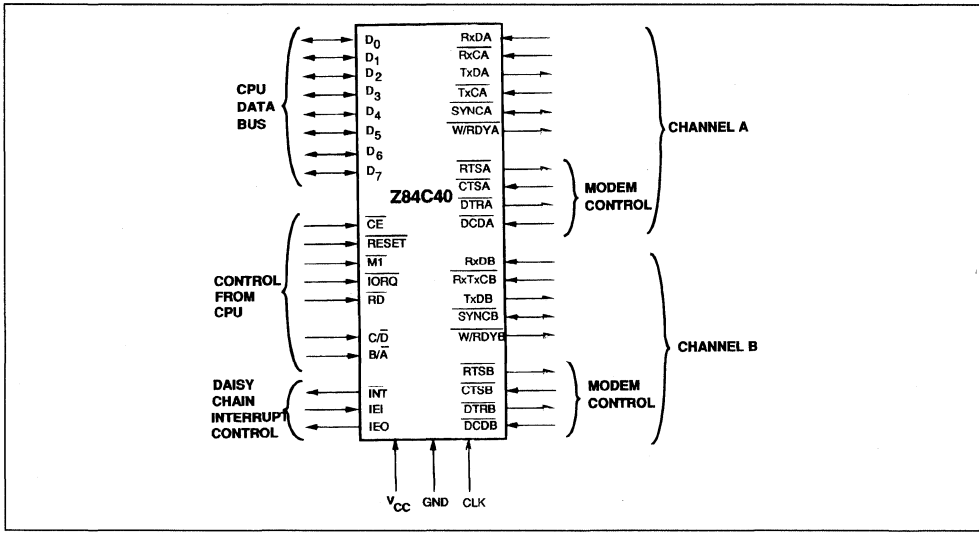


Figure 6 : Z80C SIO-0 Dual in Line Pin Configuration.

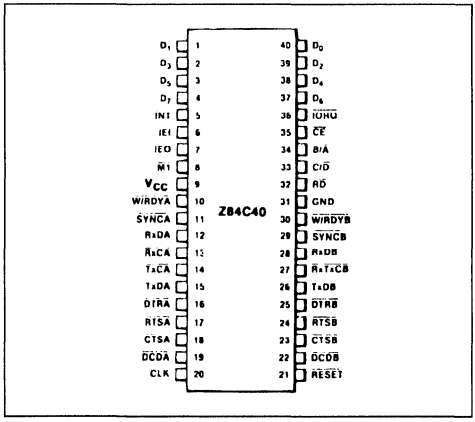
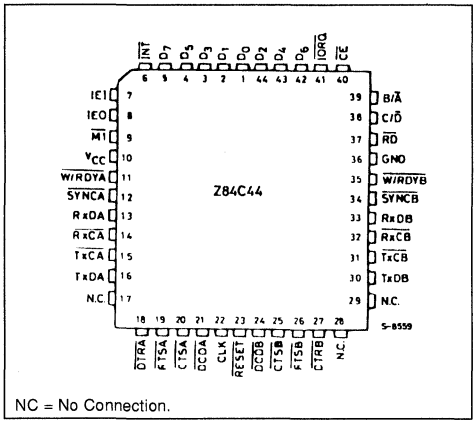


Figure 7 : Chip Carrier Pin Configuration.



NC = No Connection.

trols High and disables all interrupts. The control registers must be rewritten after the SIO is reset and before data is transmitted or received.

RTSA, **RTSB**. *Request To Send* (Outputs, Active Low). When the RTS bit in Write register 5 (figure 14) is set, the RTS output goes Low. When the RTS bit is reset in the Asynchronous mode, the output goes High after the transmitter is empty. In Synchronous modes, the RTS pin strictly follows the state of the RTS bit. Both pins can be used as general-purpose outputs.

SYNCA, **SYNCB**. *Synchronization* (Inputs/Outputs, Active Low). These pins can act either as inputs or outputs. In the asynchronous receive mode, they are inputs similar to CTS and DCD. In this mode, the transitions on these lines affect the state of the Sync/Hunt status bits in Read Register 0 (figure 14), but have no other function. In the External Sync mode, these lines also act as inputs. When external synchronization is achieved, SYNC must be driven Low on the second rising edge of RxC after that rising edge of RxC on which the last bit of the sync character was received. In other words, after the sync pattern is detected, the external logic must wait for two full Receive Clock cycles to activate the SYNC input. Once SYNC is forced Low, it should be kept Low until the CPU informs the external synchronization detect logic that synchronization has been lost or a new message is about to start. Character assembly begins on the rising edge of RxC that immediately precedes the falling edge of SYNC in the External Sync mode.

In the internal synchronization mode (Monosync and Bisync) these pins act as outputs that are active during the part of the receive clock (RxC) cycle in which sync characters are recognized. The sync condition is not latched so these outputs are active each time a sync pattern in recognized, regardless of character boundaries.

In the Z80C SIO-2 bonding option, SYNCB is omitted.

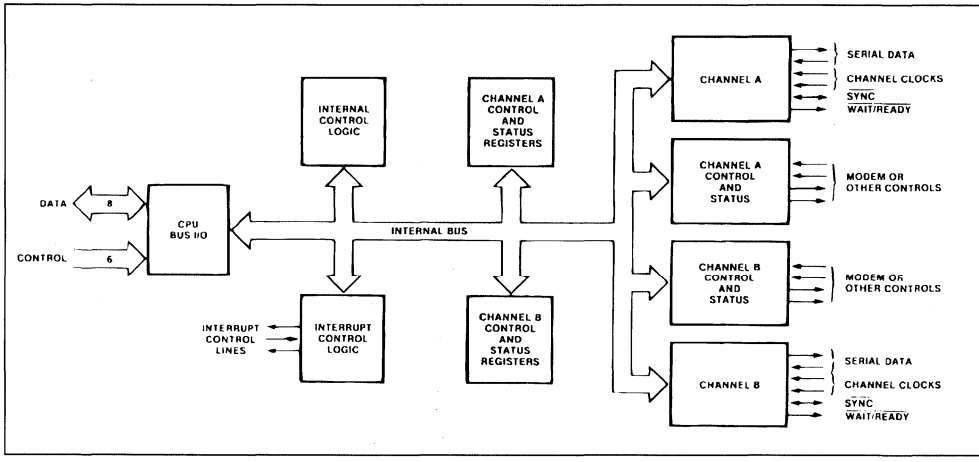
TxCA, **TxCB**. *Transmitter Clocks* (Inputs). In asynchronous modes, the Transmitter Clocks may be 1, 16, 32 or 64 times the data rate ; however, the clock multiplier for the transmitted and the receiver must be the same. The transmit Clock inputs are Schmitt-trigger buffered for relaxed rise- and fall-time requirements (no noise level margin is specified).

Transmitter Clocks may be driven by the Z80C CTC Counter Timer Circuit for programmable baud rate generation. In the Z80C SIO-0 bonding option, TxCB is bonded together with RxCB.

TxDA, **TxDB**. *Transmit Data* (Outputs, Active High). Serial data at TTL levels. TxD changes from the falling edge of TxC.

W/RDYA, **W/RDYB**. *Wait/Ready A, Wait/Ready B* (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 SIO data rate. The reset state is open drain.

Figure 8 : Block Diagram.



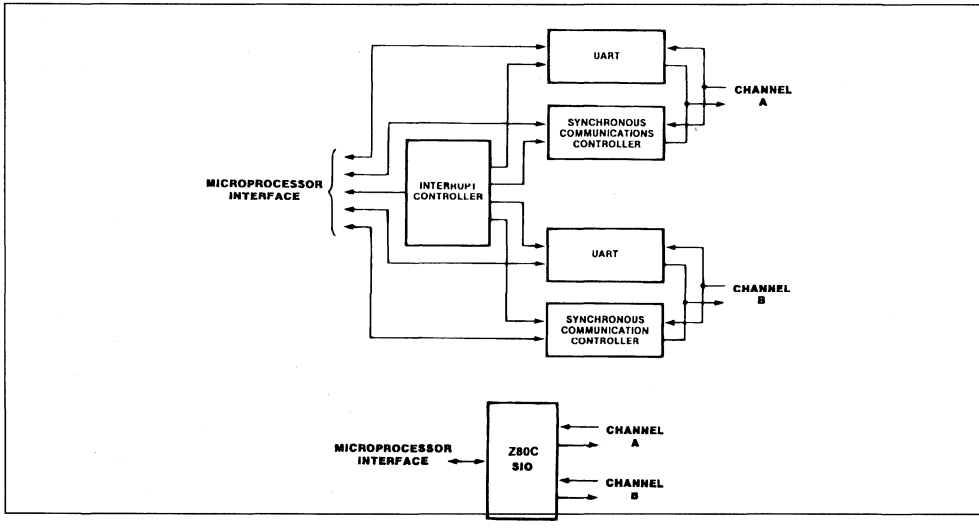
FUNCTIONAL DESCRIPTION

The functional capabilities of the Z80C SIO can be described from two different points of view : as a data communication device, it transmits and receives serial data in a wide variety of data-communication protocols ; as a Z80C family peripheral, it interacts with the Z80C CPU and other peripheral circuits, sharing the data, address and control buses, as well as being a part of the Z80C interrupt structure. As a peripheral to other microprocessors, the SIO offers valuable features such as non-vectored interrupts, polling and simple handshake capability.

Figure 9 illustrates the conventional devices that the SIO replaces.

The first part of the following discussion covers SIO data-communication capabilities ; the second part describes interactions between the CPU and the SIO.

Figure 9 : Conventional Devices Replaced by Z80C SIO.



DATA COMMUNICATION CAPABILITIES

The SIO provides two independent full-duplex channels that can be programmed for use in any common asynchronous or synchronous data-communication protocol. Figure 10 illustrates some of these protocols. The following is a short description of them. A more detailed explanation of these modes can be found in the *Z80 Family Technical Manual.*

ASYNCHRONOUS MODES

Transmission and reception can be done independently on each channel with five to eight bits per character, plus optional even or odd parity. The transmitters 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 spikerejection mechanism that checks the signal one-half a bit time after a Low level is detected on the receive data input (RxDA or RxDB in figure 6). 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 partial character on which they occurred. Vectored interrupts allow fast servicing of error 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 SIO does not require symmetric transmit and receive clock signals – a feature that allows it to be used with a Z80C CTC or many other clock sources. The transitter 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.

In asynchronous modes, the $\overline{\text{SYNC}}$ pin may be programmed as an input that can be used for functions such as monitoring a ring indicator.

SYNCHRONOUS MODES

The SIO supports both byte-oriented and bit oriented synchronous communication.

Synchronous byte-oriented protocols can be handled in several modes that allow character synchronization with an 8-bit sync character (Monosync), any 16-bit sync pattern (Bysinc), or with an external sync signal. Leading sync characters can be removed without interrupting the CPU.

Five-, six- or seven-bit sync characters are detected with 8- or 16-bit patterns in the SIO by overlapping the larger pattern across multiple in-coming sync characters, as shown in figure 11.

CRC checking for synchronous byte-oriented modes is delayed by one character time so the CPU may disable CRC checking on specific characters. This permits implementation of protocols such as IBM Bisync.

Both CRC-16 ($X^{16} + X^{15} + X^2 + 1$) and CCITT ($X^{16} + X^{12} + X^5 + 1$) error checking polynomials are supported. In all non-SDLC modes, the CRC generator is initialized to 0's ; in SDLC modes, it is initialized to 1's. The SIO can be used for interfacing to peripherals such as hard-sectored floppy disk, but it cannot generate or check CRC for IBM-compatible soft-sectored disks. The SIO also provides a feature that automatically transmits CRC data when no other data is available for transmissions. This allows very high-speed transmissions under DMA control with no need for CPU intervention at the end of a message. When there is no data or CRC to send in synchronous modes, the transmitter inserts 8- or 16-bit sync characters regardless of the programmed character length.

The SIO supports synchronous bit-oriented protocols such as SDLC and HDLC by performing automatic flag seding, zero insertion and CRC generation. A special command can be used to abort a frame in transmission. At the end of a message the SIO automatically transmits the CRC and trailing flag when the transmit buffer becomes empty. If a transmit underrun occurs in the middle of a message, an external/status interrupt warns the CPU of this status change so that an abort may be issued. One to eight bits per character can be sent, which allows reception of a message with no prior information about the character structure in the information field of a frame.

The receiver automatically synchronizes on the leading flag of a frame in SDLC or HDLC, and provides a synchronization signal on the $\overline{\text{SYNC}}$ pin ; an interrupt can also be programmed. The receiver can be programmed to search for frames addressed by a single byte to only a specified user-selected address or to a global broadcast address. In this mode, frames that do not match either the user-selected or broadcast address are ignored. The number of address bytes can be extended under software control. For transmitting data, an interrupt on the first received character or on every character can be selected. The receiver automatically deletes all zeroes inserted by the transmitter during character assembly. It also calculates and automatically checks the CRC to validate frame transmission. At the end of

transmission, the status of a received frame is available in the status registers.

The SIO can be conveniently used under DMA control to provide high-speed reception or transmission. In reception, for example, the SIO can interrupt the CPU when the first character of a message is received. The CPU then enables the DMA to transfer the message to memory. The SIO then issues an end-of-frame interrupt and the CPU can check the status of the received message. Thus, the CPU is freed for other service while the message is being received.

Figure 10 : Some Z80C SIO Protocols.

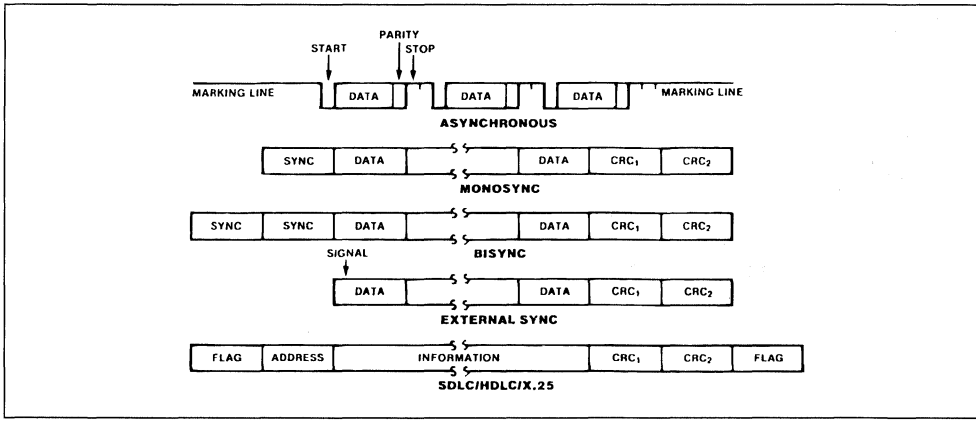
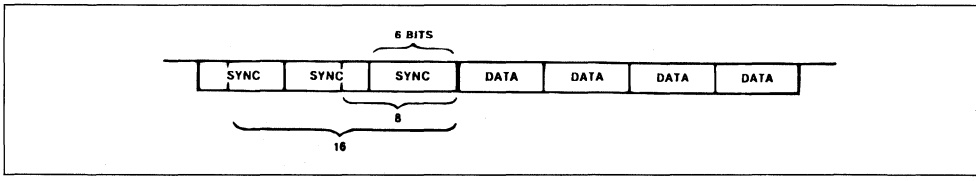


Figure 11 : Six Bit Sync Character Recognition.



STATUS FLOW-CHART

Figure 12a : Status Flowchart.

Figure 12b : Status Flowchart.

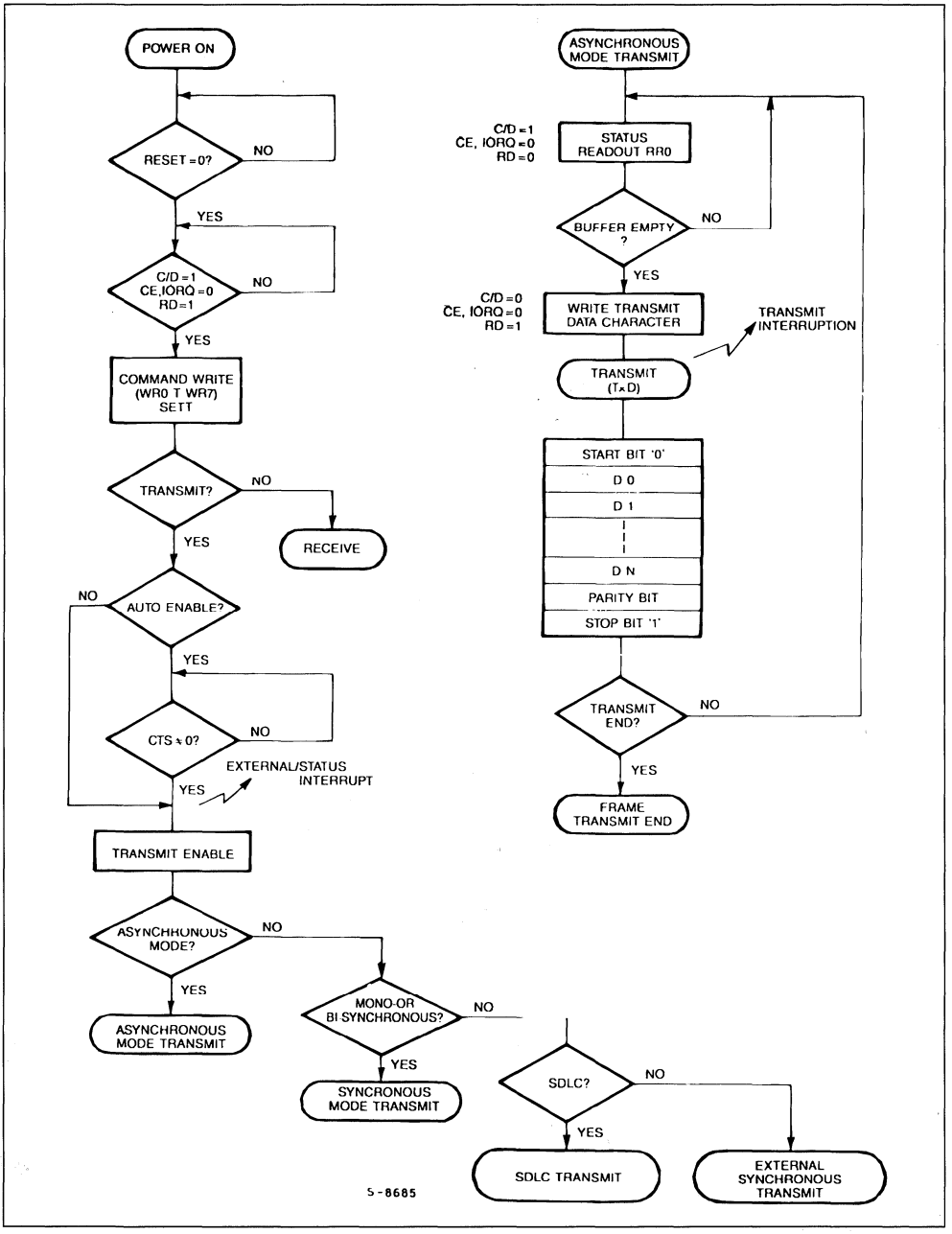


Figure 12c : Status Change Flowchart.

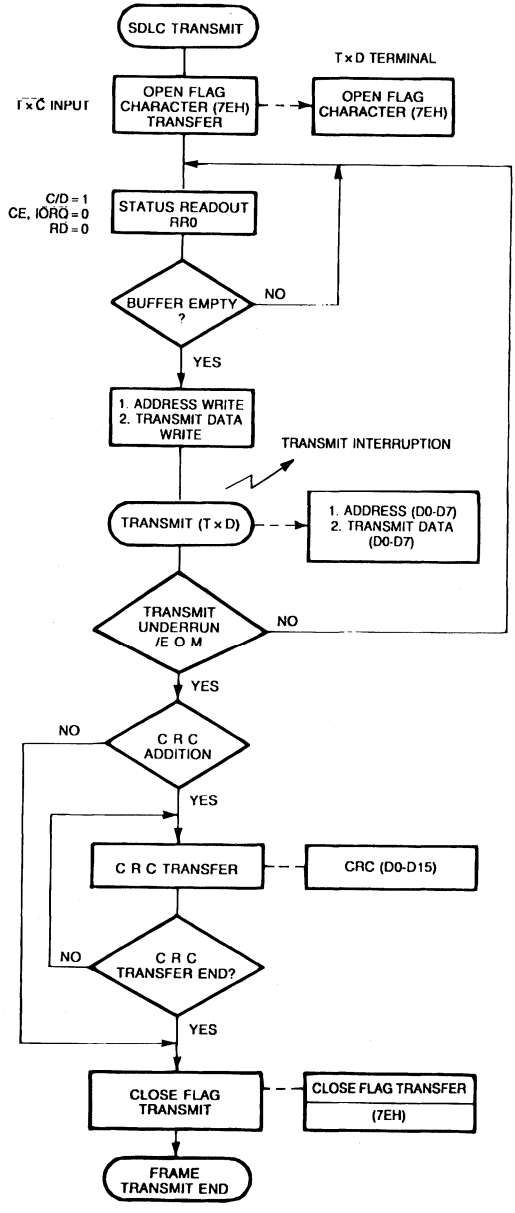


Figure 12d : Status Change Flowchart.

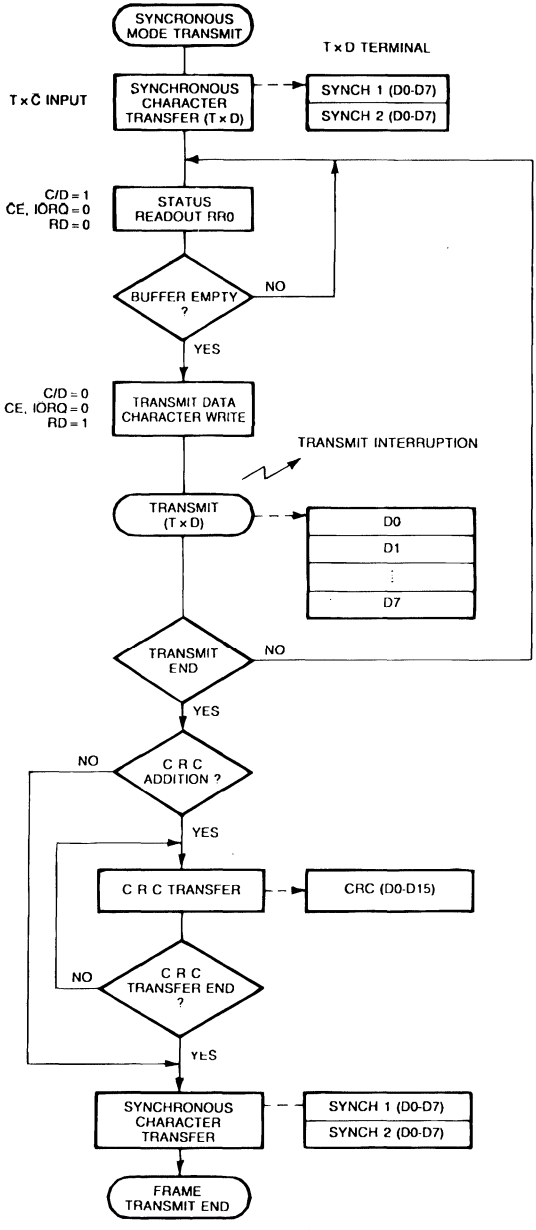
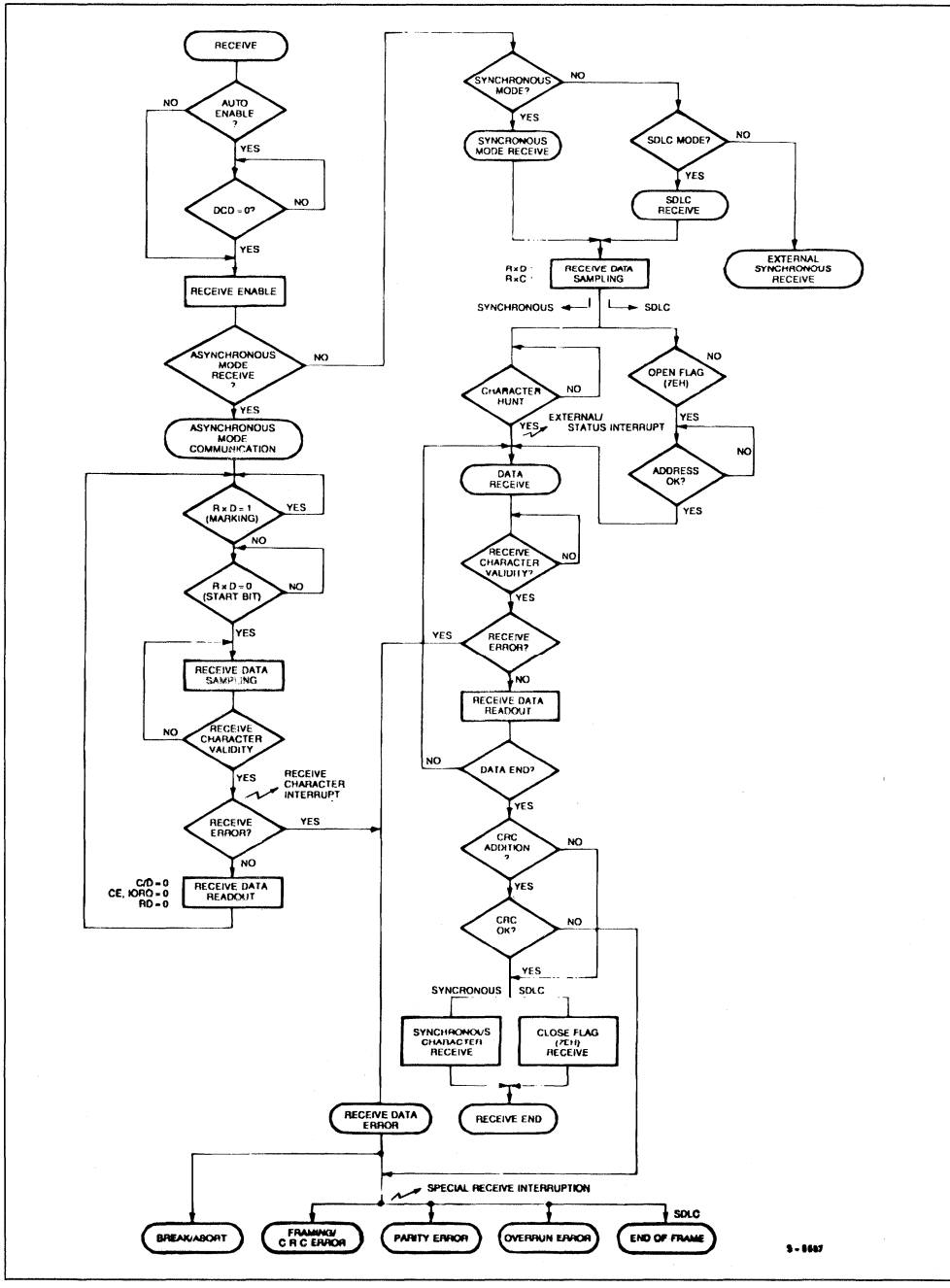


Figure 12e : Status Flowchart.



I/O INTERFACE CAPABILITIES

The SIO offers the choice of polling, interrupt, (vectored or non-vectored) and block-transfers modes to transfer data, status and control information to and from the CPU. The block-transfer mode can also be implemented under DMA control.

POLLING

Two status registers are updated at appropriate times for each function being performed (for example, CRC error-status valid at the end of a message). When the CPU is operated in a polling fashion, one of the SIO's two status registers is used to indicate whether the SIO has some data or needs some data.

Depending on the contents of this register, the CPU will either write data, read data, or just go on. Two bits in the register indicate that a data transfer is needed. In addition, error and other conditions are indicate. The second status register (special receive conditions) does not have to be read in a polling sequence, until a character has been received. All interrupt modes are disabled when operating the device in a polled environment.

INTERRUPTS

The SIO has an elaborate interrupt scheme to provide fast interrupt service in real-time applications. A control register and a status register in Channel B contain the interrupt vector. When programmed to do so, the SIO can modify three bits of the interrupt vector in the status register so that it points directly to one of eight interrupt service routines in memory, thereby servicing conditions in both channels and eliminating most of the needs for a status-analysis routine.

Transmitt 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 receive, 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). The receiver can interrupt the CPU in one or two ways :

- Interrupt on first received character
- Interrupt on all received characters

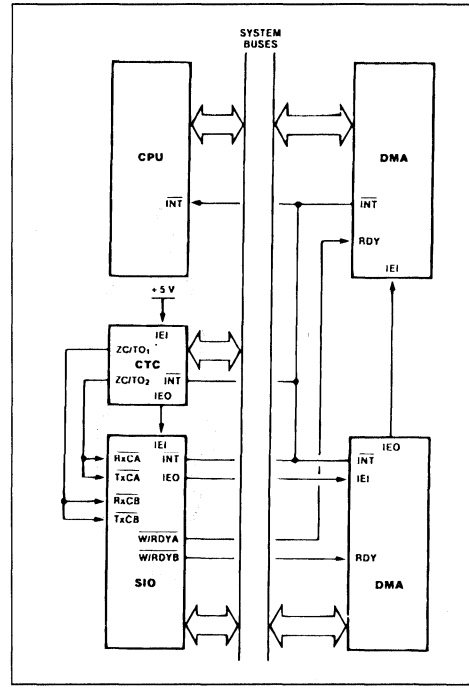
Interrupt-on-first-received-character is typically used with the block-transfer mode. Interrupt-on-all-received-characters has the option of modifying the interrupt vector in the event of a parity error. Both of these interrupt modes will also interrupt under special receive conditions on a character or message basis (end-of-frame interrupt in SDLC, for example).

This means that the special-receive condition can cause an interrupt only if the interrupt-on-first-received-character or interrupt-on-all-received-characters mode is selected. In interrupt-on-first-received-character, an interrupt can occur from special-receive conditions (except parity error) after the first-received-character interrupt (example : receive-overrun interrupt).

The main function of the external/status interrupt is to monitor the signal transitions of the Clear To Send (CTS), Data Carrier Detect (DCD) and Synchronization (SYNC) pins (figures 1 through 6). In addition, an external/status interrupt is also caused by a CRC-sending condition or by the detection of a break sequence (asynchronous mode) or abort sequence (SDLC mode) in the data stream.

The interrupt caused by the break/abort sequence allows the SIO to interrupt when the break/abort sequence is detected or terminated. This feature facilitates the proper temination of the current message,

Figure 13 : Typical Z80C Environment.



correct initialization of the next message, and the accurate timing of the break/abort condition in external logic.

In a Z80C CPU environment (figure 13), SIO interrupt vectoring is "automatic" : the SIO passes its internally-modificable 8-bit interrupt vector to the CPU, which adds an additional 8 bits from its interrupt-vector (I) register to from the memory address of the interrupt-routine table. This table contains the address of the beginning of the interrupt routine itself. The process entails an indirect transfer or CPU control to the interrupt routine, so that the next instruction executed after an interrupt acknowledge by the CPU is the first instruction of the interrupt routine itself.

CPU/DMA BLOCK TRANSFER

The SIO's block-transfer mode accommodates both CPU block transfers and DMA controllers (Z80C DMA or other designs). The block-transfer mode uses the Wait/Ready output signal, which is selected with three bits in an internal control register. The Wait/Ready output signal can be programmed 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 SIO READY output indicates that the SIO is ready to transfer data to or from memory. To the CPU, the WAIT output indicates the SIO is not ready to transfer data, thereby requesting the CPU to extend the I/O cycle.

INTERNAL STRUCTURE

The internal structure of the device includes a Z80C CPU interface, internal control and interrupt logic, and two full-duplex channels.

Each channel contains its own set of control and status (write and read) registers, and control and status logic that provides the interface to modems or other external devices.

The registers for each channel are designated as follows :

WR0-WR7 - Write Registers 0 through 7
RR0-RR2 - Read Register 0 through 2

The register group includes five 8-bit control registers, two sync-character 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 another 8-bit register (Read Register 2) in Channel B. The bit assignment and functional grouping of each register is configured to simplify and organize the programming process. Table 1 list the functions assigned to each read or write register.

The logic for both channels provides formats, synchronization and validation for data transferred to and from the channel interface. The modem control inputs, Clear To Send (CTS) and Data Carrier Detect (DCD), are monitored by the external control and status logic under program control. All external control-and-status-logic signals are general-purpose in nature and can be used for functions other than modem control.

DATA PATH

The transmit and receive data path illustrated for Channel A in figure 13 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.

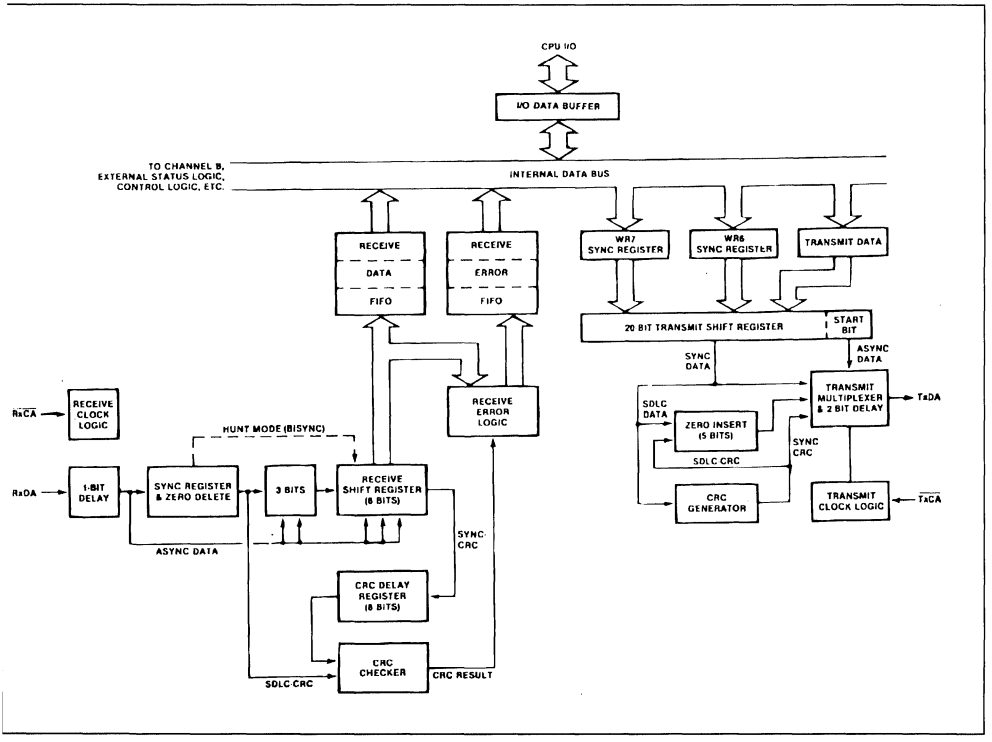
This scheme creates additional time for the CPU to service an interrupt at the beginning of a block of high-speed data. Incoming data is routed through one of several paths (data or CRC) depending on the selected mode andin asynchronous modesthe character length.

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)
Write Register Functions	
WR0	Register pointers, CRC initialize, 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
WR6	Sync Character or SDLC Address Field
WR7	Sync Character or SDLC Flag

The transmitter has an 8-bit transmit data buffer register that is loaded from the internal data bus, and a 20-bit transmit shift register that can be loaded from the sync-character buffers or from the transmit data register.

Depending on the operational mode, outgoing data is routed throught one of four main paths before it is transmitted from the Transmit Data output TxD).

Figure 14 : Transmitt and Receive Data Path (channel A).



PROGRAMMING

The system program first issues a series of commands that initialize the basic mode of operation and then other commands that qualify conditions within the selected mode. For example, the asynchronous mode, character length, clock rate, number of stop bits, even or odd parity might be set first ; then the interrupt mode ; and finally, receiver or transmitter enable.

Both channels contain registers that must be programmed via the system program prior operation. The channel-select input (B/A) and the control/data input (C/D) are the command-structure addressing controls, and are normally controlled by the CPU address bus. Figures 17 and 18 illustrate the timing relationships for programming the write registers and transfering data and status.

READ REGISTER

The SIO contains three read registers for Channel B and two read registers for Channel A (RR0-RR2 in figure 14) that can be to obtain the status information ; RR2 contains the internally-modifiable interrupt vector and is only in the Channel B register set. 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 a read 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 form a single register (RRI).

WRITE REGISTERS

The SIO contains eight write registers for Channel B and seven write registers for Channel A (WR0-WR7 in figure 15) that are programmed separately to configure the functional personality of the channels ; WR2 contains the interrupt vector for both channels and is only in the Channel B register set. With the exception of WR0, programming the write registers requires two bytes. The first byte is to WR0 and 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 SIO.

WR0 is a special case in that all of the basic commands can be written to it with a single byte. Reset (internal or external) initializes the pointer bits D_0-D_2 to point to WR0. This implies that a channel reset must not be combined with the pointing to any register.

Figure 15 : Read Register Bit Functions.

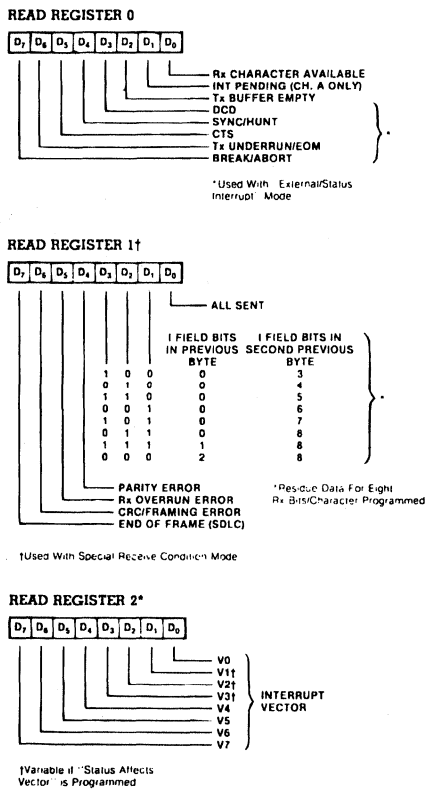
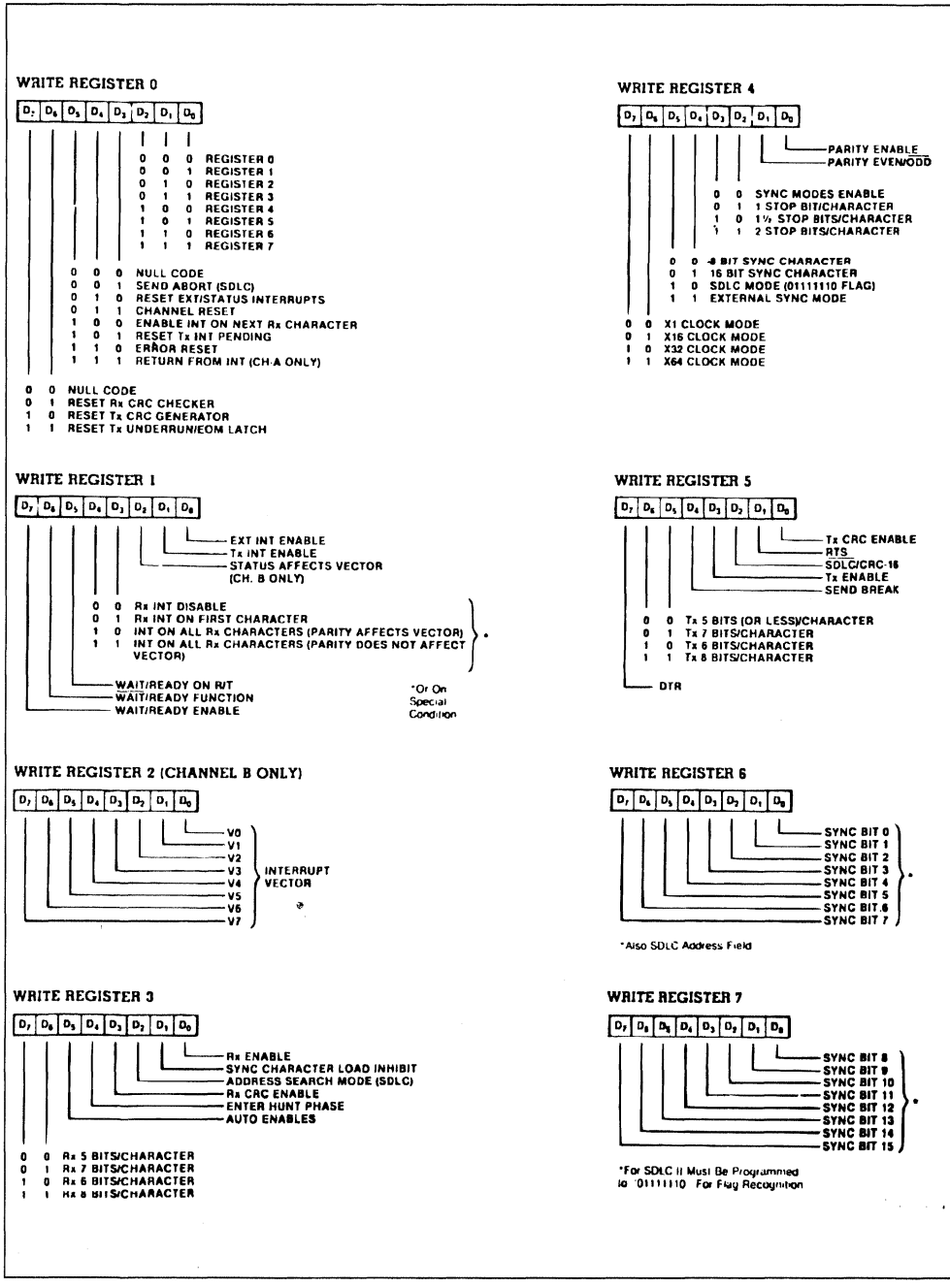


Figure 16 : Write Register Bit Functions.



TIMING

The SIO must have the same clock as the CPU (same phase and frequency relationship, not necessarily the same driver).

READ CYCLE

The timing signals generated by a Z80C CPU input instruction to read a data or status byte from the SIO are illustrated in figure 16.

WRITE CYCLE

Figure 16 illustrates the timing and data signals generated by a Z80C CPU output instruction to write a data or control byte into the SIO.

INTERRUPT-ACKNOWLEDGE CYCLE

After receiving an interrupt-request signal from an SIO (INT pulled Low), the Z80C CPU sends an interrupt-acknowledge sequence (M1 Low, and IORQ Low a few cycles later) as in figure 18.

The SIO contains an internal daisy-chained interrupt structure for prioritizing nested interrupts for the various functions of its two channels, and this structure can be used within an external user-defined daisy chain that prioritizes several peripheral circuits.

The IEI of the highest-priority device is terminated High. A device that has an interrupt pending or under service forces its IEO Low. For devices with no interrupt pending or under service, IEO = IEI.

To insure stable conditions in the daisy chain, all interrupt status signals are prevented from changing while M1 is Low. When IORQ is Low, the highest priority interrupt requestor (the one with IEI High) places its interrupt vector on the data bus and sets its internal interrupt-under-service latch.

RETURN FROM INTERRUPT CYCLE

Figure 19 illustrates the return form interrupt cycle. Normally, the Z80C CPU issues a RETI (Return From Interrupt) instruction at the end of an interrupt sevice routine. RETI is a 2-byte opcode (ED-4D) that resets the interrupt-under-service latch in the SIO to terminate the interrupt that has just been processed. This is accomplished by manipulating the daisy chain in the following way.

The normal daisy-chain operation can be used to detect a pending interrupt ; however, it cannot distinguish between an interrupt under service and a pending unacknowledged interrupt of a higher priority. Whenever "ED" is decoded, the daisy chain is modified by forcing High the IEO of any interrupt that has not yet been acknowledged. Thus the daisy chain identifies the device presently under service as the only one with an IEI High and an IEO Low. If the next opcode byte is "4D", the interrupt-under-service latch is reset.

The ripple time of the interrupt daisy chain (both the High-to-Low and the Low-to-High transitions) limits the number of devices that can be placed in the daisy chain. Ripple time can be improved with carry-look-ahead, or by extending the interrupt-acknowledge cycle.

For further information about techniques for increasing the number of daisy-chained devices, refer to the Z80C CPU Data Sheet.

Figure 17 : Read Cycle.

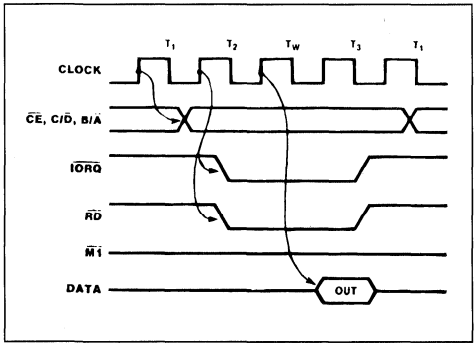


Figure 18 : Write Cycle.

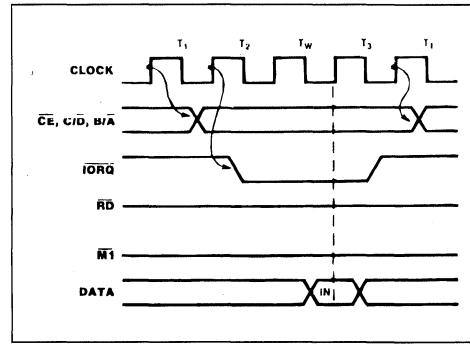


Figure 19 : Interrupt Acknowledge Cycle.

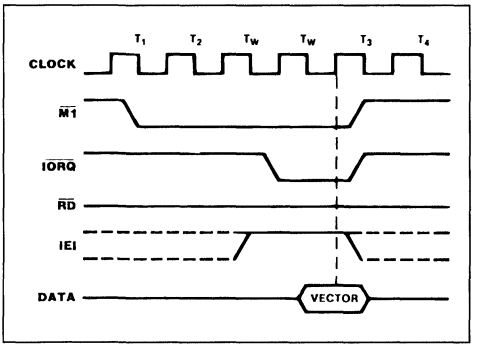


Figure 20 : Return from Interrupt Cycle.

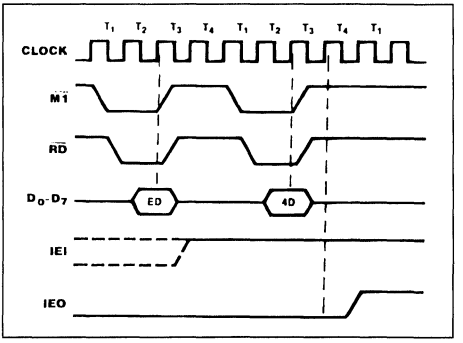
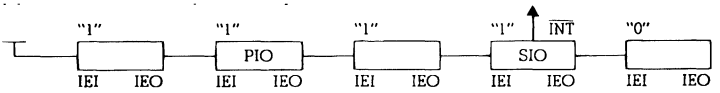
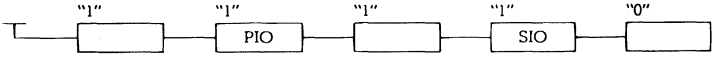


Figure 21 : Daisy chain at RETI Instruction.

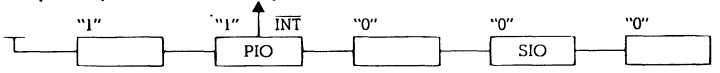
[1] Status where SIOs Request Interruption.



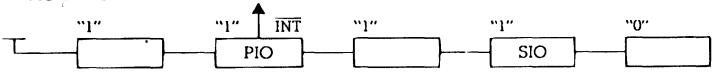
[2] SIOs are under Interruption Service



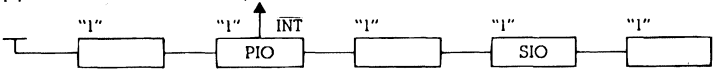
[3] Status where PIO has requested interruption immediately before SIOs decode "ED". By the request of PIO for interruption IEO of PIO becomes "0".



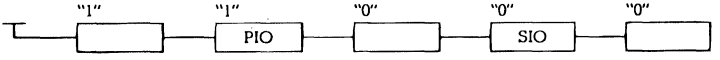
[4] As "EDH" has been decoded, the interruption request of PIO is not acknowledged. Therefore, IEO of PIO returns to "1".



[5] As "4DH" has been decoded, IEO of SIOs becomes "1".



[6] Interruption request by PIO is acknowledged, and the IEO of PIO becomes "0".



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_{CC}	V_{CC} Supply Voltage with Respect to V_{SS}	– 0.5 to 7	V
V_{IN}	Input Voltage	– 0.5 to V_{CC} + 0.5	V
P_D	Power Dissipation (T_A = 85 °C)	250	mW
T_{SOLDER}	Soldering Temperature (soldering time 10 sec)	260	°C
T_{stg}	Storage Temperature	– 65 to 150	°C
T_{op}	Operating Temperature	– 40 to 85	°C

CAPACITANCE (T_A = 25°C)

Symbol	Parameter	Test Condition	Min.	Typ.	Max.	Unit
C_{CLOCK}	Clock Capacitance	f = 1MHz	–	–	7	pF
C_{IN}	Input Capacitance		–	–	5	pF
C_{OUT}	Output Capacitance		–	–	10	pF

DC CHARACTERISTICS (T_A = – 40 °C to 85 °C, V_{CC} = 5 V ± 10 %, V_{SS} = 0 V)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_{ILC}	Clock Input Low Voltage		– 0.3	–	0.6	V
V_{IHC}	Clock Input High Voltage		V_{CC} – 0.6	–	V_{CC} + 0.3	V
V_{IL}	Input Low Voltage (except CLK)		– 0.5	–	0.8	V
V_{IH}	Input High Voltage (except CLK)		2.2	–	V_{CC}	V
V_{OL}	Output Low Voltage	I_{OL} = 2.0 mA	–	–	0.4	V
V_{OH1}	Output High Voltage (1)	I_{OH} = – 1.6 mA	2.4	–	–	V
V_{OH2}	Output High Voltage (2)	I_{OH} = – 250 µA	V_{CC} – 0.8	–	–	V
I_{LI}	Input Leakage Current	$V_{SS} \le V_{IN} \le V_{CC}$	–	–	± 10	µA
I_{LO}	3-State Output Leakage Current in Float	V_{SS} + 0.4 ≤ $V_{OUT} \le V_{CC}$	–	–	± 10	µA
$I_{L(SY)}$	$\overline{SYNC}$ Pin Leakage Current	V_{SS} + 0.4 ≤ $V_{OUT} \le V_{CC}$ V_{CC} = 5 V, CLK = 4 MHz	– 40	–	10	µA
I_{CC1}	Operating Supply Current : 4 MHz 6 MHz	$V_{IH} = V_{IHC} = V_{CC}$ – 0.2 V $V_{IL} = V_{ILC}$ = 0.2 V V_{CC} = 5 V, fCLK = $1/T_C C(min)$	 – –	 2.5 4	 6 10	 mA mA
I_{CC2}	Stand-by Supply Current (except SYNC pin)	$V_{IH} = V_{CC}$ – 0.2 V CLK = V_{IL} = 0.2 V, V_{CC} = 5 V	–	–	10	µA

TEST CONDITIONS

T_A = – 40 °C to + 85 °C

V_{CC} = 5 V ± 10 %

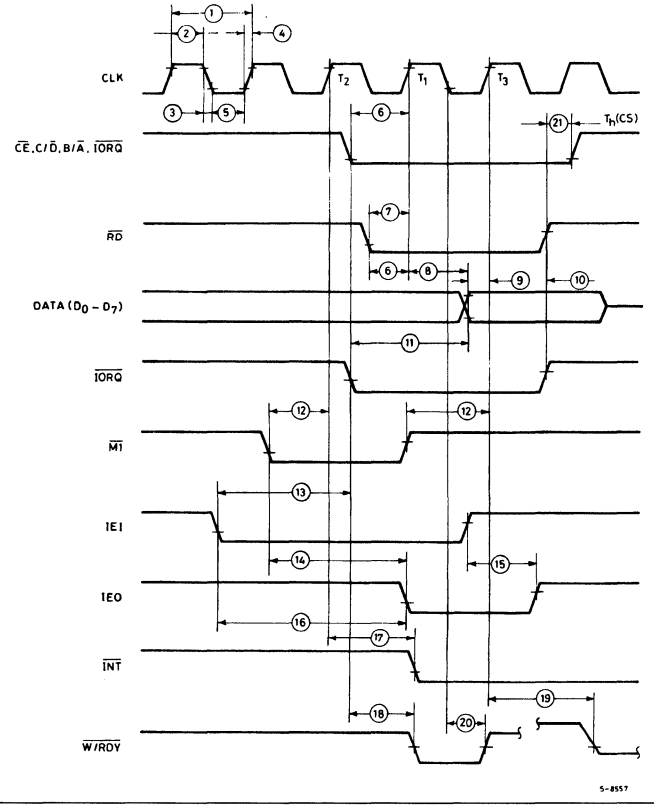
V_{SS} = 0 V

AC TEST CONDITIONS

- Inputs except CLK (clock) are driven at 2.4 V for a logic "1" and 0.4 V for a logic "0". Clock input is driven at V_{CC} – 0.6 V for a logic "1" and 0.6 V for a logic "0".
- Timing measurements are made at 2.2 V for a logic "1" and 0.8 V for a logic "0".

All AC parameters assume a load capacitance of 100 pF

AC CHARACTERISTICS

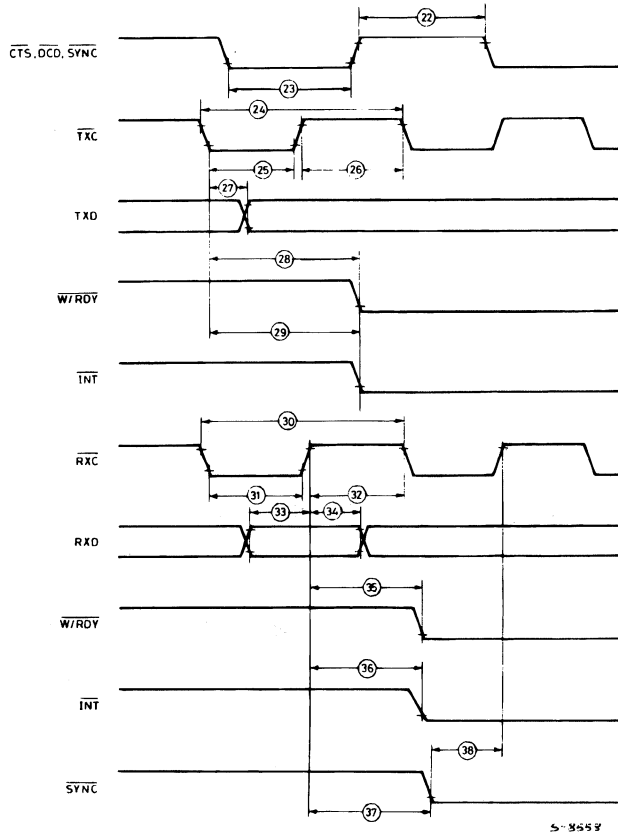


AC CHARACTERISTICS (continued)

N°	Symbol	Parameter	Z84C40 /1/2A		Z84C40 /1/2B		Unit
			Min.	Max.	Min.	Max.	
1	TcC	Clock Cycle Time	250	DC	165	DC	ns
2	TwCh	Clock Width (high)	105	DC	70	DC	ns
3	TfC	Clock Fall Time		30		15	ns
4	TrC	Clock Rise Time		30		15	ns
5	TwCl	Clock Width (low)	105	DC	70	DC	ns
6	TsCS(C)	$\overline{CE}$, C/$\overline{D}$, B/$\overline{A}$, $\overline{IORQ}$ to Clock ↑ Setup Time	145		60		ns
7	TsRD(C)	$\overline{RD}$ to Clock ↑ Setup Time	115		60		ns
8	TdC(D0)	Clock ↑ to Data Out Delay		220		150	ns
9	TsDI(C)	Data In to Clock ↑ Setup (write or MI Cycle)	50		30		ns
10	TdRD(D02)	RD ↑ to Data Out Float Delay		110		90	ns
11	TdI0(D01)	$\overline{IORQ}$ ↓ to Data Out Delay (INTACK cycle)		160		120	ns
12	TsMI(C)	$\overline{M1}$ to Clock ↑ Setup Time	90		75		ns
13	TsIEI(IO)	IEI to $\overline{IORQ}$ ↓ Setup Time (INTACK cycle)	140		120		ns
14	TdMI(IEO)	$\overline{M1}$ ↓ to IEO ↓ Delay (interrupt before M1)		190		160	ns
15*	TdIEI(IEOr)	IEI ↑ to IEO ↑ Delay (after ED decode)		160		110	ns
16	TdIEI(IEOf)	IEI ↓ to IEO ↓ Delay		100		70	ns
17	TdC(INT)	Clock ↑ to $\overline{INT}$ ↓ Delay		200		150	ns
18	TdILO(W/RWf)	$\overline{IORQ}$ ↓ or CE ↓ to $\overline{W/RDY}$ ↓ Delay (wait mode)		210		175	ns
19	TdC(W/RR)	Clock ↑ to $\overline{W/RDY}$ ↓ Delay (ready mode)		120		100	ns
20	TdC(W/RWZ)	Clock ↓ to $\overline{W/RDY}$ Float Delay (wait mode)		130		110	ns
21	Th, Th(CS)	Any unspecified hold when setup is specified.	0		0		ns

Note : * Not compatible with NMOS Specifications.

AC CHARACTERISTICS (continued)



AC CHARACTERISTICS (continued)

N°	Symbol	Parameter	Z84C40 /1/2A		Z84C40 /1/2B		Unit
			Min.	Max.	Min.	Max.	
22	TwPH	Pulse Width (high)	200		200		ns
23	TwPl	Pulse Width (low)	200		200		ns
24	TcTxc	$\overline{Txc}$ Cycle Time	400		330		ns
25	TwTxcl	$\overline{Txc}$ Width (low)	180	∞	100	∞	ns
26	TwTxch	$\overline{Txc}$ Width (high)	180	∞	100	∞	ns
27	TdTxC(TxD)	$\overline{TXC}$ ↓ to TxD Delay (X1 mode)		300		220	ns
28	TdTxC(W/RRf)	$\overline{TXC}$ ↓ to $\overline{W/RDY}$ ↓ Delay (ready mode)	5	9	5	9	CLK Periods
29	TdTxC(INT)	$\overline{TXC}$ ↓ to $\overline{INT}$ ↓ Delay	5	9	5	9	CLK Periods
30	TcRxC	$\overline{Rxc}$ Cycle Time	400	∞	330	∞	ns
31	TwRxCl	$\overline{Rxc}$ Width (low)	180	∞	100	∞	ns
32	TwRxCh	$\overline{Rxc}$ Width (high)	180	∞	100	∞	ns
33	TsRxD(RxC)	RxD to RxC ↑ Setup Time (xl mode)	0		0		ns
34	ThRxD(RxC)	$\overline{RxC}$ ↑ to RxD Hold Time (xl mode)	140		100		ns
35	TdRxC(W/RRf)	$\overline{RxC}$ ↑ to $\overline{W/RDY}$ ↓ Delay (ready mode)	10	13	10	13	CLK Periods
36	TdRxC(INT)	$\overline{RxC}$ ↑ to $\overline{INT}$ ↓ Delay	10	13	10	13	CLK Periods
37	TdRxC(SYNC)	$\overline{RxC}$ ↑ to $\overline{SYNC}$ ↓ Delay (output modes)	4	7	4	7	CLK Periods
38	TsSYNC(RxC)	$\underline{SYNC}$ ↓ to $\underline{RxC}$ ↑ Setup (external sync modes)	−100		100		ns

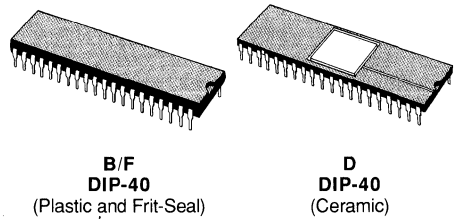
ORDERING INFORMATION

Type	Package	Temp.	Clock	Description
Z84C40/1/2AB6	DIP-40 (plastic)	−40/+ 85°C	4 MHz	Z80C Serial I/O Controller
Z84C40/1/2AD6	DIP-40 (ceramic)	−40/+ 85°C		
Z84C40/1/2AD2	DIP-40 (ceramic)	−55/+125°C		
Z84C44AC6	PLCC44 (plastic chip-carrier)	−40/+ 85°C		
Z84C40/1/2BB6	DIP-40 (plastic)	−40/+ 85°C	6 MHz	
Z84C40/1/2BD6	DIP-40 (ceramic)	−40/+ 85°C		
Z84C40/1/2BD2	DIP-40 (ceramic)	−55/+125°C		
Z84C44BC6	PLCC44 (plastic chip-carrier)	−40/+ 85°C		

NMOS FAMILY DATASHEETS

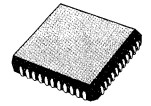
Z80 CPU CENTRAL PROCESS UNIT

- THE INSTRUCTION SET CONTAINS 158 INSTRUCTIONS. THE 78 INSTRUCTIONS OF THE 8080A ARE INCLUDED AS A SUBSET ; 8080A AND Z80* SOFTWARE COMPATIBILITY IS MAINTAINED
- 8MHz, 6MHz, 4MHz AND 2.5MHz CLOCKS FOR THE Z80H, Z80B, Z80A, THE Z80 CPU, RESULT IN RAPID INSTRUCTION EXECUTION WITH CONSEQUENT HIGH DATA THROUGHPUT
- THE EXTENSIVE INSTRUCTION SET INCLUDES STRING, BIT, BYTE, AND WORD OPERATIONS. BLOCK SEARCHES AND BLOCK TRANSFERS TOGETHER WITH INDEXED AND RELATIVE ADDRESSING RESULT IN THE MOST POWERFUL DATA HANDLING CAPABILITIES IN THE MICROCOMPUTER INDUSTRY
- THE Z80 MICROPROCESSORS AND ASSOCIATED FAMILY OF PERIPHERAL CONTROLLERS ARE LINKED BY A VECTORED INTERRUPT SYSTEM. THIS SYSTEM MAY BE DAISY-CHAINED TO ALLOW IMPLEMENTATION OF A PRIORITY INTERRUPT SCHEME. LITTLE, IF ANY, ADDITIONAL LOGIC IS REQUIRED FOR DAISY-CHAINING
- DUPLICATE SETS OF BOTH GENERAL-PURPOSE AND FLAG REGISTERS ARE PROVIDED, EASING THE DESIGN AND OPERATION OF SYSTEM SOFTWARE THROUGH SINGLE-CONTEXT SWITCHING, BACKGROUND-FOREGROUND PROGRAMMING, AND SINGLE-LEVEL INTERRUPT PROCESSING. IN ADDITION, TWO 16-BIT INDEX REGISTERS FACILITATE PROGRAM PROCESSING OF TABLES AND ARRAYS
- THERE ARE THREE MODES OF HIGH SPEED INTERRUPT PROCESSING : 8080 SIMILAR, NON-Z80 PERIPHERAL DEVICE, AND Z80 FAMILY PERIPHERAL WITH OR WITHOUT DAISY CHAIN
- ON-CHIP DYNAMIC MEMORY REFRESH COUNTER



B/F
DIP-40
(Plastic and Frit-Seal)

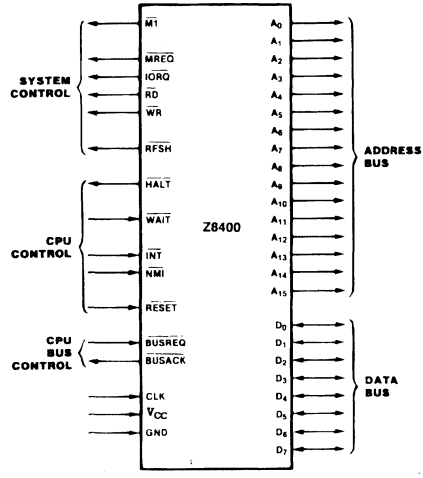
D
DIP-40
(Ceramic)



C
PLCC44
(Plastic)

(Ordering Information at the end of the datasheet)

LOGIC FUNCTIONS



DESCRIPTION

The Z80, Z80A, Z80B and Z80H CPUs are third-generation single-chip microprocessors with exceptional computational power. They offer higher system throughput and more efficient memory utilization than comparable second-and third-generation microprocessors. The internal registers contain 208 bits of read/write memory that are accessible to the programmer. These registers include two sets of six general-purpose registers which may be used individually as either 8-bit registers or as 16-bit register pairs. In addition, there are two sets of accumulator and flag registers. A group of "Exchange" instructions makes either set of main or alternate registers accessible to the programmer. The alternate set allows operation in foreground-background mode or it may be reserved for very fast interrupt response.

The Z80 also contains a Stack Pointer, Program Counter, two index registers, a Refresh register (counter), and an Interrupt register.

The CPU is easy to incorporate into a system since it requires only a single + 5V power source. All output signals are fully decoded and timed to control standard memory or peripheral circuits, and it is supported by an extensive family of peripheral controllers. The internal block diagram (figure 3) shows the primary functions of the Z80 processors. Subsequent text provides more detail on the Z80 I/O controller family, registers, instruction set, interrupts and daisy chaining, and CPU timing.

Figure 1 : Dual in Line Pin Configuration.

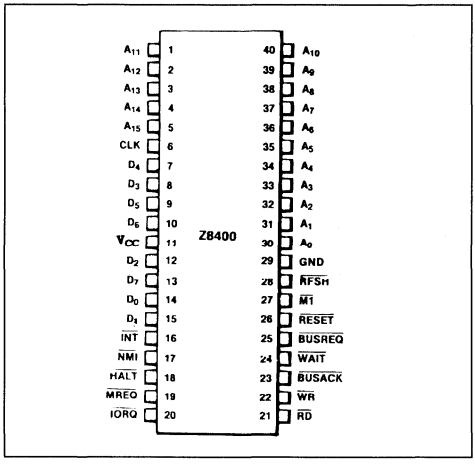
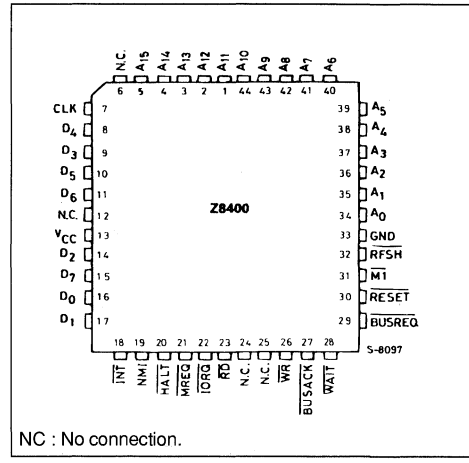
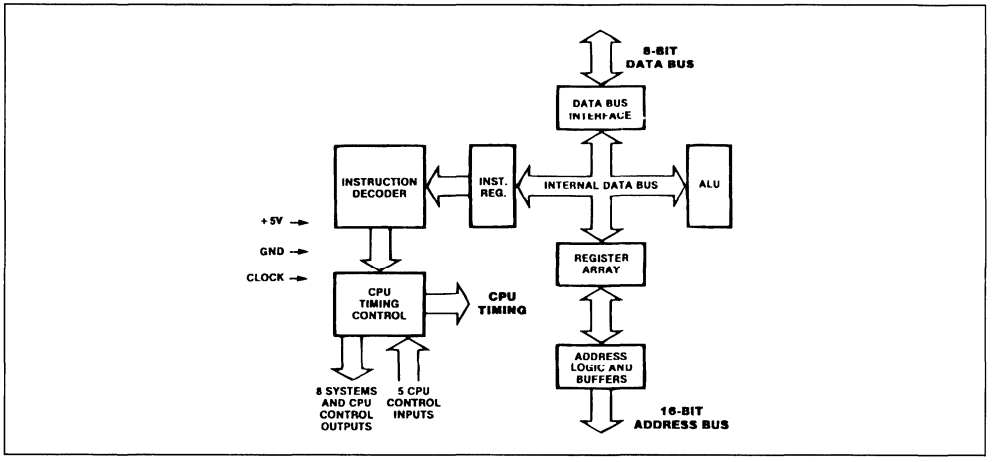


Figure 2 : Chip Carrier Pin Configuration.



NC : No connection.

Figure 3 : CPU Block Diagram.



Z80 MICROPROCESSOR FAMILY

The Z80, Z80A, Z80B and Z80H microprocessor is the central element of a comprehensive microprocessor product family. This family works together in most applications with minimum requirements for additional logic, facilitating the design of efficient and cost-effective microcomputer-base systems.

Five components to provide extensive support for the Z80 microprocessor. These are :

- The CTC (Counter/Timer Circuit) features four programmable 8-bit counter/timers, each of which has an 8-bit prescaler. Each of the four channels may be configurated to operate in either counter or timer mode.
- The PIO (Parallel Input/Output) operates in both data-byte I/O transfer mode (with handshaking) and in bit mode (without handshaking). The PIO may be configured to interface with standard parallel peripheral devices such as printers, tape punches, and keyboards.
- The DMA (Direct Memory Access) controller provides dual port data transfer operations and the ability to terminate data transfer as a result of a pattern match.
- The SIO (Serial Input/Output) controller offers two channels. It is capable of operating in a variety of programmable medes for both synchronous and asynchronous communication, including Bi-Synch and SDLC.
- The DART (Dual Asynchronous Receiver/Transmitter) device provides low cost asynchronous serial communication. It has two channels and a full modem control interface.

Z80 CPU REGISTERS

Figure 4 shows three groups of registers within the Z80 CPU. The first group consists of duplicate sets of 8-bit registers : a principal set and an alternate set (designated by' [prime], e.g., A'). Both sets consist of the Accumulator Register, the Flag Register, and six general-purpose registers. Transfer of data between these duplicate sets of registers is accomplished by use of "Exchange" instructions. The result is faster response to interrupts and easy, efficient implementation of such versatile programming techniques as background-foreground data processing. The second set of registers consists of six registers with assigned functions. These are the I (Interrupt Register), the R (Refresh Register), the IX and IY (Index Registers), the SP (Stack Pointer), and the PC (Program Counter). The third group consists of two interrupt status flip-flops, plus and additional pair of flip-flops which assists in identifying the interrupt mode at any particular time. Table 1 provides further information on these registers.

Figure 4 : CPU Registers

Main Register Set		Alternate Register Set	
A Accumulator	F Flag Register	A' Accumulator	F' Flag Register
B General Purpose	C General Purpose	B' General Purpose	C' General Purpose
D General Purpose	E General Purpose	D' General Purpose	E' General Purpose
H General Purpose	L General Purpose	H' General Purpose	L' General Purpose

← 8 Bits →

← 16 Bits →

IX Index Register	
IY Index Register	
SP Stack Pointer	
PC Program Counter	
I Interrupt Vector	R Memory Refresh

← 8 Bits →

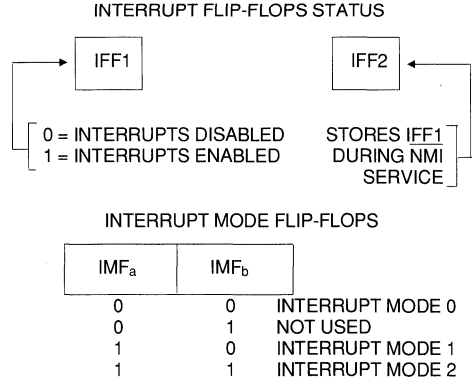


Table 1. CPU Registers.

Register		Size (Bits)	Remarks
A, A'	Accumulator	8	Stores an Operand or the Results of an Operation
F, F'	Flags	8	See Instruction Set.
B, B'	General Purpose	8	Can be used separately or as a 16-bit register with C.
C, C'	General Purpose	8	See B, above.
D, D'	General Purpose	8	Can be used separately or as a 16-bit register with E.
E, E'	General Purpose	8	See D, above.
H, H'	General Purpose	8	Can be used separately or as a 16-bit register with L.
L, L'	General Purpose	8	See H, above. Note : The (B, C), (D, E), and (H, L) sets are combined as follows : B–High Byte C–Low Byte D–High Byte E–Low Byte H–High Byte L–Low Byte
I	Interrupt Register	8	Stores upper eight bits of memory address for vectored interrupt processing.
R	Refresh Register	8	Provides user-transparent dynamic memory refresh. Lower seven bits are automatically incremented and all eight are placed on the address bus during each instruction fetch cycle refresh time.
IX	Index Register	16	Used for indexed addressing.
IY	Index Register	16	Same as IX, above.
SP	Stack Pointer	16	Holds address of the top of the stack. See Push or Pop in Instruction Set.
PC	Program Counter	16	Holds address of next instruction.
IFF_1–IFF_2	Interrupt Enable	Flip-Flops	Set or reset to indicate interrupt status (see figure 4).
IMFa–IMFb	Interrupt Mode	Flip-Flops	Reflect Interrupt Mode (see figure 4).

INTERRUPTS : GENERAL OPERATION

The CPU accepts two interrupt input signals : $\overline{NMI}$ and $\overline{INT}$. The NMI is a non-maskable interrupt and has the highest priority. $\overline{INT}$ is a lower priority interrupt and it requires that interrupts be enabled in software in order to operate. $\overline{INT}$ can be connected to multiple peripheral devices in a wired-OR configuration.

The Z80 has a single response mode for interrupt service for the non-maskable interrupt. The maskable interrupt, $\overline{INT}$, has three programmable response modes available.

These are :

- Mode 0 – similar with the 8080 microprocessor.
- Mode 1 – Peripheral Interrupt service, for use with non-8080/Z80 systems.
- Mode 2 – a vectored interrupt scheme, usually daisy-chained, for use with Z80 Family and compatible peripheral devices.

The CPU services interrupts by sampling the $\overline{NMI}$ and $\overline{INT}$ signals at the rising edge of the last clock of an instruction. Further interrupt service processing depends upon the type of interrupt that was detected. Details on interrupt responses are shown in the CPU Timing Section.

NON-MASKABLE INTERRUPT ($\overline{NMI}$)

The non-maskable interrupt cannot be disabled by program control and therefore will be accepted at all times by the CPU. $\overline{NMI}$ is usually reserved for servicing only the highest priority type interrupts, such as that for orderly shut-down after power failure has been detected.

After recognition of the $\overline{NMI}$ signal (providing $\overline{BUSREQ}$ is not active), the CPU jumps to restart location 0066H. Normally, software starting at this address contains the interrupt service routine.

MASKABLE INTERRUPT ($\overline{INT}$)

Regardless of the interrupt mode set by the user, the Z80 response to a maskable interrupt input follows a common timing cycle. After the interrupt has been detected by the CPU (provided that interrupts are enabled and $\overline{BUSREQ}$ is not active) a special interrupt processing cycle begins. This is a special fetch ($\overline{M1}$) cycle in which $\overline{IORQ}$ becomes active rather than $\overline{MREQ}$, as in normal $\overline{M1}$ cycle. In addition, this special $\overline{M1}$ cycle is automatically extended by two $\overline{WAIT}$ states, to allow for the time required to acknowledge the interrupt request.

MODE 0 INTERRUPT OPERATION

This mode is similar to the 8080 microprocessor interrupt service procedures. The interrupting device places an instruction on the data bus. This is normally a Restart Instruction, which will initiate a call to the selected one of eight restart locations in page zero of memory. Unlike the 8080, the Z80 CPU responds to the Call instruction with only one interrupt acknowledge cycle followed by two memory read cycles.

MODE 1 INTERRUPT OPERATION

Mode 1 operation is very similar to that for the $\overline{NMI}$. The principal difference is that the Mode 1 interrupt has a restart location of 0038H only.

MODE 2 INTERRUPT OPERATION

This interrupt mode has been designed to utilize most effectively the capabilities of the Z80 microprocessor and its associated peripheral family. The interrupting peripheral device selects the starting address of the interrupt service routine. It does this by placing an 8-bit vector on the data bus during the interrupt acknowledge cycle. The CPU forms a pointer using this byte as the lower 8-bits and the contents of the I register as the upper 8-bits. This points to an entry in a table of addresses for interrupt service routines. The CPU then jumps to the routine at that address. This flexibility in selecting the interrupt service routine address allows the peripheral device to use several different types of service routines. These routines may be located at any available location in memory. Since the interrupting device supplies the low-order byte of the 2-byte vector, bit 0 (A_0) must be a zero.

Table 2 : State of Flip-Flops.

Action	IFF_2	IFF_2	Comments
CPU Reset	0	0	Maskable Interrupt $\overline{INT}$ Disabled
DI Instruction execution	0	0	Maskable Interrupt $\overline{INT}$ Disabled
EI Instruction execution	1	1	Maskable Interrupt $\overline{INT}$ Enabled
LD A, I Instruction execution	•	•	$IFF_2 \rightarrow$ Parity Flag
LD A, R Instruction execution	•	•	$IFF_2 \rightarrow$ Parity Flag
Accept $\overline{NMI}$	0	IFF_1	$IFF_1 \rightarrow IFF_2$ (maskable interrupt $\overline{INT}$ disabled)
RETN Instruction execution	IFF_2	•	$IFF_2 \rightarrow IFF_1$ at Completion of an $\overline{NMI}$ Service Routine.

INTERRUPT PRIORITY (Daisy Chaining and Nested Interrupts)

The interrupt priority of each peripheral device is determined by its physical location within a daisy-chain configuration. Each device in the chain has an interrupt enable input line (IEI) and an interrupt enable output line (IEO), which is fed to the next lower priority device. The first device in the daisy chain has its IEI input hardwired to a High level. The first device has highest priority, while each succeding device has a corresponding lower priority. This arrangement permits the CPU to select the highest priority interrupt from several simultaneously interrupting peripherals.

The interrupting device disables its IEO line to the next lower priority peripheral until it has been serviced. After servicing, its IEO line is raised, allowing lower priority peripherals to demand interrupt servicing.

The Z80 CPU will nest (queue) any pending interrupts or interrupts received while a selected peripheral is being serviced.

INTERRUPT ENABLE/DISABLE OPERATION

Two flip-flops, IFF_1 and IFF_2, referred to in the register description are used to signal the CPU interrupt status. Operation of the two flip-flops is described in table 2.

INSTRUCTION SET

The Z80 microprocessor has one of the most powerful and versatile instruction sets available in any 8-bit microprocessor. It includes such unique operations as a block move for fast, efficient data transfers within memory or between memory and I/O. It also allows operations on any bit in any location in memory.

The following is a summary of the Z80 instruction set and shows the assembly language mnemonic, the operation, the flag status, and gives comments on each instruction. The *Z80 CPU Technical Manual and Z80 CPU Programming Manual* contain significantly more details for programming use.

The instructions are divided into the following categories :

- 8-BIT LOADS
- 16-BIT LOADS
- EXCHANGES, BLOCK TRANSFERS, AND SEARCHES
- 8-BIT ARITHMETIC AND LOGIC OPERATIONS
- GENERAL-PURPOSE ARITHMETIC AND CPU CONTROL
- 16-BIT ARITHMETIC OPERATIONS
- ROTATES AND SHIFT
- BIT SET, RESET, AND TEST OPERATIONS
- JUMPS
- CALLS, RETURNS, AND RESTARTS
- INPUT AND OUTPUT OPERATIONS.

A variety of addressing modes are implemented to permit efficient and fast data transfer between various registers, memory locations, and input/output devices. These addressing modes include :

- IMMEDIATE
- IMMEDIATE EXTENDED
- MODIFIED PAGE ZERO
- RELATIVE
- EXTENDED
- INDEXED
- REGISTER
- REGISTER INDIREDT
- IMPLIED
- BIT

INSTRUCTION SET (continued)

8-BIT LOAD GROUP

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210					
LD r, r'	r ← r'	•	•	X	•	X	•	•	•	01 r r'		1	1	4	r, r' Reg.
LD r, n	r ← n	•	•	X	•	X	•	•	•	00 r 110 ← n →		2	2	7	000 B 001 C 010 D 011 E 100 H 101 L 111 A
LD r, (HL)	r ← (HL)	•	•	X	•	X	•	•	•	01 r 110	DD	1	2	7	
LD r, (IX+d)	r ← (IX + d)	•	•	X	•	X	•	•	•	11 011 101 01 r 101 ← d →		3	5	19	
LD r, (IY+d)	r ← (IY + d)	•	•	X	•	X	•	•	•	11 111 101 01 r 110 ← d →	FD	3	5	19	
LD (HL), r	(HL) ← r	•	•	X	•	X	•	•	•	01 110 r		1	2	7	
LD (IX+d), r	(IX + d) ← r	•	•	X	•	X	•	•	•	11 011 101 01 110 r ← d →	DD	3	5	19	
LD (IY+d), r	(IY + d) ← r	•	•	X	•	X	•	•	•	11 111 101 01 110 r ← d →	FD	3	5	19	
LD (HL), n	(HL) ← n	•	•	X	•	X	•	•	•	00 110 110 ← n →	36	2	3	10	
LD (IX+d), n	(IX + d) ← n	•	•	X	•	X	•	•	•	11 011 101 00 110 110 ← d → ← n →	DD 36	4	5	19	
LD (IY+d), n	(IY + d) ← n	•	•	X	•	X	•	•	•	11 111 101 00 110 110 ← d → ← n →	FD 36	4	5	19	
LD A, (BC)	A ← (BC)	•	•	X	•	X	•	•	•	00 001 010	0A	1	2	7	
LD A, (DE)	A ← (DE)	•	•	X	•	X	•	•	•	00 011 010	1A	1	2	7	
LD A, (nn)	A ← (nn)	•	•	X	•	X	•	•	•	00 111 010 ← n → ← n →	3A	3	4	13	
LD (BC), A	(BC) ← A	•	•	X	•	X	•	•	•	00 000 010	02	1	2	7	
LD (DE), A	(DE ← A	•	•	X	•	X	•	•	•	00 010 010	12	1	2	7	
LD (nn), A	(nn) ← A	•	•	X	•	X	•	•	•	00 110 010 ← n → ← n →	32	3	4	13	
LD A, I	A ← I	↕	↕	X	0	X	IFF	0	•	11 101 101 01 010 111	ED 57	2	2	9	
LD A, R	A ← R	↕	↕	X	0	X	IFF	0	•	11 101 101 01 011 111	ED 5F	2	2	9	
LD I, A	I ← A	•	•	X	•	X	•	•	•	11 101 101 01 000 111	ED 47	2	2	9	
LD R, A	R ← A	•	•	X	•	X	•	•	•	11 101 101 01 001 111	ED 4F	2	2	9	

Notes : r, r' means any of the registers A, B, C, D, E, H, L. IFF the content of the interrupt enable flip-flop, (IFF) is copied into the P/V flag.
For an explanation of flag notation and symbols for mnemonic tables, see Symbolic Notation section following tables.

INSTRUCTION SET (continued)

16-BIT LOAD GROUP

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210					
LD dd, nn	dd ← nn	•	•	X	•	X	•	•	•	00 dd0 001 ← n → ← n →		3	3	10	dd Pair 00 BC 01 DE 10 HL 11 SP
LD IX, nn	IX ← nn	•	•	X	•	X	•	•	•	11 011 101 00 100 001 ← n → ← n →	DD 21	4	4	14	
LD IY, nn	IY ← nn	•	•	X	•	X	•	•	•	11 111 101 00 100 001 ← n → ← n →	FD 21	4	4	14	
LD HL, (nn)	H ← (nn + 1) L ← (nn)	•	•	X	•	X	•	•	•	00 101 010 ← n → ← n →	2A	3	5	16	
LD dd, (nn)	dd_H ← (nn + 1) dd_L ← (nn)	•	•	X	•	X	•	•	•	11 101 101 01 dd1 011 ← n → ← n →	ED	4	6	20	
LD IX, (nn)	IX_H ← (nn + 1) IX_L ← (nn)	•	•	X	•	X	•	•	•	11 011 101 01 101 010 ← n → ← n →	DD 2A	4	6	20	
LD IY, (nn)	IY_H ← (nn + 1) IY_L ← (nn)	•	•	X	•	X	•	•	•	11 111 101 00 101 010 ← n → ← n →	FD 2A	4	6	20	
LD (nn), HL	(nn + 1) ← H (nn) ← L	•	•	X	•	X	•	•	•	00 100 010 ← n → ← n →	22	3	5	16	
LD (nn), dd	(nn + 1) ← dd_H (nn) ← dd_L	•	•	X	•	X	•	•	•	11 101 101 01 dd0 011 ← n → ← n →	ED	4	6	20	
LD (nn), IX	(nn + 1) ← IX_H (nn) ← IX_L	•	•	X	•	X	•	•	•	11 011 101 00 100 010 ← n → ← n →	DD 22	4	6	20	
LD (nn), IY	(nn + 1) ← IY_H (nn) ← IY_L	•	•	X	•	X	•	•	•	11 111 101 00 100 010 ← n → ← n →	FD 22	4	6	20	

Notes : dd is any of the register pairs BC, DE, HL, SP.
qq is any of the register pairs AF, BC, DE, HL.
$(PAIR)_H$, $(PAIR)_L$ refer to high order and low order eight bits of the register pair respectively,
e.g., BC_L = C, AF_H = A.

INSTRUCTION SET (continued)

16-BIT LOAD GROUP (continued)

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210					
LD SP, HL	$SP \leftarrow HL$	•	•	X	•	X	•	•	•	11 111 001	F9	1	1	6	
LD SP, IX	$SP \leftarrow IX$	•	•	X	•	X	•	•	•	11 011 101 11 111 001	DD F9	2	2	10	
LD SP, IY	$SP \leftarrow IY$	•	•	X	•	X	•	•	•	11 111 101 11 111 001	FD F9	2	2	10	
PUSH qq	$(SP - 2) \leftarrow qq_L$ $(SP - 1) \leftarrow qq_H$ $SP \rightarrow SP - 2$	•	•	X	•	X	•	•	•	11 qq0 101		1	3	11	qq Pair 00 BC 01 DE 10 HL
PUSH IX	$(SP - 2) \leftarrow IX_L$ $(SP - 1) \leftarrow IX_H$ $SP \rightarrow SP - 2$	•	•	X	•	X	•	•	•	11 011 101 11 100 101	DD E5	2	4	15	
PUSH IY	$(SP - 2) \leftarrow IY_L$ $(SP - 1) \leftarrow IY_H$ $SP \rightarrow SP - 2$	•	•	X	•	X	•	•	•	11 111 101 11 100 101	FD E5	2	4	15	
POP qq	$qq_H \leftarrow (SP + 1)$ $qq_L \leftarrow (SP)$ $SP \rightarrow SP + 2$	•	•	X	•	X	•	•	•	11 qq0 001		1	3	10	
POP IX	$IX_H \leftarrow (SP + 1)$ $IX_L \leftarrow (SP)$ $SP \rightarrow SP + 2$	•	•	X	•	X	•	•	•	11 011 101 11 100 001	DD E1	2	4	14	
POP IY	$IY_H \leftarrow (SP + 1)$ $IY_L \leftarrow (SP)$ $SP \rightarrow SP + 2$	•	•	X	•	X	•	•	•	11 111 101 11 100 001	FD E1	2	4	14	

Notes : dd is any of the register pairs BC, DE, HL, SP.
qq is any of the register pairs AF, BC, DE, HL.
$(PAIR)_H$, $(PAIR)_L$ refer to high order and low order eight bits of the register pair respectively,
e.g., $BC_L = C$, $AF_H = A$.

EXCHANGE, BLOCK TRANSFER, BLOCK SEARCH GROUPS

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210					
EX DE, HL, EX AF, AF' EXX	DE ↔ HL, AF ↔ AF' BC ↔ BC' DE ↔ DE' HL ↔ HL'	•	•	X	•	X	•	•	•	11 101 011 00 001 000 11 011 001	EB 08 D9	1 1 1	1 1 1	4 4 4	Register Bank and Auxiliary Register Bank Exchange
EX (SP), HL	H ↔ (SP + 1) L ↔ (SP)	•	•	X	•	X	•	•	•	11 100 011	E3	1	5	19	
EX (SP), IX	IX_H ↔ (SP + 1) IX_L ↔ (SP)	•	•	X	•	X	•	•	•	11 011 101 11 100 011	DD E3	2	6	23	
EX (SP), IY	IY_H ↔ (SP + 1) IY_L ↔ (SP)	•	•	X	•	X	•	•	•	11 111 101 11 100 011	FD E3	2	6	23	

Notes : 1. If the result of B – 1 is zero the Z flag is set, otherwise it is reset.
2. Z flag is set upon instruction completion only.

INSTRUCTION SET (continued)

EXCHANGE, BLOCK TRANSFER, BLOCK SEARCH GROUPS (continued)

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210					
LDI	(DE) ← (HL) DE ← DE + 1 HL ← HL + 1 BC ← BC − 1	•	•	X	0	X	① ↕	0	•	11 101 101 10 100 000	ED A0	2	4	6	Load (HL) into (DE), increment the pointers and decrement the byte counter (BC)
LDIR	(DE) ← (HL) DE ← DE + 1 HL ← HL + 1 BC ← BC − 1 Repeat Until BC = 0	•	•	X	0	X	① 0	0	•	11 101 101 10 110 000	ED B0	2 2	5 4	21 16	If BC ≠ 0 If BC = 0
LDD	(DE) ← (HL) DE ← DE + 1 HL ← HL + 1 BC ← BC − 1	•	•	X	0	X	① ↕	0	•	11 101 101 10 101 000	ED A8	2	4	16	
LDDR	(DE) ← (HL) DE ← DE + 1 HL ← HL + 1 BC ← BC − 1 Repeat Until BC = 0	•	•	X	0	X	② 0	0	•	11 101 101 10 111 000	ED B8	2 2	5 4	21 16	If BC ≠ 0 If BC = 0
CPI	A ← (HL) HL ← HL + 1 BC ← BC − 1	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 100 001	ED A1	2	4	16	
CPIR	A − (HL) HL ← HL + 1 BC ← BC − 1 Repeat Until A = (HL) or BC = 0	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 110 001	ED B1	2 2	5 4	21 16	If BC ≠ 0 and A ≠ (HL) If BC = 0 or A = (HL)
CPD	A − (HL) HL ← HL + 1 BC ← BC − 1	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 101 001	ED A9	2	4	16	
CPDR	A − (HL) HL ← HL + 1 BC ← BC − 1 Repeat Until A = (HL) or BC = 0	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 111 001	ED B9	2 2	5 4	21 16	If BC ≠ 0 and A ≠ (HL) If BC = 0 or A = (HL)

Notes : ①. If the result of B − 1 is zero the Z flag is set, otherwise it is reset.
②. Z flag is set upon instruction completion only.

INSTRUCTION SET (continued)

8–BIT ARITHMETIC AND LOGICAL GROUP

Symbol	Symbolic Operation	Flags								Opcode			Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76	543	210					
ADD A, r	A ← A + r	↕	↕	X	↕	X	V	0	↕	10	[000]	r		1	1	4	r Reg.
ADD A, n	A ← A + n	↕	↕	X	↕	X	V	0	↕	11	[000]	110		2	2	7	000 B
										←	n	→					001 C
ADD A, (HL)	A ← A + (HL)	↕	↕	X	↕	X	V	0	↕	10	[000]	110		1	2	7	010 D
ADD A, (IX+d)	A ← A + (IX + d)	↕	↕	X	↕	X	V	0	↕	11	011	101	DD	3	5	19	011 E
										10	[000]	110					100 H
										←	d	→					101 L
ADD A, (IY+d)	A ← A + (IY + d)	↕	↕	X	↕	X	V	0	↕	11	111	101	FD	3	5	19	111 A
										10	[000]	110					
										←	d	→					
ADC A, s	A ← A + s + CY	↕	↕	X	↕	X	V	0	↕		[001]						s is any of r, n, (HL), (IX+d), (IY+d) as shown for ADD instruction. The indicated bits replace the [000] in the ADD set above.
SUB s	A ← A − s	↕	↕	X	↕	X	V	1	↕		[010]						
SBC A, s	A ← A − s − CY	↕	↕	X	↕	X	V	1	↕		[011]						
AND s	A ← A ∧ s	↕	↕	X	1	X	P	0	0		[100]						
OR s	A ← A ∨ s	↕	↕	X	0	X	P	0	0		[110]						
XOR s	A ← A ⊕ s	↕	↕	X	0	X	P	0	0		[101]						
CP s	A − s	↕	↕	X	↕	X	V	1	↕		[111]						
INC r	r ← r + 1	↕	↕	X	↕	X	V	0	•	00	r	[100]		1	1	4	
INC (HL)	(HL) ← (HL) + 1	↕	↕	X	↕	X	V	0	•	00	110	[100]		1	3	11	
INC (IX + d)	(IX + d) ←	↕	↕	X	↕	X	V	0	•	11	011	101	DD	3	6	23	
	(IX + d) + 1									00	110	[100]					
										←	d	→					
INC (IY+d)	(IY + d) ←	↕	↕	X	↕	X	V	0	•	11	111	101	FD	3	6	23	
	(IY + d) +1									00	110	[100]					
										←	d	→					
DEC m	m ← m − 1	↕	↕	X	↕	X	V	1	•			[101]					m is any of r, (HL), (IX+d), (IY+d) as shown for INC. DEC same format and states as INC. Replace [100] with [101] in opcode.

INSTRUCTION SET (continued)

GENERAL-PURPOSE ARITHMETIC AND CPU CONTROL GROUPS

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210					
DAA	Converters acc content into packed BCD following add or subtract with packed BCD operands	↕	↕	X	↕	X	P	•	↕	00 100 111	27	1	1	4	Decimal Adjust Accumulator.
CPL	$A \leftarrow \overline{A}$	•	•	X	1	X	•	1	•	00 101 111	2F	1	1	4	Complement Accumulator (one's complement)
NEG	$A \leftarrow 0 - A$	↕	↕	X	↕	X	V	1	↕	11 101 101 01 000 100	ED 44	2	2	8	Negate Acc. (two's complement).
CCF	$CY \leftarrow \overline{CY}$	•	•	X	X	X	•	0	↕	00 111 111	3F	1	1	4	Complement Carry Flag.
SCF	CY ← 1	•	•	X	0	X	•	0	1	00 110 111	37	1	1	4	Set Carry Flag.
NOP	No Operation	•	•	X	•	X	•	•	•	00 000 000	00	1	1	4	
HALT	CPU Halted	•	•	X	•	X	•	•	•	01 110 110	76	1	1	4	
DI*	IFF ← 0	•	•	X	•	X	•	•	•	11 110 011	F3	1	1	4	
EI*	IFF ← 1	•	•	X	•	X	•	•	•	11 111 011	FB	1	1	4	
IM 0	Set Interrupt Mode 0	•	•	X	•	X	•	•	•	11 101 101 01 000 110	ED 46	2	2	8	
IM 1	Set Interrupt Mode 1	•	•	X	•	X	•	•	•	11 101 101 01 010 110	ED 56	2	2	8	
IM2	Set Interrupt Mode 2	•	•	X	•	X	•	•	•	11 101 101 01 011 110	ED 5E	2	2	8	

Notes : IFF indicates the interrupt enable flip-flop.
CY indicates the carry flip-flop.
* indicates interrupts are not sampled at the end of EI or DI.

INSTRUCTION SET (continued)

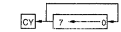
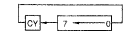
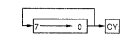
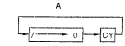
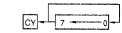
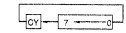
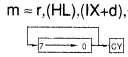
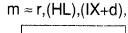
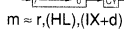
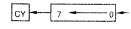
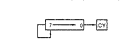
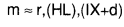
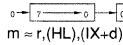
16-BIT ARITHMETIC GROUP

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210					
ADD HL, ss	HL ← HL + ss	•	•	X	X	X	•	0	↕	00 ss1 001		1	3	11	ss Reg.
ADC HL, ss	HL ← HL + ss + CY	↕	↕	X	X	X	V	0	↕	11 101 101	ED	2	4	15	00 BC
										01 ss1 010					01 DE
SBC HL, ss	HL ← HL + ss − CY	↕	↕	X	X	X	V	1	↕	11 101 101	ED	2	4	15	10 HL
										01 ss0 010					11 SP
ADD IX, pp	IX ← IX + pp	•	•	X	X	X	•	0	↕	11 011 101	DD	2	4	15	pp Reg.
										01 pp1 001					00 BC
ADD IY, rr	IY ← IY + rr	•	•	X	X	X	•	0	↕	11 111 101	FD	2	4	15	01 DE
										00 rr1 001					
INC ss	ss ← ss + 1	•	•	X	•	X	•	•	•	00 ss0 011		1	1	6	10 IX
INC IX	IX ← IX + 1	•	•	X	•	X	•	•	•	11 011 101	DD	2	2	10	11 SP
										00 100 011	23				rr Reg.
INC IY	IY ← IY + 1	•	•	X	•	X	•	•	•	11 111 101	FD	2	2	10	00 BC
										00 100 011	23				01 DE
DEC ss	ss ← ss − 1	•	•	X	•	X	•	•	•	00 ss1 011		1	1	6	10 IY
DEC IX	IX ← IX − 1	•	•	X	•	X	•	•	•	11 011 101	DD	2	2	10	11 SP
										00 101 011	2B				
DEC IY	IY ← IY − 1	•	•	X	•	X	•	•	•	11 111 101	FD	2	2	10	
										00 101 011	2B				

Notes : ss is any of the register pairs BC, DE, HL, SP.
pp is any of the register pairs BC, DE, IX, SP.
rr is any of the register pairs BC, DE, IY, SP.

INSTRUCTION SET (continued)

ROTATE AND SHIFT GROUP

Mnemonic	Symbolic Operation	S	Z	H	P/V	N	C	Opcode 76 543 210	Hex	No.of Bytes	No.of M Cycles	No.of T States	Comments
RLCA		•	•	X	0	X	•	0	↕	00 000 111	07	1	1	4	Rotate left circular accumulator.
RLA		•	•	X	0	X	•	0	↕	00 010 111	17	1	1	4	Rotate left accumulator.
RRCA		•	•	X	0	X	•	0	↕	00 001 111	0F	1	1	4	Rotate right circular accumulator.
RRA		•	•	X	0	X	•	0	↕	00 011 111	1F	1	1	4	Rotate right accumulator.
RLC r		↕	↕	X	0	X	P	0	↕	11 001 011 00 [000] r	CB	2	2	8	Rotate left circular register r.
RLC (HL)		↕	↕	X	0	X	P	0	↕	11 001 011 00 [000] 110	CB	2	4	15	r Reg 000 B
RLC (IX + d)		↕	↕	X	0	X	P	0	↕	11 011 101 11 001 011 ← d → 00 [000] 110	DD CB	4	6	23	001 C 010 D 011 E 100 H
RLC (IY + d)	r, (HL), (IX+d), (IY+d)	↕	↕	X	0	X	P	0	↕	11 111 101 11 001 011 ← d → 00 [000] 110	FD CB	4	6	23	101 L 111 A Instruction format
RL m	 m ≈ r,(HL),(IX+d),(iY+d)	↕	↕	X	0	X	P	0	↕	[010]					and states are as shown for RLC's. To form new
RRC.m	 m ≈ r,(HL),(IX+d),(iY+d)	↕	↕	X	0	X	P	0	↕	[001]					opcode replac e [000] or RLC's with shown code.
RR m	 m ≈ r,(HL),(IX+d),(iY+d)	↕	↕	X	0	X	P	0	↕	[011]					
SLA m	 m ≈ r,(HL),(IX+d),(iY+d)	↕	↕	X	0	X	P	0	↕	[100]					
SRA m	 m ≈ r,(HL),(IX+d),(iY+d)	↕	↕	X	0	X	P	0	↕	[101]					
SRL m	 m ≈ r,(HL),(IX+d),(iY+d)	↕	↕	X	0	X	P	0	↕	[111]					
RLD		↕	↕	X	0	X	P	0	•	11 101 101 01 101 111	ED 6F	2	5	18	Rotate digit left and right between the accumulator and location (HL). The contentof the upper half of the accumulator is unaffected.
RRD		↕	↕	X	0	X	P	0	•	11 101 101 01 100 111	ED 67	2	5	18	

INSTRUCTION SET (continued)

BIT SET, RESET AND TEST GROUP

Symbol	Symbolic Operation	Flags							Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210				
BIT b, r	$Z \leftarrow r_b$	X	↕	X	1	X	X	0	•	11 001 011 01 b r	CB	2	2	8	r, r' Reg. 000 B 001 C 010 D 011 E 100 H 101 L 111 A b Bit Tested 000 0 001 1 010 2 011 3 100 4 101 5 110 6 111 7
BIT b, (HL)	$Z \leftarrow \overline{(HL)_b}$	X	↕	X	1	X	X	0	•	11 001 011 01 b 110	CB	2	3	12	
BIT b, (IX+d)$_b$	$Z \leftarrow \overline{(IX+d)_b}$	X	↕	X	1	X	X	0	•	11 011 101 11 001 011 ← d → 01 b 110	DD CB	4	5	20	
BIT b, (IY+d)$_b$	$Z \leftarrow \overline{(IY+d)_b}$	X	↕	X	1	X	X	0	•	11 111 101 11 001 011 ← d → 01 b 110	FD CB	4	5	20	
SET b, r	$r_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 001 011 [11] b r	CB	2	2	8	
SET b, (HL)	$(HL)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 001 011 [11] b 110	CB	2	4	15	
SET b, (IX+d)	$(IX'+d)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 011 101 11 001 011 ← d → [11] b 110	DD CB	4	6	23	
SET b, (IY+d)	$(IY+d)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 111 101 11 001 011 ← d → [11] b 110	FD CB	4	6	23	
RES b, m	$m_b \leftarrow 0$ m ≡ r, (HL), (IX + d), (IY + d)	•	•	X	•	X	•	•	•	[10]					To form new opcode replace [11] of SET b, s with [10]. Flags and time states for SET instruction.

Note : The notation m_b indicates bit b (0 to 7) or location m.

INSTRUCTION SET (continued)

JUMP GROUP

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210					
JP nn	PC ← nn	•	•	X	•	X	•	•	•	11 000 011 ← n → ← n →	C3	3	3	10	
JP cc, nn	If condition cc is true PC ← nn, otherwise continue	•	•	X	•	X	•	•	•	11 cc 010 ← n → ← n →		3	3	10	cc Condition 000 NZ non-zero 001 Z zero 010 NC non-carry 011 C carry 100 PO parity odd 101 PE parity even 110 P sign positive 111 M sign negative
JR e	PC ← PC + e	•	•	X	•	X	•	•	•	00 011 000 ← e–2 →	18	2	3	12	
JR C, e	If C = 0 continue If C = 1 PC ← PC+e	•	•	X	•	X	•	•	•	00 111 000 ← e–2 →	38	2 2	2 3	7 12	If condition not met. If condition is met.
JR NC, e	If C = 1, continue If C = 0, PC ← PC+e	•	•	X	•	X	•	•	•	00 110 000 ← e–2 →	30	2 2	2 3	7 12	If condition not met. If condition is met.
JP Z, e	If Z = 0 continue If Z = 1 PC ← PC+e	•	•	X	•	X	•	•	•	00 101 000 ← e–2 →	28	2 2	2 3	7 12	If condition not met. If condition is met.
JR NZ, e	If Z = 1, continue If Z = 0, PC ← PC+e	•	•	X	•	X	•	•	•	00 100 000 ← e–2 →	20	2 2	2 3	7 12	If condition not met. If condition is met.
JP (HL)	PC ← HL	•	•	X	•	X	•	•	•	11 101 001	E9	1	1	4	
JP (IX)	PC ← IX	•	•	X	•	X	•	•	•	11 011 101 11 101 001	DD E9	2	2	8	
JP (IY)	PC ← IY	•	•	X	•	X	•	•	•	11 111 101 11 101 001	FD E9	2	2	8	
DJNZ, e	B ← B – 1 If B = 0, continue If B ≠ 0, PC ← PC+e	•	•	X	•	X	•	•	•	00 010 000 ← e–2 →	10	2 2	2 3	8 13	If B = 0. If B ≠ 0.

Notes : e represents the extension in the relative addressing mode.
e is a signed two's complement number in the range <– 126, 129>.
e – 2 in the opcode provides an effective address of pc + e as PC is incremented by 2 prior to the addition of e.

INSTRUCTION SET (continued)

CALL AND RETURN GROUP

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210					
CALL nn	$(SP - 1) \leftarrow PC_H$ $(SP - 2) \leftarrow PC_L$ $PC \leftarrow nn$	•	•	X	•	X	•	•	•	11 001 101 ← n → ← n →	CD	3	5	17	
CALL cc, nn	If condition cc is false continue, otherwise same as CALL nn	•	•	X	•	X	•	•	•	11 cc 100 ← n → ← n →		3 3	3 5	10 17	If cc is false. If cc is true.
RET	$PC_L \leftarrow (SP)$ $PC_H \leftarrow (SP + 1)$	•	•	X	•	X	•	•	•	11 001 001	C9	1	3	10	
RET cc	If condition cc is fase continue, otherwise same as RET	•	•	X	•	X	•	•	•	11 cc 000		1 1	1 3	5 11	If cc is false. If cc is true cc Condition 000 NZ Non-zero 001 Z Zero 010 NC Non-carry 011 C Carry 100 PO Parity Odd 101 PE Parity Even 110 P Sign Positive 111 M Sign Negative
RETI	Return from interrupt	•	•	X	•	X	•	•	•	11 101 101 01 001 101	ED 4D	2	4	14	
RETN [1]	Return from non-maskable interrupt	•	•	X	•	X	•	•	•	11 101 101 01 000 101	ED 45	2	4	14	
RSTp	$(SP - 1) \leftarrow PC_H$ $(SP - 2) \leftarrow PC_L$ $PC_H \leftarrow 0$ $PC_L \leftarrow p$	•	•	X	•	X	•	•	•	11 t 111		1	3	11	t p 000 00H 001 08H 010 10H 011 18H 100 20H 101 28H 110 30H 111 38H

Note : 1. RETN loads $IFF_2 \rightarrow IFF_1$.

INSTRUCTION SET (continued)

INPUT AND OUTPUT GRUP

Symbol	Symbolic Operation	Flags								Opcode	Hex	N° of Bytes	N° of M Cycles	N° of T States	Comments
		S	Z		H		P/V	N	C	76 543 210					
IN A, (n)	A ← (n)	•	•	X	•	X	•	•	•	11 011 011 ← n →	DB	2	3	11	n to A_0 ~ A_7 Acc. to A_8 ~ A_{15}
IN r, (C)	r ← (C) If r = 110 only the flags will be affected	↕	↕	X	↕	X	P	0	•	11 101 101 01 r 000	ED	2	3	12	C to A_0 ~ A_7 B to A_8 ~ A_{15}
INI	(HL) ← (C) B ← B – 1 HL ← HL + 1	X	① ↕	X	X	X	X	1	X	11 101 101 10 100 010	ED A2	2	4	16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
INIR	(HL) ← (C) B ← B – 1 HL ← HL + 1 Repeat unil B = 0	X	1	X	X	X	X	1	X	11 101 101 10 110 010	ED B2	2 2	5 (if B≠0) 4 (if B=0)	21 16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
IND	(HL) ← (C) B ← B – 1 HL ← HL – 1	X	① ↕	X	X	X	X	1	X	11 101 101 10 101 010	ED AA	2	4	16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
INDR	(HL) ← (C) B ← B – 1 HL ← HL – 1 Repeat until B = 0	X	1	X	X	X	X	1	X	11 101 101 10 111 010	ED BA	2 2	5 (if B≠0) 4 (if B=0)	21 16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
OUT (n), A	(n) ← A	•	•	X	•	X	•	•	•	11 010 011 ← n →	D3	2	3	11	n to A_0 ~ A_7 Acc.to A_8~A_{15}
OUT (C), r	(C) ← r	•	•	X	•	X	•	•	•	11 101 101 01 r 001	ED	2	3	12	C to A_0 ~ A_7 B to A_8 ~ A_{15}
OUTI	(C) ← (HL) B ← B – 1 HL ← HL + 1	X	① ↕	X	X	X	X	1	X	11 101 101 10 100 011	ED A3	2	4	16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
OTIR	(C) ← (HL) B ← B – 1 HL ← HL + 1 Repeat until B = 0	X	1	X	X	X	X	1	X	11 101 101 10 110 011	ED B3	2 2	5 (if B≠0) 4 (if B=0)	21 16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
OUTD	(C) ← (HL) B ← B – 1 HL ← HL – 1	X	① ↕	X	X	X	X	1	X	11 101 101 10 101 011	ED AB	2	4	16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
OTDR	(C) ← (HL) B ← B – 1 HL ← HL + 1 Repeat until B = 0	X	① 1	X	X	X	X	1	X	11 101 101 10 111 011	ED	2 2	5 (if B≠0) 4 (if B=0)	21 16	C to A_0 ~ A_7 B to A_8 ~ A_{15}

Note : ①. If the result of B – 1 is zero the Z flag is set, otherwise it is reset.

INSTRUCTION SET (continued)

SUMMARY OF FLAG OPERATION

Instruction	D_7 S	Z		H		P/V	N	D_0 C	Comments
ADD A, s ; ADC A, s	↕	↕	X	↕	X	V	0	↕	8-bit Add or Add with Carry.
SUB s; SBC A, s ; CP s ; NEG	↕	↕	X	↕	X	V	1	↕	8-Bit subtract, subtract with carry, compare and negate accumulator.
AND s OR s, XOR s	↕ ↕	↕ ↕	X X	1 0	X X	P P	0 0	0 0	Logical Operations
INC s	↕	↕	X	↕	X	V	0	•	8-bit Increment
DEC s	↕	↕	X	↕	X	V	1	•	8-bit Decrement
ADD DD, ss	•	•	X	X	X	•	0	↕	16-bit Add
ADC HL, ss	↕	↕	X	X	X	V	0	↕	16-bit Add with Carry
SBC HL, ss	↕	↕	X	X	X	V	1	↕	16-bit Subtract with Carry.
RLA, RLCA, RRA ; RRCAA	•	•	X	0	X	•	0	↕	Rotate Accumulator.
RL m ; RLC m ; RR m ; RRC m ; SLA m SRA m ; SRL m	↕	↕	X	0	X	P	0	↕	Rotate and Shift Locations.
RLD ; RRD	↕	↕	X	0	X	P	0	•	Rotate Digit Left and Right
DAA	↕	↕	X	↕	X	P	•	↕	Decimal Adjust Accumulator.
CPL	•	•	X	1	X	•	1	•	Complement Accumulator
SCF	•	•	X	0	X	•	0	1	Set Carry
CCF	•	•	X	X	X	•	0	↕	Complement Carry
IN r, (C)	↕	↕	X	0	X	P	0	•	Input Register Indirect
INI, IND, OUTI ; OUTD INIR ; INDR ; OTIR ; OTDR	X X	↕ 1	X X	X X	X X	X X	1 1	• •	Block Input and Output. Z = 0 if B ≠ otherwise Z = 0
LDI ; LDD LDIR ; LDDR	X X	X X	X X	0 0	X X	↕ 0	0 0	• •	Block Transfer Instructions. P/V = 1 if BC ≠ 0, otherwise P/V = 0
CPI ; CPIR ; CPD ; CPDR	X	↕	X	X	X	↕	1	•	Block Search Instructions. Z = 1 if A = (HL), otherwise Z = 0. P/V = 1 if BC ≠ 0, otherwise P/V = 0.
LD A, I, LD A, R	↕	↕	X	0	X	IFF	0	•	The content of the interrupt enable flip-flop (IFF) is copied into P/V flag.
BIT b, s	X	↕	X	1	X	X	0	•	The state of bit b of location is copied into the Z flag.

INSTRUCTION SET (continued)

SUMMARY OF FLAG OPERATION

Symbol	Operation
S	Sign Flag. S = 1 if the MSB of the result is 1.
Z	Zero Flag. Z = 1 if the result of the operation is 0.
P/V	Parity or Overflow Flag. Parity (P) and overflow (V) share the same flag. Logical operations affect this flag with the parity of the result while arithmetic operations affect this flag with the overflow of the result. If P/V holds parity, P/V = 1 if the result of the operation is even, P/V = 0 if result is odd. If P/V holds overflow, P/V = 1 if the result of the operation produced an overflow.
H	Half-carry Flag. H = 1 if the add or subtract operation produced a carry into or borrow from bit 4 of the accumulator.
N	Add/Subtract Flag. N = 1 if the previous operation was a subtract.
H & N	H and N flags are used in conjunction with the decimal adjust instruction (DAA) to properly correct the result into packed BCD format following addition or subtraction using operands with packed BCD format.
C	Carry/Link Flag. C = 1 if the operation produced a carry from the MSB of the operand or result.
↕	The flag is affected according to the result of the operation.
•	The flag is unchanged by the operation.
0	The flag is reset by the operation.
1	The flag is set by the operation.
X	The flag is a "don't care".
V	P/V flag affected according to the overflow result of the operation.
P	P/V flag affected according to the parity result of the operation.
r	Any one of the CPU Registers A, B, C, D, E, H, L.
s	Any 8-bit location for all the addressing modes allowed for the particular instruction.
ss	Any 16-bit location for all the addressing modes allowed for that instruction.
ii	Any one of the two Index registers IX or IY.
R	Refresh Counter
n	8-bit Value in Range < 0.255 >
nn	16-bit Value in Range < 0.65535 >

PIN DESCRIPTIONS

A_0-A_{15}. *Address Bus* (Output, Active High, 3-state). A_0-A_{15} form a 16-bit address bus. The Address Bus provides the address for memory data bus exchanges (up to 64K bytes) and for I/O device exchanges.

$\overline{\text{BUSACK}}$. *Bus Acknowledge* (Output, Active Low). Bus Acknowledge indicates to the requesting device that the CPU address bus, data bus, and control signals MREQ, IORQ, RD, and WR have entered their high-impedance states. The external circuitry can now control these lines.

$\overline{\text{BUSREQ}}$. *Bus Request* (Input, Active Low). Bus Request has a higher priority than NMI and is always recognized at the end of the current machine cycle. BUSREQ forces the CPU address bus, data bus, and control signals MREQ, IORQ, RD, and WR to go to a high-impedance state so that other devices can control these lines. BUSREQ is normally wire-ORed and requires an external pullup for these applications. Extended BUSREQ periods due to extensive DMA operations can prevent the CPU from properly refreshing dynamic RAMs.

D_0-D_7. *Data Bus* (Input/Output, Active High, 3-state). D_0-D_7 constitute an 8-bit bidirectional data bus, used for data exchanges with memory and I/O.

$\overline{\text{HALT}}$. *Halt State* (Output, Active Low). $\overline{\text{HALT}}$ indicates that the CPU has executed a Halt instruction and is awaiting either a non-maskable or a maskable interrupt (with the mask enabled) before operation can resume. While halted, the CPU executes NOPs to maintain memory refresh.

$\overline{\text{INT}}$. *Interrupt Request* (Input, Active Low). Interrupt Request is generated by I/O devices. The CPU honors a request at the end of the current instruction if the internal software-controlled interrupt enable flip-flop (IFF) is enabled. $\overline{\text{INT}}$ is normally wire-ORed and requires an external pullup for these applications.

$\overline{\text{IORQ}}$. *Input/Output Request* (Output, Active Low, 3-state). IORQ indicates that the lower half of the address bus holds a valid I/O address for an I/O read or write operation. IORQ is also generated concurrently with M1 during an interrupt acknowledge cycle to indicate that an interrupt response vector can be placed on the data bus.

$\overline{\text{M1}}$. *Machine Cycle One* (Output, Active Low). $\overline{\text{M1}}$, together with MREQ, indicates that the current machine cycle is the opcode fetch cycle of an instruction execution. M1, together with IORQ, indicates an interrupt acknowledge cycle.

$\overline{\text{MREQ}}$. *Memory Request* (Output, Active Low, 3-state). MREQ indicates that the address bus holds a valid address for a memory read or memory write operation.

$\overline{\text{NMI}}$. *Non-Maskable Interrupt* (Input, negative edge-triggered). NMI has a higher priority than $\overline{\text{INT}}$. NMI is always recognized at the end of the current instruction, independent of the status of the interrupt enable flip-flop, and automatically forces the CPU to restart at location 0066H.

$\overline{\text{RD}}$. *Read* (Output, Active Low, 3-state). $\overline{\text{RD}}$ indicates that the CPU wants to read data from memory or an I/O device. The addressed I/O device or memory should use this signal to gate data onto the CPU data bus.

$\overline{\text{RESET}}$. *Reset* (Input, Active Low). $\overline{\text{RESET}}$ initializes the CPU as follows : it resets the interrupt enable flip-flop, clears the PC and Registers I and R, and sets the interrupt status to Mode 0. During reset time, the address and data bus go to a high-impedance state, and all control output signals go to the inactive state.

Note that $\overline{\text{RESET}}$ must be active for a minimum of three full clock cycles before the reset operation is complete.

$\overline{\text{RFSH}}$. *Refresh* (Output, Active Low). $\overline{\text{RFSH}}$, together with MREQ, indicates that the lower seven bits of the system's address bus can be used as a refresh address to the system's dynamic memories.

$\overline{\text{WAIT}}$. *Wait* (Input, Active Low). $\overline{\text{WAIT}}$ indicates to the CPU that the addressed memory or I/O devices are not ready for a data transfer. The CPU continues to enter a Wait state as long as this signal is active. Extended $\overline{\text{WAIT}}$ periods can prevent the CPU from refreshing dynamic memory properly.

$\overline{\text{WR}}$. *Write* (Output, Active Low, 3-state). $\overline{\text{WR}}$ indicates that the CPU data bus holds valid data to be stored at the addressed memory or I/O location.

CPU TIMING

The Z80 CPU executes instructions by proceeding through a specific sequence of operations :

- Memory read or write
- I/O device read or write
- Interrupt acknowledge

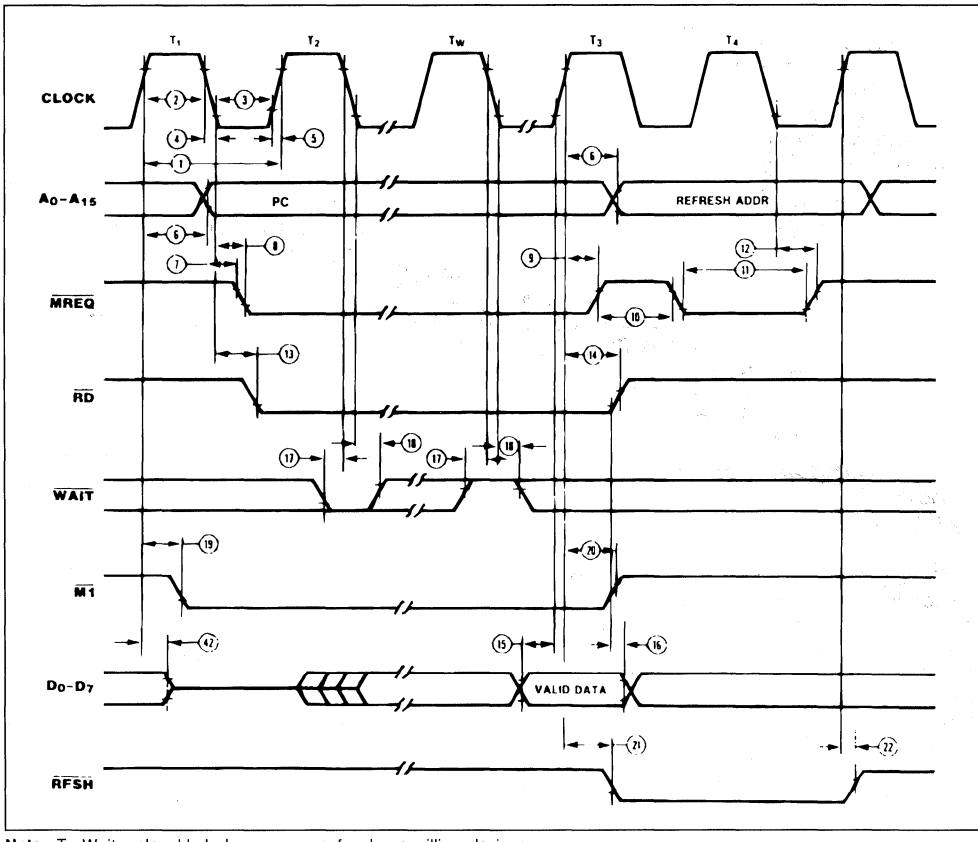
The basic clock period is referred to as a T time or cycle, and three or more T cycles make up a machine cycle (M1, M2 or M3 for instance). Machine cycles can be extended either by the CPU automatically inserting one or more Wait states or by the insertion of one or more Wait states by the user.

INSTRUCTION OPCODE FETCH

The CPU places the contents of the Program Counter (PC) on the address bus at the start of the cycle (figure 5). Approximately one-half clock cycle later, $\overline{MREQ}$ goes active. When active, $\overline{RD}$ indicates that the memory data can be enabled onto the CPU data bus.

The CPU samples the $\overline{WAIT}$ input with the falling edge of clock state T_2. During clock states T_3 and T_4 of an M1 cycle dynamic RAM refresh can occur while the CPU starts decoding and executing the instruction. When the Refresh Control signal becomes active, refreshing of dynamic memory can take place.

Figure 5 : Instruction Opcode Fetch.



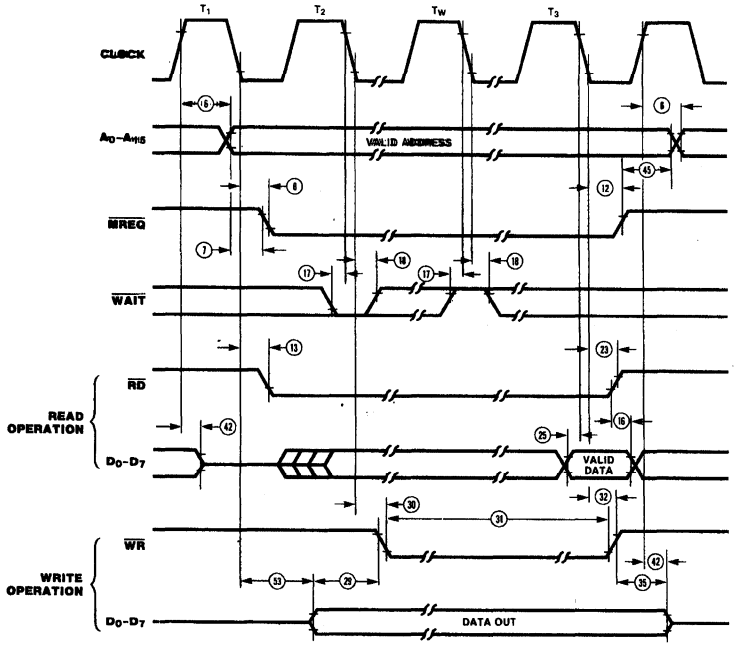
Note : T_W-Wait cycle added when necessary for slow ancilliary devices.

CPU TIMING (continued)

MEMORY READ OR WRITE CYCLES

Figure 6 shows the timing of memory read or write cycles other than an opcode fetch (M1) cycle. The MREQ and RD signals function exactly as in the fetch cycle. In a memory write cycle, $\overline{MREQ}$ also becomes active when the address bus is stable. The $\overline{WR}$ line is active when the data bus is stable, so that it can be used directly as an R/$\overline{W}$ pulse to most semiconductor memories.

Figure 6 : Memory Read or Write Cycles.



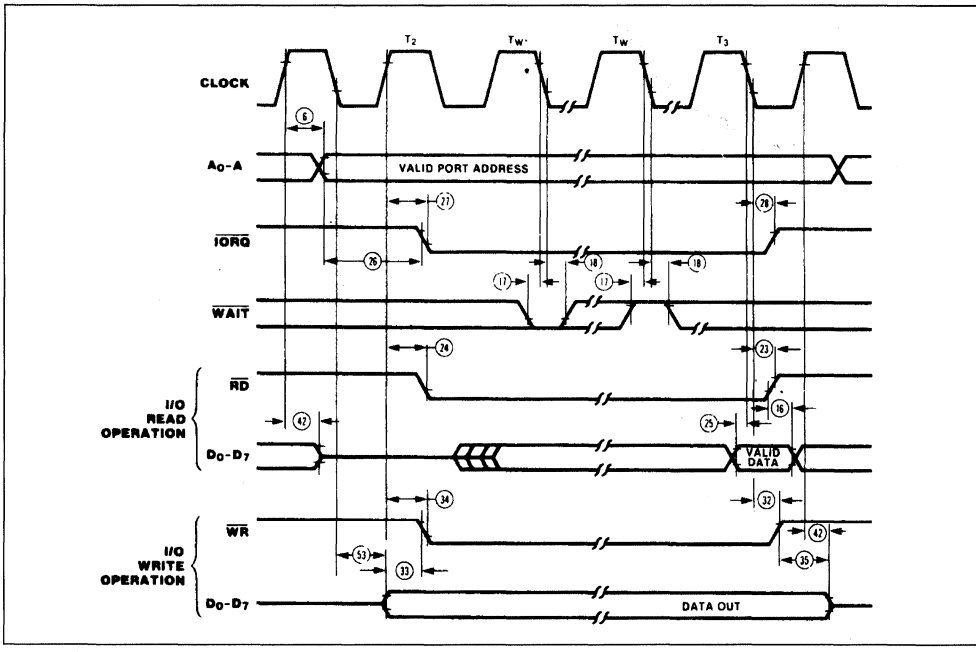
CPU TIMING (continued)

INPUT OR OUTPUT CYCLES

Figure 7 shows the timing for an I/O read or I/O write operation. During I/O operations, the CPU automatically inserts a single Wait state (T_W).

This extra Wait state allows sufficient time for an I/O port to decode the address from the port address lines.

Figure 7 : Input or Output Cycles.



Note : T_W* = One wait cycle automatically inserted by CPU.

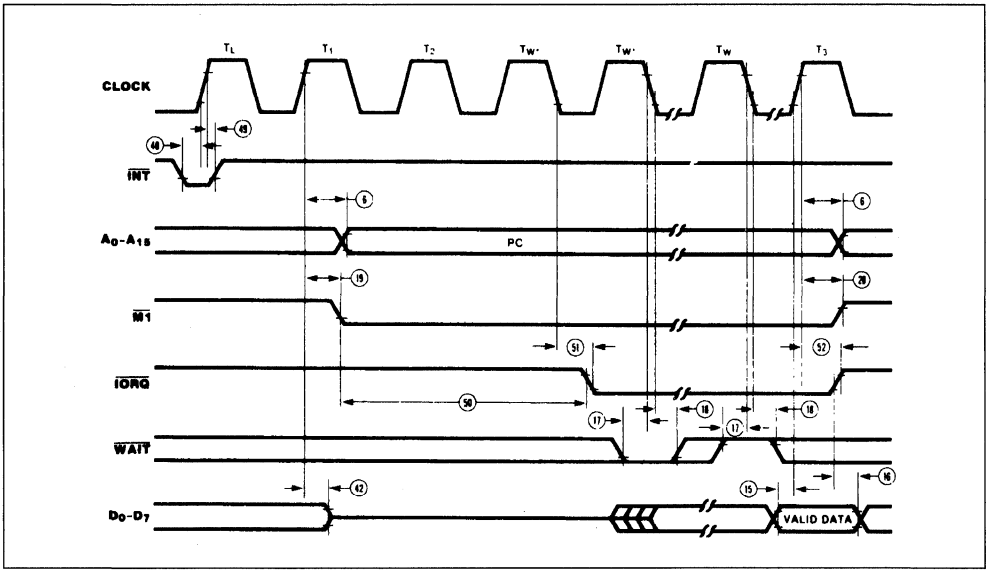
CPU TIMING (continued)

INTERRUPT REQUEST/ACKNOWLEDGE CYCLE

The CPU samples the interrupt signal with the rising edge of the last clock cycle at the end of any instruction (figure 8). When an interrupt is accepted, a special $\overline{M1}$ cycle is generated.

During this $\overline{M1}$ cycle, $\overline{IORQ}$ becomes active (instead of $\overline{MREQ}$) to indicate that the interrupting device can place an 8-bit vector on the data bus. The CPU automatically adds two Wait states to this cycle.

Figure 8 : Interrupt Request/Acknowledge Cycle.



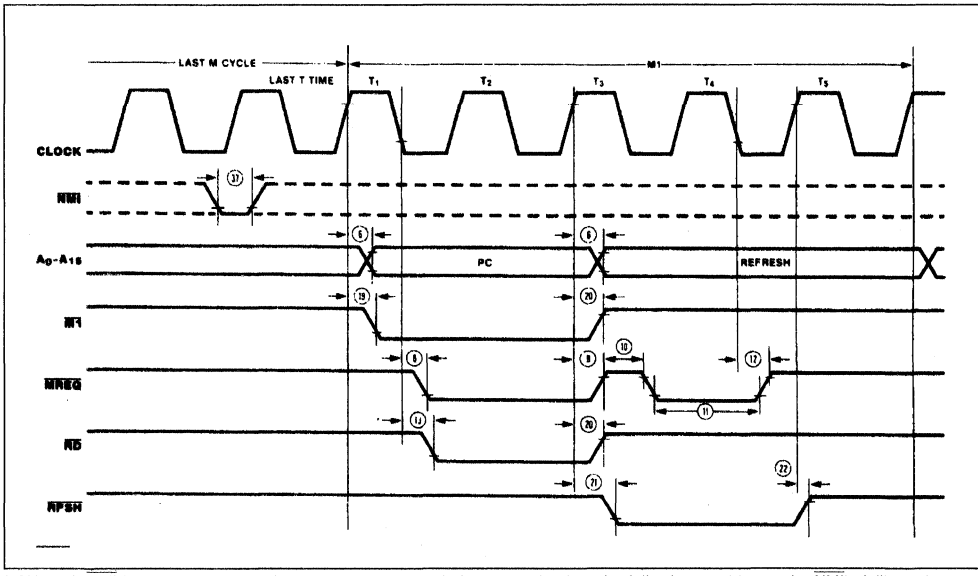
Notes : 1. TL = Last state of previous instruction.
2. Two Wait cycles automatically inserted by CPU (*).

CPU TIMING (contiued)

NON-MASKABLE INTERRUPT REQUEST CYCLE

$\overline{NMI}$ is sampled at the same time as the maskable interrupt input $\overline{INT}$ but has higher priority and cannot be disabled under software control. The subsequent timing is similar to that of a normal instruction fetch except that data put on the bus by the memory is ignored. The CPU instead executes a restart (RST) operation and jumps to the $\overline{NMI}$ service routine located at address 0066H (figure 9).

Figure 9 : Non-Maskable Interrupt Request Operation.



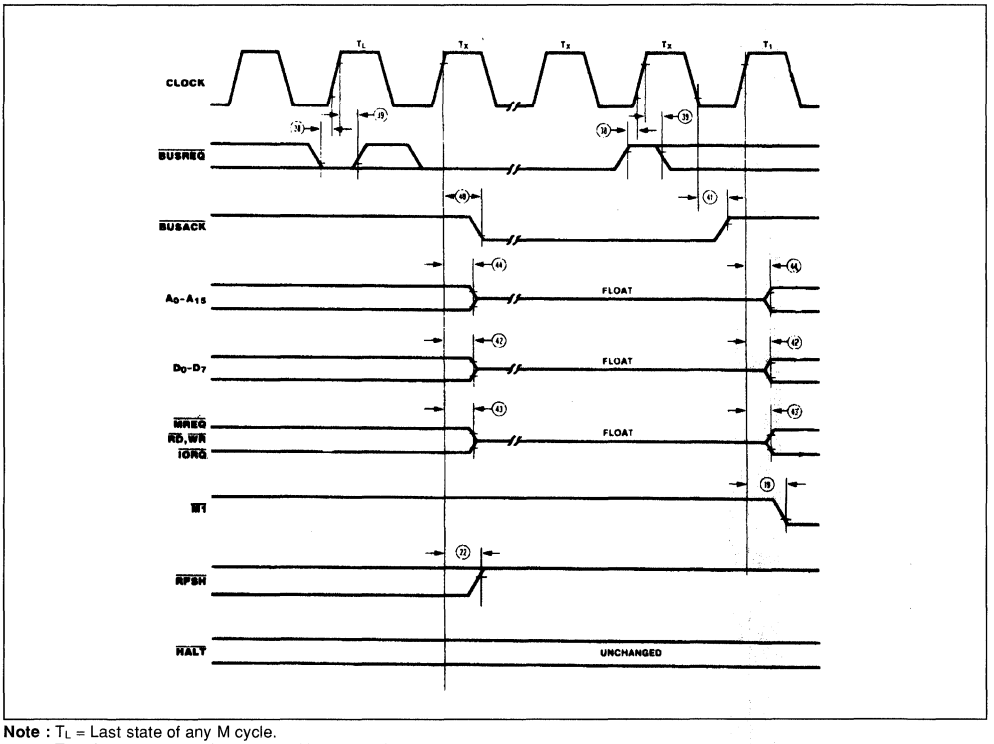
* Although $\overline{NMI}$ is an asynchronous input, to guarantee its being recognized on the following machine cycle, $\overline{NMI}$'s falling edge must occur no later than the rising edge of the clock cycle preceding T_{LAST}.

CPU TIMING (continued)

BUS REQUEST/ACKNOWLEDGE CYCLE

The CPU samples $\overline{BUSREQ}$ with the rising edge of the last clock period of any machine cycle (figure 10). If $\overline{BUSREQ}$ is active, the CPU sets its address, data, and $\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$, and $\overline{WR}$ lines to a high-impedance state with the rising edge of the next clock pulse. At that time, any external device can take control of these lines, usually to transfer data between memory and I/O devices.

Figure 10 : Z-Bus Request/Acknowledge Cycle.



Note : T_L = Last state of any M cycle.
T_X = An arbitrary clock cycle used by requesting device.

CPU TIMING (continued)

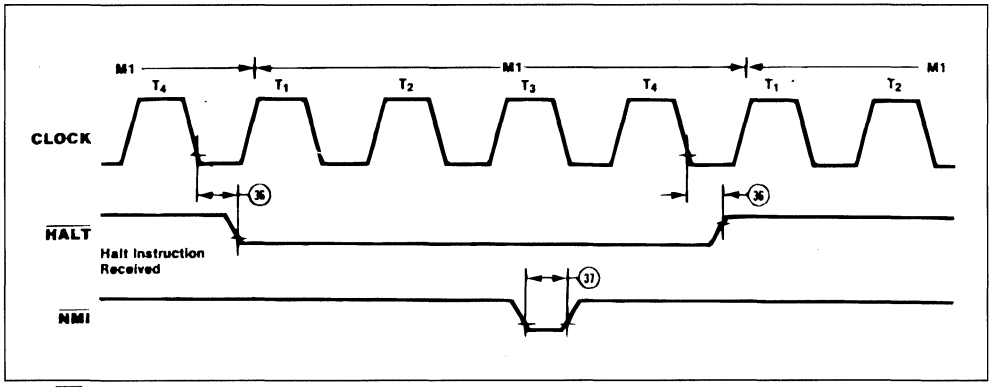
HALT ACKNOWLEDGE CYCLE

When the CPU receives an Halt instruction, it executes NOP states until either an INT or NMI input is received. When in the Halt state, the HALT output is active and remains so until an interrupt is received (figure 11).

RESET CYCLE

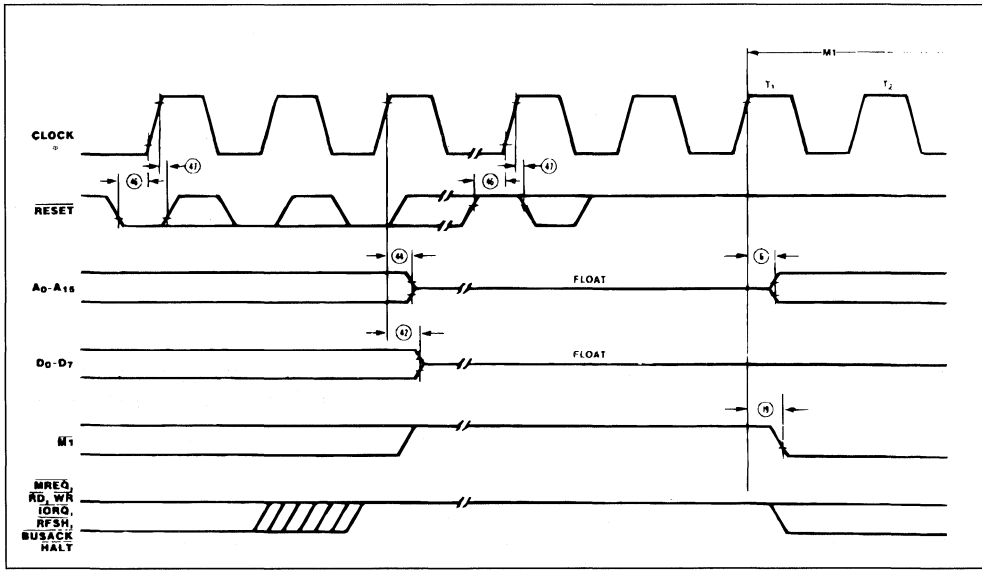
RESET must be active for at least three clock cycles for the CPU to properly accept it. As long as RESET remains active, the address and data buses float, and the control outputs are inactive. Once RESET goes inactive, three internal T cycles are consumed before the CPU resumes normal processing operation. RESET clears the PC register, so the first opcode fetch will be to location 0000 (figure 12).

Figure 11 : Halt Acknowledge Cycle.



Note : INT will also force a Halt exit.
* See note, Figure 9.

Figure 12 : Reset Cycle.



AC CHARACTERISTICS

N°	Symbol	Parameter	Z8400		Z8400A		Z8400B		Z8400H	
			Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
1	TcC	Clock Cycle Time	400*		250*		165*		125*	
2	TwCh	Clock Pulse Width (high)	180*		110*		65*		55*	
3	TwCl	Clock Pulse Width (low)	180	2000	110	2000	65	2000	55	2000
4	TfC	Clock Fall Time		30		30		20		10
5	TrC	Clock Rise Time		30		30		20		10
6	TdCr(A)	Clock ↑ to Address Valid Delay		145		110		90		80
7	TdA (MREQf)	Address Valid to $\overline{MREQ}$ ↓ Delay	125*		65*		35*		20*	
8	TdCf (MREQf)	Clock ↓ to $\overline{MREQ}$ ↓ Delay		100		85		70		60
9	TdCr (MREQr)	Clock ↑ to $\overline{MREQ}$ ↑ Delay		100		85		70		60
10	TwMREQh	$\overline{MREQ}$ Pulse Width (high)	170*		110*		65*		45*	
11	TwMREQl	$\overline{MREQ}$ Pulse Width (low)	360*		220*		135*		100*	
12	TdCf(MREQr)	Clock ↓ to $\overline{MREQ}$ ↑ Delay		100		85		70		60
13	TdCf (RDf)	Clock ↓ to $\overline{RD}$ ↓ Delay		130		95		80		70
14	TdCr(RDr)	Clock ↑ to $\overline{RD}$ ↑ Delay		100		85		70		60
15	TsD(Cr)	Data Setup Time to Clock ↑	50		35		30		30	
16	ThD(RDr)	Data Hold Time to $\overline{RD}$ ↑		0		0		0		0
17	TsWAIT(Cf)	$\overline{WAIT}$ Setup Time to Clock ↓	70		70		60		50	
18	ThWAIT(Cf)	$\overline{WAIT}$ Hold Time to Clock ↓		0		0		0		0
19	TdCr(Mlf)	Clock ↑ to $\overline{M1}$ ↓ Delay		130		100		80		70
20	TdCr(Mlr)	Clock ↑ to $\overline{M1}$ ↑ Delay		130		100		80		70
21	TdCr(RFSHf)	Clock ↑ to $\overline{RFSH}$ ↓ Delay		180		130		110		95
22	TdCr(RFSHr)	Clock ↑ to $\overline{RFSH}$ ↑ Delay		150		120		100		85
23	TdCf(RDr)	Clock ↓ to $\overline{RD}$ ↑ Delay		110		85		70		60
24	TdCr(RDf)	Clock ↑ to $\overline{RD}$ ↓ Delay		110		85		70		60
25	TsD(Cf)	Data Setup to Clock ↓ during M_2, M_3, M_4 or M_5 Cycles	60		50		40		30	
26	TdA(IORQf)	Address Stable Prior to $\overline{IORQ}$ ↓	320*		180*		110*		75*	
27	TdCr(IORQf)	Clock ↑ to $\overline{IORQ}$ ↓ Delay		90		75		65		55
28	TdCf(IORQr)	Clock ↓ to $\overline{IORQ}$ ↑ Delay		110		85		70		60
29	TdCf(WRf)	Data Stable Prior to $\overline{WR}$ ↓	190*		80*		25*		5*	
30	TdDf(WRf)	Clock ↓ to $\overline{WR}$ ↓ Delay		90		80		70		60
31	TwWR	$\overline{WR}$ Pulse Width	360*		220*		135*		100*	
32	TdCf(WRr)	Clock ↓ to $\overline{WR}$ ↑ Delay		100		80		70		60
33	TdD(WRf)	Data Stable Prior to $\overline{WR}$ ↓	20*		− 10*		− 55*		55*	
34	TdCr(WRf)	Clock ↑ to $\overline{WR}$ ↓ Delay		80		65		60		55
35	TdWRr(D)	Data Stable from $\overline{WR}$ ↑	120*		60*		30*		15*	
36	TdCf(HALT)	Clock ↓ to $\overline{HALT}$ ↑ or ↓		300		300		260		225
37	TwNMI	$\overline{NMI}$ Pulse Width	80		80		70		60*	

* For clock periods other than the minimums shown in the table, calculate parameters using the expressions in the table on the following page.

All timings are preliminary and subject to change.

AC CHARACTERISTICS (continued)

N°	Symbol	Parameter	Z8400		Z8400A		Z8400B		Z8400H	
			Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
38	TsBUSREQ(Cr)	$\overline{BUSREQ}$ Setup Time to Clock ↑	80		50		50		40	
39	TcBUSREQ(Cr)	$\overline{BUSREQ}$ Hold Time after Clock ↑	0		0		0		0	
40	TdCr(BUSACKf)	Clock ↑ to $\overline{BUSACK}$ ↓ Delay	–	120		100		90		80
41	TdCf(BUSACKr)	Clock ↓ to $\overline{BUSACK}$ ↑ Delay		110		100		90		80
42	TdCr(Tz)	Clock ↑ to Data Float Delay		90		90		80		70
43	TdCr(CTz)	Clock ↑ to Control Outputs Float Delay (MREQ, IORQ, RD, and WR)		110		80		70		60
44	TdCr(Az)	Clock ↑ to Address Float Delay		110		90		80		70
45	TdCTr(A)	$\overline{MREQ}$ ↑, $\overline{IORQ}$ ↑, $\overline{RD}$ ↑, and $\overline{WR}$ ↑ to Address Hold Time	160*		80*		35*		20*	
46	TsRESET(Cr)	$\overline{RESET}$ to Clock ↑ Setup Time	90		60		60		45	
47	ThRESET(Cr)	$\overline{RESET}$ to Clock ↑ Hold Time		0		0		0		0
48	TsINTf(Cr)	$\overline{INT}$ to Clock ↑ Setup Time	80		80		70		55	
49	ThINTr(Cr)	$\overline{INT}$ to Clock ↑ Hold Time		0		0		0		0
50	TdMIf(IORQf)	$\overline{MI}$ ↓ to $\overline{IORQ}$ ↓ Delay	920*		565*		365*		270*	
51	TdCf(IORQf)	Clock ↓ to $\overline{IORQ}$ ↓ Delay		110		85		70		60
52	TdCf(IORQr)	Clock ↑ to $\overline{IORQ}$ ↑ Delay		100		85		70		60
53	TdCf(D)	Clock ↓ to Data Valid Delay		230		150		130		115

* For clock periods other than the minimums shown in the table, calculate parameters using the expressions on the following table
All timings are preliminary and subject to change.

FOOTNOTES TO AC CHARACTERISTICS

N°	Symbol	Z8400	Z8400A	Z8400B
1	TcC	TwCh + TwCl + TrC + TfC	TwCh + TwCl + TrC + TfC	TwCh + TwCl + TrC + TfC
2	TwCh	Although static by design, TwCh of greater than 200 µs is not guaranteed.	Although static by design, TwCh of greater than 200 µs is not guaranteed.	Although static by design, TwCh of greater than 200 µs is not guaranteed.
7	TdA(MREQf)	TwCh + TfC – 75	TwCh + TfC – 65	TwCh + TfC – 50
10	TwMREQh	TwCh + TfC – 30	TwCh + TfC – 20	TwCh + TfC – 20
11	TwMREQl	TcC – 40	TcC – 30	TcC – 30
26	TdA(IORQf)	TcC – 80	TcC – 70	TcC – 55
29	TdD(WRf)	TcC – 210	TcC – 170	TcC – 140
31	TwWR	TcC – 40	TcC – 30	TcC – 30
33	TdD(WRf)	TwCl + TrC – 180	TwCl + TrC – 140	TwCl + TrC – 140
35	TdWRr(D)	TwCl + TrC – 80	TwCl + TrC – 70	TwCl + TrC – 55
45	TdCTr(A)	TwCl + TrC – 40	TwCl + TrC – 50	TwCl + TrC – 50
50	TdMIf(IORQf)	2TcC + TwCh + TfC – 80	2TcC + TwCh + TfC – 65	2TcC + TwCh + TfC – 50

AC Test Conditions :
V_{IH} = 2.0 V
V_{IL} = 0.8 V
$V_{IHC} = V_{CC} - 0.6$ V
V_{ILC} = 0.45 V
V_{OH} = 2.0 V
V_{OL} = 0.8 V
FLOAT = ± 0.5 V

ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
T_{stg}	Storage Temperature Range	– 65 to + 150	°C
T_A	Temperature under Bias Specified Operating Range		
V_I	Voltages on all Inputs and Outputs with respect to GND	– 0.3 + 7.0	V
P_D	Power Dissipation	1.5	W

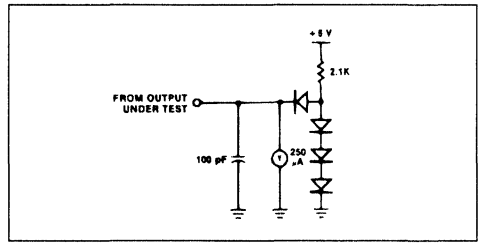
Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only ; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

STANDARD TEST CONDITIONS

The characteristics below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin. Available operating temperature ranges are :

- 0 °C to + 70 °C,
  + 4.75 V ≤ V_{CC} ≤ + 5.25 V
- – 40 °C to + 85 °C,
  + 4.75 V ≤ V_{CC} ≤ + 5.25 V
- – 55 °C to + 125 °C,
  + 4.75 V ≤ V_{CC} ≤ + 5.25 V

All ac parameters assume a load capacitance of 50pF. Add 10ns delay for each 50pF increase in load up to a maximum of 200pF for the data bus and 100pF for address and control lines.



DC CHARACTERISTICS

Symbol	Parameter	Test Conditions	Min.	Max.	Unit
V_{ILC}	Clock Input Low Voltage		− 0.3	0.45	V
V_{IHC}	Clock Input High Voltage		V_{CC} − 0.6	V_{CC} + 0.3	V
V_{IL}	Input Low Voltage		− 0.3	0.8	V
V_{IH}	Input High Voltage		2.0	V_{CC}	V
V_{OL}	Output Low Voltage	I_{OL} = 2 mA		0.4	V
V_{OH}	Output High Voltage	I_{OH} = − 250 µA	2.4		V
I_{CC}	Power Supply Current Z80 Z80A Z80B Z80H			 150 [1] 200 [2] 200 200	 mA mA mA mA
I_{LI}	Input Leakage Current	V_{IN} = 0 to V_{CC}	−	10	µA
I_{LO}	3-State Output Leakage Current in Float	V_{OUT} = 0.4 to V_{CC}	− 10	10	µA

Notes : 1. For military grade parts, I_{CC} is 200 mA.
2. Typical rate for Z8400A is 90 mA.
3. A_{15}-A_0, D_7-D_0, MREQ, IORQ, RD, and WR.

CAPACITANCE

Symbol	Parameter	Note	Min.	Max.	Unit
C_{CLOCK}	Clock Capacitance	Unmeasured pins returned to ground		35	pF
C_{IN}	Input Capacitance			5	pF
C_{OUT}	Output Capacitance			10	pF

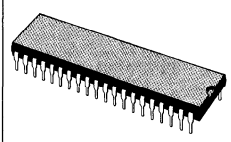
T_A = 25 °C, f = 1 MHz.

ORDERING INFORMATION

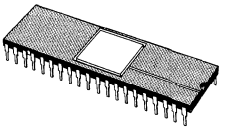
Type	Package	Temp.	Clock	Description
Z8400B1	DIP-40 (plastic)	0/+ 70°C	2.5 MHz	Z80 Central Processing Unit
Z8400F1	DIP-40 (frit seal)	0/+ 70°C		
Z8400D1	DIP-40 (ceramic)	0/+ 70°C		
Z8400D6	DIP-40 (ceramic)	−40/+ 85°C		
Z8400D2	DIP-40 (ceramic)	−55/+125°C		
Z8400C1	PLCC44 (plastic chip-carrier)	0/+ 70°C		
Z8400AB1	DIP-40 (plastic)	0/+ 70°C	4 MHz	
Z8400AF1	DIP-40 (frit seal)	0/+ 70°C		
Z8400AD1	DIP-40 (ceramic)	0/+ 70°C		
Z8400AD6	DIP-40 (ceramic)	−40/+ 85°C		
Z8400AD2	DIP-40 (ceramic)	−55/+125°C		
Z8400AC1	PLCC44 (plastic chip-carrier)	0/+ 70°C		
Z8400BB1	DIP-40 (plastic)	0/+ 70°C	6 MHz	
Z8400BF1	DIP-40 (frit seal)	0/+ 70°C		
Z8400BD1	DIP-40 (ceramic)	0/+ 70°C		
Z8400BD6	DIP-40 (ceramic)	−40/+ 85°C		
Z8400BD2	DIP-40 (ceramic)	−55/+125°C		
Z8400BC1	PLCC44 (plastic chip-carrier)	0/+ 70°C		
Z8400HB1	DIP-40 (plastic)	0/+ 70°C	8 MHz	
Z8400HF1	DIP-40 (frit seal)	0/+ 70°C		
Z8400HD1	DIP-40 (ceramic)	0/+ 70°C		
Z8400HD6	DIP-40 (ceramic)	−40/+ 85°C		
Z8400HC1	PLCC44 (plastic chip-carrier)	0/+ 70°C		

Z80 DMA DIRECT MEMORY ACCESS CONTROL

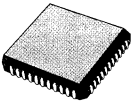
- TRANSFERS, SEARCHES AND SEARCH/TRANSFERS IN BYTE-AT-A-TIME, BURST OR CONTINUOUS MODES. CYCLE LENGTH AND EDGE TIMING CAN BE PROGRAMMED TO MATCH THE SPEED OF ANY PORT.
- DUAL PORT ADDRESSES (sources and destination) GENERATED FOR MEMORY-TO-I/O, MEMORY-TO-MEMORY, OR I/O-TO-I/O OPERATIONS
  ADDRESSES MAY BE FIXED OR AUTOMATICALLY INCREMENTED/DECREMENTED
- NEXT-OPERATION LOADING WITHOUT DISTURBING CURRENT OPERATIONS VIA BUFFERED STARTING ADDRESS REGISTERS. AN ENTIRE PREVIOUS SEQUENCE CAN BE REPEATED AUTOMATICALLY
- EXTENSIVE PROGRAMMABILITY OF FUNCTIONS. CPU CAN READ COMPLETE CHANNEL STATUS
- STANDARD Z80 FAMILY BUS-REQUEST AND PRIORITIZED INTERRUPT-REQUEST DAISY CHAINS IMPLEMENTED WITHOUT EXTERNAL LOGIC. SOPHISTICATED, INTERNALLY MODIFIABLE INTERRUPT VECTORING
- DIRECT INTERFACING TO SYSTEM BUSES WITHOUT EXTERNAL LOGIC



B/F
DIP-40
(Plastic and Frit-Seal)



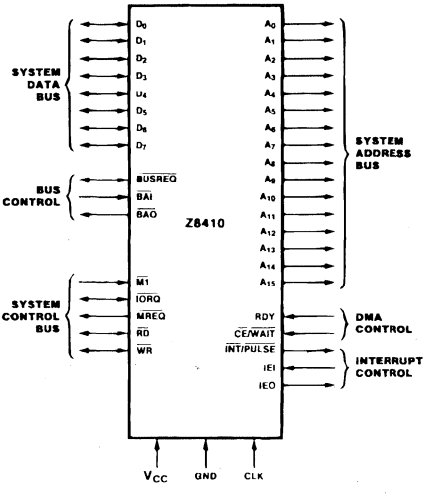
D
DIP-40
(Ceramic)



C
PLCC44
(Plastic)

(Ordering Information at the end of the datasheet)

LOGIC FUNCTIONS



DESCRIPTION

The Z80 DMA (Direct Memory Access) is a powerful ad versatile device for controlling and processing transfers of data. Its basic function of managing CPU-independent transfers between two ports is augmented by an array of features that optimize transfer speed and control with little or no external logic in systems using an 8- or 16-bit data bus and a 16-bit address bus.

Transfers can be done between any two ports (source and destination), including memory-to-I/O, memory-to-memory, and I/O-to-I/O. Dual port addresses are automatically generated for each transaction and may be either fixed or incrementing/decrementing. In addition, bit-maskable byte searches can be performed either concurrently with transfers or as an operation in itself.

The Z80 DMA contains direct interfacing to and independent control of system buses, as well as soph-

isticated bus and interrupt controls. Many programmable features, including variable cycle timing and auto-restart, minimize CPU software overhead. They are especially useful in adapting this special-purpose transfer processor to a broad variety of memory, I/O and CPU environments.

The Z80 DMA is an n-channel silicon-gate depletion-load device packaged in a 40-pin plastic or ceramic DIP. It uses a single + 5 V power supply and the standard Z80 Family single-phase clock.

Figure 1 : Dual in Line Pin Configuration.

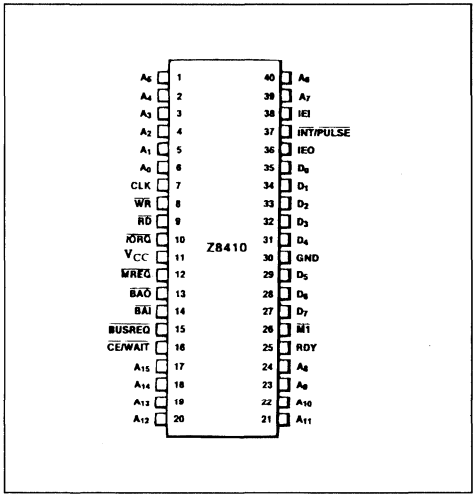
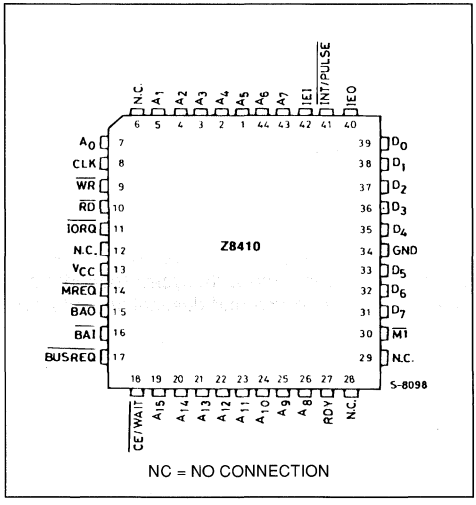


Figure 2 : Chip Carrier Pin Configuration.



FUNCTIONAL DESCRIPTION

Classes of Operation. The Z80 DMA has three basic classes of operation :

- Transfers of data between two ports (memory or I/O peripheral)
- Searches for a particular 8-bit maskable byte at a single port in memory or an I/O peripheral
- Combined transfers with simultaneous search between two ports

Figure 4 illustrates the basic functions served by these classes of operation.

During a transfer, the DMA assumes control of the system address and data buses. Data is read from one addressable port and written to the other addressable port, byte by byte. The ports may be programmed to be either system main memory or peripheral I/O devices. Thus, a block of data may be written from one peripheral to another, from one area of main memory to another, or from a peripheral to main memory and vice versa.

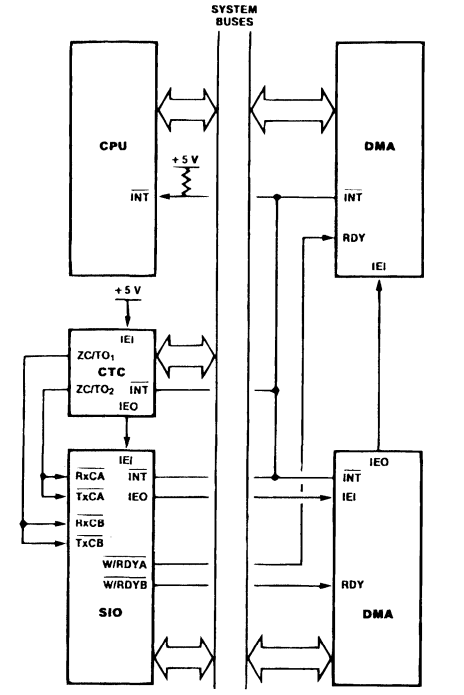
During a search-only operation, data is read from the source port and compared byte by byte with a DMA-internal register containing a programmable match byte. This match byte may optionally be masked so that only certain bits within the match byte are compared.
Search rates up to 1.25M bytes per second can be obtained with the 2.5 MHz Z80 DMA or 2M bytes per second with the 4 MHz Z80 DMA.
In combined searches and transfers, data is transferred between two ports while simultaneously searching for a bit-maskable byte match.
Data transfers or searches can be programmed to stop or interrupt under various conditions. In addition, CPU-readable status bits can be programmed to reflect the condition.

MODES OF OPERATION

The Z80 DMA can be programmed to operate in one of three transfer and/or search modes :

- *Byte-at-a-Time* : data operations are performed one byte at a time. Between each byte operation the system buses are released to the CPU. The buses are requested again for each succeeding byte operation.
- *Burst* : data operations continue until a port's Ready line to the DMA goes inactive. The DMA then stops and releases the system buses after completing its current byte operation.
- *Continuous* : data operations continue until the end of the programmed block of data is reached before the system buses are releases. If a port's Ready line goes inactive before this occurs, the DMA simply pauses until the Ready line comes active again.

Figure 3 : Typical Z80 Environment.



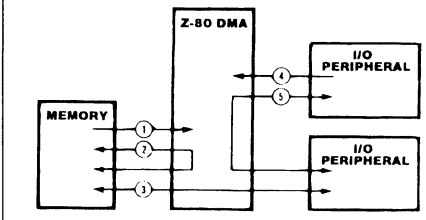
In all modes, once a byte of data is read into the DMA, the operation on the byte will be completed in an orderly fashion, regardless of the state of other signals (including a port's Ready line).

Due to the DMA's high-speed buffered method of reading data, operations on one byte are not completed until the next byte is read in. This means that total transfer or search block lengths must be two or more bytes, and that block lengths programmed into the DMA must be one byte less than the desired block length (cont is N-1 where N is the block length).

COMMANDS AND STATUS

The Z80 DMA has several writable control registers and readable status registers available to the CPU. Control bytes can be written to the DMA whenever the DMA is not controlling the system buses, but the act of writing a control byte to the DMA disables the

Figure 4 : Function of the Z80 DMA.



1. Search memory
2. Transfer memory-to-memory (optional search)
3. Transfer memory-to-I/O (optional search)
4. Search I/O
5. Transfer I/O-to-I/O (optional search)

DMA until it is again enabled by a specific command. Status bytes can also be read at any such time, but writting the Read Status Byte command or the Initiate Read Sequence command disables the DMA.

Control bytes to the DMA include those which effect immediate command actions such as enable, disable, reset, load starting-address buffers, continue, clear counters, clear status bits and the like. In addition, many mode-setting control bytes can be written, including mode and class of operation, port configuration, starting addresses, block length, address counting rule, match and match-mask byte, interrupt conditions, interrupt vector, status-affects-vector condition, pulse counting, auto restart, Ready-line and Wait-line rules, and read mask.

Readable status registers include a general status byte reflecting Ready-line, end-of-block, byte-match and interrupt conditions, as well as 2-byte registers for the current byte count, Port A address and Port B address.

VARIABLE CYCLE

The Z80 DMA has the unique feature of programmable operation-cycle length. This is valuable in tailoring the DMA to the particular requirements of other system components (fast or slow) and maximizes the data-transfer rate. It also eliminates external logic for signal conditioning.

There are two aspects to the variable cycle feature. First, the entire read and write cycles (periods) associated with the source and destination ports can be independently programmed as 2, 3 or 4 T-cycles long (more if Wait cycles are used), thereby increasing or decreasing the speed with which all DMA signals change (figure 5).

Second, the four signals in each port specifically associated with transfers of data (I/O Request, Memory Request, Read, and Write) can each have its active trailing edge terminated one-half T-cycle early. This adds a further dimension of flexibility and speed, allowing such things as shorter-than-normal Read or Write signals that go inactive before data starts to change.

ADDRESS GENERATION

Two 16-bit addresses are generated by the Z80 DMA for every transfer operation, one address for the source port and another for the destination port. Each address can be either variable or fixed. Variable addresses can increment or decrement from the programmed starting address. The fixed-address capability eliminates the need for separate enabling wires to I/O ports.

Port addresses are multiplexed onto the system address bus, depending on whether the DMA is reading the source port or writing to the destination port. Two readable address counters (2 bytes each) keep the current address of each port.

AUTO RESTART

The starting addresses of either port can be reloaded automatically at the end of a block. This option is selected by the Auto Restart control bit. The byte counter is cleared when the addresses are reloaded.

The Auto Restart feature relieves the CPU of software overhead for repetitive operations such as CRT refresh and many others. Moreover, when the CPU has access to the buses during byte-at-a-time or burst transfers, different starting addresses can be written into buffer registers during transfers, causing the Auto Restart to begin at a new location.

INTERRUPTS

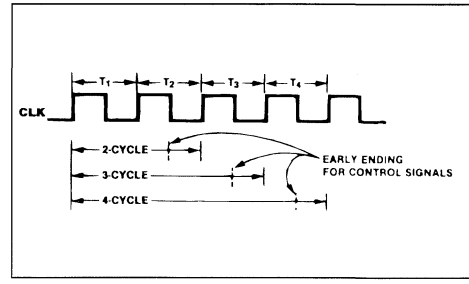
The Z80 DMA can be programmed to interrupt the CPU on three conditions :

- Interrupt on Ready (before requesting bus)
- Interrupt on Match
- Interrupt on End of Block

Any of these interrupts cause an interrupt-pending status bit to be set, and each of them can optionally alter the DMA's interrupt vector. Due to the buffered constraint mentioned under "Modes of Operation", interrupts on Match at End of Block are caused by matches to the byte just prior to the last byte in the block.

The DMA shares the Z80 Family's elaborate interrupt scheme, which provides fast interrupt service in real-time applications. In a Z80 CPU environment,

Figure 5 : Variable Cycle Lenght..



the DMA passes its internally modifiable 8-bit interrupt vector to the CPU, which adds an additional eight bits to form the memory address of the interrupt routine table. This table contains the address of the beginning of the interrupt routine itself. In this process ; CPU control is transferred directly to the interrupt routine, so that the next instruction executed after an interrupt acknowledge is the first instruction of the interrupt routine itself.

PULSE GENERATION

External devices can keep track of how many bytes have been transferred by using the DMA's pulse output, which provides a signal at 256-byte intervals. The interval sequence may be offset at the beginning by 1 to 255 bytes.

The Interrupt line outputs the pulse signal in a manner that prevents misinterpretation by the CPU as an interrupt request, since it only appears when the Bus Request and Bus Acknowledge lines are both active.

PIN DESCRIPTIONS

A_0-A_{15}. *System Address Bus* (Output, 3-state). Addresses generated by the DMA are sent to both source and destination ports (main memory or I/O peripherals) on these lines.

$\overline{BAI}$. *Bus Acknowledge In* (Input, Active Low). Signals that the system buses have been released for DMA control. In multiple-DMA configurations, the BAI pin of the highest priority DMA is normally connected to the Bus Acknowledge pin of the CPU. Lower-priority DMAs have their BAI connected to the BAO of a higher-priority DMA.

$\overline{BAO}$. *Bus Acknowledge Out* (Output, Active Low). In a multiple-DMA configuration, this pin signals that non other higher-priority DMA has requested the system buses. BAI and BAO from daisy chain for multiple-DMA priority resolution over bus control.

BUSREQ. *Bus Request* (Bidirectional, Active Low, Open Drain). As and output, it sends requests for control of the system address bus, data bus and control bus to the CPU. As an input, when multiple DMAs are strung together in a priority daisy chain via BAI and BAO, it senses when another DMA has requested the buses and causes this DMA to refrain from bus requesting until the other DMA is finished. Because it is a bidirectional pin, there cannot be any buffers between this DMA and any other DMA. It can, however, have a buffer between it and the CPU because it is unidirectional into the CPU. A pull-up resistor is connected to this pin.

CE/WAIT. *Chip Enable and Wait* (Input, Active Low). Normally this functions only as a CE line, but it can also be programmed to serve a WAIT function. As a CE line from the CPU, it becomes active when WR and IORQ are active and the I/O port address on the system address bus is the DMA's address, thereby allowing a transfer of control or command bytes from the CPU to the DMA. As a WAIT line from memory or I/O devices, after the DMA has received a bus-request acknowledge from the CPU, it causes wait states to be inserted in the DMA's operation cycles thereby slowing the DMA to a speed that matches the memory or I/O device.

CLK. System Clock (Input). Standard Z80 single-phase clock at 2.5MHz (Z80 DMA) or 4.0MHz (Z80 DMA). For slower system clocks, a TTL gate with a pullup resistor may be adeguate to meet the timing and voltage level specification. For higher-speed systems, use a clock driver with an active pullup to meet the V_{IH} specification and risetime requirements. In all cases there should be a resistive pullup to the power supply of 10Kohms (max) to ensure proper power when the DMA is reset.

D_0-D_7. *System Data Bus* (Bidirectional, 3-state). Commands from the CPU, DMA status, and data from memory or I/O peripherals are transferred on these lines.

IEI. *Interrupt Enable In* (Input, Active High). This is used with IEO to from 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 DMA. Thus, this signal block lower-priority devices from interrupting while a higher-priority device is being serviced by its CPU interrupt service routine.

INT/PULSE. *Interrupt Request* (Output, Active Low, Open Drain). This requests a CPU interrupt. The CPU acknowledges the interrupt by pulling its IORQ output Low during an M1 cycle. It is typically connected to the INT pin of the CPU with a pullup resistor and tied to all other INT pins in the system. This pin can also be used to generate periodic pulses to an external device. It can be used this way only when the DMA is bus master (i.e., the CPU's BUSREQ and BUSACK lines are both Low and the CPU cannot see interrupts).

IORQ. *Input/Output Request* (Bidirectional ; Active Low, 3-state). As an input, this indicates that the lower half of the address bus holds a valid I/O port address for transfer of control or status bytes from or to the CPU, respectively ; this DMA is the addressed port if its CE pin and its WR or RD pins are simultaneously active. As an output, after the DMA has taken control of the system buses, it indicates that the 8-bit or 16-bit address bus holds a valid port address for another I/O device involved in a DMA transfer of data. When IORQ and M1 are both active simultaneously, an interrupt acknowledge is indicated.

M1. *Machine Cycle One* (Input, Active Low). Indicates that the current CPU machine cycle is an instruction fetch. It is used by the DMA to decode the return-from-interrupt instruction (RETI) (ED-4D) sent by the CPU. During two-byte instruction fetches, M1 is active as each opcode byte is fetched. An interrupt acknowledge is indicated when both M1 and IORQ are active.

MREQ. *Memory Request* (Output, Active Low, 3-state). This indicates that the address bus holds a valid address for a memory read or write operation. After the DMA has taken control of the system buses, it indicates a DMA transfer request from or to memory.

RD. *Read* (Bidirectional, Active Low, 3-state). As an input, this indicates that the CPU wants to read status bytes from the DMA's read registers. As an output, after the DMA has taken control of the system buses, it indicates a DMA-controlled read from a memory or I/O port address.

RDY. *Ready* (Input, Programmable Active Low or High). This is monitored by the DMA to determine when a peripheral device associated with a DMA port is ready for a read or write operation. Depending on the mode of DMA operation (Byte, Burst or Continuous), the RDY line indirectly controls DMA activity by causing the BUSREQ line to go Low or High.

WR. *Write* (Bidirectional, Active Low, 3-state). As and input, this indicates that the CPU wants to write control or command bytes to the DMA write registers. As an output, after the DMA has taken control of the system buses, it indicates a DMA-controlled write to a memory or I/O port address.

INTERNAL STRUCTURE

The internal structure of the Z80 DMA includes driver and receiver circuitry for interfacing with an 8-bit system data bus, a 16-bit system address bus, and system control lines (figure 6). In a Z80 CPU environment, the DMA can be tied directly to the analogous pins on the CPU (figure 7) with no additional buffering, except for the CE/WAIT line.

The DMA's internal data bus interfaces with the system data bus and services all internal logic and registers. Addresses generated from this logic for Ports A and B (source and destination) of the DMA's single transfer channel are multiplexed onto the system address bus.

Specialized logic circuits in the DMA are dedicated to the various functions of external bus interfacing, internal bus control, byte matching, byte counting, periodic pulse generation, CPU interrupts, bus request, and address generation. A set of twenty-one writable control registers and seven readable status registers provides the means by which the CPU governs and monitors the activities of these logic circuits. All registers are eight bits wide, with

Figure 6 : Block Diagram.

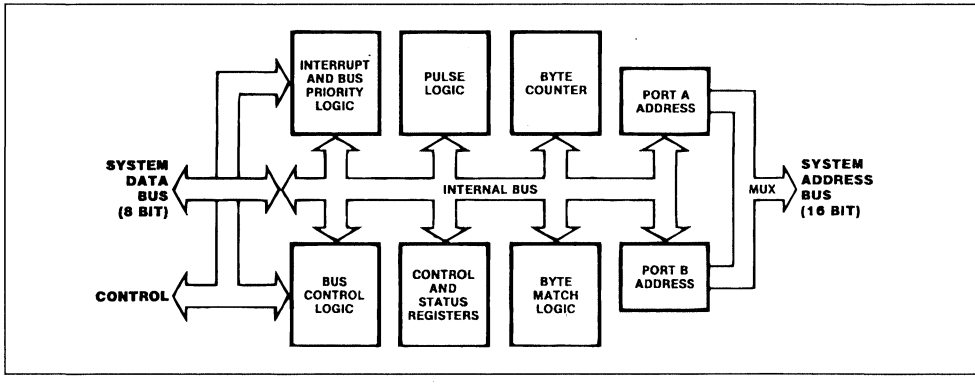
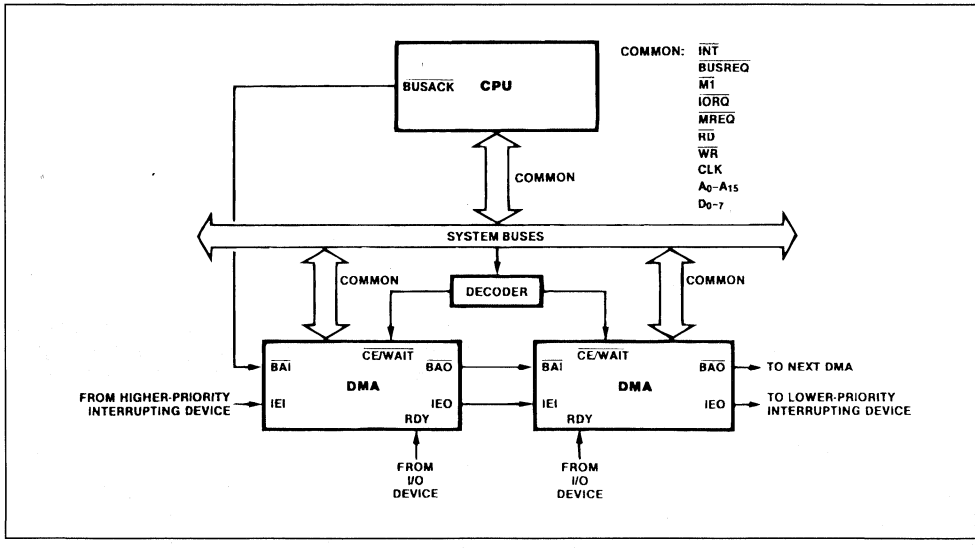


Figure 7 : Multiple-DMA Interconnection to the Z80 CPU.



double-byte information stored in adjacent registers. The two address counters (two bytes each) for Ports A and B are buffered by the two stating addresses.

The 21 writable control register are organized into seven base-register groups, most of which have multiple registers. The base registers in each writable group contain both control/command bits and pointer bits that can be set to address other registers within the group. The seven readable status registers have no analogous second-level registers.

The registers are designated as follows, according to their base-register groups :

WR0-WR6 - Write Register groups 0 through 6 (7 base registers plus 14 associated registers)

RR0-RR6 - Read Registers 0 through 6

Writing to a register within a write-register group involves first writing to the base register, with the appropriate pointer bits set, then writing to one or more of the other register within the group. All seven of the readable status registers are accessed sequentially according to a programmable mask contained in one of the writable registers. The section entitled "Programming" explains this in more detail.

A pipelining scheme is used for reading data in. The programmed block length is the number of bytes compared to the byte counter, which increments at the end of each cycle. In searches, data byte comparisons with the match byte are made during the read cycle of the next byte. Matches are, therefore, discovered only after the next byte is read in.

In multiple-DMA configurations, interrupt request daisy chains are priorized by the order in which their IEI and IEO lines are connected. The system bus, however, may not be pre-empted.

Any DMA that gains access to the system bus keeps the bus until it is finished.

PROGRAMMING

The Z80 DMA has two programmable fundamental states : (1) an enabled state, in which it can gain control of the system buses and direct the transfer of data between ports, and (2) a disabled state, in which it can initiate neither bus requests nor data transfers. When the DMA is powered up or reset by any means, it is automatically placed into the disabled state. Program commands can be written to it by the CPU in either state, but this automatically puts the DMA in the disabled state, which is maintained until an enable command is issued by the CPU. The CPU must program the DMA in advance of any data search or transfer by addressing it as an I/O port and sending a sequence of control bytes using an Output instruction (such as OTIR for the Z80 CPU).

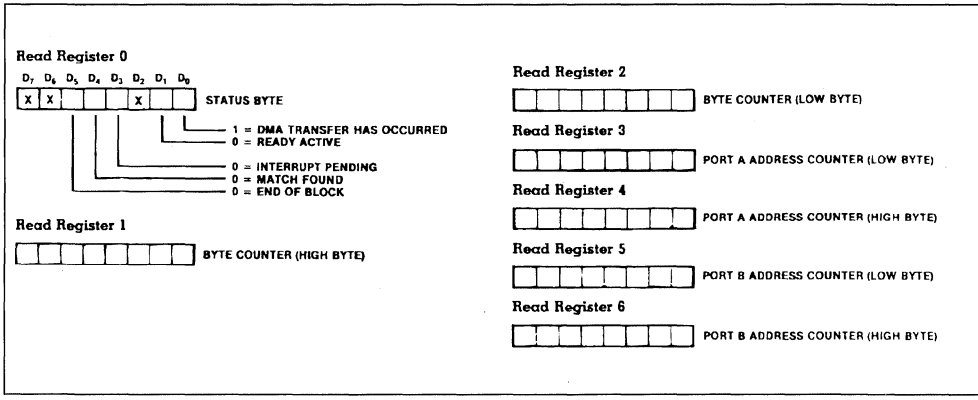
Write Registers	
WR0	Base Register Byte
	Port A starting Address (low byte)
	Port A starting Address (high byte)
	Block Length (low byte)
	Block Length (high byte)
WR1	Base Register Byte
	Port A Variable-timing Byte
WR2	Base Register Byte
	Port B Variable-timing Byte
WR3	Base Register Byte
	Mask Byte
	Match Byte
WR4	Base Register Byte
	Port B starting Address (low byte)
	Port B starting Address (high byte)
	Interrupt Control Byte
	Pulse Control Byte
	Interrupt Vector
WR5	Base Register Byte
WR6	Base Register Byte
	Read Mask
Read Registers	
RR0	Status Byte
RR1	Byte Counter (low byte)
RR2	Byte Counter (high byte)
RR3	Port A Address Counter (low byte)
RR4	Port A Address Counter (high byte)
RR5	Port B Address Counter (low byte)
RR6	Port B Address Counter (high byte)

WRITING.

Control or command bytes are written into one or more of the Write Register groups (WR0-WR6) by first writing to the base register byte in that group. All groups have base registers and most groups have additional associated registers. The associated registers in a group are sequentially accessed by first writing a byte to the base register containing register-group identification and ponter bits (1's) to one or more of that base register's associated registers.

This is illustrated in figure 8b. In this figure, the sequence in which associated registers within a group can be written to is shown by the vertical position of the associated registers. For example, if a byte written to the DMA contains the bits that identify WR0 (bits D0, D1 and D7), and also contains 1's in the bit positions that point to the associated "Port A Starting Address (low byte)" and "Port A Starting Address (high byte)", then the next two byte written to the DMA will be stored in these two registers, in that order.

Figure 8a : Read Registers.



READING

The Read Registers (RR0-RR6) are read by the CPU by addressing the DMA as an I/O port using and Input instruction (such as INIR for the Z80 CPU). The readable bytes contain DMA status, byte-counter values, and port addresses since the last DMA reset. The registers are always read in a fixed sequence beginning with RR0 and ending with RR6. However, the register read in this sequence is determined by programming the Read Mask in WR6. The sequence of reading is initialized by writing an Initiate Read Sequence or Set Read Status command to WR6. After a Reset DMA, the sequence must be initialized with the Initiate Read Sequence command or a Read Status command. The sequence of reading all registers that are not excluded by the Read Mask register must be completed before a new Initiate Read Sequence or Read Status command.

FIXED-ADDRESS PROGRAMMING

A special circumstance arises when programming a destination port to have a fixed address. The load command in WR6 only loads a fixed address to a port selected as the source, not to a port selected as the destination.

Therefore, a fixed destination address must be loaded by temporarily declaring it a fixed-source address and subsequently declaring the true source as such, thereby implicitly making the other a destination.

The following example illustrates the steps in this procedure, assuming that transfers are to occur from a variable-address source (Port A) to a fixed-address destination (Port B) :

- 1. Temporarily declare Port B as source in WR0.
- 2. Load Port B address in WR6.
- 3. Declare Port A as source in WR0.
- 4. Load Port A address in WR6.
- 5. Enable DMA in WR6.

Figure 9 illustrates a program to transfer data from memory (Port A) to a peripheral device (Port B). In this example, the Port A memory starting address is 1050_H and the Port B peripheral fixed address is 05_H. Note that the data flow is 1001_H bytes - one more than specified by the block length. The table of DMA commands may be stored in consecutive memory locations and transferred to the DMA with an output instruction such as the Z80 CPU's OTIR instruction.

Figure 8b : Write Registers.

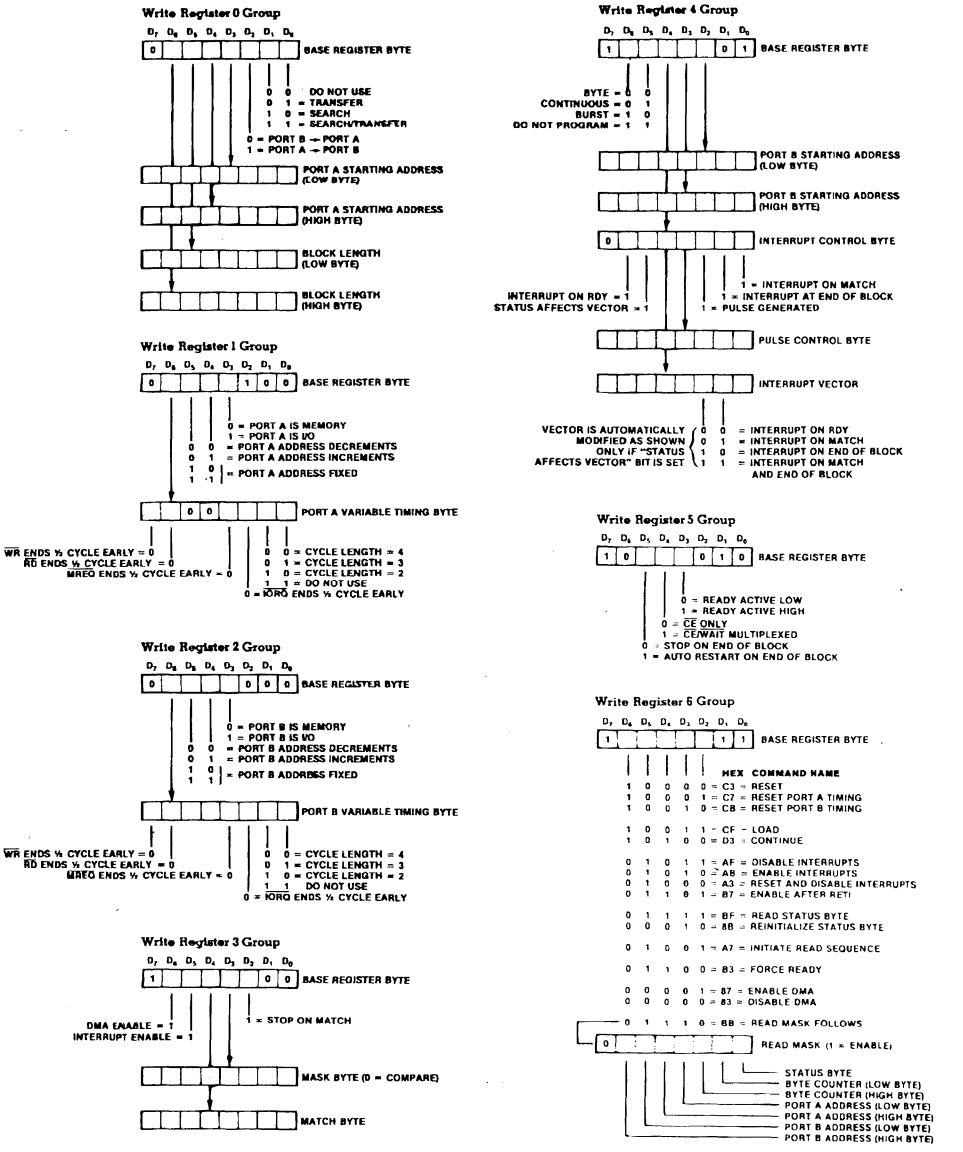


Figure 9 : Sample DMA Program.

Comments	D_7	D_6	D_5	D_4	D_3
WR0 sets DMA to receive block length. Port A starting address and temporarily sets Port B as source.	0	1 Block Length Upper Follows	1 Block Length Lower Follows	1 Port A Upper Address Follows	1 Port A Lower Address Follows
Port A Address (lower)	0	1	0	1	0
Port A Address (upper)	0	0	0	1	0
Block Length (lower)	0	0	0	0	0
Block Length (upper)	0	0	0	1	0
WR1 defines Port A as memory with fixed incrementing address.	0	0 No Timing Follows	0 Address Changes	1 Address Increments	0 Port is Memory
WR2 defines Port B as peripheral with fixed address.	0	0 No Timing Follows	1 Fixed Address	0	1 Port is I/O
WR4 sets mode to Burst, sets DMA to expect Port B address.	1	1	0	0 No Interrupt Control Byte Follows	0 No Upper Address
		Burst Mode			
Port B Address (lower)	0	0	0	0	0
WR5 Sets Ready Active High.	1	0	0 No Auto Restart	0 No Wait States	1 RDY Active High
WR6 Loads Port B Address and Resets Block Counter. *	1	1	0	0	1
WR0 Sets Port A as Source. *	0	0	0	0	0
		No Address or Block Length Bytes			
WR6 Loads Port A Address and Resets Block Counter.	1	1	0	0	1
WR6 Enables DMA to start operation.	1	0	0	0	

Note : The actual number of bytes transferred is one more than specified by the block length.
* These entries are necessary only in the case of a fixed destination address.

Figure 9 : Sample DMA Program (continued).

Comments	D_2	D_1	D_0	HEX
WR0 sets DMA to receive block length. Port A starting address and temporarily sets Port B as source.	0 B → A Temporary for Loading B Address *	0	1	79
		Transfer, No Search		
Port A Address (lower)	0	0	0	50
Port A Address (upper)	0	0	0	10
Block Length (lower)	0	0	0	00
Block Length (upper)	0	0	0	10
WR1 defines Port A as memory with fixed incrementing address.	1	0	0	14
WR2 defines Port B as peripheral with fixed address.	0	1	0	28
WR4 sets mode to Burst, sets DMA to expect Port B address.	1 Port B Lower Address Follows	0	1	C5
Port B Address (lower)	1	0	1	05
WR5 Sets Ready Active High.	0	1	0	8A
WR6 Loads Port B Address and Resets Block Counter. *	1	1	1	CF
WR0 Sets Port A as Source. *	1 A → B	0	1	05
		Transfer, No Search		
WR6 Loads Port A Address and Resets Block Counter.	1	1	1	CF
WR6 Enables DMA to start operation.	1	1	1	87

Note : The actual number of bytes transferred is one more than specified by the block length.
* These entries are necessary only in the case of a fixed destination address.

INACTIVE STATE TIMING (DMA as CPU Peripheral).

In its disabled or inactive state, the DMA is addressed by the CPU as an I/O peripheral for write and read (control and status) operations. Write timing is illustrated in figure 10.

Reading of the DMA's status byte, byte counter or port address counters is illustrated in figure 11. These operations require less than three T-cycles. The CE, IORQ and RD lines are made active over two rising edges of CLK, and data appears on the bus approximately one T-cycle after they become active.

ACTIVE STATE TIME (DMA as Bus Controller).

DEFAULT READ AND WRITE CYCLES

By default, and after reset, the DMA's timing of read and write operations is exactly the same as the Z80 CPU's timing of read and write cycles for memory and I/O peripherals, with one exception : during a read cycle, data is latched on the falling edge of T_3 and held on the data bus across the boundary between read and write cycles, through the end of the following write cycle.

Figure 12 illustrates the timing for memory-to-I/O port transfers and figure 13 illustrates I/O-to-memory transfers. Memory-to-memory and I/O-to-I/O transfers timings are simply permutations of these diagrams.

Figure 10 : CPU-to-DMA Write Cycle

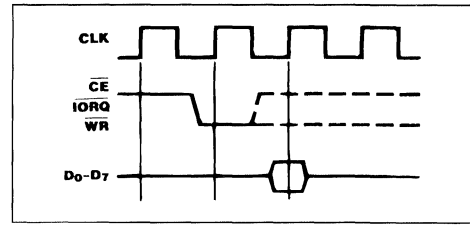
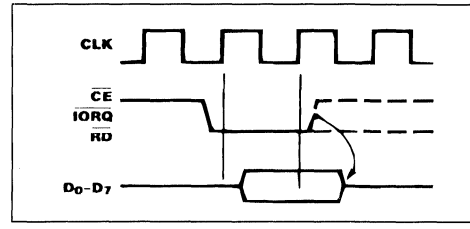
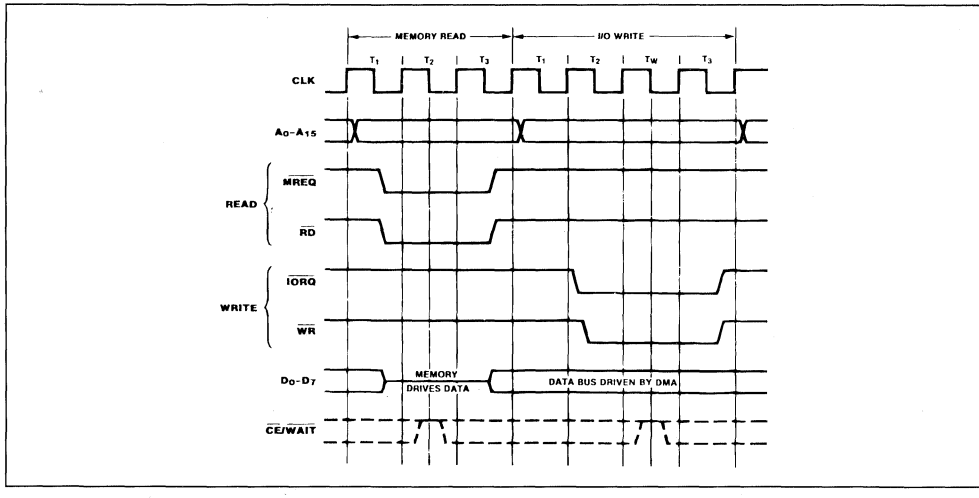


Figure 11 : CPU-to-DMA Read Cycle.



The default timing uses three T-cycles for memory transactions and four T-cycles for I/O transactions, which include one automatically inserted wait cycle between T_2 and T_3. If the CE/WAIT line is programmed to act a WAIT line during the DMA's active state, it is sampled on the falling edge of T_2 for memory transactions and the falling edge of T_W for I/O transactions. If CE/WAIT is Low during this time another T-cycle is added, during which the CE/WAIT line will again be sampled. The duration of transactions can thus be indefinitely extended.

Figure 12 : Memory-to-I/O Transfer.



VARIABLE CYCLE AN EDGE TIMING

The Z80 DMA's default operation-cycle length for the source (read) port and destination (write) port can be independently programmed. This variable-cycle feature allows read or write cycles consisting of two, three of four-T-cycle (more if Wait cycles are inserted), thereby increasing or decreasing the speed of all signals generated by the DMA. In addition, the trailing edges of the IORQ, MREQ, RD and WR signals can be independently terminated one-half cycle early. Figure 14 illustrates this.

In the variable-cycle mode, unlike default timing, IORQ comes active one-half cycle before MREQ, RD and WR. CE/WAIT can be used to extend only the 3 or 4 T-cycle variable memory cycles and only the 4-cycle variable I/O cycle. The CE/WAIT line is sampled at the falling edge of T_2 for 3- or 4-cycle memory cycles, and at the falling edge of T_3 for 4-cycle I/O cycles.

During transfers, data is latched on the clock edge causing the rising edge of RD and held through the end of the write cycle.

BUS REQUESTS

Figure 15 illustrates the bus request and acceptance timing. The RDY line, which may be programmed active High or Low, is sampled on every rising edge of CLK. If it is found to be active, and if the bus is not in use by any other device, the following rising edge of CLK drives BUSREQ low. After receiving BUSREQ the CPU acknowledges on the BAI input either directly or through a multiple-DMA daisy chain. When a Low is detect on BAI for two consecutive rising edges of CLK, the DMA will begin transferring data on the next rising edge of CLK.

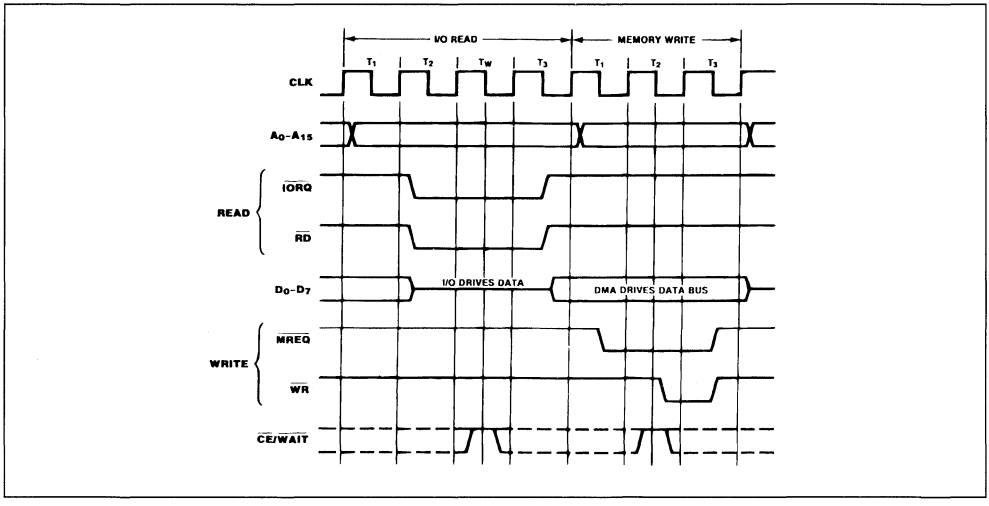
BUS RELEASE BYTE-AT-A-TIME

In Byte-at-a-Time mode, BUSREQ is brought High on the rising edge of CLK prior to the end of each read cycle (search-only) or write cycle (transfer and transfer/search) as illustrated in figure 16. This is done regardless of the state of RDY. There is no possibility of confusion when a Z80 CPU is used since the CPU cannot begin an operation until the following T-cycle. Most other CPUs are not bothered by this either, although note should be taken of it. The next bus request for the next byte will come after both BUSREQ and BAI have returned High.

BUS RELEASE AT END OF BLOCK

In Burst and Continuous modes, an end of block causes BUSREQ to go High usually on the same rising edge of CLK in which the DMA completes the transfer of the data block (figure 17). The last byte in the block is transferred even if RDY goes inactive before completion of the last byte transfer.

Figure 13 : I/O-to-Memory Transfer.



BUS RELEASE AND NOT READY

In Burst mode, when RDY goes inactive it causes BUSREQ to go High on the next rising edge of CLK after the completion of its current byte operation (figure 18). The action on BUSREQ is thus somewhat delayed from action on the RDY line. The DMA always completes its current byte operation in an orderly fashion before releasing the bus.

By contrast, BUSREQ is not released in Continuous mode when RDY goes inactive. Instead, the DMA idles after completing the current byte operation, awaiting an active RDY again.

BUS RELEASE ON MATCH

If the DMA is programmed to stop on match in Burst or Continuous modes, a match causes BUSREQ to go inactive on the next DMA operation, i.e., at the end of the next read in a search or at the end of the following write in a transfer (figure 19). Due to the pipelining scheme, matches are determined while the next DMA read or write is being performed.

The RDY line can go inactive after the matching operation begins without affecting this bus-release timing.

INTERRUPTS

Timings for interrupt acknowledge and return from interrupt are the same as timings for these in other Z80 peripherals.

Interrupt on RDY (interrupt before requesting bus) does not directly affect the BUSREQ line. Instead, the interrupt service routine must handle this by issuing the following commands to WR6 :

1. Enable after Return From Interrupt (RETI) Command - Hex B7
2. Enable DMA - Hex 87
3. A RETI instruction that reset the interrupt Under Service latch in the Z80 DMA.

Figure 14 : Variable-Cycle and Edge Timing.

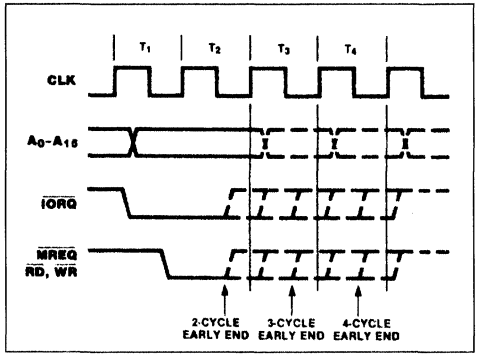


Figure 15 : Bus Request and Acceptance.

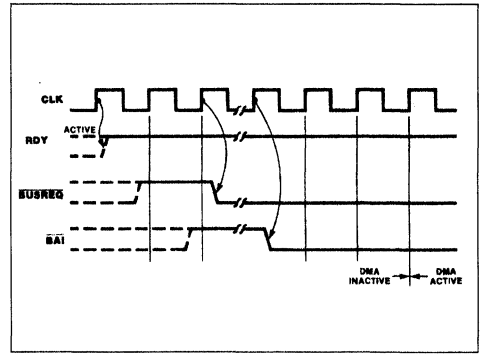


Figure 16 : Bus Release (Byte-at-a-Time-Mode).

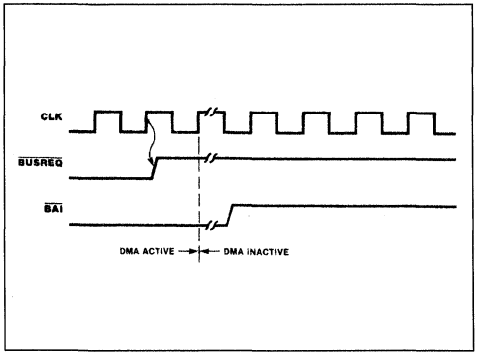


Figure 17 : Bus Release at End of Block (Burst and Continuous Modes).

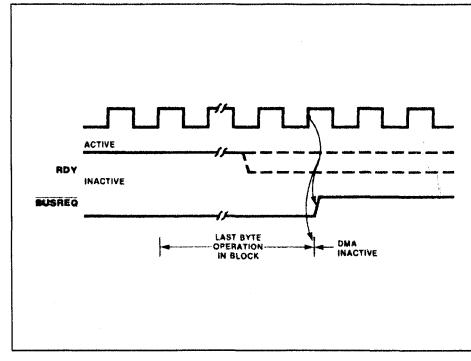


Figure 18 : Bus Release When Not Ready (Burst Mode).

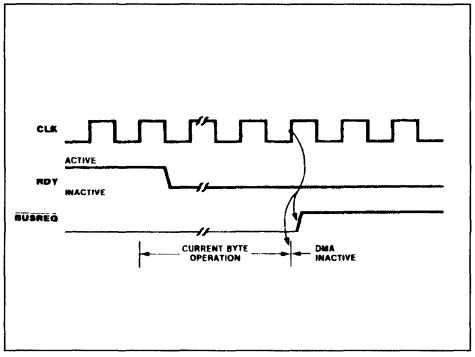
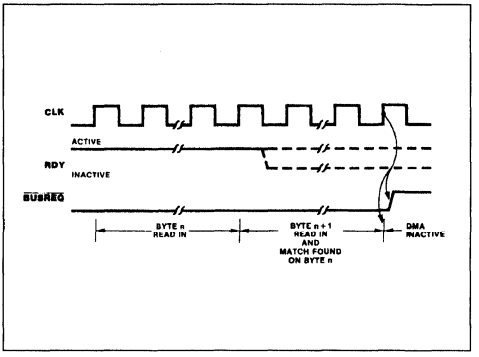


Figure 19 : Bus Release on Match (Burst and Continuous Modes).



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter		Value	Unit
T_A	Operating Ambient Temperature Under Bias	As Specified Under "Order Codes"		
T_{stg}	Storage Temperature		– 65 to + 150	°C
V_I	Voltage on Any Pin with Respect to ground		– 3 to + 7	V
P_D	Power Dissipation		1.5	W

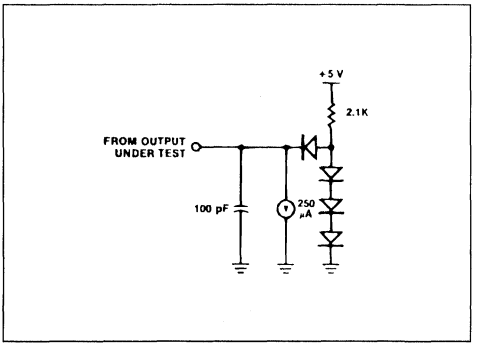
Stresses greater then those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only ; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

STANDARD TEST CONDITIONS

The characteristics below apply for the following test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin. Available operating temperature ranges are :

- 0 °C to + 70 °C,
  + 4.75 V ≤ V_{CC} ≤ + 5.25 V
- – 40 °C to + 85 °C,
  + 4.75 V ≤ V_{CC} ≤ + 5.25 V
- – 55 °C to + 125 °C,
  + 4.75 V ≤ V_{CC} ≤ + 5.25 V

All ac parameters assume a load capacitance of 100pF max. Timing references between two output signals assume a load difference of 50pF max.



DC CHARACTERISTICS

Symbol	Parameter	Test Conditions	Min.	Max.	Unit
V_{ILC}	Clock Input Low Voltage		– 0.3	0.45	V
V_{IHC}	Clock Input High Voltage		V_{CC} – 0.6	5.5	V
V_{IL}	Input Low Voltage		– 0.3	0.8	V
V_{IH}	Input High Voltage		2.0	5.5	V
V_{OL}	Output Low Voltage	I_{OL} = 3.2 mA for $\overline{BUSREQ}$ I_{OL} = 3.2 mA for all others		0.4	V
V_{OH}	Output High Voltage	I_{OH} = – 250 µA	2.4		V
I_{CC}	Power Supply Current Z8410 Z80 DMA Z8410A Z80A DMA			 150 200	 mA mA
I_{LI}	Input Leakage Current	V_{IN} = 0 to V_{CC}		10	µA
I_{LO}	3-State Output Leakage Current in Float	V_{OUT} = 0.4 to V_{CC}	– 10	10	µA
I_{LD}	Data Bus Leakage Current in Input Mode	$0 \leq V_{IN} \leq V_{CC}$	– 10	10	µA

CAPACITANCE

Symbol	Parameter	Test Conditions	Min.	Max.	Unit
C	Clock Capacitance	Unmeasured Pins		35	pF
C_{IN}	Input Capacitance	Returned to Ground		5	pF
C_{OUT}	Output Capacitance			10	pF

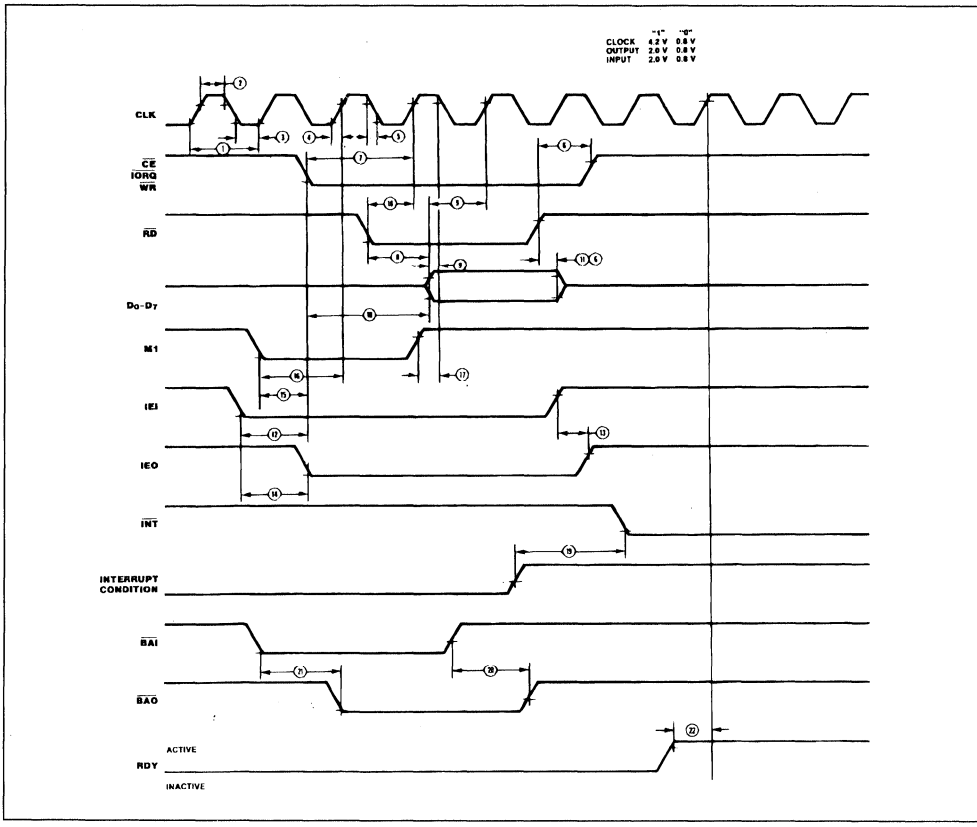
Over specified temperature range ; f = 1 MHz.

INACTIVE STATE AC CHARACTERISTICS

N°	Symbol	Parameter	Z8410		Z8410A	
			Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
1	TcC	Clock Cycle Time	400	4000	250	4000
2	TwCh	Clock Width (high)	170	2000	110	2000
3	TwCl	Clock Width (low)	170	2000	110	2000
4	TrC	Clock Rise Time		30		30
5	TfC	Clock Fall Time		30		30
6	Th	Hold Time for any Specified Setup Time	0		0	
7	TsC(Cr)	$\overline{IORQ}$, $\overline{ER}$, $\overline{CE}$ ↓ to Clock ↑ Setup	280		145	
8	TdDO(RDf)	$\overline{RD}$ ↓ to Data Output Delay		500		380
9	TsWM(Cr)	Data in to Clock ↑ Setup ($\overline{WR}$ or $\overline{MI}$)	50		50	
10	TdCf(DO)	$\overline{IORQ}$ ↓ to Data Out Delay (INTA cycle)		340		160
11	TdRD(Dz)	$\overline{RD}$ ↑ to Data Float Delay (output buffer disable)		160		110
12	TsIEI(IORQ)	IEI ↓ to $\overline{IORQ}$ ↓ Setup (INTA cycle)	140		140	
13	TdIEOr(IEIr)	IEI ↑ to IEO ↑ Delay		210		160
14	TdIEO(IEIf)	IEI ↓ to IEO ↓ Delay		190		130
15	TdMI(IEO)	$\overline{MI}$ ↓ to IEO ↓ Delay (interrupt just prior to $\overline{MI}$ ↓)		300		190
16	TsMIf(Cr)	$\overline{MI}$ ↓ to Clock ↑ Setup	210		90	
17	TsMIr(Cf)	$\overline{MI}$ ↑ to Clock ↓ Setup	20		− 10	
18	TsRD(Cr)	$\overline{RD}$ ↓ to Clock ↑ Setup ($\overline{MI}$ cycle)	240		115	
19	TdI(INT)	Interrupt Cause to $\overline{INT}$ ↓ Delay ($\overline{INT}$ generated only when DMA is inactive)		500		500
20	TdBAIr(BAOr)	$\overline{BAI}$ ↑ to $\overline{BAO}$ ↑ Delay		200		150
21	TdBAIf(BAOf)	$\overline{BAI}$ ↓ to $\overline{BAO}$ ↓ Delay		200		150
22	TsRDY(Cr)	RDY Active to Clock ↑ Set up	150		100	

Note : 1. Negative minimum setup values mean that the first mentioned event can come after the second mentioned event.

INACTIVE STATE CHARACTERISTICS (continued).



ACTIVE STATE AC CHARACTERISTICS

N°	Symbol	Parameter	Z8410		Z8410A	
			Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
1	TcC	Clock Cycle Time	400		250	
2	TwCh	Clock Width (high)	180	2000	110	2000
3	TwCl	Clock Width (low)	180	2000	110	2000
4	TrC	Clock Rise Time		30		30
5	TfC	Clock Fall Time		30		30
6	TdA	Address Output Delay		145		110
7	TdC(Az)	Clock ↑ to Address Float Delay		110		90
8	TsA(MREQ)	Address to $\overline{MREQ}$ ↓ Setup (memory cycle)	(2)+(5)–75		(2)+(5)–75	9
9	TsA(IRW)	Address Stable to $\overline{IORQ}$, $\overline{RD}$, $\overline{WR}$ ↓ Setup (I/O cycle)	(1)–80		(1)–70	
*10	TdRW(A)	$\overline{RD}$, $\overline{WR}$ ↑ to Addr. Stable Delay	(3)+(4)–40		(3)+(4)–50	
*11	TdRW(Az)	$\overline{RD}$, $\overline{WR}$ ↑ to Addr. Float	(3)+(4)–60		(3)+(4)–45	
12	TdCf(DO)	Clock ↓ to data Out Delay		230		150
*13	TdCr(Dz)	Clock ↑ to Data Float Delay (write cycle)		90		90
14	TsDI(Cr)	Data in to Clock ↑ Setup (read cycle when rising edge ends read)	50		35	
15	TsDI(Cf)	Data in to Clock ↓ Setup (read cycle when falling edge ends read)	60		50	
*16	TsDO(WfM)	Data out to $\overline{WR}$ ↓ Setup (memory cycle)	(1)–210		(1)–170	
17	TsDO(WfI)	Data Out to $\overline{WR}$ ↓ Setup (I/O cycle)	100		100	
*18	TdWr(DO)	$\overline{WR}$ ↑ to Data Out Delay	(3)+(4)–80		(3)+(4)–70	
19	Th	Hold Time for Any Specified Setup Time	0		0	
20	TdCr(Mf)	Clock ↓ to $\overline{MREQ}$ ↓ Delay		100		85
21	TdCf(Mr)	Clock ↑ to $\overline{MREQ}$ ↑ Delay		100		85
22	TdCr(Mr)	Clock ↓ to $\overline{MREQ}$ ↑ Delay		100		85
23	TdCf(Mr)	Clock ↓ to $\overline{MREQ}$ ↑ Delay		100		85
24	TwMl	$\overline{MREQ}$ Low Pulse Width	(1)–40		(1)–30	
*25	TwMh	$\overline{MREQ}$ High Pulse Width	(2)+(5)–30		(2)+(5)–20	
26	TdCf(If)	Clock ↓ to $\overline{IORQ}$ ↓ Delay		110		85
27	TdCr(If)	Clock ↑ to $\overline{IORQ}$ ↓ Delay		90		75
28	TdCr(Ir)	Clock ↑ to $\overline{IORQ}$ ↑ Delay		100		85
*29	TdCf(Ir)	Clock ↓ to $\overline{IORQ}$ ↑ Delay		110		85

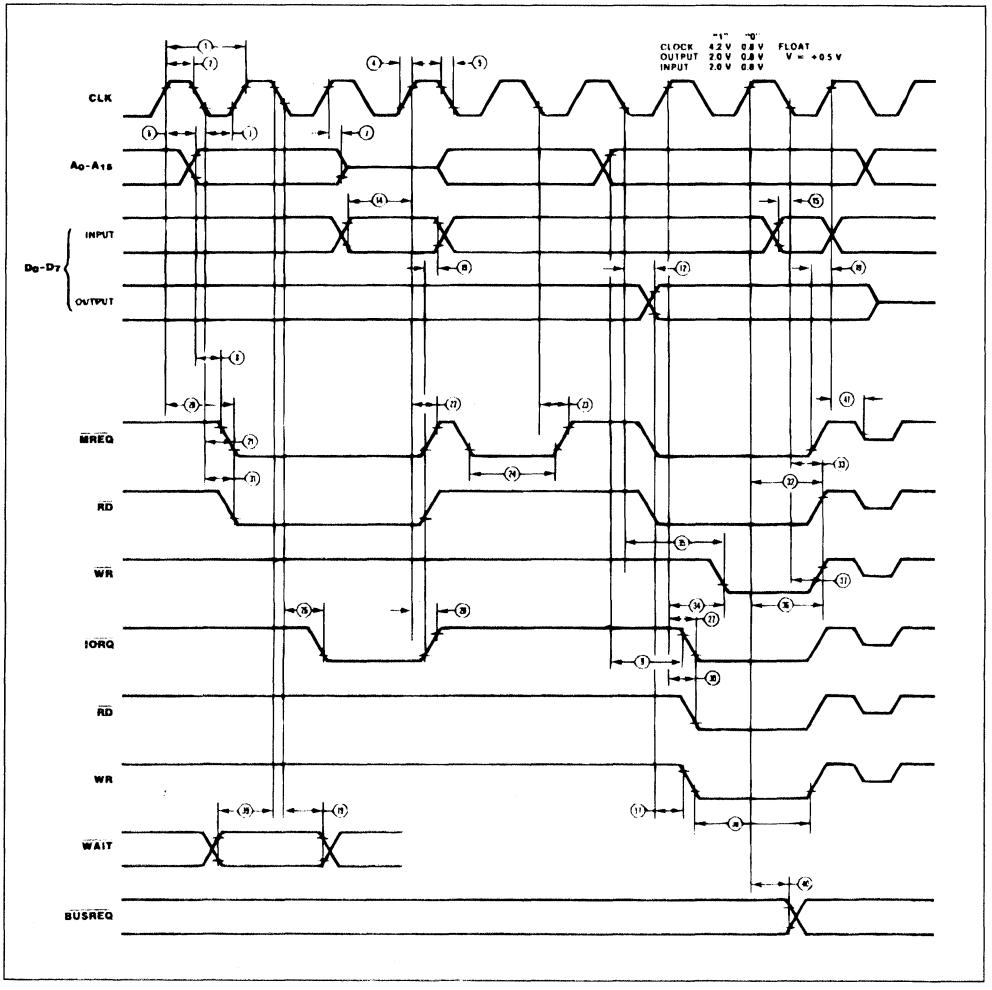
Notes : 1. Numbers in parentheses are other parameter-numbers in this table ; their values should be substituted in equations.
2. All equations imply DMA default (standard) timing.
3. Data must be enabled into data bus when $\overline{RD}$ is active.
4. Asterisk (*) before parameter number means the parameter is not illustrated in the AC Timing Diagrams.

ACTIVE STATE AC CHARACTERISTICS (continued)

N°	Symbol	Parameter	Z8410		Z8410A	
			Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
30	TdCr(Rf)	Clock ↑ to $\overline{RD}$ ↓ Delay		100		85
31	TdCf(Rf)	Clock ↓ to $\overline{RD}$ ↓ Delay		130		95
32	TdCr(Rr)	Clock ↑ to $\overline{RD}$ ↓ Delay		100		85
33	TdCf(Rr)	Clock ↓ to $\overline{RD}$ ↑ Delay		110		85
34	TdCr(Wf)	Clock ↑ to $\overline{WR}$ ↓ Delay		80		65
35	TdCf(Wf)	Clock ↓ to $\overline{WR}$ ↓ Delay		90		80
36	TdCr(Wr)	Clock ↑ to $\overline{WR}$ ↓ Delay		100		80
37	TdCf(Wr)	Clock ↓ to $\overline{WR}$ ↑ Delay		100		80
38	TwWI	$\overline{WR}$ Low Pulse Width	(1)–40		(1)–30	
39	TsWA(Cf)	$\overline{WAIT}$ to Clock ↓ Setup	70		70	
40	TdCr(B)	Clock ↑ to $\overline{BUSREQ}$ Delay		150		100
41	TdCr(Iz)	Clock ↑ to $\overline{IORQ}$, $\overline{MREQ}$, $\overline{RD}$, $\overline{WR}$ Float Delay		100		80

Notes : 1. Numbers in parentheses are other parameter-numbers in this table ; their values should be substituted in equations.
2. All equations imply DMA default (standard) timing.
3. Data must be enabled into data bus when $\overline{RD}$ is active.
4. Asterisk (*) before parameter number means the parameter is not illustrated in the AC Timing Diagrams.

ACTIVE STATE AC CHARACTERISTICS (continued).

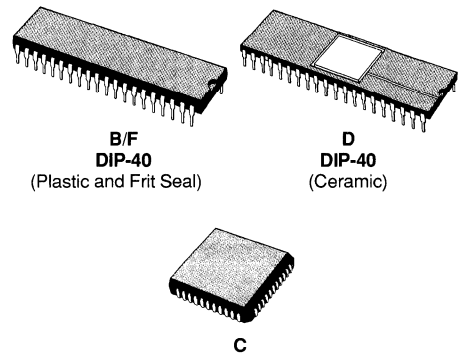


ORDERING INFORMATION

Type	Package	Temp.	Clock	Description
Z8410B1	DIP-40 (plastic)	0/+ 70°C	2.5 MHz	Z80 Direct Memory Access Unit
Z8410F1	DIP-40 (frit seal)	0/+ 70°C		
Z8410D1	DIP-40 (ceramic)	0/+ 70°C		
Z8410D6	DIP-40 (ceramic)	−40/+ 85°C		
Z8410D2	DIP-40 (ceramic)	−55/+125°C		
Z8410C1	PLCC44 (plastic chip-carrier)	0/+ 70°C		
Z8410AB1	DIP-40 (plastic)	0/+ 70°C	4 MHz	
Z8410AF1	DIP-40 (frit seal)	0/+ 70°C		
Z8410AD1	DIP-40 (ceramic)	0/+ 70°C		
Z8410AD6	DIP-40 (ceramic)	−40/+ 85°C		
Z8410AD2	DIP-40 (ceramic)	−55/+125°C		
Z8410AC1	PLCC44 (plastic chip-carrier)	0/+ 70°C		

Z80 PIO PARALLEL INPUT/OUTPUT CONTROLLER

- PROVIDES A DIRECT INTERFACE BETWEEN Z80 MICROCOMPUTER SYSTEMS AND PERIPHERAL DEVICES
- BOTH PORTS HAVE INTERRUPT-DRIVEN HANDSHAKE FOR FAST RESPONSE
- FOUR PROGRAMMABLE OPERATING MODES : BYTE INPUT, BYTE OUTPUT, BYTE INPUT/OUTPUT (Port A only), AND BIT INPUT/OUTPUT
- PROGRAMMABLE INTERRUPTS ON PERIPHERAL STATUS CONDITIONS
- STANDARD Z80 FAMILY BUS-REQUEST AND PRIORITIZED INTERRUPT-REQUEST DAISY CHAINS IMPLEMENTED WITHOUT EXTERNAL LOGIC
- THE EIGHT PORT B OUTPUTS CAN DRIVE DARLINGTON TRANSISTORS (1.5 mA at 1.5 V)



B/F
DIP-40
(Plastic and Frit Seal)

D
DIP-40
(Ceramic)

C
PLCC44
(Plastic)

(Ordering Information at the end of the datasheet)

DESCRIPTION

The Z80 PIO Parallel I/O Circuit is a programmable, dual-port device that provides a TTL-compatible interface between peripheral devices and the Z80 CPU. The CPU configures the Z80 PIO to interface with a wide range of peripheral devices with no other external logic. Typical peripheral devices that are compatible with the Z80 PIO include most keyboards, paper tape readers and punches, printers, PROM programmers, etc.

One characteristic of the Z80 peripheral controllers that separates them from other interface controllers is that all data transfer between the peripheral device and the CPU is accomplished under interrupt control. Thus, the interrupt logic of the PIO permits full use of the efficient interrupt capabilities of the Z80 CPU during I/O transfers. All logic necessary to implement a fully nested interrupt structure is included in the PIO.

Another feature of the PIO is the ability to interrupt the CPU upon occurrence of specified status conditions in the peripheral device. For example, the PIO can be programmed to interrupt if any specified peripheral alarm conditions should occur. This interrupt capability reduces the time the processor must spend in polling peripheral status.

The Z80 PIO interfaces to peripherals via two independent general-purpose I/O ports, designated port

LOGIC FUNCTIONS

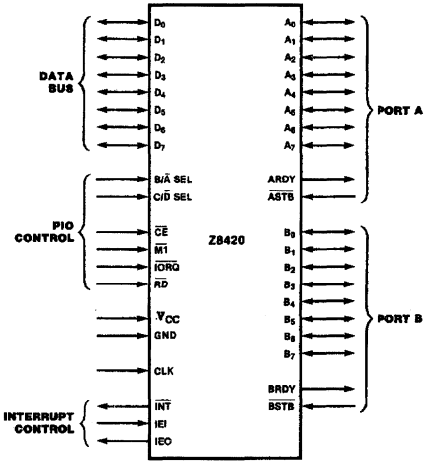


Figure 1 : Dual in Line Pin Configuration.

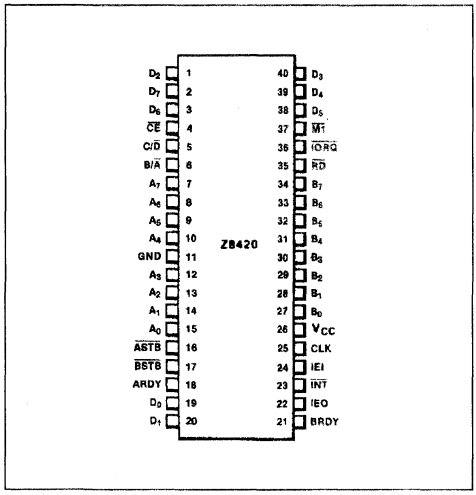
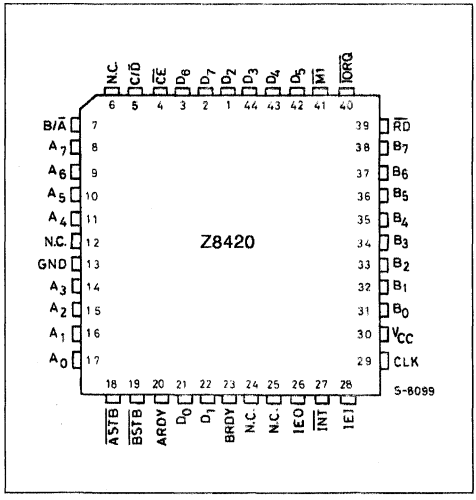


Figure 2 : Chip Carrier pin Configuration.



A and Port B. Each port has eight data bits and two handshake signals, Ready and Strobe, which control data transfer. The Ready output indicates to the peripheral that the port is ready for a data transfer. Strobe is an input from the peripheral that indicates when a data transfer has occured.

OPERATING MODES

The Z80 PIO ports can be programmed to operate in four modes : byte output (mode 0), byte input (mode 1), byte input/output (mode 2) and bit input/output (mode 3).

In mode 0, either port A or port B can be programmed to output data. Both ports have output registers that are individually addressed by the CPU ; data can be written to either port at any time. When data is written to a port, an active Ready output indicates to the external device that data is available at the associated port and is ready for transfer to the external device. After the data transfer, the external device responds with an active Strobe input, which generates an interrupt, if enabled.

In mode 1, either port A or port B can be configured in the input mode. Each port has an input register addressed by the CPU. When the CPU reads data from a port, the PIO sets the Ready signal, which is detected by the external device. The external device then places data on the I/O lines and strobes the I/O port, which latches the data into the Port Input Register, resets Ready, and triggers the Interrupt Request, if enabled. The CPU can read the input data at any time, which again sets Ready.

Mode 2 is bidirectional and uses port A, plus the interrupts and handshake signals from both ports. Port B must be set to mode 3 and masked off. In operation, port A is used for both data input and output. Output operation is similar to mode 0 except that data is allowed out onto the port A bus only when ASTB is Low. For input, operation is similar to mode 1, except that the data input uses the port B handshake signals and the port B interrupt (if enabled).

Both ports can be used in mode 3. In this mode, the individual bits are defined as either input or output bits. This provides up to eight separate, individually defined bits for each port. During operation, Ready and Strobe are not used. Instead, an interrupt is generated if the condition of one input changes, or if all inputs change. The requirements for generating an interrupt are defined during the programming operation ; the active level is specified as either High or Low, and the logic condition is specified as either one input active (OR) or all inputs active (AND). For example, if the port is programmed for active Low inputs and the logic function is AND, then all inputs at the specified port must go Low to generate an interrupt.

Data outputs are controlled by the CPU and can be written or changed at any time.

- Individual bits can be masked off.
- The handshake signals are not used in mode 3, Ready is held Low, and Strobe is disabled.
- When using the Z80 PIO interrupts, the Z80 CPU interrupt mode must be set to mode 2.

INTERNAL STRUCTURE

The internal structure of the Z80 PIO consists of a Z80 CPU bus interface, internal control logic, port A I/O logic, port B I/O logic, and interrupt control logic (figure 3). The CPU bus interface logic allows the Z80 PIO to interface directly to the Z80 CPU with no other external logic. The internal control logic synchronizes the CPU data bus to the peripheral device interfaces (port A and port B). The two I/O ports (A and B) are virtually identical and are used to interface directly to peripheral devices.

PORT LOGIC

Each port contains separate input and output registers, handshake control logic, and the control registers shown in figure 4. All data transfers between the peripheral unit and the CPU use the data input and output registers. The handshake logic associated with each port controls the data transfers through the input and the output registers. The mode control register (two bits) selects one of the four programmable operating modes.

The control mode (mode 3) uses the remaining registers. The input/output control register specifies which of the eight data bits in the port are to be outputs and enables these bits ; the remaining bits are inputs. The mask register and the mask control register control mode 3 interrupt conditions. The mask register specifies which of the bits in the port are active and which are masked or inactive.

The mask control register specifies two conditions : first, whether the active state of the input bits is High or Low, and second, whether an interrupt is generated when any one unmasked input bit is active (OR condition) or if the interrupt is generated when all unmasked input bits are active (AND condition).

INTERRUPT CONTROL LOGIC

The interrupt control logic section handles all CPU interrupt protocol for nested-priority interrupt structures. Any device's physical location in a daisy-chain configuration determines its priority. Two lines (IEI and IEO) are provided in each PIO to form this daisy chain. The device closest to the CPU has the highest priority. Within a PIO, port A interrupts have higher priority than those of port B. In the byte input, byte output, or bidirectional modes, an interrupt can be generated whenever the peripheral requests a new byte transfer. In the bit control mode, an interrupt can be generated when the peripheral status matches a programmed value. The PIO provides for complete control of nested interrupts. That is, lower priority devices may not interrupt higher priority devices that have not had their interrupt service routines completed by the CPU. Higher priority devices may interrupt the servicing of lower priority devices.

If the CPU (in interrupt mode 2) accepts an interrupt, the interrupting device must provide an 8-bit interrupt vector for the CPU. This vector forms a pointer to a location in memory where the address of the interrupt service routine is located. The 8-bit vector from the interrupting device forms the least significant eight bits of the indirect pointer while the I Register in the CPU provides the most significant eight bits of the pointer. Each port (A and B) has an independent interrupt vector. The least significant bit of the vector is automatically set to 0 within the PIO because the pointer must point to two adjacent memory locations for a complete 16-bit address.

Unlike the other Z80 peripherals, the PIO does not enable interrupts immediately after programming. It waits until M1 goes Low (e.g., during an opcode fetch). This condition is unimportant in the Z80 en-

Figure 3 : Block Diagram.

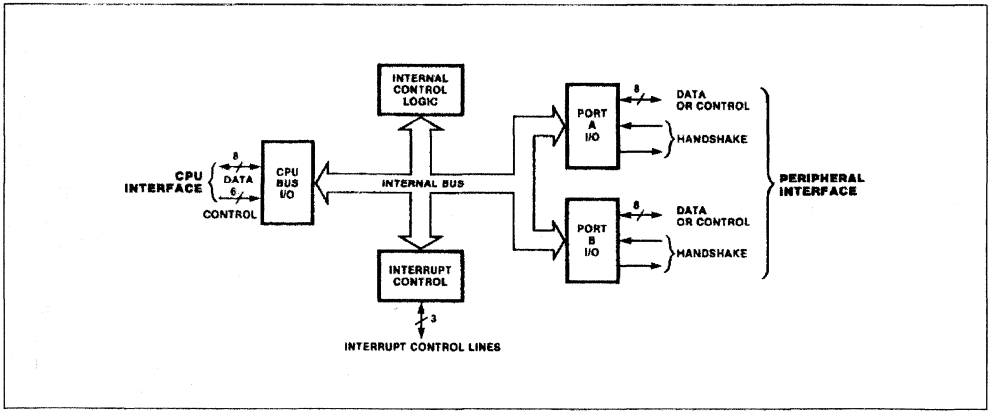
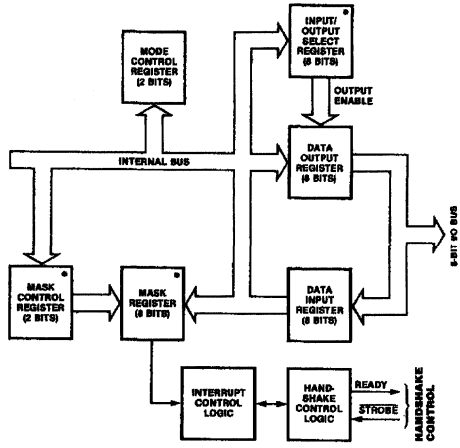


Figure 4 : Typical Port I/O Block Diagram.



* Used in the bit mode only to allow generation of an interrupt if the peripheral I/O pins go to the specified state.

vironment but might not be if another type of CPU is used.

The PIO decodes the RETI (Return From Interrupt) instruction directly from the CPU data bus so that each PIO in the system knows at all times whether it is being serviced by the CPU interrupt service routine. No other communication with the CPU is required.

CPU BUS I/O LOGIC

The CPU bus interface logic interfaces the Z80 PIO directly to the Z80 CPU, so no external logic is necessary. For large systems, however, address decoders and/or buffers may be necessary.

INTERNAL CONTROL LOGIC

This logic receives the control words for each port during programming and, in turn, controls the operating functions of the Z80 PIO. The control logic synchronizes the port operations, controls the port mode, port addressing, selects the read/write function, and issues appropriate commands to the ports and the interrupt logic. The Z80 PIO does not receive a write input from the CPU ; instead, the $\overline{RD}$, $\overline{CE}$, C/$\overline{D}$ and $\overline{IORQ}$ signals generate the write input internally.

PROGRAMMING

MODE 0, 1, OR 2

(Byte Input, Output, or Bidirectional). Programming a port for mode 0, 1, or 2 requires two words per port. These words are :

A MODE CONTROL WORD. Selects the port operating mode (figure 5). This word may be written any time.

AN INTERRUPT VECTOR. The Z80 PIO is designed for use with the Z80 CPU in interrupt mode 2 (figure 6). When interrupts are enabled, the PIO must provide an interrupt vector.

MODE 3

(Bit Input/Output). Programming a port for mode 3 operation requires a control word, a vector (if interrupts are enabled), and three additional words, described as follows :

I/O REGISTER CONTROL. When mode 3 is selected, the mode control word must be followed by another control word that sets the I/O control register, which in turn defines which port lines are inputs and which are outputs (figure 7).

Figure 5 : Mode Control Word.

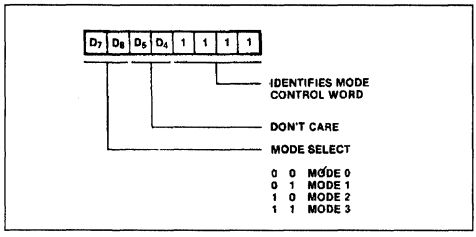
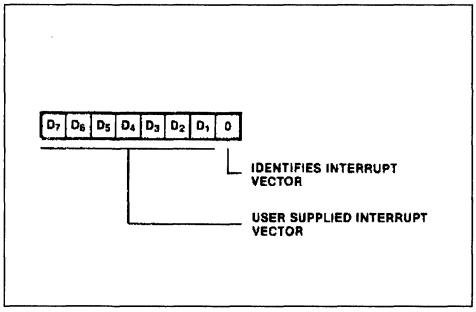


Figure 6 : Interrupt Vector Word.



INTERRUPT CONTROL WORD. In mode 3, handshake is not used. Interrupts are generated as a logic function of the input signal levels. The interrupt control word sets the logic conditions and the logic levels required for generating an interrupt. Two logic conditions or functions are available : AND (if all input bits change to the active level, an interrupt is triggered), and OR (if any one of the input bits changes to the active level, an interrupt is triggered). Bit D_6 sets the logic function, as shown in figure 8. The active level of the input bits can be set either High or Low. The active level is controlled by bit D_5.

Figure 9 : Mask Control Word.

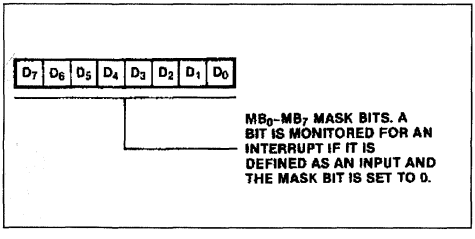


Figure 7 : I/O Register Control Word.

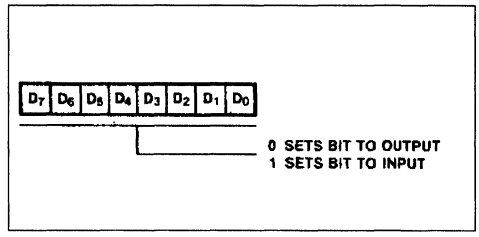
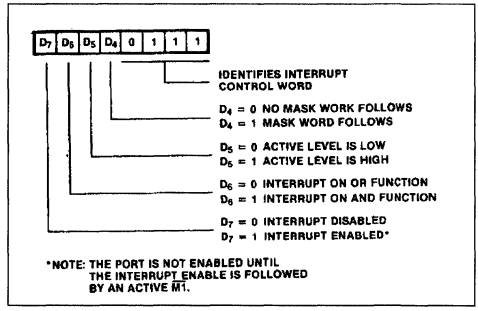


Figure 8 : Interrupt Control Word.

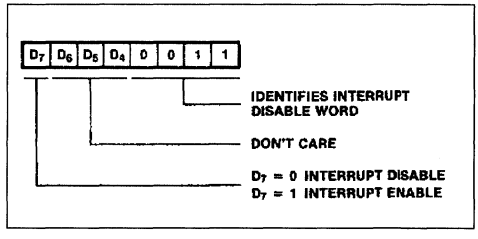


MASK CONTROL WORD. This word sets the mask control register, allowing any unused bits to be masked off. If any bits are to be masked, then D_4 must be set. When D_4 is set, the next word written to the port must be a mask control word (figure 9).

INTERRUPT DISABLE

There is one other control word which can be used to enable or disable a port interrupt. It can be used without changing the rest of the interrupt control word (figure 10).

Figure 10 : Interrupt Disable Word.



PIN DESCRIPTIONS

A_0-A_7. *Port A Bus* (Bidirectional, 3-state). This 8-bit bus transfers data, status, or control information between port A of the PIO and a peripheral device. A_0 is the least significant bit of the port A data bus.

ARDY. *Register A Ready* (Output, Active High). The meaning of this signal depends on the mode of operation selected for port A as follows :

OUTPUT MODE. This signal goes active to indicate that the port A output register has been loaded and the peripheral data bus is stable and ready for transfer to the peripheral device.

INPUT MODE. This signal is active when the port A input register is empty and ready to accept data from the peripheral device.

BIDIRECTIONAL MODE. This signal is active when data is available in the port A output register for transfer to the peripheral device. In this mode, data is not placed on the port A data bus, unless ASTB is active.

CONTROL MODE. This signal is disabled and forced to a Low state.

$\overline{\text{ASTB}}$. *Port A Strobe Pulse From Peripheral Device* (Input, Active Low). The meaning of this signal depends on the mode of operation selected for port A as follows :

OUTPUT MODE. The positive edge of this strobe is issued by the peripheral to acknowledge the receipt of data made available by the PIO.

INPUT MODE. The strobe is issued by the peripheral to load data from the peripheral into the port A input register. Data is loaded into the PIO when this signal is active.

BIDIRECTIONAL MODE. When this signal is active, data from the Port A output register is gated into the port A bidirectional data bus. The positive edge of the strobe acknowledges the receipt of the data.

CONTROL MODE. The strobe is inhibited internally.

B_0-B_7. Port B Bus (Bidirectional, 3-state). This 8-bit bus transfers data, status, or control information between port B and a peripheral device. The port B data bus can supply 1.5 mA at 1.5 V to drive Darlington transistors. B_0 is the least significant bit of the bus.

B/$\overline{\text{A}}$. *Port B Or A Select* (Input, High = B). This pin defines which port is accessed during a data transfer between the CPU and the PIO. A low on this pin selects port A ; a High selects port B. Often address bit A_0 from the CPU is used for this selection function.

BRDY. *Register B Ready* (Output, Active High). This signal is similar to ARDY, except that in the port A bidirectional mode this signal is High when the port A input register is empty and ready to accept data from the peripheral device.

$\overline{\text{BSTB}}$. *Port B Strobe Pulse From Peripheral Device* (Input, Active Low). This signal is similar to ASTB, except that in the port A bidirectional mode this signal strobes data from the peripheral device into the port A input register.

C/$\overline{\text{D}}$. *Control Or Data Select* (Input, High = C). This pin defines the type of data transfer to be performed between the CPU and the PIO. A High on this pin during a CPU write to the PIO causes the Z80 data bus to be interpreted as a command for the port selected by the B/$\overline{\text{A}}$ Select line. A low on this pin means that the Z80 data bus is being used to transfer data between the CPU and the PIO. Often address bit A_1 from the CPU is used for this function.

$\overline{\text{CE}}$. *Chip Enable* (Input, Active Low). A low on this pin enables the PIO to accept command or data inputs from the CPU during a write cycle or to transmit data to the CPU during a read cycle. This signal is generally decoded from four I/O port numbers for ports A and B, data, and control.

CLK. *System Clock* (Input). The Z80 PIO uses the standard single-phase Z80 system clock.

D_0-D_7. *Z80 CPU Data Bus* (Bidirectional, 3-state). This bus is used to transfer all data and commands between the Z80 CPU and the Z80 PIO. D_0 is the least significant bit.

IEI. *Interrupt Enable In* (Input, Active High). This signal is used to form a priority-interrupt daisy chain when more than one interrupt-driven device is being used. A High level on this pin indicates that no other devices of higher priority are being serviced by a CPU interrupt service routine.

IEO. *Interrupt Enable Out* (Output, Active High). The IEO signal is the other signal required to form a daisy chain priority scheme. It is High only if IEI is High and the CPU is not servicing an interrupt from this PIO. Thus this signal blocks lower priority devices from interrupting while a higher priority device is being serviced by its CPU interrupt service routine.

$\overline{\text{INT}}$. *Interrupt Request* (Output, Open Drain, Active Low). When $\overline{\text{INT}}$ is active the Z80 PIO is requesting an interrupt from the Z80 CPU.

$\overline{\text{IORQ}}$. *Input/Output Request* (Input from Z80 CPU, Active Low). $\overline{\text{IORQ}}$ is used in conjunction with B/A, C/D, $\overline{\text{CE}}$, and $\overline{\text{RD}}$ to transfer commands and data between the Z80 CPU and the Z80 PIO. When $\overline{\text{CE}}$, $\overline{\text{RD}}$, and $\overline{\text{IORQ}}$ are active, the port addressed by B/A transfers data to the CPU (a read operation).

Conversely, when $\overline{\text{CE}}$ and $\overline{\text{IORQ}}$ are active but $\overline{\text{RD}}$ is not, the port addressed by B/A is written into from the CPU with either data or control information, as specified by C/D.

Also, if $\overline{\text{IORQ}}$ and $\overline{\text{M1}}$ are active simultaneously, the CPU is acknowledging an interrupt ; the interrupting port automatically places its interrupt vector on the CPU data bus if it is the highest priority device requesting an interrupt.

$\overline{\text{M1}}$. Machine Cycle (Input from CPU, Active Low). This signal is used as a sync pulse to control several internal PIO operations. When both the $\overline{\text{M1}}$ and $\overline{\text{RD}}$ signals are active, the Z80 CPU is fetching an instruction from memory. Conversely, when both $\overline{\text{M1}}$ and $\overline{\text{IORQ}}$ are active, the CPU is acknowledging an interrupt. In addition, $\overline{\text{M1}}$ has two other functions within the Z80 PIO : it synchronizes the PIO interrupt logic ; when $\overline{\text{M1}}$ occurs without an active $\overline{\text{RD}}$ or $\overline{\text{IORQ}}$ signal, the PIO is reset.

$\overline{\text{RD}}$. Read Cycle Status (Input from Z80 CPU, Active Low). If $\overline{\text{RD}}$ is active, or an I/O operation is in progress, $\overline{\text{RD}}$ is used with B/A, C/D, $\overline{\text{CE}}$, and $\overline{\text{IORQ}}$ to transfer data from the Z80 PIO to the Z80 CPU.

TIMING

The following timing diagrams show typical timing in a Z80 CPU environment. For more precise specifications refer to the composite ac timing diagram.

WRITE CYCLE

Figure 11 illustrates the timing for programming the Z80 PIO or for writing data to one of its ports. No Wait states are allowed for writing to the PIO other than the automatically inserted T_{WA}. The PIO does not receive a specific write signal ; it internally generates its own from the lack of an active $\overline{\text{RD}}$ signal.

READ CYCLE

Figure 12 illustrates the timing for reading the data input from an external device to one of the Z80 PIO ports. No Wait states are allowed for reading the PIO other than the automatically inserted T_{WA}.

OUTPUT MODE (mode 0)

An output cycle (figure 13) is always started by the execution of an output instruction by the CPU. The $\overline{\text{WR}}$* pulse from the CPU latches the data from the CPU data bus into the selected port's output register. The $\overline{\text{WR}}$* pulse sets the Ready flag after a Low-going edge of CLK, indicating data is available. Ready stays active until the positive edge of the strobe line is received, indicating that data was taken by the peripheral. The positive edge of the strobe pulse generates an $\overline{\text{INT}}$ if the interrupt enable flip-flop has been set and if this device has the highest priority.

INPUT MODE (mode 1)

When $\overline{\text{STROBE}}$ goes Low, data is loaded into the selected port input register (figure 14). The next rising edge of strobe activates $\overline{\text{INT}}$, if Interrupt Enable is set and this is the highest-priority requesting device. The following falling edge of CLK resets Ready to an inactive state, indicating that the input register is full and cannot accept any more data until the CPU completes a read. When a read is complete, the positive edge of $\overline{\text{RD}}$ sets Ready at the next Low-going transition of CLK. At this time new data can be loaded into the PIO.

BIDIRECTIONAL MODE (mode 2)

This is a combination of Modes 0 and 1 using all four handshake lines and the eight Port A I/O lines (figure 15). Port B must be set to the bit mode and its inputs must be masked. The Port A handshake lines are used for output control and the Port B lines are used for input control. If interrupts occur, Port A's vector will be used during port output and Port B's will be used during port input. Data is allowed out onto the Port A bus only when $\overline{\text{ASTB}}$ is Low. The rising edge of this strobe can be used to latch the data into the peripheral.

BIT MODE (mode 3)

The bit mode does not utilize the handshake signals, and a normal port write or port read can be executed at any time. When writing, the data is latched into the output registers with the same timing as the output mode (figure 16).

When reading the PIO, the data returned to the CPU is composed of output register data from those port data lines assigned as outputs and input register data from those port data lines assigned as inputs. The input register contains data that was present immediately prior to the falling edge of $\overline{\text{RD}}$. An interrupt is generated if interrupts from the port are enabled and the data on the port data lines satisfy the logical equation defined by the 8-bit mask and 2-bit mask control registers. However, if Port A is programmed in bidirectional mode, Port B does not issue an interrupt in bit mode and must therefore be polled.

Figure 11 : Write Cycle Timing.

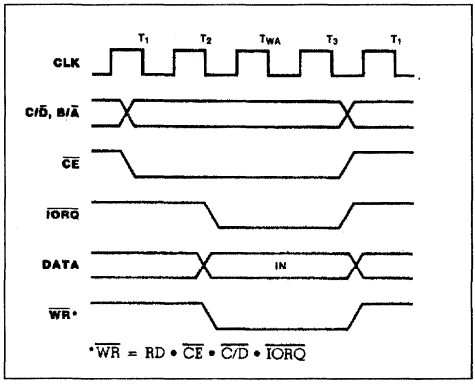


Figure 12 : Read Cycle Timing.

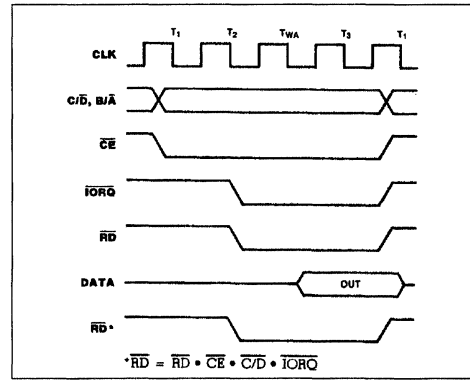
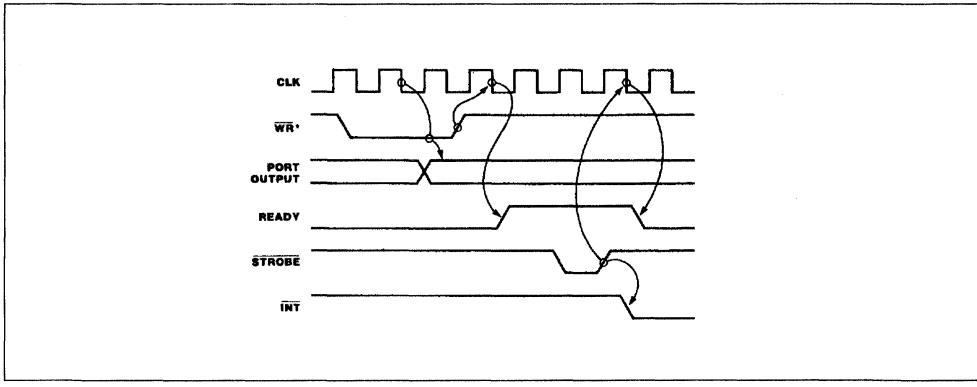


Figure 13 : Mode 0 Output Timing.



INTERRUPT ACKNOWLEDGE TIMING

During M1 time, peripheral controllers are inhibited from changing their interrupt enable status, permitting the Interrupt Enable signal to ripple through the daisy chain. The peripheral with IEI High and IEO Low during INTACK places a preprogrammed 8-bit interrupt vector on the data bus at this time (figure 17). IEO is held Low until a Return From Interrupt (RETI) instruction is executed by the CPU while IEI is High. The 2-byte RETI instruction is decoded internally by the PIO for this purpose.

RETURN FROM INTERRUPT CYCLE

If a Z80 peripheral has no interrupt pending and is not under service, then its IEO = IEI. If it has an interrupt under service (i.e., it has already interrupted and received an interrupt acknowledge) then its IEO is always Low, inhibiting lower priority devices from interrupting. If it has an interrupt pending which has not yet been acknowledged, IEO is Low unless an "ED" is decoded as the first byte of a 2-byte opcode (figure 18). In this case, IEO goes High until the next opcode byte is decoded, where upon it goes Low again. If the second byte of the opcode was a "4D", then the opcode was an RETI instruction.

After an "ED" opcode is decoded, only the peripheral device which has interrupted and is currently under service has its IEI High and its IEO Low. This device is the highest-priority device in the daisy chain that has received an interrupt acknowledge. All other peripherals have IEI = IEO. If the next opcode byte decoded is "4D", this peripheral device resets its "interrupt under service" condition.

Figure 14 : Mode 1 Input Timing.

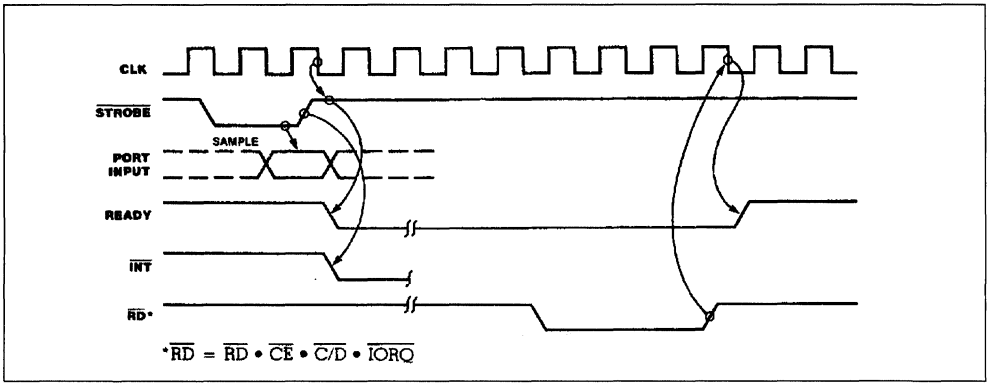


Figure 15 : Mode 2 Bidirectional Timing.

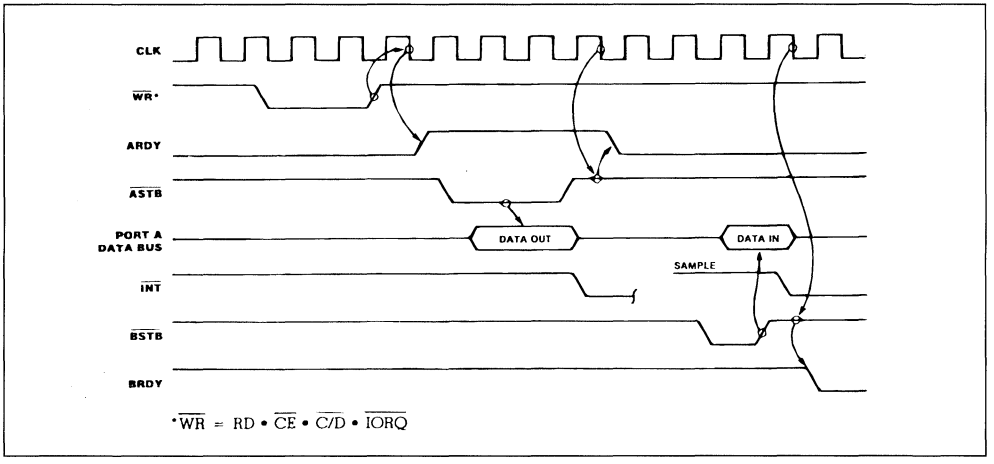


Figure 16 : Mode 3 Bit Mode Timing.

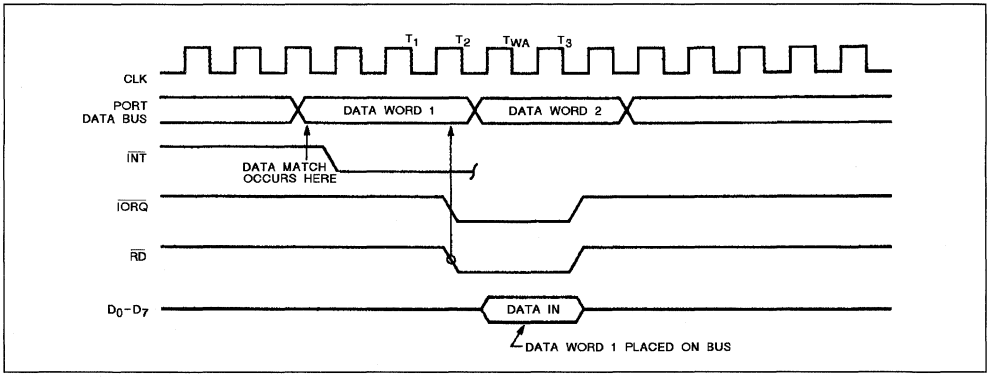


Figure 17 : Interrupt Acknowledge Timing.

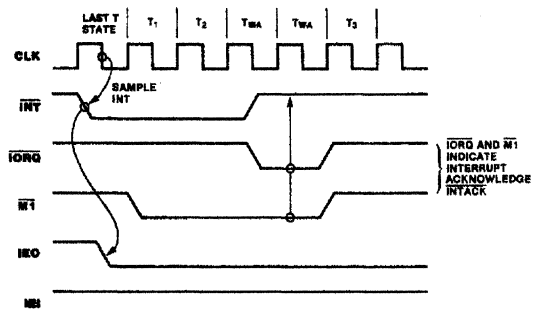
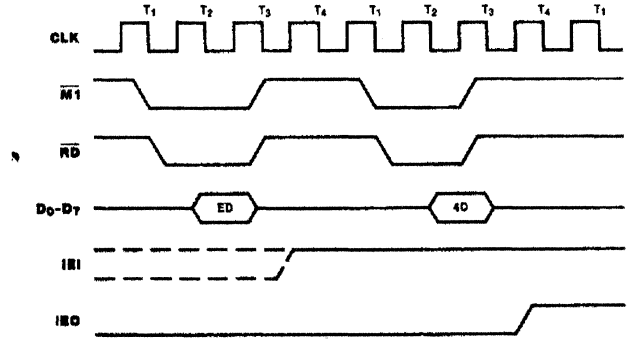


Figure 18 : Return From Interrupt.



AC CHARACTERISTICS

N°	Symbol	Parameter	Comment	Z8420		Z8420A		Z8420B	
				Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
1	TcC	Clock Cycle Time		400	(1)	250	(1)	165	(1)
2	TwCh	Clock Width (high)		170	2000	105	2000	65	2000
3	TwCl	Clock Width (low)		170	2000	105	2000	65	2000
4	TfC	Clock Fall Time			30		30		20
5	TrC	Clock Rise Time			30		30		20
6	TsCS(RI)	$\overline{CE}$, B/A, C/D to $\overline{RD}$, $\overline{IORQ}$ ↓ Setup Time	(6)	50		50		50	
7	Th	Any Hold Times for Specified Setup Time		0		0		0	0
8	TsRI(C)	$\overline{RD}$, $\overline{IORQ}$ to Clock ↑ Setup Time		115		115		70	
9	TdRI(DO)	$\overline{RD}$, $\overline{IORQ}$ ↓ to Data Out Delay	(2)		430		380		300
10	TdRI(DOs)	$\overline{RD}$, $\overline{IORQ}$ ↑ to Data Out Float Delay			160		110		70
11	TsDI(C)	Data in to Clock ↑ Setup Time	CL = 50 pF	50		50		40	
12	TdIO(DOI)	$\overline{IORQ}$ ↓ to Data Out Delay (INTACK cycle)	(3)		340		160		120
13	TsMI(Cr)	$\overline{MI}$ ↓ to Clock ↑ Setup Time		210		90		70	
14	TsMI(Cf)	$\overline{MI}$ ↑ to Clock ↓ setup Time ($\overline{MI}$ cycle)	(8)	0		0		0	
15	TdMI(IEO)	$\overline{MI}$ ↓ to IEO ↓ Delay (interrupt immediately preceding $\overline{MI}$ ↓)	(5, 7)		300		190		100
16	TsIEI(IO)	IEI to $\overline{IORQ}$ ↓ Setup Time (INTACK cycle)	(7)	140		140		100	
17	TdIEI(IEOf)	IEI ↓ to IEO ↓ Delay	(5) CL = 50 pF		190		130		120
18	TdIEI(IEOr)	IEI ↑ to IEO ↑ Delay (after ED decode)	(5)		210		160		160
19	TcIO(C)	$\overline{IORQ}$ ↑ to Clock ↓ Setup Time (to activate READY on next clock cycle)		220		200		170	
20	TdC(RDYr)	Clock ↓ to READY ↑ Delay	(5) CL = 50 pF		200		190		170
21	TdC(RDYf)	Clock ↓ to READY ↓ Delay	(5)		150		140		120
22	TwSTB	$\overline{STROBE}$ Pulse Width	(4)	150		150		120	
23	TsSTB(C)	$\overline{STROBE}$ ↑ to Clock ↓ Setup Time (to activate READY on next clock cycle)	(5)	220		220		150	

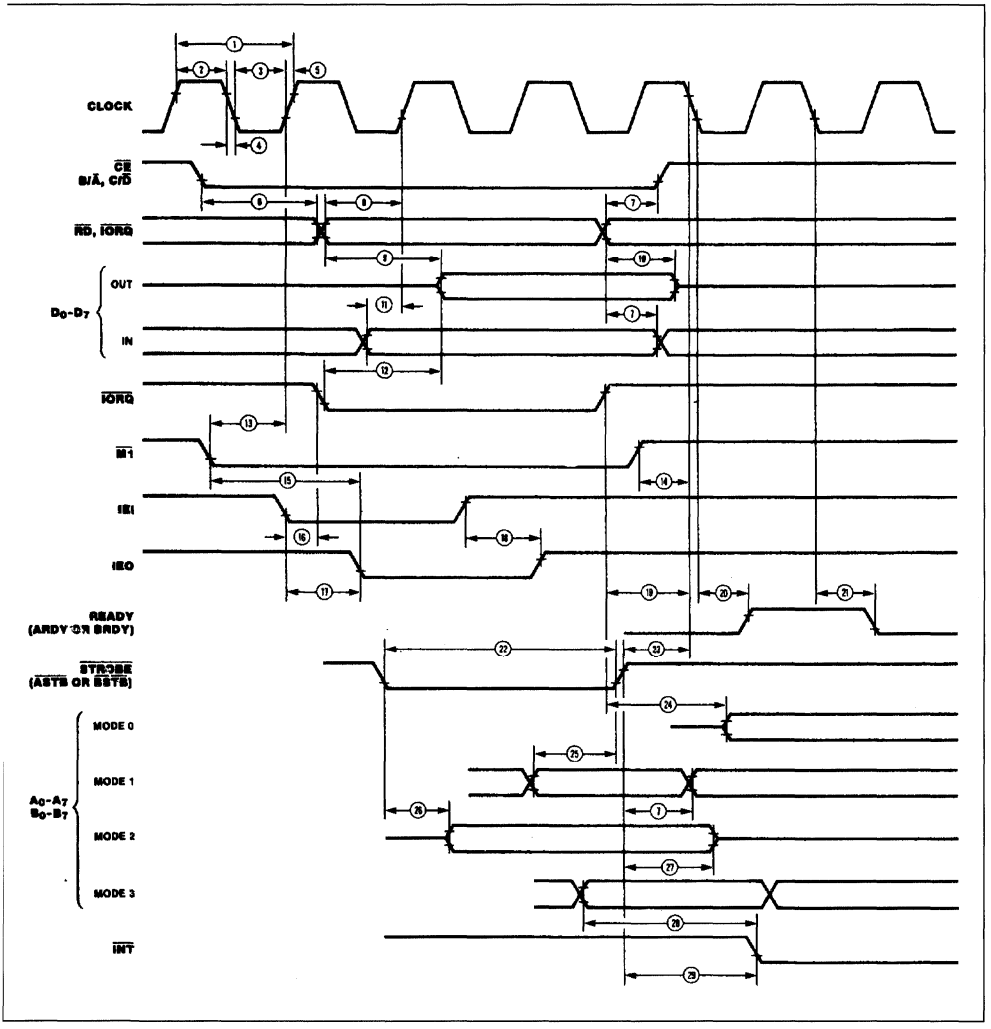
Notes :
1 TcC = TwCh + TwC1 + TrC + TfC.
2 Increase TdRi(DO) by 10 ns for each 50 pF increase in load up to 200 pF max.
3 Increase TdIO(DO) by 10 ns for each 50 pF increase in loading up to 200 pF max.
4 For Mode 2 TwSTB > TsPD(STB).
5 Increase these values by 2 ns for each 10 pF increase in loading up to 100 pF max.
6 TsCS(RI) may be reduced. However the time subtracted from TsCS(RI) may be added to TdRI(DO).
7 2.5 TdC > (N-2)TdIEI(IEOf) + TdMI(IEO) + TsIEI(IO) + TTL Buffer Delay if any.
8 M1 must be active for a minimum of two clock cycles to reset the PIO.

AC CHARACTERISTICS (continued)

N°	Symbol	Parameter	Comment	Z8420		Z8420A		Z8420B	
				Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
24	TdIO(PD)	$\overline{IORQ}$ ↑ to PORT DATA Stable Delay (mode 0)	(5)		200		180		160
25	tsPD(STB)	PORT DATA to $\overline{STROBE}$ ↑ Setup Time (mode 1)		260		230		190	
26	TdSTB(PD)	$\overline{STROBE}$ ↓ to PORT DATA Stable (mode 2)	(5)		230		210		180
27	TdSTB(PDr)	$\overline{STROBE}$ ↑ to PORT DATA Float Delay (mode 2)	CL = 50 pF		200		180		160
28	TdPD(INT)	PORT DATA Match to $\overline{INT}$ ↓ Delay (mode 3)			540		490		430
29	TdSTB(INT)	$\overline{STROBE}$ ↑ to $\overline{INT}$ ↑ Delay			490		440		350

Notes :
1 TcC = TwCh + TwC1 + TrC + TfC.
2 Increase TdRi(DO) by 10 ns for each 50 pF increase in load up to 200 pF max.
3 Increase TdIO(DO) by 10 ns for each 50 pF increase in loading up to 200 pF max.
4 For Mode 2 TwSTB > TsPD(STB).
5 Increase these values by 2 ns for each 10 pF increase in loading up to 100 pF max.
6 TsCS(RI) may be reduced. However the time subtracted from TsCS(RI) may be added to TdRI(DO).
7 2.5 TdC > (N-2)TdIEI(IEOf) + TdMI(IEO) + TsIEI(IO) + TTL Buffer Delay if any.
8 M1 must be active for a minimum of two clock cycles to reset the PIO.

AC CHARACTERISTICS (continued)



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_I	Voltage on all Input and Outputs with Respect to GND	− 0.3 to + 7.0	V
T_A	Operation Ambient Temperature As Specified in Order Codes		
T_{stg}	Storage Temperature Range	− 65 to + 150	°C

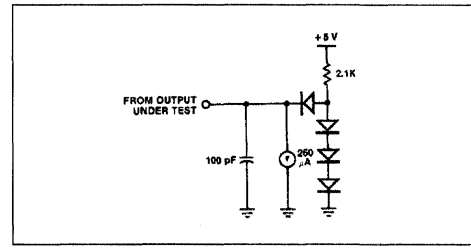
Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only ; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

STANDARD TEST CONDITIONS

The characteristics below apply for the following test conditions, unless otherwise noted. All voltages are referenced to GND (0 V). Positive current flows into the referenced pin. Available operating temperature ranges are :

- 0 °C to + 70 °C,
  + 4.75 V ≤ V_{CC} ≤ + 5.25 V
- - 40 °C to + 85 °C,
  + 4.75 V ≤ VCC ≤ + 5.25 V
- - 55 °C to + 125 °C,
  + 4.75 V ≤ VCC ≤ + 5.25 V

All ac parameters assume a load capacitance of 100pF max.



DC CHARACTERISTICS

Symbol	Parameter	Test Conditions	Min.	Max.	Unit
V_{ILC}	Clock Input Low Voltage		− 0.3	0.45	V
V_{IHC}	Clock Input High Voltage		V_{CC} − 0.6	V_{CC}+0.3	V
V_{IL}	Input Low Voltage		− 0.3	0.8	V
V_{IH}	Input High Voltage		2.0	V_{CC}	V
V_{OL}	Output Low Voltage	I_{OL} = 2.0 mA	−	0.4	V
V_{OH}	Output High Voltage	I_{OH} = − 250 µA	2.4	−	V
I_{LI}	Input Leakage Current	V_{IN} = 0 to V_{CC}	− 10	10	µA
I_{LO}	3-State Output Leakage Current in Float	V_{OUT} = 0.4 V to V_{CC}	− 10	10	µA
I_{CC}	Power Supply Current	V_{OH} = 1.5 V		100	mA
I_{OHD}	Darlington Drive Current	R_{EXT} = 390 Ω	− 1.5	3.8	mA

Over specified temperature and voltage range.

CAPACITANCE

Symbol	Parameter	Test Conditions	Min.	Max.	Unit
C	Clock Capacitance	Unmeasured pins returned to ground.		10	pF
C_{IN}	Input Capacitance			5	pF
C_{OUT}	Output Capacitance			10	pF

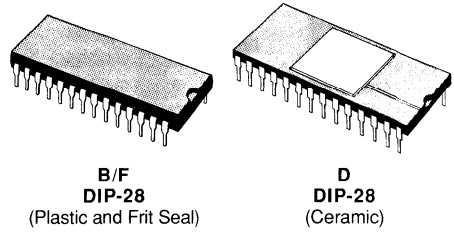
Over specified temperature range ; f = 1 MHz.

ORDERING INFORMATION

Type	Package	Temp.	Clock	Description
Z8420B1	DIP-40 (plastic)	0/+ 70°C	2.5 MHz	Z80 Parallel Input/Output Unit
Z8420F1	DIP-40 (frit seal)	0/+ 70°C		
Z8420D1	DIP-40 (ceramic)	0/+ 70°C		
Z8420D6	DIP-40 (ceramic)	−40/+ 85°C		
Z8420D2	DIP-40 (ceramic)	−55/+125°C		
Z8420C1	PLCC44 (plastic chip-carrier)	0/+ 70°C		
Z8420AB1	DIP-40 (plastic)	0/+ 70°C	4 MHz	
Z8420AF1	DIP-40 (frit seal)	0/+ 70°C		
Z8420AD1	DIP-40 (ceramic)	0/+ 70°C		
Z8420AD6	DIP-40 (ceramic)	−40/+ 85°C		
Z8420AD2	DIP-40 (ceramic)	−55/+125°C		
Z8420AC1	PLCC44 (plastic chip-carrier)	0/+ 70°C		
Z8420BB1	DIP-40 (plastic)	0/+ 70°C	6 MHz	
Z8420BF1	DIP-40 (frit seal)	0/+ 70°C		
Z8420BD1	DIP-40 (ceramic)	0/+ 70°C		
Z8420BD6	DIP-40 (ceramic)	−40/+ 85°C		
Z8420BD2	DIP-40 (ceramic)	−55/+125°C		
Z8420BC1	PLCC44 (plastic chip-carrier)	0/+ 70°C		

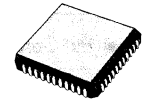
Z80 CTC COUNTER TIMER CIRCUIT

- FOUR INDEPENDENTLY PROGRAMMABLE COUNTER/TIMER CHANNELS, EACH WITH A READABLE DOWNCOUNTER AND A SELECTABLE 16 OR 256 PRESCALER. DOWNCOUNTERS ARE RELOADED AUTOMATICALLY AT ZERO COUNT
- THREE CHANNELS HAVE ZERO COUNT/TIMEOUT OUTPUTS CAPABLE OF DRIVING DARLINGTON TRANSISTORS
- SELECTABLE POSITIVE OR NEGATIVE TRIGGER INITIATES TIMER OPERATION
- STANDARD Z80 FAMILY DAISY-CHAIN INTERRUPT STRUCTURE PROVIDES FULLY VECTORED, PRIORITIZED INTERRUPTS WITHOUT EXTERNAL LOGIC. THE CTC MAY ALSO BE USED AS AN INTERRUPT CONTROLLER
- INTERFACE DIRECTLY TO THE Z80 CPU OR FOR BAUD RATE GENERATION - TO THE Z80 SIO



B/F
DIP-28
(Plastic and Frit Seal)

D
DIP-28
(Ceramic)



C
PLCC44
(Plastic)

(Ordering Information at the end of the datasheet)

DESCRIPTION

The Z80 CTC four-channel counter/timer can be programmed by system software for a broad range of counting and timing applications. The four independently programmable channels of the Z80 CTC satisfy common microcomputer system requirements for event counting, interrupt and interval timing, and general clock rate generation.

System design is simplified because the CTC connects directly to both the Z80 CPU and the Z80 SIO with no additional logic. In larger systems, address decoders and buffers may be required.

Programming the CTC is straightforward : each channel is programmed with two bytes ; a third is necessary when interrupts are enabled. Once started, the CTC counts down, reloads its time constant automatically, and resumes counting. Software timing loops are completely eliminated. Interrupt processing is simplified because only one vector need be specified ; the CTC internally generates a unique vector for each channel.

The Z80 CTC requires a single + 5V power supply and the standard Z80 single-phase system clock. It is fabricated with n-channel silicon-gate depletion-load technology, and packaged in a 28-pin plastic or ceramic DIP.

LOGIC FUNCTIONS

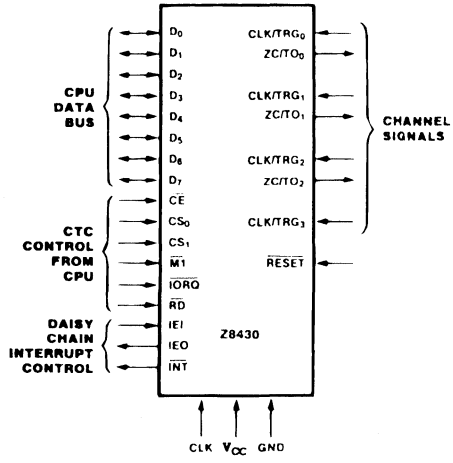


Figure 1 : Dual in Line Pin Configuration.

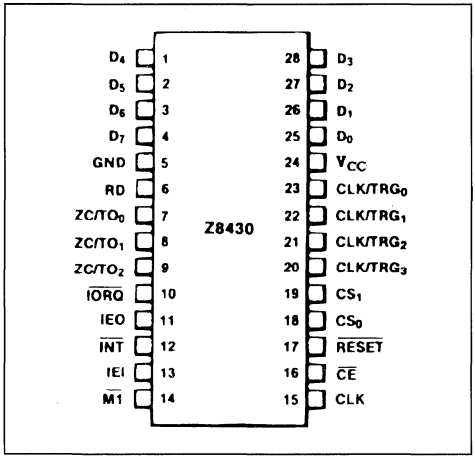
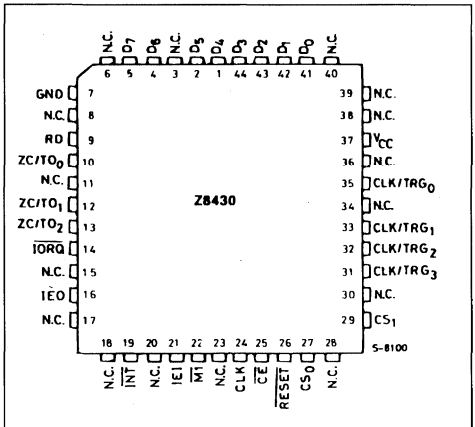


Figure 2 : Chip-Carrier Pin Configuration.



NC = No connection.

FUNCTIONAL DESCRIPTION

The Z80 CTC has four independent counter/timer channels. Each channel is individually programmed with two words : a control word and a time-constant word. The control word selects the operating mode (counter or timer), enables or disables the channel interrupt, and selects certain other operating parameters. If the timing mode is selected, the control word also sets a prescaler, which divides the system clock by either 16 or 256. The time-constant word is a value from 1 to 256.

During operation, the individual counter channel counts down from the preset time constant value. In counter mode operation the counter decrements on each of the CLK/TRG input pulses until zero count is reached. Each decrement is synchronized by the system clock. For counts greater than 256, more than one counter can be cascaded. At zero count, the down-counter is automatically reset with the time constant value.

The timer mode determines time intervals as small as 4 µs (Z80A) or 6.4 µs (Z80) without additional logic or software timing loops. Time intervals are generated by dividing the system clock with a prescaler that decrements a preset down-counter.

Thus, the time interval is an integral multiple of the clock period, the prescaler value (16 or 256) and the time constant that is preset in the down-counter. A timer is triggered automatically when its time constant value is programmed, or by an external CLK/TRG input.

Three channels have two outputs that occur at zero count. The first output is a zero-count/timeout pulse at the ZC/TO output. The fourth channel (Channel 3) does not have a ZC/TO output ; interrupt request is the only output available from Channel 3.

The second output is Interrupt Request ($\overline{INT}$), which occurs if the channel has its interrupt enabled during programming. When the Z80 CPU acknowledges Interrupt Request, the Z80 CTC places an interrupt vector on the data bus.

The four channels of the Z80 CTC are fully prioritized and fit into four contiguous slots in a standard Z80 daisy-chain interrupt structure. Channel 0 is the highest priority and Channel 3 the lowest. Interrupts can be individually enabled (or disabled) for each of the four channels.

ARCHITECTURE

The CTC has four major elements, as shown in figure 3.

- CPU bus I/O
- Channel control logic
- Interrupt logic
- Counter/timer circuits

CPU BUS I/O

The CPU bus I/O circuit decodes the address inputs, and interfaces the CPU data and control signals to the CTC for distribution on the internal bus.

INTERNAL CONTROL LOGIC

The CTC internal control logic controls overall chip operating functions such as the chip enable, reset, and read/write logic.

INTERRUPT LOGIC

The interrupt control logic ensures that the CTC interrupts interface properly with the Z80 CPU interrupt system. The logic controls the interrupt priority of the CTC as a function of the IEI signal. If IEI is High, the CTC has priority. During interrupt processing, the interrupt logic holds IEO Low, which inhibits the interrupt operation on lower priority devices. If the IEI input goes Low, priority is relinquished and the interrupt logic drives IEO Low.

If a channel is programmed to request an interrupt, the interrupt logic drives IEO Low at the zero count, and generates an $\overline{INT}$ signal to the Z80 CPU. When the Z80 CPU responds with interrupt acknowledge ($\overline{M1}$ and $\overline{IORQ}$), then the interrupt logic arbitrates the CTC internal priorities, and the interrupt control logic places a unique interrupt vector on the data bus.

If an interrupt is pending, the interrupt logic holds IEO Low. When the Z80 CPU issues a Return From Interrupt (RETI) instruction, each peripheral device decodes the first byte (ED_{16}). If the device has a pending interrupt, it raises IEO (High) for one $\overline{M1}$ cycle. This ensures that all lower priority devices can decode the entire RETI instruction and reset properly.

COUNTER/TIMER CIRCUITS

The CTC has four independent counter/timer circuits, each containing the logic shown in figure 4.

CHANNEL CONTROL LOGIC

The channel control logic receives the 8-bit channel control word when the counter/timer channel is programmed. The channel control logic decodes the control word and sets the following operating conditions :

- Interrupt enable (or disable)
- Operating mode (timer or counter)
- Timer mode prescaler factor (16 or 256)
- Active slope for CLK/TRG input
- Timer mode trigger (automatic or CLK/TRG input)
- Time constant data word to follow
- Software reset

TIME CONSTANT REGISTER

When the counter/timer channel is programmed, the time constant register receives and stores an 8-bit time constant value, which can be anywhere from 1 to 256 (0 = 256). This constant is automatically loaded into the down-counter when the counter/time channel is initialized, and subsequently after each zero count.

PRESCALER

The prescaler, which is used only in timer mode, divides the system clock frequency by a factor of either 16 or 256.

The prescaler output clocks the down-counter during timer operation. The effect of the prescaler on the down-counter is a multiplication of the system clock period by 16 or 256. The prescaler factor is programmed by bit 5 of the channel control word.

Figure 3 : Functional Block Diagram.

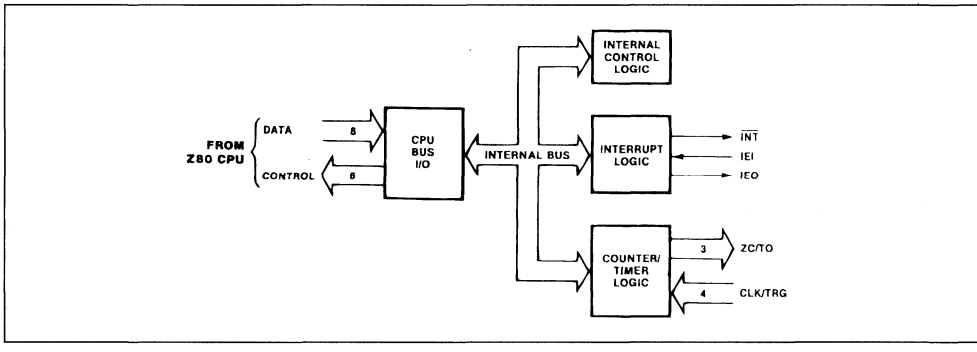
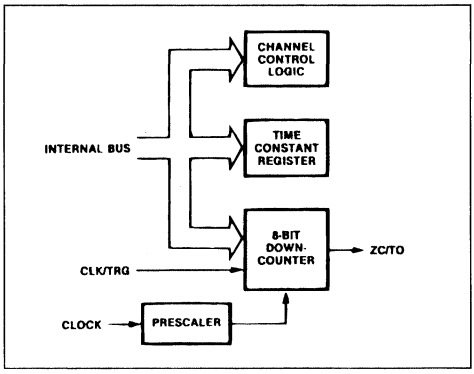


Figure 4 : Counter/Timer Block Diagram.



DOWN-COUNTER

Prior to each count cycle, the down-counter is loaded with the time constant register contents. The counter is then decremented one of two ways, depending on operating mode :

- By the prescaler output (timer mode)
- By the trigger pulses into the CLK/TRG input (counter mode)

Without disturbing the down-count, the Z80 CPU can read the count remaining at any time by performing an I/O read operation at the port address assigned to the CTC channel. When the down-counter reaches the zero count, the ZC/TO output generates a positive-going pulse. When the interrupt is enabled, zero count also triggers an interrupt request signal (INT) from the interrupt logic.

PROGRAMMING

Each Z80 CTC channel must be programmed prior to operation. Programming consists of writing two words to the I/O port that corresponds to the desired channel. The first word is a control word that selects the operating mode and other parameters ; the second word is a time constant, which is a binary data word with a value from 1 to 256. A time constant word must be preceded by a channel control word.

After initialization, channels may be reprogrammed at any time. If updated control and time constant words are written to a channel during the count operation, the count continues to zero before the new time constant is loaded into the counter.

If the interrupt on any Z80 CTC channel is enabled, the programming procedure should also include an interrupt vector. Only one vector is required for all four channels, because the interrupt logic automatically modifies the vector for the channel requesting service.

A control word is identified by a 1 in bit 0. A 1 in bit 2 indicates a time constant word is to follow. Interrupt vectors are always addressed to Channel 0, and identified by a 0 in bit 0.

ADDRESSING

During programming, channels are addressed with the channel select pins CS_1 and CS_2. A 2-bit binary code selects the appropriate channel as shown in the following table.

Channel	CS_1	CS_0
0	0	0
1	0	1
2	1	0
3	1	1

RESET

The CTC has both hardware and software resets. The hardware reset terminates all down-counts and disables all CTC interrupts by resetting the interrupt bits in the control registers. In addition, the ZC/TO and Interrupt outputs go inactive, IEO reflects IEI, and D_0-D_7 go to the high-impedance state. All channels must be completely reprogrammed after a hardware reset.

The software reset is controlled by bit 1 in the channel control word. When a channel receives a software reset, it stops counting. When a software reset is used, the other bits in the control word also change the contents of the channel control register. After a software reset a new time constant word must be written to the same channel.

If the channel control word has both bits D_1 and D_2 set to 1, the addressed channel stops operating, pending a new time constant word. The channel is ready to resume after the new constant is programmed. In timer mode, if $D_3 = 0$, operation is triggered automatically when the time constant word is loaded.

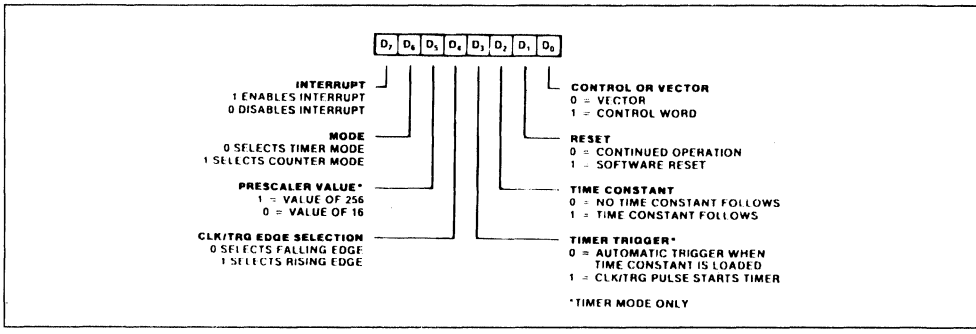
CHANNEL CONTROL WORD PROGRAMMING

The channel control word is shown in figure 5. It sets the modes and parameters described below.

INTERRUPT ENABLE. D_7 enables the interrupt, so that an interrupt output (INT) is generated at zero count. Interrupts may be programmed in either mode and may be enabled or disabled at any time.

OPERATING MODE. D_6 selects either timer or counter mode.

Figure 5 : Channel Control Word.



PRESCALER FACTOR. *(Timer Mode Only).* D_5 selects factor - either 16 or 256.

TRIGGER SLOPE. D_4 selects the active edge or slope of the CLK/TRG input pulses. Note that reprogramming the CLK/TRG slope during operation is equivalent to issuing an active edge. If the trigger slope is changed by a control word update while a channel is pending operation in timer mode, the result is the same as a CLK/TRG pulse and the timer starts. Similarly, if the channel is in counter mode, the counter decrements.

TRIGGER MODE. *(Timer Mode Only).* D_3 selects the trigger mode for timer operation. When D_3 is reset to 0, the timer is triggered automatically. The time constant word is programmed during an I/O write operation, which takes one machine cycle. At the end of the write operation there is a setup delay of one clock period. The timer starts automatically (decrements) on the rising edge of the second clock pulse (T_2) of the machine cycle following the write operation. Once started, the timer runs continuously. At zero count the timer reloads automatically and continues counting without interruption or delay, until stopped by a reset.

When D_3 is set to 1, the timer is triggered externally through the CLK/TRG input. The time constant word is programmed during an I/O write operation, which takes one machine cycle. The timer is ready for operation on the rising edge of the second clock pulse (T_2) of the following machine cycle. Note that the first timer decrement follows the active edge of the CLK/TRG pulse by a delay time of one clock cycle if a minimum setup time to the rising edge of clock is met. If this minimum is not met, the delay is extended by another clock period. Consequently, for immediate triggering, the CLK/TRG input must precede T_2 by one clock cycle plus its minimum setup time. If the minimum time is not met, the timer will start on the third clock cycle (T_3).

Once started the timer operates continuously, without interruption or delay, until stopped by a reset.

TIME CONSTANT TO FOLLOW. A 1 in D_2 indicates that the next word addressed to the selected channel is a time constant data word for the time constant register. The time constant word may be written at any time.

A 0 in D_2 indicates no time constant word is to follow. This is ordinarily used when the channel is already in operation and the new channel control word is an update. A channel will not operate without a time constant value. The only way to write a time constant value is to write a control word with D_2 set.

SOFTWARE RESET. Setting D_1 to 1 causes a software reset, which is described in the Reset section.

CONTROL WORD. Setting D_0 to 1 identifies the word as a control word.

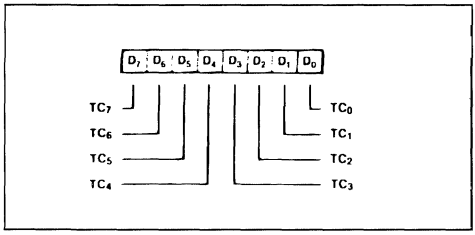
TIME CONSTANT PROGRAMMING

Before a channel can start counting it must receive a time constant word from the CPU. During programming or reprogramming, a channel control word in which bit 2 is set must precede the time constant word to indicate that the next word is a time constant. The time constant word can be any value from 1 to 256 (figure 6). Note that 00_{16} is interpreted as 256.

In timer mode, the time interval is controlled by three factors :

- The system clock period (ϕ)
- The prescaler factor (P), which multiplies the interval by either 16 or 256
- The time constant (T), which is programmed into the time constant register.

Figure 6 : Time Constant Word.

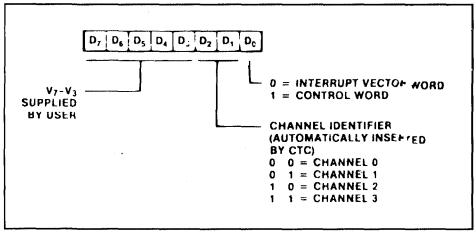


Consequently, the time interval is the product of ϕ x P x T. The minimum timer resolution is 16 x ϕ (4 µs with a 4 MHz clock). The maximum timer interval is 256 x ϕ x 256 (16.4 ms with a 4 MHz clock). For longer intervals timers may be cascaded.

INTERRUPT VECTOR PROGRAMMING

If the Z80 CTC has one or more interrupts enabled, it can supply interrupt vectors to the Z80 CPU. To do so, the Z80 CTC must be preprogrammed with the most-significant five bits of the interrupt vector. Programming consists of writing a vector word to the I/O port corresponding to the Z80 CTC Channel 0. Note that D_0 of the vector word is always zero, to distinguish the vector from a channel control word. D_1 and D_2 are not used in programming the vector word. These bits are supplied by the interrupt logic to identify the channel requesting interrupt service with a unique interrupt vector (figure 7). Channel 0 has the highest priority.

Figure 7 : Interrupt Vector Word.



PIN DESCRIPTION

$\overline{CE}$. *Chip Enable* (Input, Active Low). When enabled the CTC accepts control words, interrupt vectors, or time constant data words from the data bus during an I/O write cycle ; or transmits the contents of the down-counter to the CPU during an I/O read cycle. In most applications this signal is decoded from the eight least significant bits of the address bus for any of the four I/O port addresses that are mapped to the four counter-timer channels.

CLK. *System Clock* (Input). Standard single-phase Z80 system clock.

CLK/TRG0-CLK/TRG3. *External Clock/Timer Trigger* (Input, user-selectable Active High or Low). Four pins corresponding to the four Z80 CTC channels. In counter mode, every active edge on this pin decrements the down-counter. In timer mode, an active edge starts the timer.

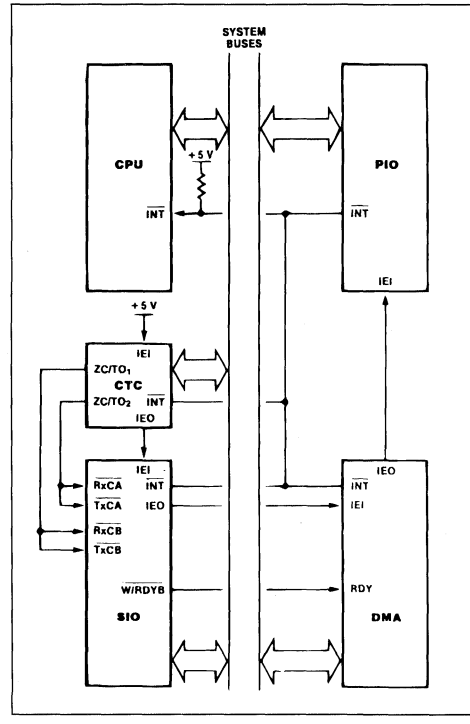
CS_0-CS_1. *Channel Select* (Inputs Active High). Two-bit binary address code selects one of the four CTC channels for an I/O write or read (usually connected to A_0 and A_1).

D_0-D_7. *System Data Bus* (Bidirectional, 3-state). Transfers all data and commands between the Z80 CPU and the Z80 CTC.

IEI. *Interrupt Enable In* (Input, Active High). A High indicates that no other interrupting devices of higher priority in the daisy chain are being serviced by the Z80 CPU.

IEO. *Interrupt Enable Out* (Output, Active High). High only if IEI is High and the Z80 CPU is not ser-

Figure 8 : A Typical Z80 Environment.



vicing an interrupt from any Z80 CTC channel. IEO blocks lower priority devices from interrupting while a higher priority interrupting device is being serviced.

INT. *Interrupt Request* (Output, Open Drain, Active Low). Low when any Z80 CTC channel that has been programmed to enable interrupts has a zero-count condition in its down-counter.

IORQ. *Input/Output Request* (Input from CPU, Active Low). Used with CE and RD to transfer data and channel control words between the Z80 CPU and the Z80 CTC. During a write cycle, IORQ and CE are active and RD inactive. The Z80 CTC does not receive a specific write signal ; rather, it internally generates its own from the inverse of an active RD signal. In a read cycle, IORQ, CE and RD are active ; the contents of the down-counter are read by the Z80 CPU. If IORQ and M1 are both true, the CPU is acknowledging an interrupt request, and the highest priority interrupting channel places its interrupt vector on the Z80 data bus.

M1. *Machine Cycle One* (Input from CPU, Active Low). When M1 and IORQ are active, the Z80 CPU is acknowledging an interrupt. The Z80 CTC then places an interrupt vector on the data bus if it has highest priority, and if a channel has requested an interrupt (INT).

RD. *Read Cycle Status* (Input, Active Low). Used in conjunction with IORQ and CE to transfer data and channel control words between the Z80 CPU and the Z80 CTC.

RESET. *Reset* (Input Active Low). Terminates all down-counts and disables all interrupts by resetting the interrupt bits in all control registers ; the ZC/TO and the Interrupt outputs go inactive ; IEO reflects IEI ; D_0-D_7 go to the high-impedance state.

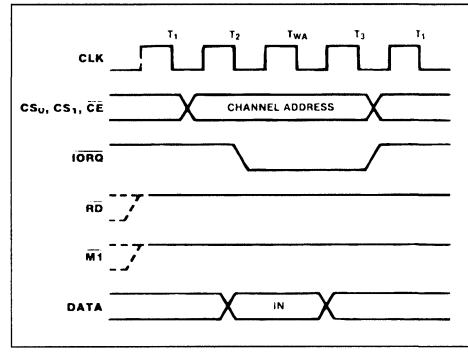
ZC/TO$_0$-ZC/TO$_2$. *Zero Count/Timeout* (Output, Active High). Three ZC/TO pins corresponding to Z80 CTC channels 2 through 0 (Channel 3 has no ZC/TO pin). In both counter and timer modes the output is an active High pulse when the down-counter decrements to zero.

TIMING

READ CYCLE TIMING

Figure 9 shows read cycle timing. This cycle reads the contents of a down-counter without disturbing the count. During clock cycle T_2, the Z80 CPU initiates a read cycle by driving the following inputs.

Figure 9 : Read Cycle Timing.

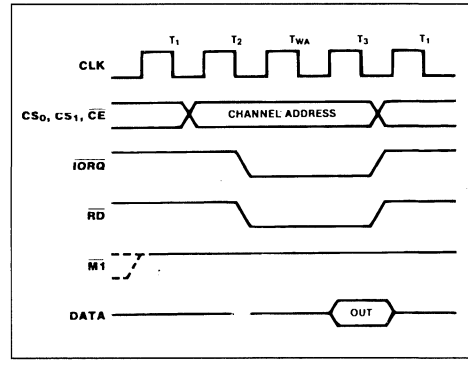


WRITE CYCLE TIMING

Figure 10 shows write cycle timing for loading control, time constant or vector words.

The CTC does not have a write signal input, so it generates one internally when the read (RD) input is High during T_1. During T_2 IORQ and CE inputs are Low : RD, IORQ, and CE. A 2-bit binary code at inputs CS1 and CS0 selects the channel to be read. M1 must be High to distinguish this cycle from an interrupt acknowledge. No additional wait states are allowed.

Figure 10 : Write Cycle Timing.

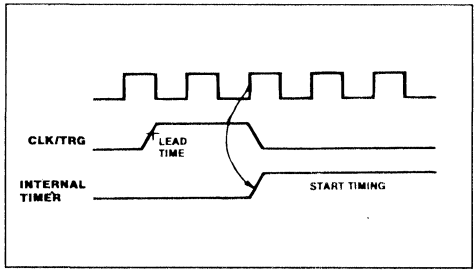


Low. M1 must be High to distinguish a write cycle from an interrupt acknowledge. A 2-bit binary code at inputs CS_1 and CS_0 selects the channel to be addressed, and the word being written is placed on the Z80 data bus. The data word is latched into the appropriate register with the rising edge of clock cycle T_3.

TIMER OPERATION

In the timer mode, a CLK/TRG pulse input starts the timer (figure 11) on the second succeeding rising edge of CLK. The trigger pulse is asynchronous and it must have a minimum width. A minimum lead time (210 ns) is required between the active edge of the CLK/TRG and the next rising edge of CLK to enable the prescaler on the following clock edge. If the CLK/TRG edge occurs closer than this, the initiation of the timer function is delayed one clock cycle. This corresponds to the startup timing discussed in the programming section. The timer can also be started automatically if so programmed by the channel control word.

Figure 11 : Timer Mode Timing.

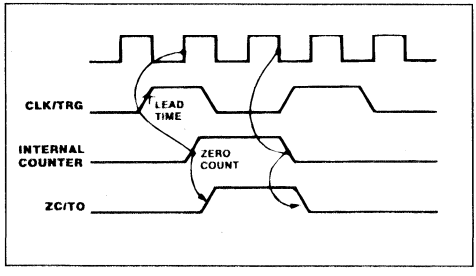


COUNTER OPERATION

In the counter mode, the CLK/TRG pulse input decrements the downcounter. The trigger is asynchronous, but the count is synchronized with CLK. For the decrement to occur on the next rising edge of CLK, the trigger edge must precede CLK by a minimum lead time as shown in figure 12. If the lead time is less than specified, the count is delayed by one clock cycle. The trigger pulse must have a minimum width, and the trigger period must be at least twice the clock period.

The ZC/TO output occurs immediately after zero count, and follows the rising CLK edge.

Figure 12 : Counter Mode Timing.



INTERRUPT OPERATION

The Z80 CTC follows the Z80 system interrupt protocol for nested priority interrupts and return from interrupt, wherein the interrupt priority of a peripheral is determined by its location in a daisy chain. Two lines - IEI and IEO - in the CTC connect it to the system daisy chain. The device closest to the + 5 V supply has the highest priority (figure 13). For additional information on the Z80 interrupt structure, refer to the Z80 CPU Technical Manual.

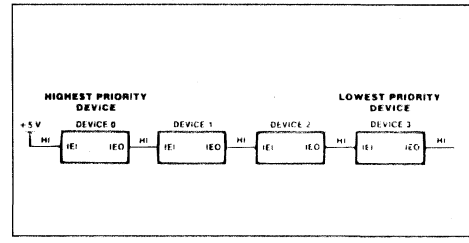
Within the Z80 CTC, interrupt priority is predetermined by channel number : Channel 0 has the highest priority, and Channel 3 the lowest. If a device or channel is being serviced with an interrupt routine, it cannot be interrupted by a device or channel with lower priority until service is complete. Higher priority devices or channels may interrupt the servicing of lower priority devices or channels.

A Z80 CTC channel may be programmed to request an interrupt every time its down-counter reaches zero. Note that the CPU must be programmed for interrupt mode 2. Some time after the interrupt request, the CPU sends an interrupt acknowledge. The CTC interrupt control logic determines the highest priority channel that is requesting an interrupt. Then, if the CTC IEI input is High (indicating that it has priority within the system daisy chain) it places an 8-bit interrupt vector on the system data bus. The high-order five bits of this vector were written to the CTC during the programming process ; the next two bits are provided by the CTC interrupt control logic as a binary code that identifies the highest priority channel requesting an interrupt ; the low-order bit is always zero.

INTERRUPT ACKNOWLEDGE TIMING

Figure 14 shows interrupt acknowledge timing. After an interrupt request, the Z80 CPU sends an interrupt acknowledge (M1 and IORQ). All channels are inhibited from changing their interrupt request status when M1 is active - about two clock cycles earlier than IORQ. RD is High to distinguish this cycle from an instruction fetch.

Figure 13 : Daisy-Chain Interrupt Priorities.



The CTC interrupt logic determines the highest priority channel requesting an interrupt. If the CTC interrupt enable input (IEI) is High, the highest priority interrupting channel within the CTC places its interrupt vector on the data bus when IORQ goes Low. Two wait states (T_{WA}) are automatically inserted at this time to allow the daisy chain to stabilize. Additional wait states may be added.

RETURN FROM INTERRUPT TIMING

At the end of an interrupt service routine the RETI (Return From Interrupt) instruction initializes the daisy chain enable lines for proper control of nested priority interrupt handling. The CTC decodes the 2-byte RETI code internally and determines whether it is intended for a channel being serviced. Figure 15 shows RETI timing.

If several Z80 peripherals are in the daisy chain, IEI settles active (High) on the chip currently being serviced when the opcode ED_{16} is decoded. If the following opcode is $4D_{16}$, the peripheral being serviced is released and its IEO becomes active. Additional wait states are allowed.

Figure 14 : Interrupt Acknowledge Timing.

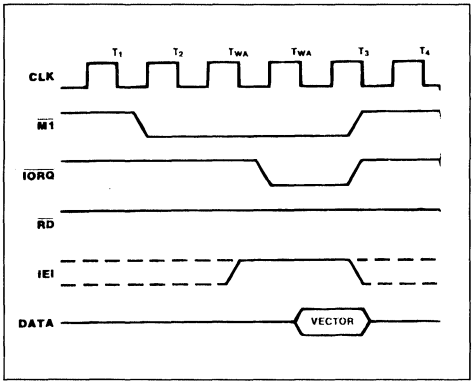
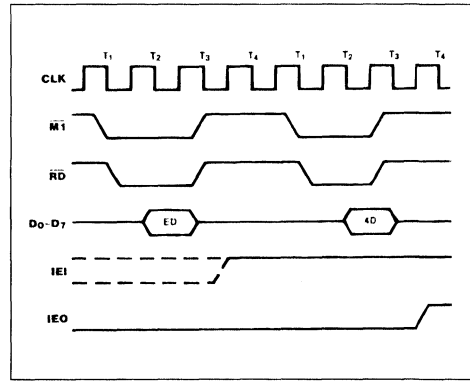


Figure 15 : Return From Interrupt Timing.



ABSOLUTE MAXIMUM RATINGS

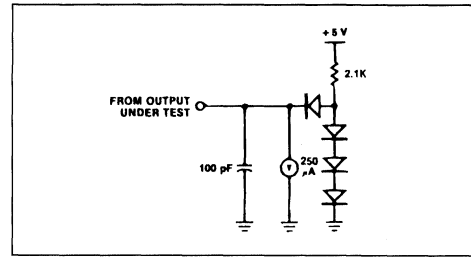
Symbol	Parameter	Value	Unit
V_I	Voltages on all Input and Outputs with respect to GND	− 0.3 to + 7.0	V
T_A	Operation Ambient Temperature As Specified in Ordering Information		
T_{stg}	Storage Temperature	− 65 to + 150	°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only ; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

STANDARD TEST CONDITIONS

The characteristics below apply for the following test conditions, unless otherwise noted. All voltages are referenced to GND (0 V). Positive current flows into the referenced pin. Available operating temperature ranges are :

- 0 °C to + 70 °C,
  + 4.75 V ≤ V_{CC} ≤ + 5.25 V
- – 40 °C to + 85 °C,
  + 4.75 V ≤ V_{CC} ≤ + 5.25 V
- – 55 °C to + 125 °C,
  + 4.75 V ≤ V_{CC} ≤ + 5.5 V



DC CHARACTERISTICS

Symbol	Parameter	Test Conditions	Min.	Max.	Unit
V_{ILC}	Clock Input Low Voltage		– 0.3	0.45	V
V_{IHC}	Clock Input High Voltage		V_{CC} – 0.6	V_{CC}+0.3	V
V_{IL}	Input Low Voltage		+ 0.3	0.8	V
V_{IH}	Input High Voltage		+ 2.0	V_{CC}	V
V_{OL}	Output Low Voltage	I_{OL} = 2.0 mA		0.4	V
V_{OH}	Output High Voltage	I_{OH} = – 250 µA	+ 2.4		V
I_{CC}	Power Supply Current			100	mA
I_{LI}	Input Leakage Current	V_{IN} = 0 to V_{CC}	– 10	10	µA
I_{LO}	3-State Output Leakage Current in Float	V_{OUT} = 0.4 to V_{CC}	– 10	10	µA
I_{OHD}	Darlington Drive Current	V_{OH} = 1.5, R_{EXT} = 390 Ω	– 1.5		mA

CAPACITANCE

Symbol	Parameter	Notes	Min.	Max.	Unit
CLK	Clock Capacitance	Unmeasured pins returned to ground.		20	pF
C_{IN}	Input Capacitance			5	pF
C_{OUT}	Output Capacitance			10	pF

T_A = 25 °C, f = 1 MHz.

AC CHARACTERISTICS

N°	Symbol	Parameter	Notes	Z8430		Z8430A		Z8430B	
				Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
1	TcC	Clock Cycle Time		400	(1)	250	(1)	165	(1)
2	TwCH	Clock Width (high)		170	2000	105	2000	65	2000
3	TwCl	Clock Width (low)		170	2000	105	2000	65	2000
4	TfC	Clock Fall Time			30		30		20
5	TrC	Clock Rise Time			30		30		20
6	th	All Hold Times		0		0		0	
7	TsCS(C)	CS to Clock ↑ Setup Time		250		160		100	
8	TsCE(C)	$\overline{CE}$ to Clock ↑ Setup Time		200		150		100	
9	TsIO(C)	$\overline{IORQ}$ ↓ ot Clock ↑ Setup Time		250		115		70	
10	TsRD(C)	$\overline{RD}$ ↓ to Clock ↑ Setup Time		240		115		70	
11	TdC(DO)	Clock ↑ to Data Out Delay	(2)		240		200		130
12	tdC(DOz)	Clock ↓ to Data Out Float Delay			230		110		90
13	TsDI(C)	Data in to Clock ↑ Setup Time		60		50		40	
14	TsMI(C)	$\overline{MI}$ to Clock ↑ Setup Time		210		90		70	
15	TdMI(IEO)	$\overline{MI}$ ↓ to IEO ↓ Delay (interrupt immediately preceding $\overline{MI}$)	(3)		300		190		130
16	TdIO(DOI)	$\overline{IORQ}$ ↓ to Data Out Delay (INTA cycle)	(2)		340		160		110
17	TdIEI(IEOf)	IEI ↓ to IEO ↓ Delay	(3)		190		130		100
18	TdIEI(IEOr)	IEI ↑ to IEO ↑ Delay (after ED decode)	(3)		220		160		110
19	TdC(INT)	Clock ↑ to $\overline{INT}$ ↓ Delay	(4)		(TcC +200)		(TcC +140)		(TcC +120)
20	TdCLK(INT)	CLK/TRG ↑ to $\overline{INT}$ ↓ TsCTR(C) Satisfied TsCTR(C) Not Satisfied	(5)		(19) +(26) 1 +(19) +(26)		(19) +(26) 1 +(19) +(26)		(19) +(26) 1 +(19) +(26)
21	TcCTR	CLK/TRG Cycle Time	(5)	(2TcC)		(2TcC)		(2TcC)	
22	TrCTR	CLK/TRG Rise Time			50		50		40
23	TfCTR	CLK/TRG Fall Time			50		50		40

(A) 2.5 TcC > (n–2) TdIEI(IEOf) + TdMI(IEO) + TsIEI(IO) + TTL buffer delay, if any.
(B) RESET must be active for a minimum of 3 clock cycles.

Notes : 1. TcC = TwCh + TwCl + TrC + TfC.
2. Increase delay by 10 ns for each 50 pF increase in loading 200 pF maximum for data lines, and 100 pF for control lines.
3. Increase delay by 2 ns for each 10 pF increase in loading 100 pF maximum.
4. Timer mode.
5. Counter mode.
6. RESET must be active for a minimum of 3 clock cycles.

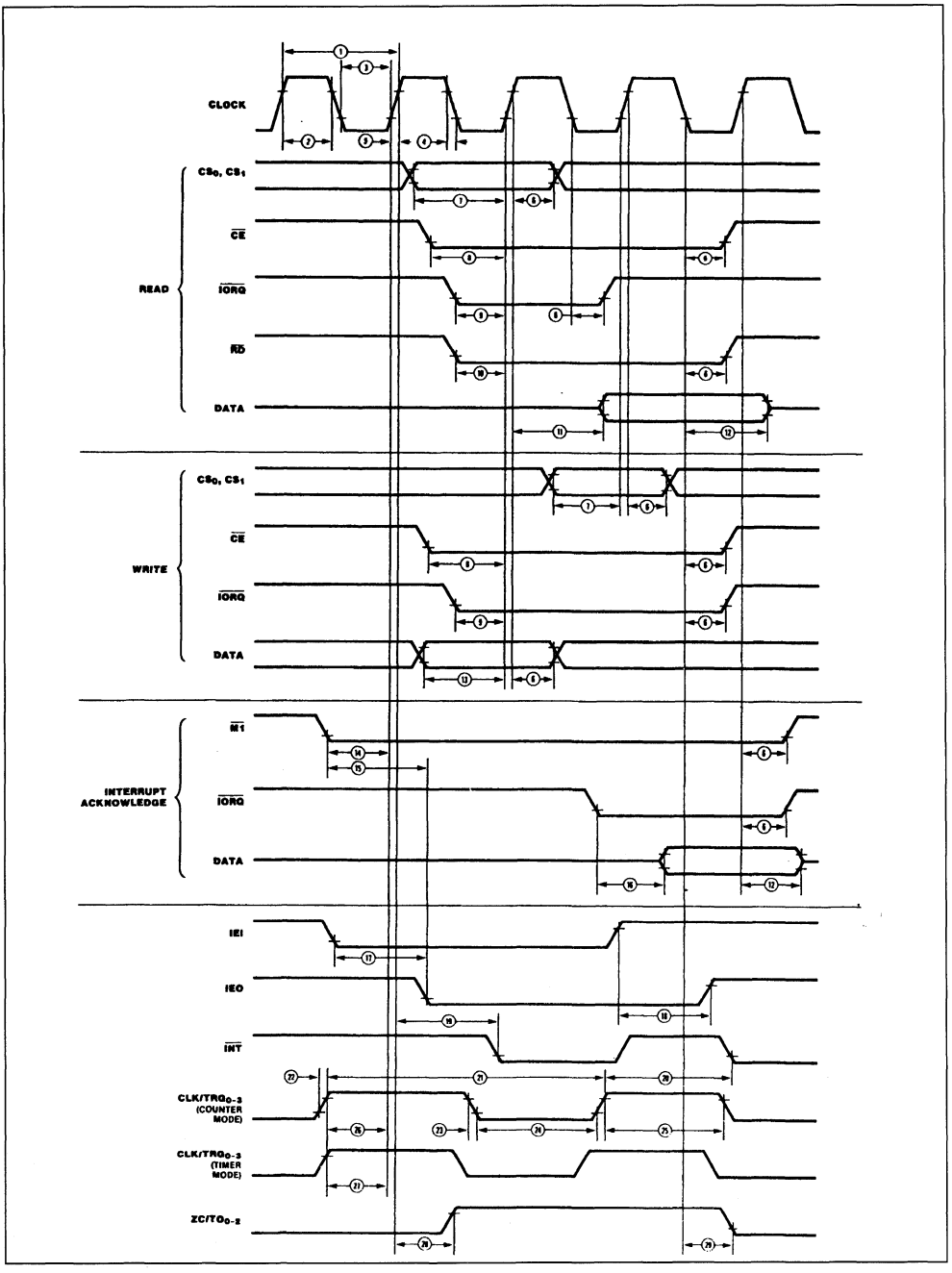
AC CHARACTERISTICS (continued)

N°	Symbol	Parameter	Notes	Z8430		Z8430A		78430B	
				Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
24	TwCTRl	CLK/TRG Width (low)		200		200		120	
25	TwCTRh	CLK/TRG Width (high)		200		200		120	
26	TsCTR(Cs)	CLK/TRG ↑ to Clock ↑ Setup Time for Immediate Count	(5)	300		210		150	
27	TsCTR(Ct)	CLK/TRG ↑ to Clock ↑ Setup Time for enabling of Prescaler on following Clock ↑	(4)	210		210		150	
28	TdC(ZC/TOr)	Clock ↑ to ZC/TO ↑ Delay			260		190		140
29	TdC(ZC/TOf)	Clock ↓ to ZC/TO ↓ Delay			190		190		140

(A) 2.5 TcC > (n–2) TdIEI(IEOf) + TdMI(IEO) + TsIEI(IO) + TTL buffer delay, if any.
(B) RESET must be active for a minimum of 3 clock cycles.

Notes :
1. TcC = TwCh + TwCl + TrC + TfC.
2. Increase delay by 10 ns for each 50 pF increase in loading 200 pF maximum for data lines, and 100 pF for control lines.
3. Increase delay by 2 ns for each 10 pF increase in loading 100 pF maximum.
4. Timer mode.
5. Counter mode.
6. RESET must be active for a minimum of 3 clock cycles.

AC CHARACTERISTICS (continued)

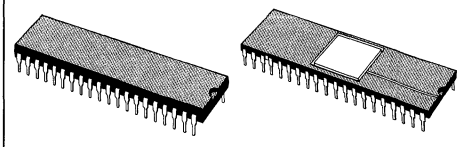


ORDERING INFORMATION

Type	Package	Temp.	Clock	Description
Z8430B1	DIP-28 (plastic)	0/+ 70°C	2.5 MHz	Z80 Counter Timer Control
Z8430F1	DIP-28 (frit seal)	0/+ 70°C		
Z8430D1	DIP-28 (ceramic)	0/+ 70°C		
Z8430D6	DIP-28 (ceramic)	−40/+ 85°C		
Z8430D2	DIP-28 (ceramic)	−55/+125°C		
Z8430C1	PLCC44 (plastic chip-carrier)	0/+ 70°C		
Z8430AB1	DIP-28 (plastic)	0/+ 70°C	4 MHz	
Z8430AF1	DIP-28 (frit seal)	0/+ 70°C		
Z8430AD1	DIP-28 (ceramic)	0/+ 70°C		
Z8430AD6	DIP-28 (ceramic)	−40/+ 85°C		
Z8430AD2	DIP-28 (ceramic)	−55/+125°C		
Z8430AC1	PLCC44 (plastic chip-carrier)	0/+ 70°C		
Z8430BB1	DIP-28 (plastic)	0/+ 70°C	6 MHz	
Z8430BF1	DIP-28 (frit seal)	0/+ 70°C		
Z8430BD1	DIP-28 (ceramic)	0/+ 70°C		
Z8430BD6	DIP-28 (ceramic)	−40/+ 85°C		
Z8430BD2	DIP-28 (ceramic)	−55/+125°C		
Z8430BC1	PLCC44 (plastic chip-carrier)	0/+ 70°C		

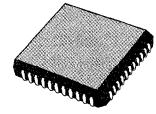
Z80 SIO SERIAL INPUT/OUTPUT CONTROLLER

- TWO INDEPENDENT FULL-DUPLEX CHANNELS, WITH SEPARATE CONTROL AND STATUS LINES FOR MODEMS OR OTHER DEVICES
- DATA RATES OF 0 TO 500K BITS/SECOND IN THE XL CLOCK MODE WITH A 2.5MHz CLOCK (Z8440), OR 0 TO 800K BITS/SECOND WITH A 4.0MHz CLOCK (Z8440A)
- ASYNCHRONOUS PROTOCOLS : EVERYTHING NECESSARY FOR COMPLETE MESSAGES IN 5, 6, 7 OR 8 BITS/CHARACTER. INCLUDES VARIABLE STOP BITS AND SEVERAL CLOCK-RATE MULTIPLIERS ; BREAK GENERATION AND DETECTION ; PARITY ; OVERRUN AND FRAMING ERROR DETECTION
- SYNCHRONOUS PROTOCOLS : EVERYTHING NECESSARY FOR COMPLETE BIT- OR BYTE-ORIENTED MESSAGES IN 5, 6, 7 OR 8 BITS/CHARACTER, INCLUDING IBM BISYNC, SDLC, HDLC, CCITT-X.25 AND OTHERS. AUTOMATIC CRC GENERATION/CHECKING SYNC CHARACTER AND ZERO INSERTION/DELETION, ABORT GENERATION/DETECTION AND FLAG INSERTION
- RECEIVER DATA REGISTERS QUADRUPLY BUFFERED, TRANSMITTER REGISTERS DOUBLY BUFFERED
- HIGHLY SOPHISTICATED AND FLEXIBLE DAISY-CHAIN INTERRUPT VECTORING FOR INTERRUPTS WITHOUT EXTERNAL LOGIC



B/F
DIP-40
(Plastic and Frit-Seal)

D
DIP-40
(Ceramic)



C
PLCC44
(Plastic)

(Ordering Information at the end of the datasheet)

DESCRIPTION

The Z80 SIO Serial Input/Output Controller is a dual-channel data communication interface with extraordinary versatility and capability. Its basic functions as a serial-to-parallel, parallel-to-serial converter/controller can be programmed by a CPU for a broad range of serial communication applications.

The device supports all common asynchronous and synchronous protocols, byte- or bit-oriented and performs all of the functions traditionally done by UARTs, USARTs and synchronous communication controllers combined, plus additional functions traditionally performed by the CPU. Moreover, it does this on two fully-independent channels, with an exceptionally sophisticated interrupt structure that allows very fast transfers.

Full interfacing is provided for CPU or DMA control. In addition to data communication, the circuit can handle virtually all types of serial I/O with fast (or slow) peripheral devices.

While designed primarily as a member of the Z80 family, its versatility makes it well suited to many other CPUs.

The Z80 SIO is an n-channel silicon-gate depletion-load device and uses a single + 5V power supply and the standard Z80 Family single-phase clock.

PIN DESCRIPTIONS

Figures 1 through 6 illustrate the three pin configurations (bonding options) available in the SIO. The constraints of a 40-pin package make it impossible to bring out the Receive Clock (RxC), Transmitt Clock (TxC), Data Terminal Ready (DTR) and Sync (SYNC) signals for both channels. Therefore, either Channel B lacks a signal or two signals are bonded together in the three bonding options offered :

- Z80 SIO-2 lacks $\overline{SYNCB}$
- Z80 SIO-1 lacks $\overline{DTRB}$
- Z80 SIO-0 as a four signal, but $\overline{TxCB}$ and $\overline{RxCB}$ are bonded together

The first bonding option above (SIO-2) is the preferred version for most applications. The Chip-Carrier package version, having a 44-pin facility, resume the three bonding option configurations. It is named Z8444 (figure 7). The pin description are as follows :

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 SIO. Address bit A_0 from the CPU is often used for the selection function.

C/$\overline{D}$. *Control Or Data Select* (Input, High selects Control). This input defines the type of information transfer performed between the CPU and the SIO. A High at this input during a CPU write to the SIO causes the information on the data bus to be interpreted as a command for the channel selected by B/A. A Low at C/D means that the information on the data bus is data. Address bit A_1 is often used for this function.

C/$\overline{E}$. *Chip Enable* (Input, Active Low). A Low level at this input enables the SIO 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 SIO uses the standard Z80 System Clock to synchronize internal signals. This is a single-phase clock.

$\overline{CTSA}$, **$\overline{CTSB}$**. *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 accomodate slow-risetime signals. The SIO detects pulses on these inputs and interrupts the CPU on both logic level transitions. The Schmitt-trigger buffering does not guarantee a specified noise-level margin.

D_0-D_7. *System Data Bus* (Bidirectional, 3-state). The system data bus transfers data and commands between the CPU and the Z80 SIO. D_0 is the least significant bit.

$\overline{DCDA}$, **$\overline{DCDB}$**. *Data Carrier Detect* (Inputs, Active Low). These pins function as receiver enables if the

Figure 1 : Z80 SIO-2 Logic Function.

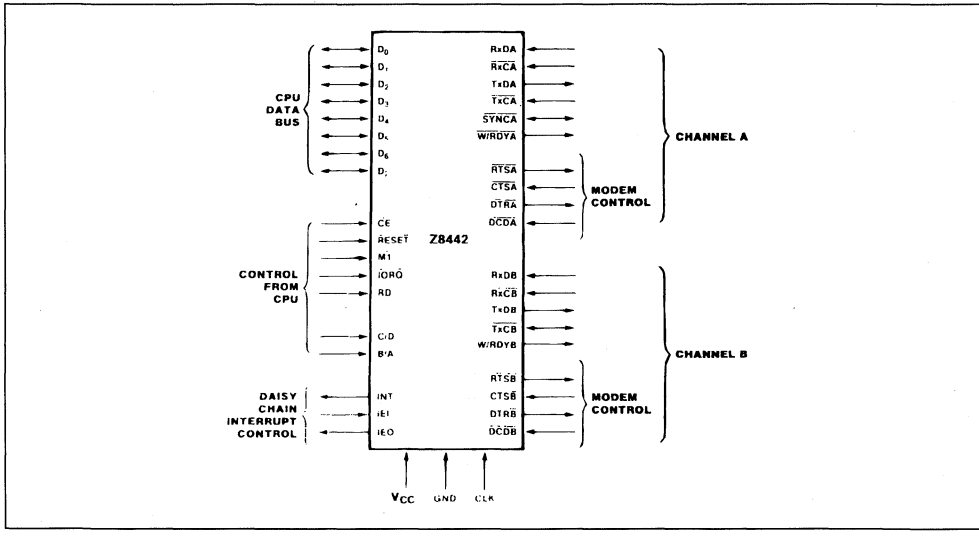
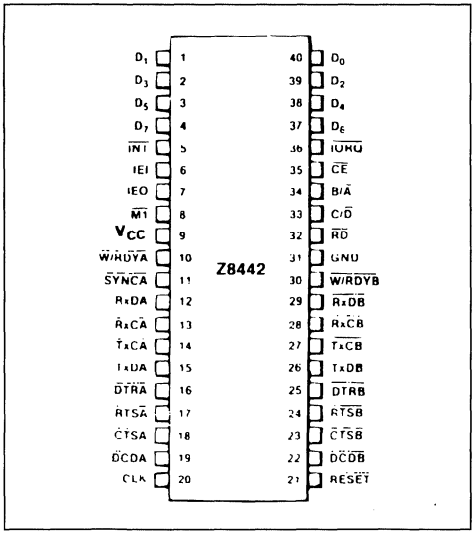


Figure 2 : Z80 SIO-2 Dual in Line Pin Configuration.



SIO is programmed for Auto Enables ; otherwise they may be used as general-purpose input pins. Both pins are Schmitt-trigger buffered to accommodate slow-risetime signals. The SIO detects pulses on these pins and interrupts the CPU on both logic level transitions. Schmitt-trigger buffering does not guarantee a specific noise-level margin.

DTRA, DTRB. *Data Terminal Ready* Outputs, Active Low). These outputs follow the state programmed into Z80 SIO. They can also be programmed as general-purpose outputs.

In the Z80 SIO-1 bonding option, DTRB is omitted.

IEI. *Interrupt Enable In* (Input, Active High). This signal 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 SIO. Thus, this signal blocks lower priority devices form interrupting while a higher priority device is being serviced by its CPU interrupt service routine.

INT. *Interrupt Request* (Output, Open Drain, Active Low). When the SIO is requesting an interrupt, it pulls INT Low.

IORQ. *Input/Output Request* (Input from CPU, Active Low). IORQ is used in conjunction with B/A, C/D, CE and RD to transfer commands and data between the CPU and the SIO. When CE, RD and

Figure 3 : Z80 SIO-I Logic Function.

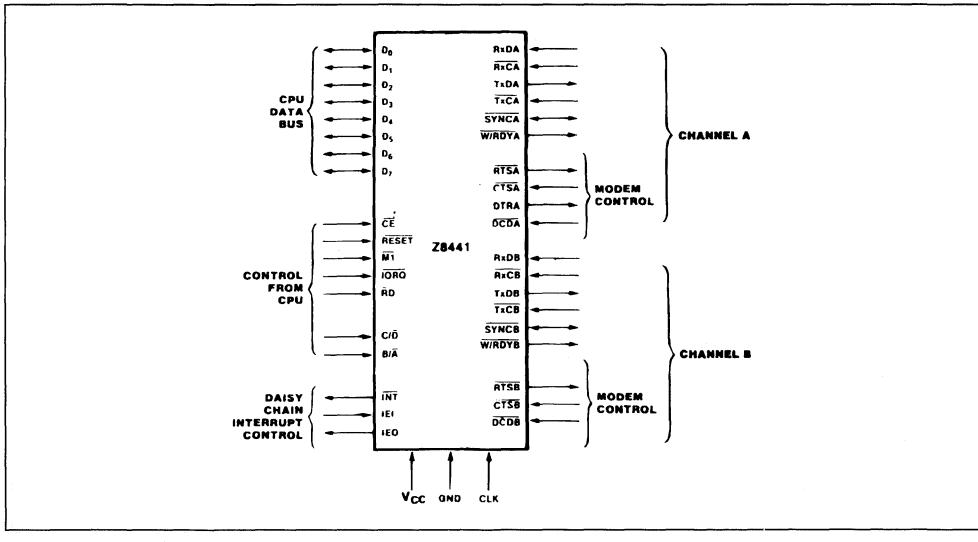
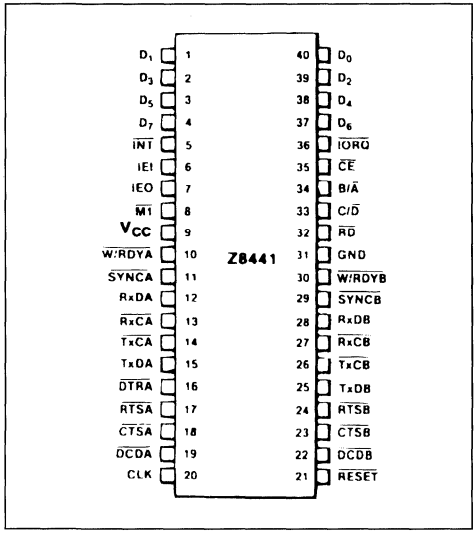


Figure 4 : Z80 SIO-I Dual in Line Pin Configuration.



IORQ are all active, the channel selected by B/A transfers data to the CPU (a read operation). When CE and IORQ are active but RD is inactive, the channel selected by B/A is written to by the CPU with either data or control information as specified by C/D. If IORQ and M1 are active simultaneously, the CPU is acknowledging an interrupt and the SIO automatically places its interrupt vector on the CPU data bus if it is the highest priority device requesting an interrupt.

M1. *Machine Cycle* (Input from Z80 CPU, Active Low). When M1 is active and RD is also active, the Z80 CPU is fetching an instruction from memory ; when M1 is active while IORQ is active, the SIO accepts M1 and IORQ as an interrupt acknowledge if the SIO is the highest priority device that has interrupted the Z80 CPU.

RxCA, **RxCB**. *Receiver Clocks* (Inputs). Receive data is sampled on the rising edge of RxC. The Receive Clocks may be 1, 16, 32 or 64 times the data rate in asynchronous modes. These clocks may be driven by the Z80 CTC Counter Timer Circuit for programmable baud rate generation. Both inputs are Schmitt-trigger buffered (no noise level margin is specified).

In the Z80 SIO-0 bonding option, RxCB is bonded together with TxCB.

RD. *Read Cycle Status* (Input from CPU, Active Low). If RD is active, a memory or I/O read operation is in progress. RD is used with B/A, CE and IORQ to transfer data from the SIO to the CPU.

RxDA, **RxDB**. *Receive Data* (Inputs, Active High). Serial data at TTL levels.

RESET. *Reset* (Input, Active Low). A Low RESET disables both receivers and transmitters, forces

Figure 5 : Z80 SIO-0 Logic Function.

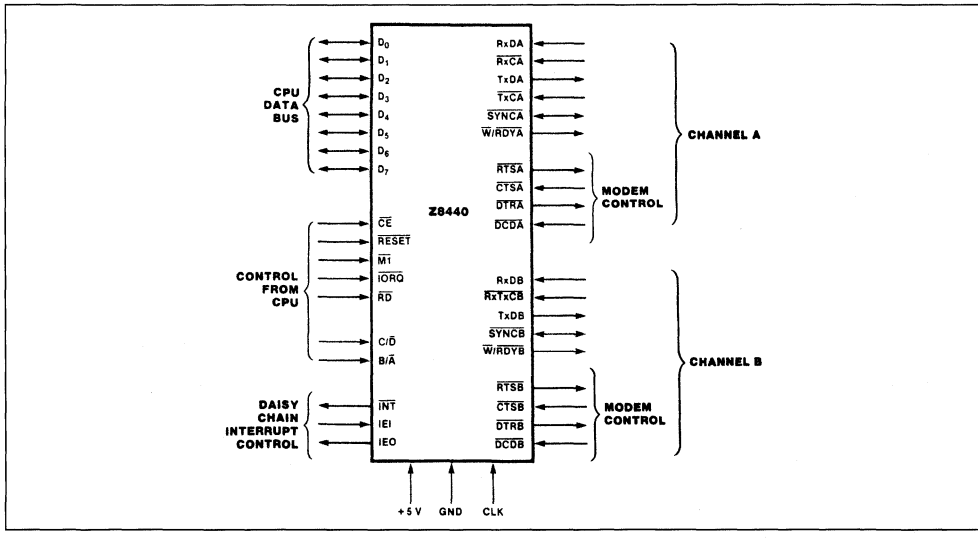


Figure 6 : Z80 SIO-0 Dual in Line Pin Configuration.

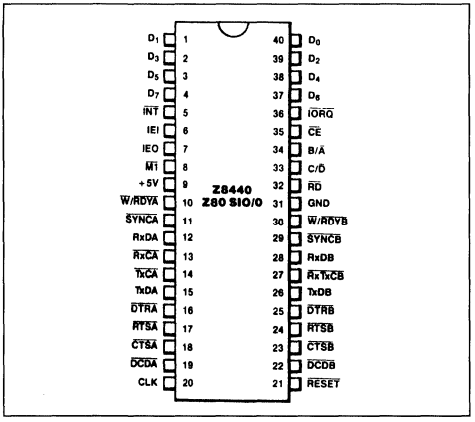
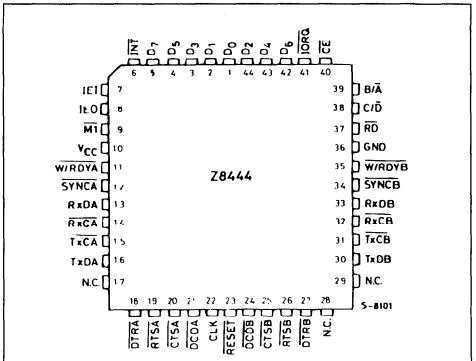


Figure 7 : Chip Carrier Pin Configuration.



Notes : NC = No Connection.
Z80 SIO-0 or Dart in Asynchronous Mode.

TxDA and TxDB marking, forces the modem controls High and disables all interrupts. The control registers must be rewritten after the SIO is reset and before data is transmitted or received.

$\overline{\text{RTSA}}$, $\overline{\text{RTSB}}$. *Request To Send* (Outputs, Active Low). When the RTS bit in Write Register 5 (figure 14) is set, the $\overline{\text{RTS}}$ output goes Low. When the RTS bit is reset in the Asynchronous mode, the output goes High after the transmitter is empty. In Synchronous modes, the $\overline{\text{RTS}}$ pin strictly follows the state of the RTS bit. Both pins can be used as general-purpose outputs.

$\overline{\text{SYNCA}}$, $\overline{\text{SYNCB}}$. *Synchronization* (Inputs/Outputs, Active Low). These pins can act either as inputs or outputs. In the asynchronous receive mode, they are inputs similar to $\overline{\text{CTS}}$ and $\overline{\text{DCD}}$. In this mode, the transitions on these lines affect the state of the Sync/Hunt status bits in Read Register 0 (figure 14), but have no other function. In the External Sync mode, these lines also act as inputs. When external synchronization is achieved, $\overline{\text{SYNC}}$ must be driven Low on the second rising edge of RxC after that rising edge of RxC on which the last bit of the sync character was received. In other words, after the sync pattern is detected, the external logic must wait for two full Receive Clock cycles to activate the $\overline{\text{SYNC}}$ input. Once $\overline{\text{SYNC}}$ is forced Low, it should be kept Low until the CPU informs the external synchronization detect logic that synchronization has been lost or a new message is about to start. Character assembly begins on the rising edge of RxC that immediately precedes the falling edge of $\overline{\text{SYNC}}$ in the External Sync mode.

In the internal synchronization mode (Monosync and Bisync) these pins act as outputs that are active during the part of the receive clock (RxC) cycle in which sync characters are recognized. The sync condition is not latched so these outputs are active each time a sync pattern is recognized, regardless of character boundaries.

In the Z80 SIO-2 bonding option, $\overline{\text{SYNCB}}$ is omitted.

$\overline{\text{TxCA}}$-$\overline{\text{TxCB}}$. *Transmitter Clocks* (Inputs). In asynchronous modes, the Transmitter Clocks may be 1, 16, 32 or 64 times the data rate ; however, the clock multiplier for the transmitted and the receiver must be the same. The Transmit Clock inputs are Schmitt-trigger buffered for relaxed rise- and fall-time requirements (no noise level margin is specified).

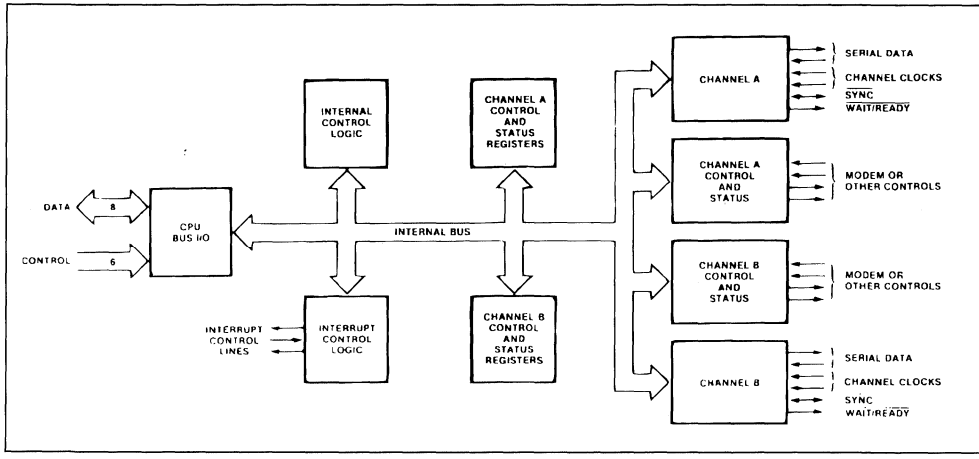
Transmitter Clocks may be driven by the Z80 CTC Counter Timer Circuit for programmable baud rate generation.

In the Z80 SIO-0 bonding option, $\overline{\text{TxCB}}$ is bonded together with RxCB.

TxDA, **TxDB**. *Transmitt Data* (Outputs, Active High). Serial data at TTL levels. TxD changes from the falling edge of TxC.

$\overline{\text{W}}$/RDYA, **$\overline{\text{W}}$/RDYB**. *Wait/Ready A, Wait/Ready B* (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 SIO data rate. The reset state is open drain.

Figure 8 : Block Diagram.



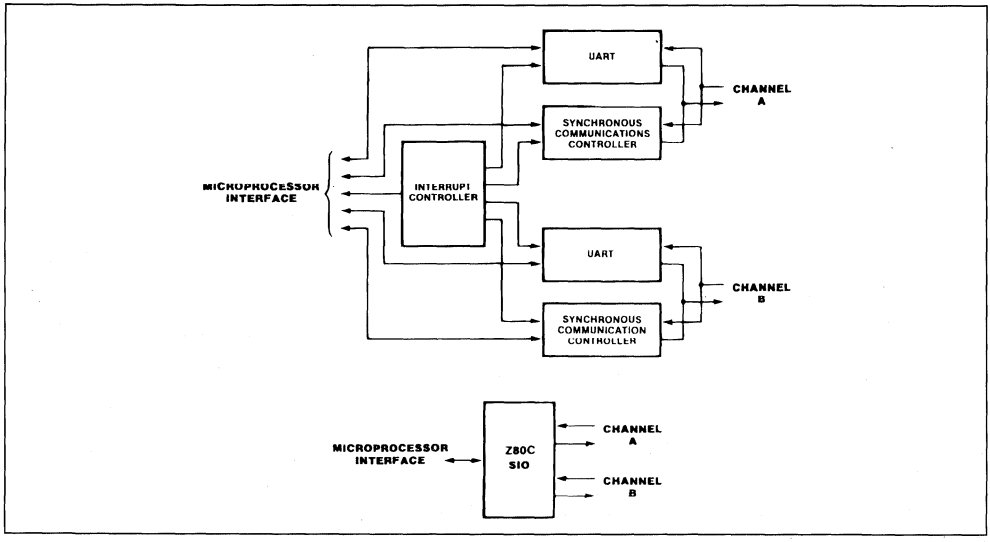
FUNCTIONAL DESCRIPTION

The functional capabilities of the Z80 SIO can be described from two different points of view : as a data communication device, it transmits and receives serial data in a wide variety of data-communication protocols ; as a Z80 family peripheral, it interacts with the Z80 CPU and other peripheral circuits, sharing the data, address and control buses, as well as being a part of the Z80 interrupt structure. As a peripheral to other microprocessors, the SIO offers valuable features such as non-vectored interrupts, polling and simple handshake capability.

Figure 9 illustrates the conventional devices that the SIO replaces.

The first part of the following discussion covers SIO data-communication capabilities ; the second part describes interactions between the CPU and the SIO.

Figure 9 : Conventional Devices Replaced by the Z80 SIO.



DATA COMMUNICATION CAPABILITIES

The SIO provides two independent full-duplex channels that can be programmed for use in any common asynchronous or synchronous data-communication protocol. Figure 10 illustrates some of these protocols. The following is a short description of them. A more detailed explanation of these modes can be found in the *Z80 Family Technical Manual.*

ASYNCHRONOUS MODES

Transmission and reception can be done independently on each channel with five to eight bits per character, plus optional even or odd parity. The transmitters 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 spikerejection mechanism that checks the signal one-half a bit time after a Low level is detected on the receive data input (RxDA or RxDB in figure 6). 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 partial character on which they occurred. Vectored interrupts allow fast servicing of error 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 SIO does not require symmetric transmit and receive clock signals-a feature that allows it to be used with a Z80 CTC or many other clock sources. 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.

In asynchronous modes, the $\overline{\text{SYNC}}$ pin may be programmed as an input that can be used for functions such as monitoring a ring indicator.

SYNCHRONOUS MODES

The SIO supports both byte-oriented and bit oriented synchronous communication.

Synchronous byte-oriented protocols can be handled in several modes that allow character synchronization with an 8-bit sync character (Monosync), any 16-bit sync pattern (Bysinc), or with an external sync signal. Leading sync characters can be removed without interrupting the CPU.

Five-, six- or seven-bit sync characters are detected with 8- or 16-bit patterns in the SIO by overlapping the larger pattern across multiple in-coming sync characters, as shown in figure 11.

CRC checking for synchronous byte-oriented modes is delayed by one character time so the CPU may disable CRC checking on specific characters. This permits implementation of protocols such as IBM Bisync.

Both CRC-16 ($X^{16} + X^{15} + X^2 + 1$) and CCITT ($X^{16} + X^{12} + X^5 + 1$) error checking polynomials are supported. In all non-SDLC modes, the CRC generator is initialized to 0's ; in SDLC modes, it is initialized to 1's. The SIO can be used for interfacing to peripherals such as hard-sectored floppy disk, but it cannot generate or check CRC for IBM-compatible soft-sectored disks. The SIO also provides a feature that automatically transmits CRC data when no other data is available for transmissions. This allows very high-speed transmissions under DMA control with no need for CPU intervention at the end of a message. When there is no data or CRC to send in synchronous modes, the transmitter inserts 8- or 16-bit sync characters regardless of the programmed character length.

The SIO supports synchronous bit-oriented protocols such as SDLC and HDLC by performing automatic flag seding, zero insertion and CRC generation. A special command can be used to abort a frame in transmission. At the end of a message the SIO automatically transmits the CRC and trailing flag when the transmit buffer becomes empty. If a transmit underrun occurs in the middle of a message, an external/status interrupt warns the CPU of this status change so that an abort may be issued. One to eight bits per character can be sent, which allows reception of a message with no prior information about the character structure in the information field of a frame.

The receiver automatically synchronizes on the leading flag of a frame in SDLC or HDLC, and provides a synchronization signal on the $\overline{\text{SYNC}}$ pin ; an interrupt can also be programmed. The receiver can be programmed to search for frames addressed by a single byte to only a specified user-selected address or to a global broadcast address. In this mode, frames that do not match either the user-selected or broadcast address are ignored. The number of address bytes can be extended under software control. For transmitting data, an interrupt on the first received character or on every character can be selected. The receiver automatically deletes all zeroes inserted by the transmitter during character assembly. It also calculates and automatically checks the CRC to validate frame transmission. At the end of transmission, the status of a received frame is available in the status registers.

The SIO can be conveniently used under DMA control to provide high-speed reception or transmission. In reception, for example, the SIO can interrupt the CPU when the first character of a message is received. The CPU then enables the DMA to transfer the message to memory. The SIO then issues an end-of-frame interrupt and the CPU can check the status of the received message. Thus, the CPU is freed for other service while the message is being received.

I/O INTERFACE CAPABILITIES

The SIO offers the choice of polling, interrupt, (vectored or non-vectored) and block-transfers modes to transfer data, status and control information to and from the CPU. The block-transfer mode can also be implemented under DMA control.

POLLING

Two status registers are updated at appropriate times for each function being performed (for example, CRC error-status valid at the end of a message). When the CPU is operated in a polling fashion, one of the SIO's two status registers is used to indicate whether the SIO has some data or needs some data. Depending on the contents of this register, the CPU will either write data, read data, or just go on. Two bits in the register indicate that a data transfer is needed. In addition, error and other conditions are indicate. The second status register (special receive conditions) does not have to be read in a polling sequence, until a character has been received. All interrupt modes are disabled when operating the device in a polled environment.

INTERRUPTS

The SIO has an elaborate interrupt scheme to provide fast interrupt service in real-time applications. A control register and a status register in Channel B contain the interrupt vector. When programmed to do so, the SIO can modify three bits of the interrupt vector in the status register so that in points directly to one of eight interrupt service routines in memory, thereby servicing conditions in both channels and eliminating most of the needs for a status-analysis routine.

Transmitt 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

Figure 11 : Six Bit Sync Character Recognition.

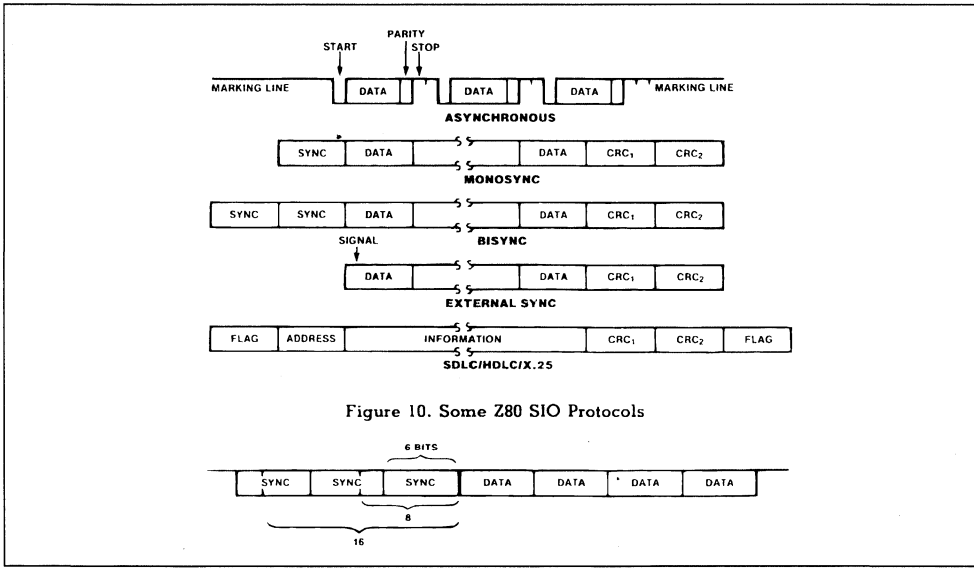


Figure 10. Some Z80 SIO Protocols

priority than Channel B, and with receive, 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). The receiver can interrupt the CPU in one or two ways :

- Interrupt on first received character
- Interrupt on all received characters

Interrupt-on-first-received-character is typically used with the block-transfer mode. Interrupt-on-all-received-characters has the option of modifying the interrupt vector in the event of a parity error. Both of these interrupt modes will also interrupt under special receive conditions on a character or message basis (end-of-frame interrupt in SDLC, for example).

This means that the special-receive condition can cause an interrupt only if the interrupt-on-first-received-character or interrupt-on-all-received-characters mode is selected. In interrupt-on-first-received-character, an interrupt can occur from special-receive conditions (except parity error) after the first-received-character interrupt (example : receive-overrun interrupt).

The main function of the external/status interrupt is to monitor the signal transitions of the Clear To Send (CTS), Data Carrier Detect (DCD) and Synchronization (SYNC) pins (figures 1 through 6). In addition, an external/status interrupt is also caused by a CRC-sending condition or by the detection of a break sequence (asynchronous mode) or abort sequence (SDLC mode) in the data stream. The interrupt caused by the break/abort sequence allows the SIO to interrupt when the break/abort 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/abort condition in external logic.

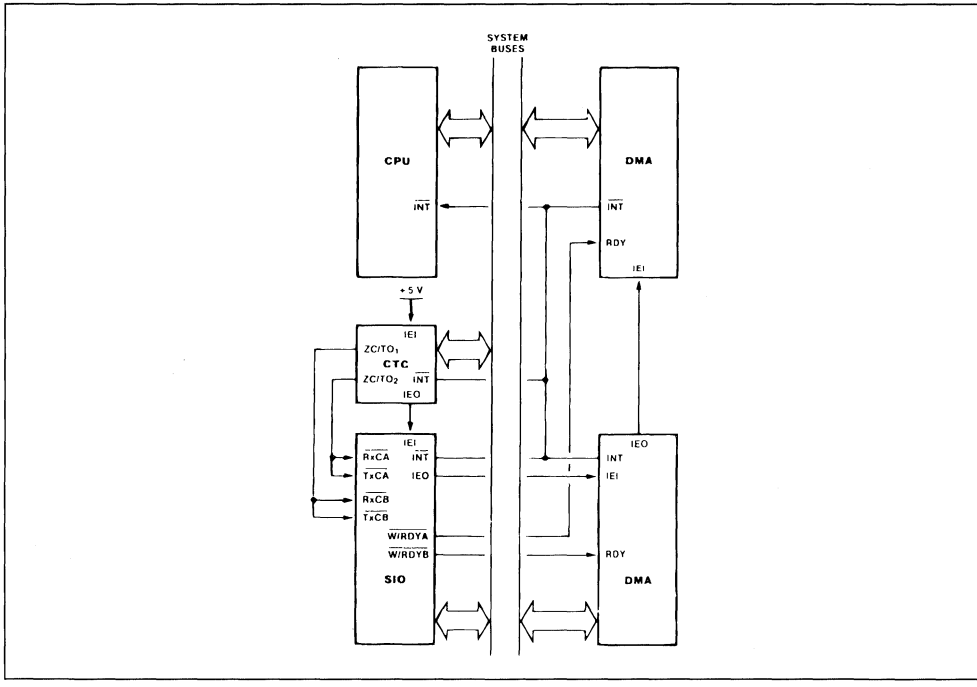
In a Z80 CPU environment (figure 12), SIO interrupt vectoring is "automatic" : the SIO passes its internally-modificable 8-bit interrupt vector to the CPU, which adds an additional 8 bits from its interrupt-vector (I) register to form the memory address of the interrupt-routine table. This table contains the address of the beginning of the interrupt routine itself. The process entails an indirect transfer or CPU control to the interrupt routine, so that the next instruction executed after an interrupt acknowledge by the CPU is the first instruction of the interrupt routine itself.

CPU/DMA BLOCK TRANSFER

The SIO's block-transfer mode accommodates both CPU block transfers and DMA controllers (Z80 DMA or other designs). The block-transfer mode uses the Wait/Ready output signal, which is selected with three bits in an internal control register. The Wait/Ready output signal can be programmed 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 SIO $\overline{\text{READY}}$ output indicates that the SIO is ready to transfer data to or from memory. To the CPU, the $\overline{\text{WAIT}}$ output indicates that the SIO is not ready to transfer data, thereby requesting the CPU to extend the I/O cycle.

Figure 12 : Typical Z80 Environment.



INTERNAL STRUCTURE

The internal structure of the device includes a Z80 CPU interface, internal control and interrupt logic, and two full-duplex channels. Each channel contains its own set of control and status (write and read) registers, and control and status logic that provides the interface to modems or other external devices.

The registers for each channel are designated as follows :

WR0-WR7 - Write Registers 0 through 7
RR0-RR2 - Read Register 0 through 2

The register group includes five 8-bit control registers, two sync-character 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 another 8-bit register (Read Register 2) in Channel B. The bit assignment and functional grouping of each register is configured to simplify and organize the programming process. Table 1 list the functions assigned to each read or write register.

The logic for both channels provides formats, synchronization and validation for data transferred to and from the channel interface. The modem control inputs, Clear To Send (CTS) and Data Carrier Detect (DCD), are monitored by the external control and status logic under program control. All external

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)
Write Register Functions	
WR0	Register pointers, CRC initialize, initialization commands for the various modes, etc.
WR1	Transmitt/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
WR6	Sync Character or SDLC Address Field
WR7	Sync Character or SDLC Flag

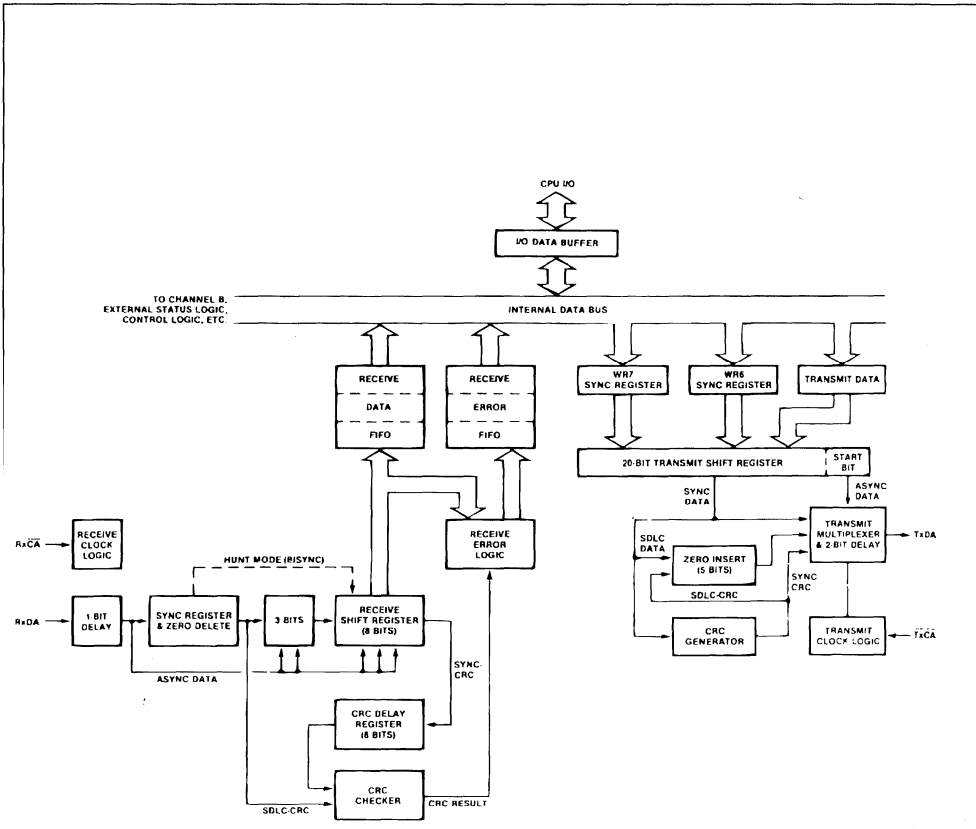
control-and-status-logic signals are general-purpose in nature and can be used for functions other than modem control.

DATA PATH

The transmit and receive data path illustrated for Channel A in figure 13 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 an interrupt at the beginning of a block of high-speed data. Incoming data is routed through one of several paths (data or CRC) depending on the selected mode and-in asynchronous modes-the character length.

The transmitter has an 8-bit transmit data buffer register that is loaded from the internal data bus, and a 20-bit transmit shift register that can be loaded from the sync-character buffers or from the transmit data register. Depending on the operational mode, outgoing data is routed throught one of four main paths before it is transmitted from the Transmit Data output (TxD).

Figure 13 : Transmit and Receive Data Path (channel A).



PROGRAMMING

The system program first issues a series of commands that initialize the basic mode of operation and then other commands that qualify conditions within the selected mode. For example, the asynchronous mode, character length, clock rate, number of stop bits, even or odd parity might be set first ; then the interrupt mode ; and finally, receiver or transmitter enable.

Both channels contain registers that must be programmed via the system program prior to operation. The channel-select input (B/A) and the control/data input (C/D) are the command-structure addressing controls, and are normally controlled by the CPU address bus. Figures 16 an 17 illustrate the timing relationships for programming the write registers and transfering data and status.

READ REGISTER

The SIO contains three read registers for Channel B and two read registers for Channel A (RR0-RR2 in figure 14) that can be to obtain the status information ; RR2 contains the internally-modifiable interrupt vector and is only in the Channel B register set. 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 a read 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 (RRI).

WRITE REGISTERS

The SIO contains eight write registers for Channel B and seven write registers for Channel A (WR0-WR7 in figure 15) that are programmed separately to configure the functional personality of the channels ; WR2 contains the interrupt vector for both channels and is only in the Channel B register set. With the exception of WR0, programming the write registers requires two bytes. The first byte is to WR0 and 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 SIO.

WR0 is a special case in that all of the basic commands can be written to it with a single byte. Reset (internal or external) initializes the pointer bits D_0-D_2 to point to WR0. This implies that a channel reset must not be combined with the pointing to any register.

Figure 14 : Read Register Bit Functions.

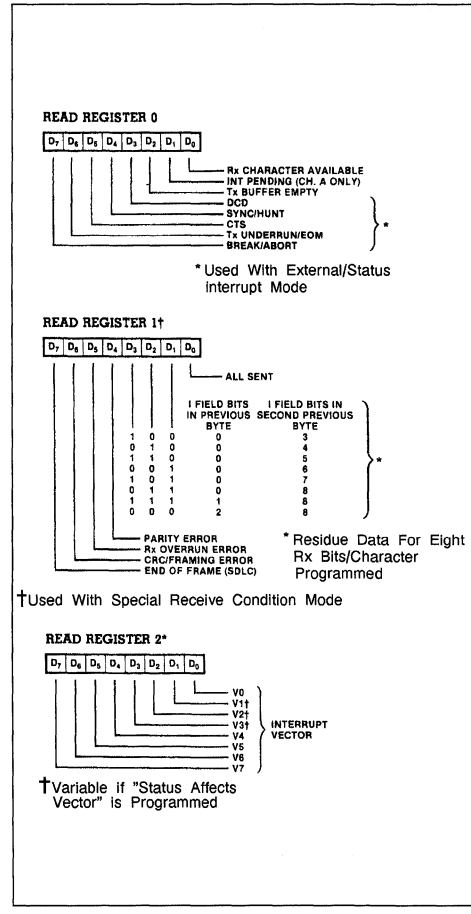
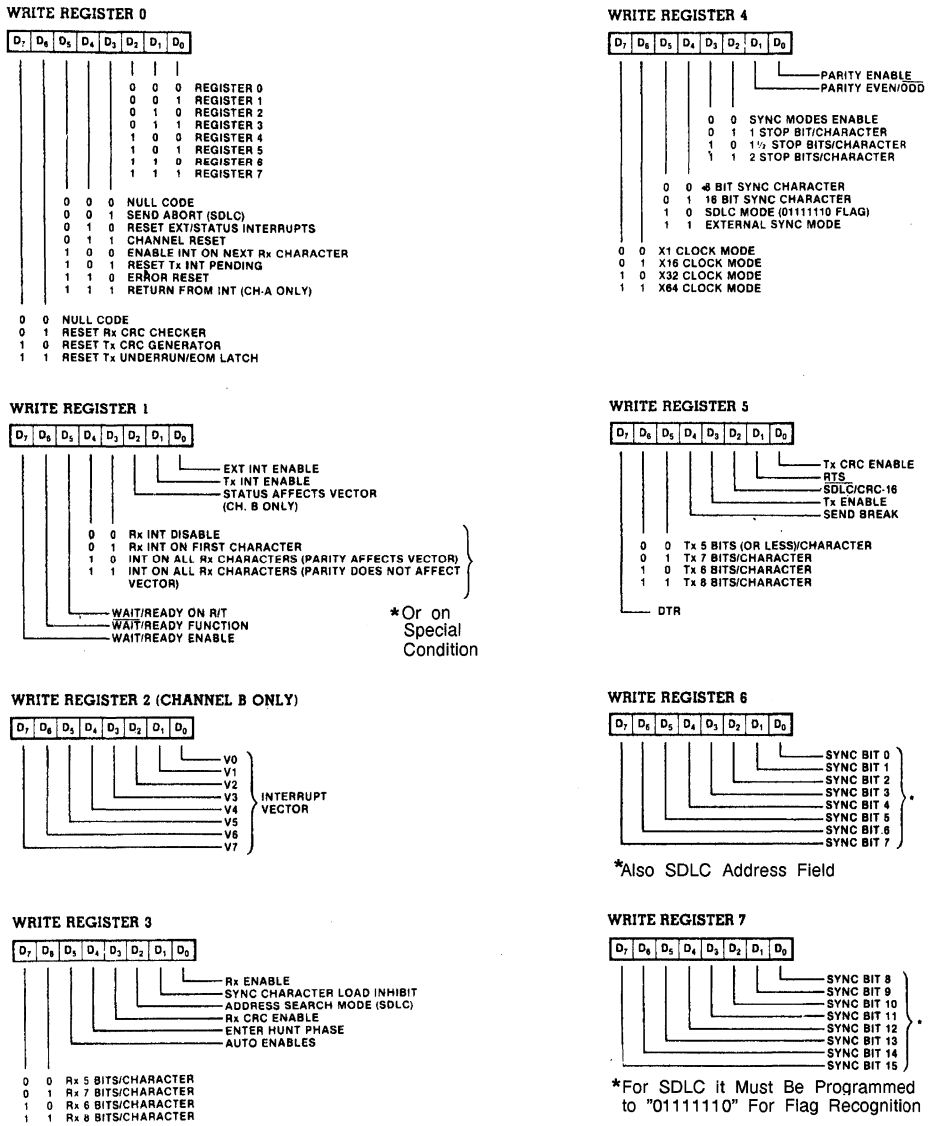


Figure 15 : Write Register Bit Functions.



TIMING

The SIO must have the same clock as the CPU (same phase and frequency relationship, not necessarily the same driver).

READ CYCLE

The timing signals generated by a Z80 CPU input instruction to read a data or status byte from the SIO are illustrated in figure 16.

WRITE CYCLE

Figure 16 illustrates the timing and data signals generated by a Z80 CPU output instruction to write a data or control byte into the SIO.

INTERRUPT-ACKNOWLEDGE CYCLE

After receiving an interrupt-request signal from an SIO (INT pulled Low), the Z80 CPU sends an interrupt-acknowledge sequence (M1 Low, and IORQ Low a few cycles later) as in figure 18.

The SIO contains an internal daisy-chained interrupt structure for prioritizing nested interrupts for the various functions of its two channels, and this structure can be used within an external user-defined daisy chain that prioritizes several peripheral circuits.

The IEI of the highest-priority device is terminated High. A device that has an interrupt pending or under service forces its IEO Low. For devices with no interrupt pending or under service, IEO = IEI.

To insure stable conditions in the daisy chain, all interrupt status signals are prevented from changing while M1 is Low. When IORQ is Low, the highest priority interrupt requestor (the one with IEI High) places its interrupt vector on the data bus and sets its internal interrupt-under-service latch.

RETURN FROM INTERRUPT CYCLE

Figure 19 illustrates the return from interrupt cycle. Normally, the Z80 CPU issues a RETI (Return From Interrupt) instruction at the end of an interrupt service routine. RETI is a 2-byte opcode (ED-4D) that resets the interrupt-under-service latch in the SIO to terminate the interrupt that has just been processed. This is accomplished by manipulating the daisy chain in the following way.

The normal daisy-chain operation can be used to detect a pending interrupt ; however, it cannot distinguish between an interrupt under service and a pending unacknowledged interrupt of a higher priority. Whenever "ED" is decoded, the daisy chain is modified by forcing High the IEO of any interrupt that has not yet been acknowledged. Thus the daisy chain identifies the device presently under service as the only one with an IEI High and an IEO Low. If the next opcode byte is "4D", the interrupt-under-service latch is reset.

The ripple time of the interrupt daisy chain (both the High-to-Low and the Low-to-High transitions) limits the number of devices that can be placed in the daisy chain. Ripple time can be improved with carry-look-ahead, or by extending the interrupt-acknowledge cycle. For further information about techniques for increasing the number of daisy-chained devices, refer to the Z80 CPU Data Sheet.

Figure 16 : Read Cycle.

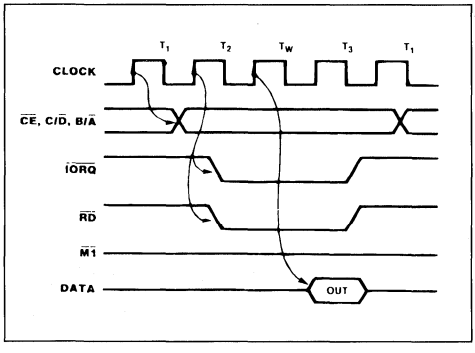


Figure 17 : Write Cycle.

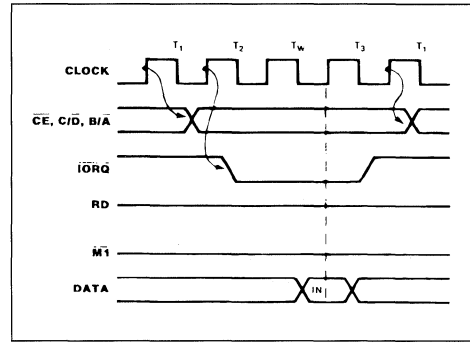


Figure 18 : Interrupt Acknowledge Cycle.

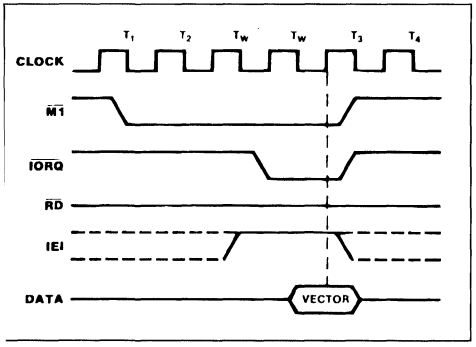
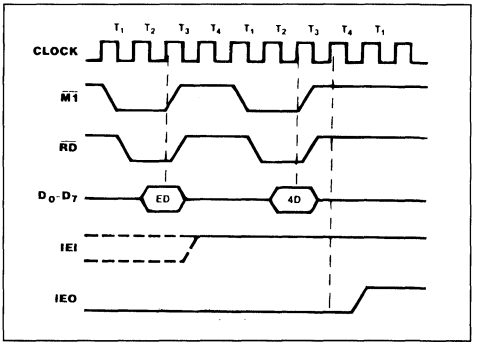


Figure 19 : Return from Interrupt Cycle.



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_I	Voltage on all Input and Outputs with Respect to GND	− 0.3 to + 7.0	V
T_A	Operating Ambient Temperature As Specified in Order Codes		
T_{stg}	Storage Temperature		°C

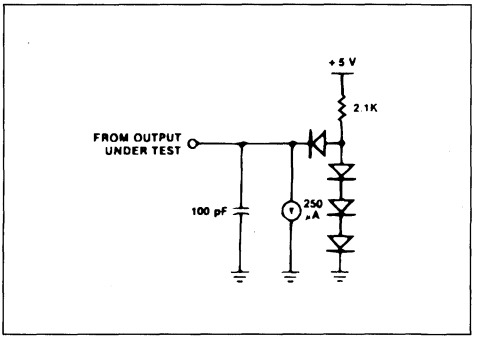
Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

TEST CONDITIONS

The characteristics below apply for the following test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin. Available operating temperature ranges are :

- 0 °C to + 70 °C,
  + 4.75V ≤ V_{CC} ≤ + 5.25V
- − 40 °C to + 85 °C,
  + 4.75V ≤ V_{CC} ≤ + 5.25V
- − 55 °C to + 125 °C,
  + 4.75V ≤ V_{CC} ≤ + 5.5V

The product number for each operating temperature range may be found in the ordering information section.



CAPACITANCE

Symbol	Parameter	Note	Min.	Max.	Unit
C	Clock Capacitance	Unmeasured Pins Returned to Ground		40	pF
C_{IN}	Input Capacitance			5	pF
C_{OUT}	Output Capacitance			10	pF

Over specified temperature range ; f = 1 MHz.

DC CHARACTERISTICS

Symbol	Parameter	Test Conditions	Min.	Max.	Unit
V_{ILC}	Clock Input Low Voltage		− 0.3	+ 0.45	V
V_{IHC}	Clock Input High Voltage		V_{CC} − 0.6	V_{CC} + 0.3	V
V_{IL}	Input Low Voltage		− 0.3	+ 0.8	V
V_{IH}	Input High Voltage		2.0	V_{CC}	V
V_{OL}	Output Low Voltage	I_{OL} = 2.0 mA	−	+ 0.4	V
V_{OH}	Output High Voltage	I_{OH} = − 250 µA	+ 2.4		V
I_{LI}	Input Leakage Current	V_{IN} = 0 to V_{CC}	− 10	+ 10	µA
I_{OL}	3-State Output Leakage Current in Float	V_{OUT} = 0.4 to V_{CC}	− 10	+ 10	µA
$I_{L(SY)}$	$\overline{SYNC}$ Pin Leakage Current	0 < V_{IN} < V_{CC}	− 40	+ 10	µA
I_{CC}	Power Supply Current			100	mA

Over specified temperature and voltage range.

AC CHARACTERISTICS

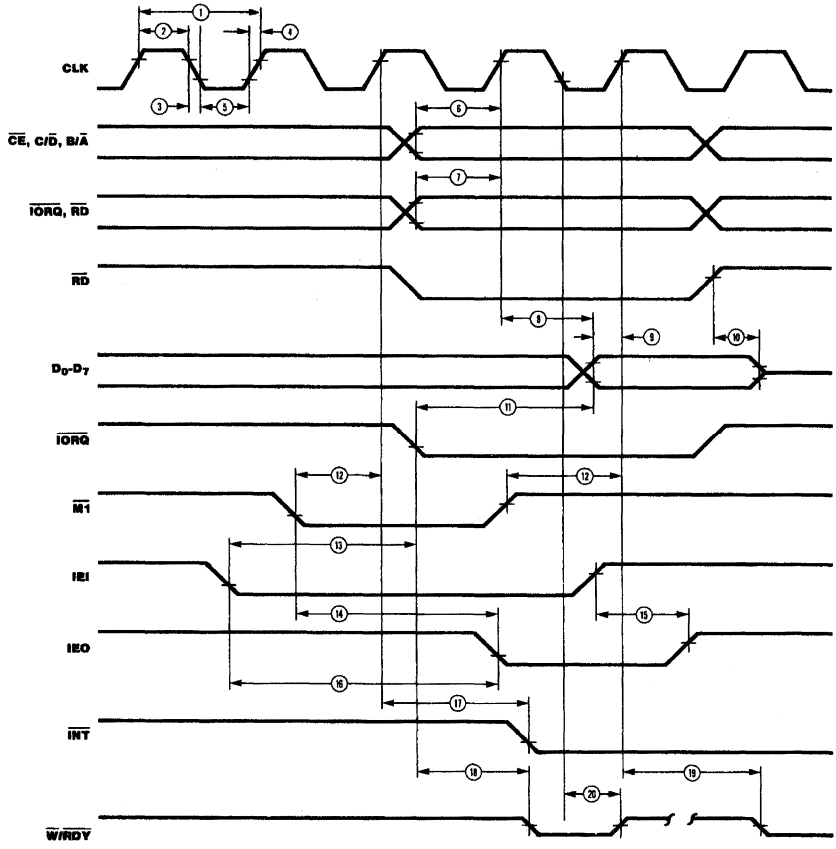
N°	Symbol	Parameter	Z8440, 1, 2		Z8440, 1, 2A		Z8440, 1, 2B	
			Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
1	TcC	Clock Cycle Time	400	4000	250	4000	165	4000
2	TwCh	Clkock Width (high)	170	2000	105	2000	70	2000
3	TfC	Clock Fall Time		30		30		15
4	TrC	Clock Rise Time		30		30		15
5	TwCl	Clock Width (low)	170	2000	105	2000	70	2000
6	TsAD(C)	$\overline{CE}$, C/$\overline{D}$, B/$\overline{A}$ to Clock ↑ Setup Time	160		145		60	
7	TsCS(Cl)	$\overline{IORQ}$, $\overline{RD}$, to Clock ↑ Setup Time	240		115		60	
8	TdC(DO)	Clock ↑ to Data Out Delay		240		220		150
9	TsDI(C)	Data in to Clock ↑ Setup (write or $\overline{MI}$ cycle)	50		50		30	
10	TdRD(DOz)	$\overline{RD}$ ↑ to Data Out Float Delay		230		110		90
11	TdIO(DOI)	$\overline{IORQ}$ ↓ to Data Out Delay (INTACK cycle)		340		160		100
12	TsMI(C)	$\overline{MI}$ to Clock ↑ Setup Time	210		90		75	
13	TsIEI(IO)	IEI to $\overline{IORQ}$ ↓ Setup Time (INTACK cycle)	200		140		120	
14	TdMI(IEO)	$\overline{MI}$ ↓ to IEO ↓ Delay (interrupt before $\overline{MI}$)		300		190		160
15	TdIEI(IEOr)	IEI ↑ to IEO ↑ Delay (after ED decode)		150		100		70
16	TdIEI(IEOf)	IEI ↓ to IEO ↓ Delay		150		100		70
17	TdC(INT)	Clock ↑ to $\overline{INT}$ ↓ Delay		200		200		150
18	TdIO(W/RWf)	$\overline{IORQ}$ ↓ or $\overline{CE}$ ↓ to W/$\overline{RDY}$ ↓ Delay (wait mode)		300		210		175
19	TdC(W/RR)	Clock ↑ to W/$\overline{RDY}$ ↓ Delay (ready mode)		120		120		100
20	TdC(W/RWz)	Clock ↓ to W/$\overline{RDY}$ Float Delay (wait mode)		150		130		110
21	Th	Any unspecified hold when setup is specified	0		0		0	

AC CHARACTERISTICS (continued)

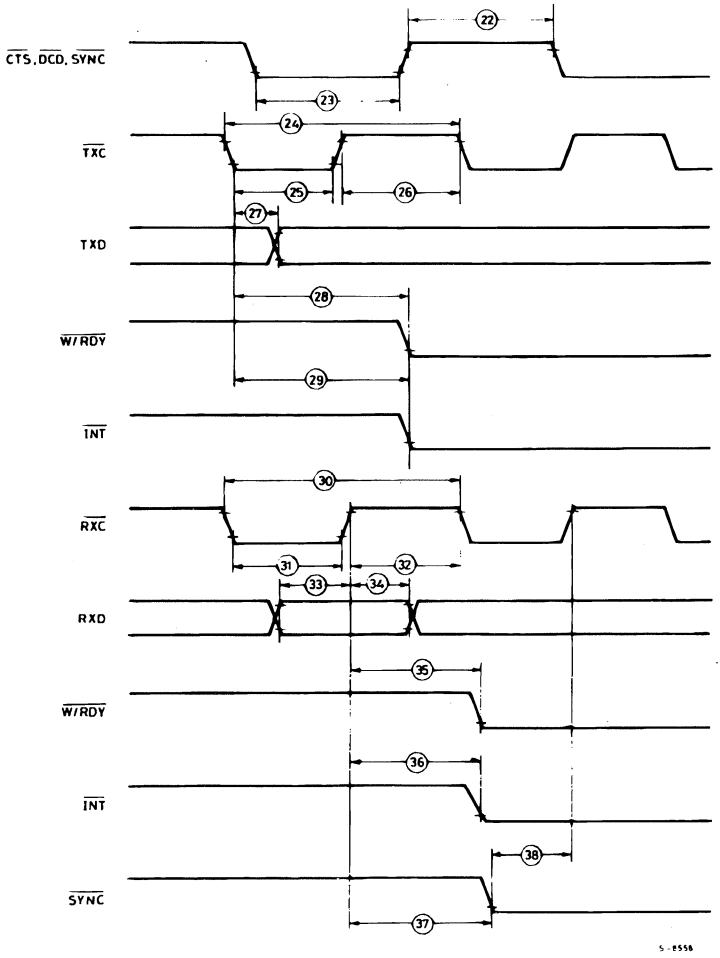
N°	Symbol	Parameter	Notes	Z8440, 1, 2		Z8440, 1, 2A		Z8440, 1, 2B	
				Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
22	TwPh	Pulse Width (high)		200		200		200	
23	TwPl	Pulse Width low)		200		200		200	
24	TcTxC	$\overline{TxC}$ Cycle Time		400	∞	400	∞	330	∞
25	TwTxCl	$\overline{TxC}$ Width (low)		180	∞	180	∞	100	∞
26	TwTxCh	$\overline{TxC}$ Width (high)		180	∞	180	∞	100	∞
27	TdTxC(TxD)	$\overline{TxC}$ ↓ to TxD Delay (xI mode)			400		300		220
28	TdTxC(W/RRf)	$\overline{TxC}$ ↓ to $\overline{W/RDY}$ ↓ Delay (ready mode)	Clk Periods*	5	9	5	9	5	9
29	TdTxC(INT)	$\overline{TxC}$ ↓ to $\overline{INT}$ ↓ Delay	Clk Periods*	5	9	5	9	5	9
30	TcRxC	$\overline{RxC}$ Cycle Time		400	∞	400	∞	330	∞
31	TwRxCl	$\overline{RxC}$ Width (low)		180	∞	180	∞	100	∞
32	TwRxCh	$\overline{RxC}$ Width (high)		180	∞	180	∞	100	∞
33	TsRxD(RxC)	RxD to $\overline{RxC}$ ↑ Setup Time (xI mode)		0		0		0	
34	ThRxD(RxC)	RxC ↑ to RxD Hold Time (xI mode)		140		140		100	
35	TdRxC(W/RRf)	$\overline{RxC}$ ↑ to $\overline{W/RDY}$ ↓ Delay (ready mode)	Clk Periods*	10	13	10	13	10	13
36	TdRxC(INT)	$\overline{RxC}$ ↑ to $\overline{INT}$ ↓ Delay	Clk Periods*	10	13	10	13	10	13
37	TdRxC(SYNC)	$\overline{RxC}$ ↑ to $\overline{SYNC}$ ↓ Delay (outputs modes)	Clk Periods*	4	7	4	7	4	7
38	TsSYNC(RxC)	$\overline{SYNC}$ ↓ to $\overline{RxC}$ ↑ Setup (external sync modes)		−100		−100		−100	

In all modes, the System Clock rate must be at least five times the maximum data rate $\overline{RESET}$ must be active a minimum of one complete Clock Cycle.
* System Clock.

AC CHARACTERISTICS



AC CHARACTERISTICS (continued)

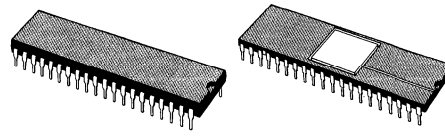


ORDERING INFORMATION

Type	Package	Temp.	Clock	Description
Z8440/1/2B1	DIP-40 (plastic)	0/+ 70°C	2.5 MHz	Z80 Dual Channel Serial I/O Controller
Z8440/1/2F1	DIP-40 (frit seal)	0/+ 70°C		
Z8440/1/2D1	DIP-40 (ceramic)	0/+ 70°C		
Z8440/1/2D6	DIP-40 (ceramic)	−40/+ 85°C		
Z8440/1/2D2	DIP-40 (ceramic)	−55/+125°C		
Z8444C1	PLCC44 (plastic chip-carrier)	0/+ 70°C		
Z8440/1/2AB1	DIP-40 (plastic)	0/+ 70°C	4 MHz	
Z8440/1/2AF1	DIP-40 (frit seal)	0/+ 70°C		
Z8440/1/2AD1	DIP-40 (ceramic)	0/+ 70°C		
Z8440/1/2AD6	DIP-40 (ceramic)	−40/+ 85°C		
Z8440/1/2AD2	DIP-40 (ceramic)	−55/+125°C		
Z8444AC1	PLCC44 (plastic chip-carrier)	0/+ 70°C		
Z8440/1/2BB1	DIP-40 (plastic)	0/+ 70°C	6 MHz	
Z8440/1/2BF1	DIP-40 (frit seal)	0/+ 70°C		
Z8440/1/2BD1	DIP-40 (ceramic)	0/+ 70°C		
Z8440/1/2BD6	DIP-40 (ceramic)	−40/+ 85°C		
Z8440/1/2BD2	DIP-40 (ceramic)	−55/+125°C		
Z8444BC1	PLCC44 (plastic chip-carrier)	0/+ 70°C		

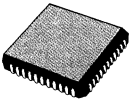
DUAL ASYNCHRONOUS RECEIVER/TRANSMITTER

- TWO INDEPENDENT FULL-DUPLEX CHANNELS WITH SEPARATE MODEM CONTROLS. MODEM STATUS CAN BE MONITORED
- RECEIVER DATA REGISTERS ARE QUADRUPLY BUFFERED ; THE TRANSMITTER IS DOUBLY BUFFERED
- INTERRUPT FEATURES INCLUDE A PROGRAMMABLE INTERRUPT VECTOR, A "STATUS AFFECTS VECTOR" MODE FOR FAST INTERRUPT PROCESSING, AND THE STANDARD Z80 PERIPHERAL DAISY-CHAIN INTERRUPT STRUCTURE THAT PROVIDES AUTOMATIC INTERRUPT VECTORING WITH NO EXTERNAL LOGIC
- IN x1 CLOCK MODE, DATA RATES ARE 0 TO 500K BITS/SECOND WITH A 2.5MHz CLOCK, OR 0 TO 800K BITS/SECOND WITH A 4.0MHz CLOCK, OR 0 TO 1200K BIT/SECOND WITH A 6.0MHz CLOCK
- PROGRAMMABLE OPTIONS INCLUDE 1, 1 1/2 OR 2 STOP BITS ; EVEN, ODD OR NO PARITY ; AND x1, x16, x32 AND x64 CLOCK MODES
- BREAK GENERATION AND DETECTION AS WELL AS PARITY-, OVERRUN- AND FRAMING-ERROR DETECTION ARE AVAILABLE



B/F
DIP-40
(Plastic and Frit-Seal)

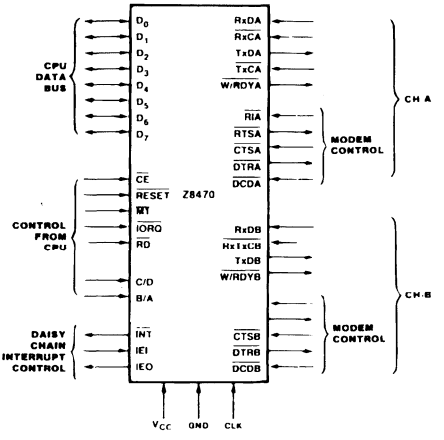
D
DIP-40
(Ceramic)



C
PLCC44
(Plastic)

(Ordering Information at the end of the datasheet)

LOGIC FUNCTIONS



DESCRIPTION

The Z80 DART (Dual-Channel Asynchronous Receiver/Transmitter) is a dual-channel multi-function peripheral component that satisfies a wide variety of asynchronous serial data communications requirements in microcomputer systems. The Z80 DART is used as a serial-to-parallel, parallel-to-serial converter/controller in asynchronous applications. In addition, the device also provides modem controls for both channels. In applications where modem controls are not needed, these lines can be used for general-purpose I/O.

The Z80 SIO, a more versatile device, provides synchronous (Bisync, HDLC and SDLC) as well as asynchronous operation.

The Z80 DART is fabricated with n-channel silicon-gate depletion-load technology, and is packaged in a 40 pin plastic or ceramic DIP.

Figure 1 : Dual ine pine Configuration.

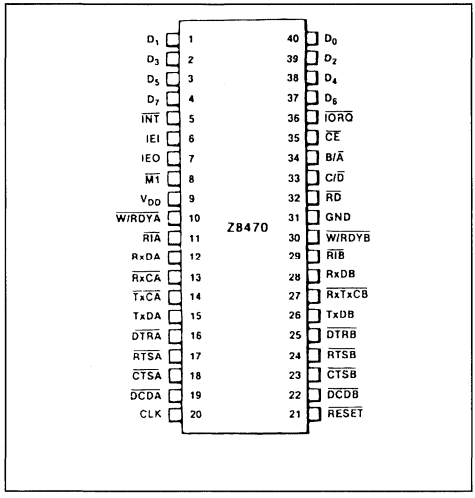
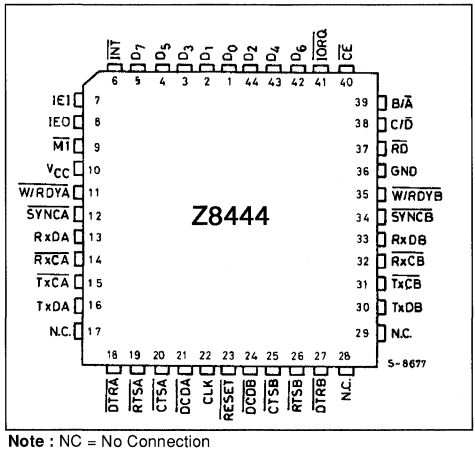


Figure 2 : Chip Carrier Pin Configuration.



Note : NC = No Connection
Z80 SIO-0 or Dart in Asynchronous Mode.

PIN DESCRIPTIONS

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 Z80 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 Z80 DART.

$\overline{CE}$. *Chip Enable* (Input, Active Low). A Low at this input enables the Z80 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 Z80 DART uses the standard Z80 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 accomodate slow-risetime signals.

D_0-D_7. *System Data Bus* (Bidirectional, 3-state) transfers data and commands between the CPU and the Z80 DART.

$\overline{DCDA}$, **$\overline{DCDB}$**. *Data Carrier Detect* (Inputs, Active Low). These pins function as receiver enables if the Z80 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 Z80 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 Z80 DART is requesting an interrupt, it pulls INT Low.

$\overline{M1}$. *Machine Cycle One* (Input from Z80 CPU, Active Low). When M1 and RD are both active, the Z80 CPU is fetching an instruction from memory ; when M1 is active while IORQ is active, the Z80 DART accepts M1 and IORQ as an interrupt acknowledge if the Z80 DART is the highest priority device that has interrupted the Z80 CPU.

$\overline{IORQ}$. *Input/Output Request* (Input from CPU, Active Low). IORQ is used in conjunction with B/A, C/D, CE and RD to transfer commands and data between the CPU and the Z80 DART. When CE, RD and

IORQ are all active, the channel selected by B/A transfers data to the CPU (a read operation). When CE and IORQ are active, but RD is inactive, the channel selected by B/A is written to by the CPU with either data or control information as specified by C/D.

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

RD. *Read Cycle Status* (Input from CPU, Active Low). If RD is active, a memory or I/O read operation is in progress.

RxDA, **RxDB**. *Receive Data* (Inputs, Active High).

RESET. *Reset* (Input, Active Low). Disables both receivers and transmitters, forces TxDA and TxDB marking, forces the modem controls High and disables all interrupts.

RIA, **RIB**. *Ring Indicator* (Inputs, Active Low). These inputs are similar to CTS and DCD. The Z80 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.

RTSA, **RTSB**. *Request to Send* (Outputs, Active Low). When the RTS bit is set, the RTS output goes Low. When the RTS bit is reset, the output goes High after the transmitter empties.

TxCA, **TxCB**. *Transmitter Clocks* (Inputs). TxD changes on the falling edge of 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 Transmitt Clock inputs are Schmitt-triggered buffered. Both the Receiver and Transmitter Clocks may be driven by the Z80 CTC Counter Time Circuit for programmable baud rate generation.

TxDA, **TxDB**. *Transmitt Data* (Outputs, Active High).

W/RDYA, **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 Z80 DART data rate. The reset state is open drain.

FUNCTIONAL DESCRIPTION

The functional capabilities of the Z80 DART can be described from two different points of view : as a data communication device, it transmits and receives serial data, and meets the requirements of asynchronous data communication protocols ; as a Z80 family peripheral, it interacts with the Z80 CPU and other Z80 peripheral circuits, and shares the data, address and control buses, as well as being a part of the Z80 interrupt structure. As a peripheral to other microprocessors, the Z80 DART offers valuable features such as non-vectored interrupts, polling and simple hand-shake capability.

The first part of the following functional description introduces Z80 DART data communications capabilities ; the second part describes the interaction between the CPU and the Z80 DART.

The Z80 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 Z80 DART also features a dual channel Ring Indicator (RIA, RIB) input to facilitate local/remote or station-to-station communication capability.

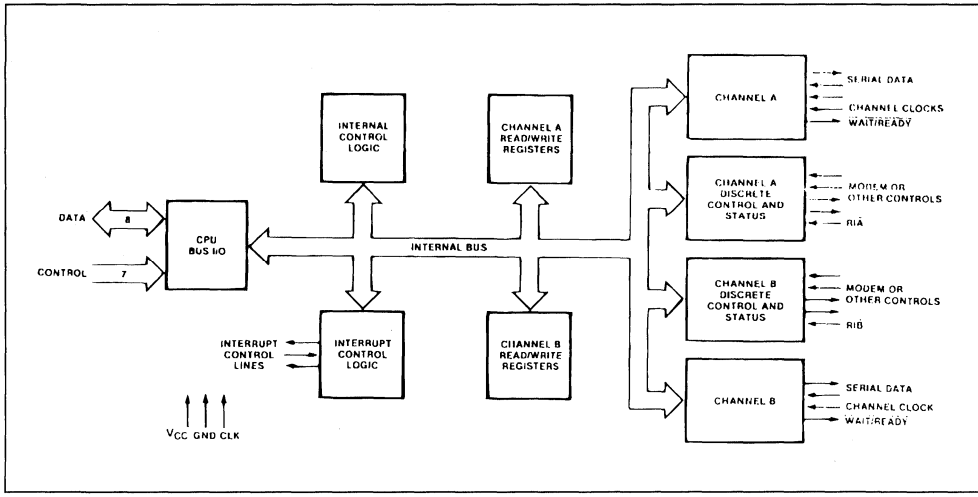
COMMUNICATIONS CAPABILITIES.

The Z80 DART provides two independent full-duplex channels receiver/transmitter. The following is a short description of receiver/transmitter capabilities. For more details, refer to the Asynchronous Mode section of the Z80 *Family Technical Manual.* The Z80 DART offers transmission and reception of five to eigh 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.

Figure 3 : Block Diagram.



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-halt a bit time to the point at which the search for the next start bit is begun.

The Z80 DART does not require symmetric Transmitt and Receive Clock signals-a-feature that allows it to be used with a Z80 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 RxC and TxC are bonded together (RxTxCB).

I/O INTERFACE CAPABILITIES

The Z80 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.

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 or the Z80 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 "Z80 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 Z80 DART offers an elaborate interrupts scheme that provides fast interrupt response in real-time applications. As a member of the Z80 family, the Z80 DART can be daisy-chained along with other Z80 peripherals or peripheral interrupt-priority resolution. In addition, the internal interrupts of the Z80 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 ponts to an interrupt service routine in the memory. To eliminate the necessity of writing a status analysis routine, the Z80 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 inter-

rupt 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 Character 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 CTS, DCD and 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 Z80 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 Z80 DART provides a Block Transfer mode to accomodate CPU block transfer functions and DMA block transfers (Z80 DMA or other design). The Block Transfer mode uses the W/RDY output in conjunction with the Wait/Ready bits of Write Register 1. The 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 Z80 DART Ready output indicates that the Z80 DART is ready to transfer data to or from memory. To the CPU the Wait output indicates that the Z80 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 Z80 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 channel are designated as follows :

WR0-WR5-Write Register 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 (CTS), Data Carrier Detect (DCD) and Ring Indicator (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.

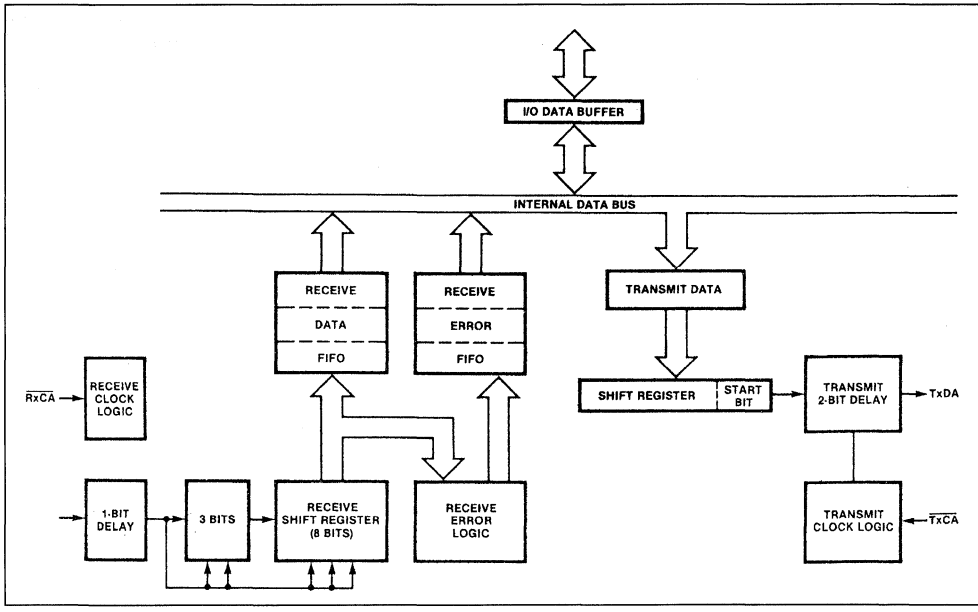
For automatic interrupt vectoring, the interrupt control logic determines which channel and which device within the channel has the highest priority. Priority is fixed with Channel A assigned a higher priority than Channel B ; Receive, Transmit and External/Status interrupts are prioritized in that order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

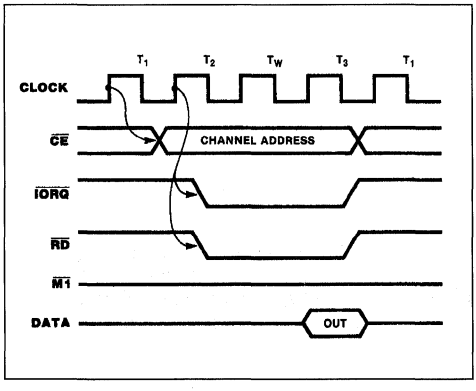


READ, WRITE AND INTERRUPT TIMING

READ CYCLE

The timing signals generated by a Z80 CPU input instruction to read a Data or Status byte from the Z80 DART are illustrated in figure 5a.

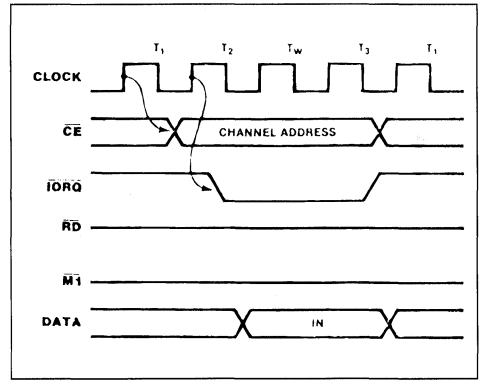
Figure 5a : Read Cycle.



WRITE CYCLE

Figure 5b illustrates the timing and data signals generated by a Z80 CPU output instruction to write a Data or Control byte into the Z80 DART.

Figure 5b : Write Cycle.



INTERRUPT ACKNOWLEDGE CYCLE

After receiving an Interrupt Request signal ($\overline{INT}$ pulled Low), the Z80 CPU sends an Interrupt Acknowledge signal ($\overline{M1}$ and $\overline{IORQ}$ both Low). The daisy-chained interrupt circuits determine the highest priority interrupt requestor. The IEI of the highest priority peripheral is terminated High. For any peripheral that has no interrupt pending or under service, IEO = IEI. Any peripheral that does have an interrupt pending or under service forces its IEO Low.

To insure stable conditions in the daisy chain, all interrupt status signals are prevented from changing while $\overline{M1}$ is Low. When $\overline{IORQ}$ is Low, the highest priority interrupt requestor (the one with IEI High) places its interrupt vector on the data bus and sets its internal interrupt-under-service latch.

Refer to the Z80 *Family Technical Manual* for additional details on the interrupt daisy chain and interrupt nesting.

RETURN FROM INTERRUPT CYCLE

Normally, the Z80 CPU issues a RETI (Return From Interrupt) instruction at the end of an interrupt service routine. RETI is a 2-byte opcode (ED-4D) that resets the interrupt-under-service latch to terminate the interrupt that has just been processed.

When used with other CPUs, the Z80 DART allows the user to return from the interrupt cycle with a special command called "Return From Interrupt" in Write Register 0 of Channel A. This command is interpreted by the Z80 DART in exactly the same way it would interpret a RETI command on the data bus.

Figure 5c : Interrupt Acknowledge Cycle.

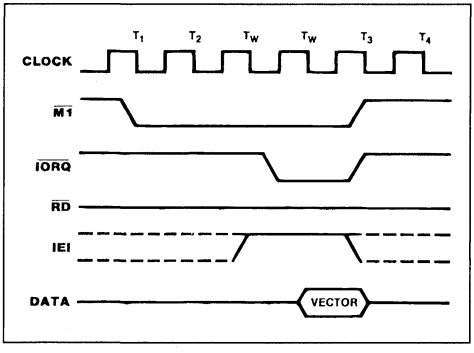
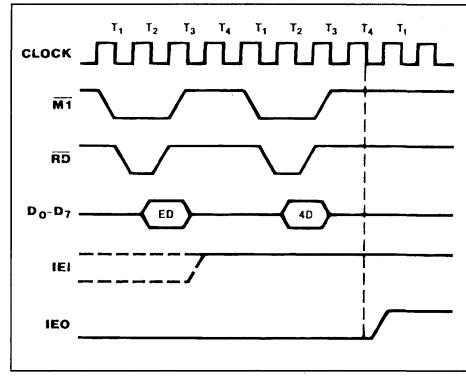


Figure 5d : Return from Interrupt Cycle.



Z80 DART PROGRAMMING

To program the Z80 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 Z80 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 Z80 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 ponter 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

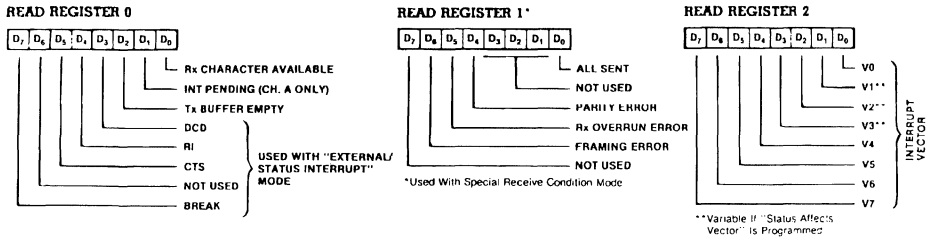
The Z80 DART contains three registers (RR0-RR2) that can be read to obtain the status information for each channel (except for RR2, which applies to Channel B only). The status information includes error conditions, interrupt vector and standard communications-interface signals.

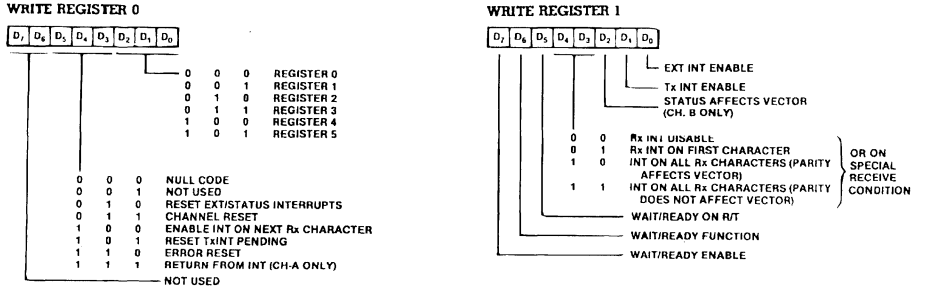
To read the contents of a selected read register other than RR0, the system program must first write the pointer byte to WR0 in exactly the same way as a write register operation. Then, by executing an input instruction, the contents of the addressed read register can be read by the CPU.

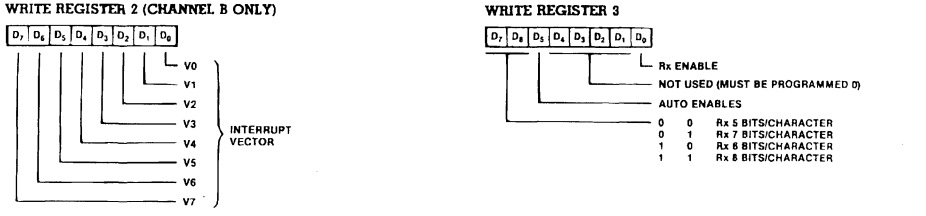
The status bits of RR0 and RR1 are carefully grouped to simplify status monitoring. For example, when the interrupt vector indicates that a Special Receive Conditions interrupt has occurred, all the appropriate error bits can be read from a single register (RR1).

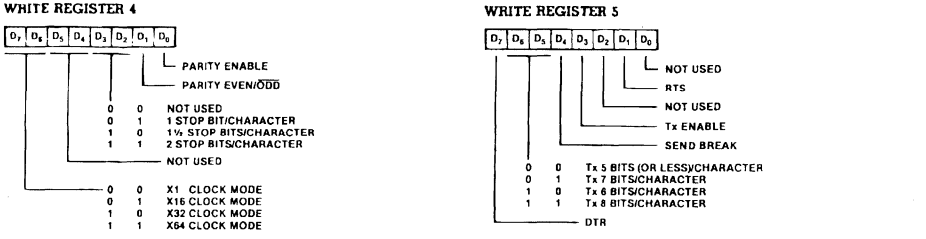
Read Register Functions	
RR0	Transmit/Receive Buffer Status, Interrupt Status and External Status
RR1	Special Receive Condition Status
RR2	Modified Interrupt Vector (channel B only)

Z80 DART READ AND WRITE REGISTERS









ABSOLUTE MAXIMUM RATINGS

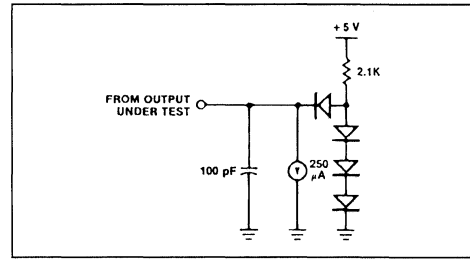
Symbol	Parameter	Value	Unit
V_I	Voltage on all Inputs and Outputs with Respect to GND	− 0.3 to + 7.0	V
T_A	Operating Ambient Temperature As Specified in Ordering Information		
T_{stg}	Storage Temperature	− 65 to + 150	°C

Stresses greater than those listed under Absolute Maximun Ratings may cause permanent damage to the device. This is a stress rating only ; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

TEST CONDITIONS

The characteristics below apply for the following test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin. Available operating temperature ranges are :

- 0 °C to + 70 °C,
  + 4.75 V ≤ V_{CC} ≤ + 5.25 V
- 40 °C to + 85 °C,
  + 4.75 V ≤ V_{CC} ≤ + 5.25 V
- 55 °C to + 125 °C,
  + 4.5 V ≤ V_{CC} ≤ + 5.5 V

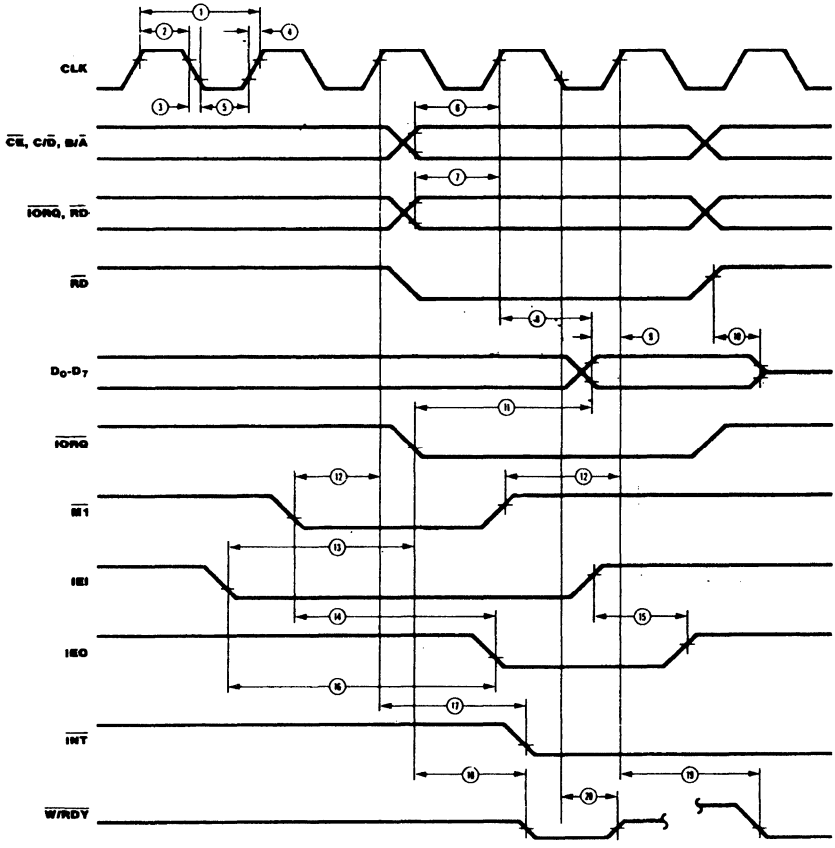


DC CHARACTERISTICS

Symbol	Parameter	Test Conditions	Min.	Max.	Unit
V_{ILC}	Clock Input Low Voltage		− 0.3	+ 0.45	V
V_{IHC}	Clock Input High Voltage		V_{CC} − 0.6	+ 5.5	V
V_{IL}	Input Low Voltage		− 0.3	+ 0.8	V
V_{IH}	Input High Voltage		+ 2.0	+ 5.5	V
V_{OL}	Output Low Voltage	I_{OL} = 2.0 mA		+ 0.4	V
V_{OH}	Output High Voltage	I_{OH} = − 250 µA	+ 2.4		V
I_L	Input/3-State Output Leakage Current	0.4 < V < 2.4 V	− 10	+ 10	µA
$I_{L(RI)}$	$\overline{RI}$ Pin Leakage Current	0 ≤ V_{IN} ≤ V_{CC}	− 40	+ 10	µA
I_{CC}	Power Supply Current			100	mA

T_A = 0 °C to 70 °C, V_{CC} = 5 V ± 5 %.

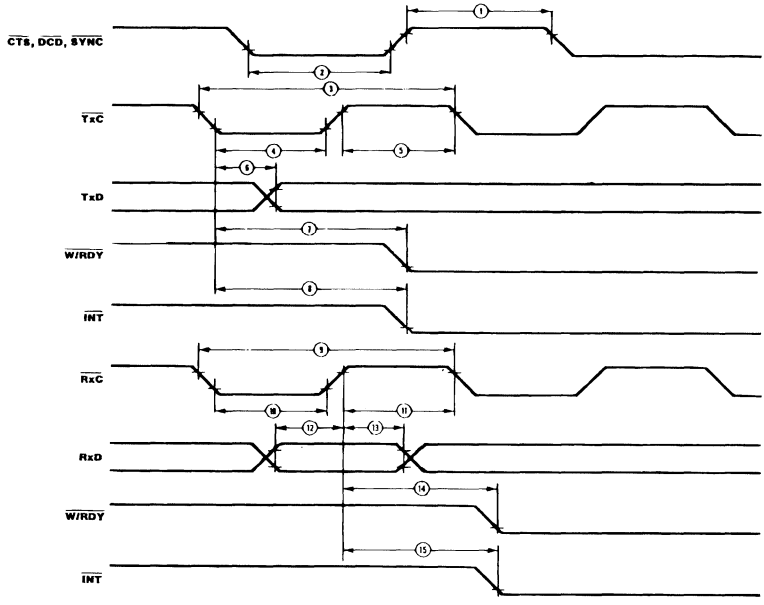
AC CHARACTERISTICS



AC CHARACTERISTICS (continued)

N°	Symbol	Parameter	Z8470,		Z8470A		Z8470B	
			Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
1	TcC	Clock Cycle Time	400	4000	250	4000	165	4000
2	TwCh	Clock Width (high)	170	2000	105	2000	70	2000
3	TfC	Clock Fall Time		30		30		15
4	TrC	Clock Rise Time		30		30		15
5	TwCl	Clock Width (low)	170	2000	105	2000	70	2000
6	TsAD(C)	$\overline{CE}$, C/$\overline{D}$, B/$\overline{A}$ to Clock ↑ Setup Time	160		145		60	
7	TsCS(C)	$\overline{IORQ}$, $\overline{RD}$, to Clock ↑ Setup Time	240		115		60	
8	TdC(DO)	Clock ↑ to Data Out Delay		240		220		150
9	TsDI(C)	Data in to Clock ↑ Setup Time (write or MI cycle)	50		50		30	
10	TdRD(DOz)	$\overline{RD}$ ↑ to Data Out Float Delay		230		110		90
11	TdIO(DOI)	$\overline{IORQ}$ ↓ to Data Out Delay (INTACK cycle)		340		160		100
12	TsMI(C)	$\overline{MI}$ to Clock ↑ Setup Time	210		90		75	
13	TsIEI(IO)	IEI to $\overline{IORQ}$ ↓ Setup Time (INTA cycle)	200		140		120	
14	TdMI(IEO)	$\overline{MI}$ ↓ to IEO ↓ Delay (interrupt before $\overline{MI}$)		300		190		160
15	TdIEI(IEOr)	IEI ↑ to IEO ↑ Delay (after ED decode)		150		100		70
16	TdIEI(IEOf)	IEI ↓ to IEO ↓ Delay		150		100		70
17	TdC(INT)	Clock ↑ to INT ↓ Delay		200		200		150
18	TdIO(W/RWf)	$\overline{IORQ}$ ↓ or $\overline{CE}$ ↓ to W/RDY ↓ Delay (wait mode)		300		210		175
19	TdC(W/RR)	Clock ↑ to W/RDY ↓ Delay (ready mode)		120		120		100
20	TdC(W/RWz)	Clock ↓ to W/RDY Float Delay (wait mode)		150		130		110

AC CHARACTERISTICS (continued)



AC CHARACTERISTICS (continued)

N°	Symbol	Parameter	Unit	Z8470		Z8470A		Z8470B	
				Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
1	TwPh	Pulse Width (high)		200		200		200	
2	TwPl	Pulse Width (low)		200		200		200	
3	TcTxC	$\overline{TxC}$ Cycle Time		400	∞	400	∞	330	∞
4	TwTxCl	$\overline{TxC}$ Width (low)		180	∞	180	∞	100	∞
5	TwTxCh	$\overline{TxC}$ Width (high)		180	∞	180	∞	100	∞
6	TdTxC(TxD)	$\overline{TxC}$ ↓ to TxD Delay			400		300		220
7	TdTxC(W/RRf)	$\overline{TxC}$ ↓ to W/RDY ↓ Delay (ready mode)	Clk Periods	5	9	5	9	5	9
8	TdTxC(INT)	$\overline{TxC}$ ↓ to $\overline{INT}$ ↓ Delay	Clk Periods	5	9	5	9	5	9
9	TcRxC	$\overline{RxC}$ Cycle Time		400	∞	400	∞	330	∞
10	TwRxCl	$\overline{RxC}$ Width (low)		180	∞	180	∞	100	∞
11	TwRxCh	$\overline{RxC}$ Width (high)		180	∞	180	∞	100	∞
12	TsRxD(RxC)	RxD to $\overline{RxC}$ ↑ Setup Time (xl mode)		0		0		0	
13	ThRxD(RxC)	RxD Hold Time (xl mode)		140		140		100	
14	TdRxC(W/RRf)	$\overline{RxC}$ ↑ to W/RDY ↓ Delay (ready mode)	Clk Periods	10	13	10	13	10	13
15	TdRxC(INT)	$\overline{RxC}$ ↑ to $\overline{INT}$ ↓ Delay	Clk Periods	10	13	10	13	10	13

ORDERING INFORMATION

Type	Package	Temp.	Clock	Description
Z8470B1	DIP-40 (plastic)	0/+ 70°C	2.5 MHz	Z80 Dual Channel Asynchronous Receiver Transmitter
Z8470F1	DIP-40 (frit seal)	0/+ 70°C		
Z8470D1	DIP-40 (ceramic)	0/+ 70°C		
Z8470D6	DIP-40 (ceramic)	−40/+ 85°C		
Z8470D2	DIP-40 (ceramic)	−55/+125°C		
Z8444C1	PLCC44 (plastic chip-carrier)	0/+ 70°C		
Z8470AB1	DIP-40 (plastic)	0/+ 70°C	4 MHz	
Z8470AF1	DIP-40 (frit seal)	0/+ 70°C		
Z8470AD1	DIP-40 (ceramic)	0/+ 70°C		
Z8470AD6	DIP-40 (ceramic)	−40/+ 85°C		
Z8470AD2	DIP-40 (ceramic)	−55/+125°C		
Z8444AC1	PLCC44 (plastic chip-carrier)	0/+ 70°C		
Z8470BB1	DIP-40 (plastic)	0/+ 70°C	6 MHz	
Z8470BF1	DIP-40 (frit seal)	0/+ 70°C		
Z8470BD1	DIP-40 (ceramic)	0/+ 70°C		
Z8470BD6	DIP-40 (ceramic)	−40/+ 85°C		
Z8470BD2	DIP-40 (ceramic)	−55/+125°C		
Z8444BC1	PLCC44 (plastic chip-carrier)	0/+ 70°C		

AUSTRALIA

NSW 2027 EDGECLIFF
Suite 211, Edgecliff centre
203-233, New South Head Road
Tel. (61-2) 327.39.22
Telex: 071 126911 TCAUS
Telefax: (61-2) 327.61.76

BRAZIL

05413 SÃO PAULO
R. Henrique Schaumann 286-CJ33
Tel. (55-11) 883-5455
Telex: (39-11) 37988 "UMBR BR"

CANADA

BRAMPTON, ONTARIO
341 Main St. North
Tel. (416) 455-0505
Telefax: 416-455-2606

CHINA

BEIJING
Beijing No. 5 Semiconductor
Device Factory
14 Wu Lu Tong Road
Da Shang Mau Wai
Tel. (861) 2024378
Telex 222722 STM CH

DENMARK

2730 HERLEV
Herlev Torv, 4
Tel. (45-2) 94.85.33
Telex: 35411
Telefax: (45-2) 948694

FRANCE

94253 GENTILLY Cedex
7 - avenue Gallieni - BP. 93
Tel.: (33-1) 47.40.75.75
Telex: 632570 STMHQ
Telefax: (33-1) 47.40.79.10

67000 STRASBURG
20, Place des Halles
Tel. (33) 88.25.49.90
Telex: 870001F
Telefax: (33) 88.22.29.32

HONG KONG

WANCHAI
22nd Floor - Hopewell centre
183 Queen's Road East
Tel. (852-5) 8615788
Telex: 60955 ESGIES HX
Telefax: (852-5) 8656589

INDIA

NEW DELHI 110 001
Liason Office
c/o Diners Business Services Pvt Ltd
World Trade Tower - First Floor
Barakhamba Lane
Tel. 331 4668 - 331 2840
Telex: 031 63421 DBSD IN
Telefax: 331 2830

ITALY

20090 ASSAGO (MI)
V.le Milanofiori - Strada 4 - Palazzo A/4/A
Tel. (39-2) 89213.1 (10 linee)
Telex: 330131 - 330141 SGSAGR
Telefax: (39-2) 8250449

40033 CASALECCHIO DI RENO (BO)
Via R. Fucini, 12
Tel. (39-51) 591914
Telex: 512442
Telefax: (39-51) 591305

00161 ROMA
Via A. Torlonia, 15
Tel. (39-6) 8443341/2/3/4/5
Telex: 620653 SGSATE I
Telefax: (39-6) 8444474

JAPAN

TOKYO 108
Nisseki - Takanawa Bld. 4F
2-18-10 Takanawa
Minato-Ku
Tel. (81-3) 280-4121
Telefax: (81-3) 280-4131

KOREA

SEOUL 121
8th floor Shinwon Building
823-14, Yuksam-Dong
Kang-Nam-Gu
Tel. (82-2) 552-0399
Telex: SGSKOR K29998
Telefax: (82-2) 552-1051

NETHERLANDS

5612 AM EINDHOVEN
Dillenburgstraat 25
Tel.: (31-40) 550015
Telex: 51186
Telefax: (31-40) 528835

SINGAPORE

SINGAPORE 2056
28 Ang Mo Kio - Industrial Park 2
Tel. (65) 4821411
Telex: RS 55201 ESGIES
Telefax: (65) 4820240

SPAIN

08021 BARCELONA
Calle Platon, 6 4th Floor, 5th Door
Tel. (34-3) 2022017-2020316
Telefax: (34-3) 2021461

28027 MADRID
Calle Albacete, 5
Tel. (34-1) 4051615
Telex: 27060 TCCEE
Telefax: (34-1) 4031134

SWEDEN

S-16421 KISTA
Borgarfjordsgatan, 13 - Box 1094
Tel.: (46-8) 7939220
Telex: 12078 THSWS
Telefax: (46-8) 7504950

SWITZERLAND

1218 GRAND-SACONNEX (GENÈVA)
Chemin Franois-Lehmann, 18/A
Tel. (41-22) 7986462
Telex: 415493 STM CH
Telefax: (41-22) 7984869

TAIWAN

TAIPEI
12th Floor
571, Tun Hua South Road
Tel. (886-2) 755-4111
Telex: 10310 ESGIE TW
Telefax: (886-2) 755-4008

UNITED KINGDOM and EIRE

MARLOW, BUCKS
Planar House, Parkway
Globe Park
Tel.: (44-628) 890800
Telex: 847458
Telefax: (44-628) 890391

SALES OFFICES

U.S.A.

NORTH & SOUTH AMERICAN
MARKETING HEADQUARTERS
1000 East Bell Road
Phoenix, AZ 85022-2699
(1)-(602) 867-6340

SALES COVERAGE BY STATE

ALABAMA
Huntsville - (205) 533-5995

ARIZONA
Phoenix - (602) 867-6340

CALIFORNIA
Irvine - (714) 250-0455
San Jos - (408) 452-8585

COLORADO
Boulder (303) 449-9000

GEORGIA
Norcross - (404) 242-7444

ILLINOIS
Schaumburg - (708) 517-1890

MARYLAND
Columbia - (301) 995-6952

MASSACHUSSETTS
Waltham - (617) 890-6688

NEW JERSEY
Voorhees - (609) 772-6222

OREGON
Tigard - (503) 620-5517

TEXAS
Austin - (512) 339-4191
Carrollton - (214) 466-8844

WASHINGTON
Seattle - (206) 524-6421

FOR RF AND MICROWAVE
POWER TRANSISTORS CONTACT
THE FOLLOWING REGIONAL
OFFICES IN THE U.S.A.

CALIFORNIA
Hawthorne - (213) 675-0742

NEW JERSEY
Totowa - (201) 890-0884

PENNSYLVANIA
Montgomeryville - (215) 362-8500

TEXAS
Carrollton - (214) 466-8844

WEST GERMANY

6000 FRANKFURT
Gutleutstrabe 322
Tel. (49-69) 237492
Telex: 176997 689
Telefax: (49-69) 231957
Teletex: 6997689=STVBP

8011 GRASBRUNN
Bretonischer Ring 4
Neukeferloh Technopark
Tel.: (49-89) 460060
Telex: 528211
Telefax: (49-89) 4605454
Teletex: 897107=STDISTR

3000 HANNOVER 1
Eckenerstrasse 5
Tel. (49-511) 634191
Telex 175118418
Teletex: 5118418 csfbeh
Telefax: (49-511) 633552

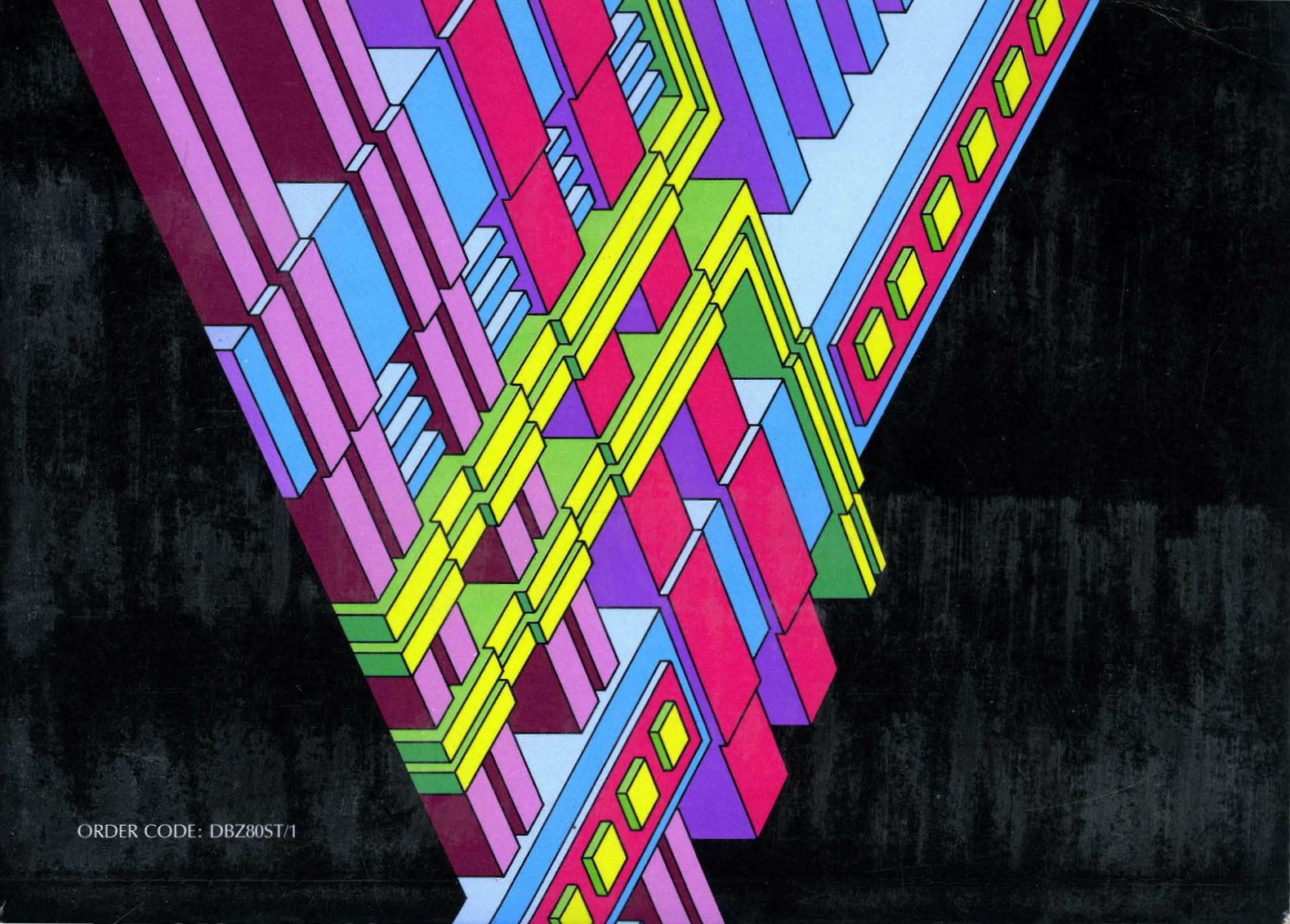
8500 NÜRNBERG 20
Erlenstegenstrasse, 72
Tel.: (49-911) 597032
Telex: 626243
Telefax: (49-911) 5980701

5200 SIEGBURG
Frankfurter Str. 22a
Tel. (49-2241) 660 84-86
Telex: 889510
Telefax: (49-2241) 67584

7000 STUTTGART
Oberer Kirchhaldenweg 135
Tel. (49-711) 692041
Telex: 721718
Telefax: (49-711) 691408

Cover design by Keit & Koppel, Segrate, Italy
Typesetting and layout on Desk Top Publishing
by AZIMUT, Hénin Bt., France
Printed by Garzanti, Cernusco S./N., Italy

SGS-THOMSON Microelectronics GROUP OF COMPANIES
Australia - Brazil - China - France - Hong Kong - Italy - Japan - Korea - Malaysia - Malta - Morocco - The Netherlands - Singapore - Spain - Sweden - Switzerland - Taiwan - United Kingdom - U.S.A. - West Germany



ORDER CODE: DBZ80ST/1