



Z80S188

GENERAL-PURPOSE INTEGRATED
MICROPROCESSOR

PRELIMINARY
PRODUCT SPECIFICATION

PS001500-ZMP0999



©1999 by ZiLOG, Inc. All rights reserved. Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZiLOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZiLOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. Except with the express written approval of ZiLOG, use of information, devices, or technology as critical components of life support systems is not authorized. No licenses are conveyed, implicitly or otherwise, by this document under any intellectual property rights.



TABLE OF CONTENTS

TABLE OF CONTENTS	3
LIST OF FIGURES	5
LIST OF TABLES	7
OVERVIEW	1
FEATURES	1
GENERAL DESCRIPTION	1
BLOCK DIAGRAM	3
PIN DESCRIPTIONS	4
OPERATIONAL DESCRIPTION	12
PROCESSOR DESCRIPTION	12
SECURITY FEATURES AND ROM OPTIONS	15
INTERRUPTS	16
MEMORY (ROM AND RAM)	29
INPUT/OUTPUT	37
CLOCKING	41
RESET CONDITIONS	43
POWER MANAGEMENT	43
PARALLEL I/O (PIOs)	45
DMA CHANNELS	57
WATCH-DOG TIMER	63
COUNTER/TIMER CHANNELS (CTCs)	63
PROGRAMMABLE RELOAD TIMERS (PRTs)	70
SERIAL I/O CHANNELS (SIOs)	72
ASYNC SERIAL COMMUNICATIONS INTERFACES (ASCIs)	95
CLOCKED SERIAL INPUT/OUTPUT MODULE (CSI/O)	106
I/O REGISTERS	109
REGISTERS SUMMARY	109
BASIC DEVICE REGISTERS	111
INTERRUPT REGISTERS	116
MMU REGISTERS	118
CHIP SELECT AND WAIT REGISTERS	123
I/O PORT (PIO) REGISTERS	128
DMA REGISTERS	132
WATCH-DOG TIMER REGISTERS	143
COUNTER/TIMER (CTC) REGISTERS	144
PROGRAMMABLE RELOAD TIMER (PRT) REGISTERS	147
SERIAL I/O (SIO) REGISTERS	152
ASYNC SERIAL COMMUNICATIONS INTERFACE (ASCI) REGISTERS	167
CLOCKED SERIAL I/O (CSI/O) REGISTERS	176
INSTRUCTION SET	177
CLASSES OF INSTRUCTIONS	178
PROCESSOR FLAGS	180
CONDITION CODES	181
NOTATION	183
ASSEMBLY LANGUAGE SYNTAX	184



INSTRUCTION SUMMARY	184
OP CODE MAP	193
ELECTRICAL CHARACTERISTICS	200
ABSOLUTE MAXIMUM RATINGS	200
STANDARD TEST CONDITIONS	200
DC CHARACTERISTICS	201
AC CHARACTERISTICS	203
CAPACITANCE	214
TIMING DIAGRAMS	215
CHARACTERISTIC CURVES	227
SYSTEM DESIGN CONSIDERATIONS	227
ERRATA	227
APPLICATION NOTES/DEVELOPMENT TOOLS	228
ZiLOG DEBUG INTERFACE	228
ORDERING INFORMATION	229
PART NUMBER DESCRIPTION	230
DISCLAIMER	232
PRECHARACTERIZATION PRODUCT	232
DOCUMENT INFORMATION	232
CHANGE LOG	232

LIST OF FIGURES

FIGURE 1.	FUNCTIONAL BLOCK DIAGRAM	4
FIGURE 2.	Z80S188 PIN DIAGRAM	5
FIGURE 3.	AN INTERRUPT ACKNOWLEDGE DAISY CHAIN	20
FIGURE 4.	INTO MODE 0 TIMING	23
FIGURE 5.	INTO MODE 1 TIMING	24
FIGURE 6.	INTO MODE 2 TIMING	26
FIGURE 7.	RETI INSTRUCTION WITH M1E=0	28
FIGURE 8.	3-CLOCK REFRESH CYCLE	37
FIGURE 9.	I/O CYCLE TIMING	40
FIGURE 10.	FUNDAMENTAL MODE CRYSTAL CIRCUIT < 20 MHz	42
FIGURE 11.	THIRD-OVERTONE CRYSTAL > 20 MHz	42
FIGURE 12.	MODE 0 OUTPUT TIMING	47
FIGURE 13.	MODE 1 INPUT TIMING	47
FIGURE 14.	MODE 2 BIDIRECTIONAL TIMING	48
FIGURE 15.	MODE 3 BIT CONTROL TIMING, BIT MODE READ	49
FIGURE 16.	PROCESSOR/DMA OPERATION WITH LEVEL-SENSE REQUEST	59
FIGURE 17.	PROCESSOR/DMA OPERATION WITH EDGE-SENSE REQUEST	59
FIGURE 18.	RESIDUE VALUES VS. LAST CHARACTERS OF FRAME (8 BITS/CHAR)	84
FIGURE 19.	RESIDUE VALUES VS. LAST CHARACTERS OF FRAME (7 BITS/CHAR)	85
FIGURE 20.	RESIDUE VALUES VS. LAST CHARACTERS OF FRAME (6 BITS/CHAR)	85
FIGURE 21.	RESIDUE VALUES VS. LAST CHARACTERS OF FRAME (5 BITS/CHAR)	86
FIGURE 22.	AN ASYNCHRONOUS CHARACTER	98
FIGURE 23.	CSI/O OPERATION	108
FIGURE 24.	TEST LOAD CIRCUIT	201
FIGURE 25.	BASIC TIMING	214
FIGURE 26.	MEMORY READ TIMING (ONE WAIT STATE)	215
FIGURE 27.	MEMORY WRITE TIMING (NO WAIT STATES)	216
FIGURE 28.	I/O READ TIMING (AUTO WAIT STATE)	217
FIGURE 29.	I/O WRITE TIMING (AUTO AND ONE WAIT STATE)	218
FIGURE 30.	BUS EXCHANGE TIMING	218
FIGURE 31.	INTERRUPT TIMING	219
FIGURE 32.	INTERRUPT ACKNOWLEDGE TIMING	220
FIGURE 33.	REFRESH TIMING (2-CLOCK CYCLE)	220
FIGURE 34.	DMA REQUEST TIMING	221
FIGURE 35.	DMA TERMINATION TIMING	221



FIGURE 36. HALT TIMING	221
FIGURE 37. SLEEP TIMING	222
FIGURE 38. PRT TIMING	222
FIGURE 39. ASCI TIMING	222
FIGURE 40. CSI/O TIMING	223
FIGURE 41. WATCH-DOG TIMER TIMING	223
FIGURE 42. PIO MODE 0 (OUTPUT) TIMING	223
FIGURE 43. PIO MODE 1 (INPUT) TIMING	224
FIGURE 44. PIO MODE 2 (BIDIRECTIONAL) TIMING	225
FIGURE 45. PIO MODE 3 (BIT CONTROL) TIMING	225
FIGURE 46. CTC TIMING	226
FIGURE 47. SIO TIMING	226
FIGURE 48. CAPACITIVE LOAD CL VS. SWITCHING TIME	227
FIGURE 49. ZDI CONNECTOR ON TARGET BOARD	227

LIST OF TABLES

TABLE 1.	PIN DESCRIPTIONS	6
TABLE 2.	INTERRUPT OFFSETS AND PRIORITIES	30
TABLE 3.	SIZE/ENABLE VALUES FOR MEMCS0-1	35
TABLE 4.	MEMCS0-1 WAIT STATES	35
TABLE 5.	REFRESH CYCLE CONTROL	37
TABLE 6.	A7-5 DECODING FOR IOCS PIN	39
TABLE 7.	LOW POWER MODES	44
TABLE 8.	SAR0B VALUE FOR A SOURCE IN I/O SPACE	60
TABLE 9.	DAR0B VALUE FOR A DESTINATION IN I/O SPACE	61
TABLE 10.	IAR1B VALUE	61
TABLE 11.	DMA0 SOURCE MODE	62
TABLE 12.	DMA0 DESTINATION MODE	62
TABLE 13.	DMA1 OPERATING MODE	62
TABLE 14.	WRITE AND READ REGISTERS PER CHANNEL	74
TABLE 15.	STATUS AFFECTS VECTOR CODES IN RR2 BITS 3-1	86
TABLE 16.	OLD BRG DIVISION FACTORS	97
TABLE 17.	CSI/O CLOCK RATE SELECTION	107
TABLE 18.	FREE-RUNNING COUNTER (0018H)	112
TABLE 19.	CLOCK MULTIPLIER REGISTER (001EH) CMR	113
TABLE 20.	CPU CONTROL REGISTER (001FH) CCR	114
TABLE 21.	REFRESH CONTROL REGISTER (0036H) RCR	115
TABLE 22.	OPERATING MODE CONTROL REGISTER (003EH) OMCR	115
TABLE 23.	I/O CONTROL REGISTER (003FH) IOCR	116
TABLE 24.	ON-CHIP MEMORY CONTROL REGISTER (xxFC OR 00FCH) OCMCR	116
TABLE 25.	INTERRUPT VECTOR LOW REGISTER (0033H) IL	117
TABLE 26.	INTERRUPT/TRAP CONTROL REGISTER (0034H) ITC	118
TABLE 27.	INTERRUPT PRIORITY REGISTER (xxF4 OR 00F4H) INTPR	119
TABLE 28.	COMMON BASE REGISTER (0038H IN CLASSIC MODE) CBR	120
TABLE 29.	BANK BASE REGISTER (003H9 IN CLASSIC MODE) BBR	120
TABLE 30.	COMMON/BANK AREA REGISTER (003AH IN CLASSIC MODE) CBAR	121
TABLE 31.	COMMON BASE REGISTER LOW (0038H IN EXTENDED MODE) CBRL	121
TABLE 32.	COMMON BASE REGISTER HIGH (0039H IN EXTENDED MODE) CBRH 122	
TABLE 33.	BANK AREA REGISTER (003AH IN EXTENDED MODE) BAR	122
TABLE 34.	COMMON AREA REGISTER (003BH IN EXTENDED MODE) CAR	123

TABLE 35.	BANK BASE REGISTER LOW (003CH IN EXTENDED MODE) BBRL	123
TABLE 36.	BANK BASE REGISTER HIGH (003DH IN EXTENDED MODE) BBRH	124
TABLE 37.	MEMORY CHIP SELECT 0 LOW REGISTER (xxF6 OR 00FH6) MCSOL	125
TABLE 38.	MEMORY CHIP SELECT 0 HIGH REGISTER (xxF7 OR 00F7H) MCSOH	126
TABLE 39.	MEMORY CHIP SELECT 1 LOW REGISTER (xxF8 OR 00F8H) MCS1L	126
TABLE 40.	MEMORY CHIP SELECT 1 HIGH REGISTER (xxF9 OR 00F9H) MCS1H	127
TABLE 41.	I/O CHIP SELECT LOW REGISTER (xxFA OR 00FAH) IOCSL	128
TABLE 42.	I/O CHIP SELECT HIGH REGISTER (xxFB OR 00FBH) IOCSH	128
TABLE 43.	PORT A DATA (xxDC OR 00DCH) PAD	129
TABLE 44.	PORT A CONTROL (xxDD OR 00DDH) PAC	129
TABLE 45.	PORT B DATA (xxDE OR 00DEH) PBD	130
TABLE 46.	PORT B CONTROL (xxDF OR 00DFH) PBC	131
TABLE 47.	PORT C DATA (xxD4 OR 00D4H) PCD	132
TABLE 48.	PORT C CONTROL (xxD5 OR 00D5H) PCC	132
TABLE 49.	PORT D DATA (xxD6 OR 00D6H) PDD	132
TABLE 50.	PORT D CONTROL (xxD7 OR 00D7H) PDC	132
TABLE 51.	DMA0 SOURCE ADDRESS REGISTER LOW (0020H) SAROL	133
TABLE 52.	DMA0 SOURCE ADDRESS REGISTER HIGH (0021H) SAROH	133
TABLE 53.	DMA0 SOURCE ADDRESS REGISTER B (0022H) SAROB	134
TABLE 54.	DMA0 DESTINATION ADDRESS REGISTER LOW (0023H) DAROL	134
TABLE 55.	DMA0 DESTINATION ADDRESS REGISTER HIGH (0024H) DAROH	135
TABLE 56.	DMA0 DESTINATION ADDRESS REGISTER B (0025H) DAROB	135
TABLE 57.	DMA0 BYTE COUNT REGISTER LOW (0026H) BCR0L	136
TABLE 58.	DMA0 BYTE COUNT REGISTER HIGH (0027H) BCR0H	136
TABLE 59.	DMA1 MEMORY ADDRESS REGISTER LOW (0028H) MAR1L	136
TABLE 60.	DMA1 MEMORY ADDRESS REGISTER HIGH (0029H) MAR1H	137
TABLE 61.	DMA1 MEMORY ADDRESS REGISTER B (002AH) MAR1B	137
TABLE 62.	DMA1 I/O ADDRESS REGISTER LOW (002BH) IAR1L	137
TABLE 63.	DMA1 I/O ADDRESS REGISTER HIGH (002CH) IAR1H	138
TABLE 64.	DMA1 I/O ADDRESS REGISTER B (002DH) IAR1B	138
TABLE 65.	DMA1 BYTE COUNT REGISTER LOW (002EH) BCR1L	140
TABLE 66.	DMA1 BYTE COUNT REGISTER HIGH (002FH) BCR1H	140
TABLE 67.	DMA STATUS REGISTER (0030H) DSTAT	141

TABLE 68.	DMA MODE REGISTER (0031H) DMODE	142
TABLE 69.	DMA/WAIT CONTROL REGISTER (0032H) DCNTL	143
TABLE 70.	WATCH-DOG TIMER MASTER REGISTER (xxF0 OR 00F0H) WDTMR	144
TABLE 71.	WATCH-DOG TIMER COMMAND REGISTER (xxF1 OR 00F1H) WDTCR	144
TABLE 72.	CTC0 (xxD0 OR 00D0H) CTC0	145
TABLE 73.	CTC1 (xxD1 OR 00D1H) CTC1	147
TABLE 74.	CTC2 (xxD2 OR 00D2H) CTC2	147
TABLE 75.	CTC3 (xxD3 OR 00D3H) CTC3	147
TABLE 76.	PRT0 TIMER DATA REGISTER LOW (000CH) TMDROL	148
TABLE 77.	PRT0 TIMER DATA REGISTER HIGH (000DH) TMDROH	148
TABLE 78.	PRT0 RELOAD REGISTER LOW (000EH) RLDR0L	148
TABLE 79.	PRT0 RELOAD REGISTER HIGH (000FH) RLDR0H	149
TABLE 80.	TIMER CONTROL REGISTER (0010H) TCR	149
TABLE 81.	TIMER PRESCALE REGISTER (0011H) TPR	150
TABLE 82.	PRT1 TIMER DATA REGISTER LOW (0014H) TMDR1L	150
TABLE 83.	PRT1 TIMER DATA REGISTER HIGH (0015H) TMDR1H	151
TABLE 84.	PRT1 RELOAD REGISTER LOW (0016H) RLDR1L	151
TABLE 85.	PRT1 RELOAD REGISTER HIGH (0017H) RLDR1H	151
TABLE 86.	SIO A DATA (xxD8 OR 00D8H) SIOAD	152
TABLE 87.	SIO A CONTROL (xxD9 OR 00D9H) SIOAC	152
TABLE 88.	SIO WR0 (WRITE TO xxD9, 00D9, xxDB, OR 00DBH w/WR0 2-0 = 000)	153
TABLE 89.	SIO WR1 (WRITE TO xxD9, 00D9, xxDB, OR 00DBH WITH WR0 2-0 = 001)	154
TABLE 90.	SIO WR3 (WRITE TO xxD9, 00D9, xxDB, OR 00DBH WITH WR0 2-0 = 011)	155
TABLE 91.	SIO WR4 (WRITE TO xxD9, 00D9, xxDB, OR 00DBH WITH WR0 2-0 = 100)	156
TABLE 92.	SIO WR5 (WRITE TO xxD9, 00D9, xxDB, OR 00DBH WITH WR0 2-0 = 101)	157
TABLE 93.	SIO WR6 (WRITE TO xxD9, 00D9, xxDB, OR 00DBH WITH WR0 2-0 = 110)	158
TABLE 94.	SIO WR7 (WRITE TO xxD9, 00D9, xxDB, OR 00DBH WITH WR0 2-0 = 111)	159
TABLE 95.	SIO RR0 (READ F/xxD9, 00D9, xxDB, OR 00DBH WITH WR0 2-0 = 000)	159
TABLE 96.	SIO RR1 (READ F/xxD9, 00D9, xxDB, OR 00DBH WITH WR0 2-0 = 001)	161
TABLE 97.	SIO B DATA (xxDA OR 00DAH) SIOBD	163
TABLE 98.	SIO B CONTROL (xxDB OR 00DBH) SIOBC	163

TABLE 99. SIO B WR2 (WRITE TO xxDB OR 00DBH WITH WR0 2-0=010)	163
TABLE 100. SIO B RR2 (READ FROM xxDB OR 00DBH WITH WR0 2-0=010)	164
TABLE 101. ASCIO CONTROL REGISTER A (0000H) CNTLA0	165
TABLE 102. ASCI1 CONTROL REGISTER A (0001H) CNTLA1	166
TABLE 103. ASCIO CONTROL REGISTER B (0002H) CNTLB0	166
TABLE 104. ASCI1 CONTROL REGISTER B (0003H) CNTLB1	167
TABLE 105. ASCIO STATUS REGISTER (0004H) STAT0	168
TABLE 106. ASCI1 STATUS REGISTER (0005H) STAT1	169
TABLE 107. ASCIO Tx DATA REGISTER (0006H) TDR0	169
TABLE 108. ASCI1 Tx DATA REGISTER (0007H) TDR1	170
TABLE 109. ASCIO Rx DATA REGISTER (0008H) RDR0	170
TABLE 110. ASCI1 Rx DATA REGISTER (0009H) RDR1	170
TABLE 111. ASCIO EXTENSION CONTROL REGISTER (0012H) ASEXTO	171
TABLE 112. ASCI1 EXTENSION CONTROL REGISTER (0013H) ASEXTO1	172
TABLE 113. ASCIO TIME CONSTANT LOW (001AH) ASTCOL	172
TABLE 114. ASCIO TIME CONSTANT HIGH (001BH) ASTCOH	173
TABLE 115. ASCI1 TIME CONSTANT LOW (001CH) ASTC1L	173
TABLE 116. ASCIO TIME CONSTANT HIGH (001DH) ASTC1H	173
TABLE 117. CSI/O CONTROL REGISTER (000AH) CNTR	174
TABLE 118. CSI/O DATA REGISTER (000BH) TRDR	176
TABLE 119. LOAD INSTRUCTIONS	177
TABLE 120. ARITHMETIC INSTRUCTIONS	177
TABLE 121. LOGICAL INSTRUCTIONS	177
TABLE 122. EXCHANGE INSTRUCTIONS	177
TABLE 123. PROGRAM CONTROL INSTRUCTIONS	178
TABLE 124. BIT MANIPULATION INSTRUCTIONS	178
TABLE 125. BLOCK TRANSFER INSTRUCTIONS	178
TABLE 126. ROTATE AND SHIFT INSTRUCTIONS	178
TABLE 127. INPUT/OUTPUT INSTRUCTIONS	179
TABLE 128. PROCESSOR CONTROL INSTRUCTIONS	179
TABLE 129. FLAG REGISTER	180
TABLE 130. FLAG SETTINGS DEFINITIONS	180
TABLE 131. CONDITION CODES	181
TABLE 132. SYMBOLS	182
TABLE 133. INSTRUCTION SUMMARY	183
TABLE 134. OP CODE MAP (1ST OP CODE)	193
TABLE 135. OP CODE MAP (2ND OP CODE AFTER 0CBH)	194
TABLE 136. OP CODE MAP (2ND OP CODE AFTER 0DDH)	195



TABLE 137. OP CODE MAP (2ND OP CODE AFTER 0EDH)	196
TABLE 138. OP CODE MAP (2ND OP CODE AFTER 0FDH)	197
TABLE 139. OP CODE MAP (4TH BYTE, AFTER 0DDH, 0CBH, AND D)	198
TABLE 140. OP CODE MAP (4TH BYTE, AFTER 0FDH, 0CBH, AND D)	199
TABLE 141. ABSOLUTE MAXIMUM RATINGS	200
TABLE 142. DC CHARACTERISTICS, TA = 0°C TO +70°C	201
TABLE 143. DC CHARACTERISTICS, TA = -40°C TO +85°C	202
TABLE 144. AC CHARACTERISTICS, TA = 0°C TO +70°C, CL = 100 pF	203
TABLE 145. AC CHARACTERISTICS, TA = -40°C TO +85°C, CL = 100 pF	208





OVERVIEW

FEATURES

- Code-compatible with Z80 and Z8x180
- Two multiprotocol Serial I/O channels (Z80-SIO)
- Four 8-bit parallel ports with handshake logic (2 Z80-PIO)
- Four Counter/Timer channels (Z80-CTC)
- Two 16-bit Programmable Reload Timers (PRTs)
- 4-KB ROM with security protection
- 1-KB RAM with security protection
- Enhanced Memory Management Unit (MMU) addresses up to 8 MB
- Two Direct Memory Access channels (DMAs)
- Two Enhanced UART channels (ASCIs)
- Clocked Serial I/O interface (CSI/O)
- Watch-Dog Timer (WDT)
- Chip select and wait state generators
- Three interrupt request inputs
- 32-Bit CRC on SIO channels
- ZiLOG Debug Interface (ZDI)
- DC to 33-MHz operating frequency @ 5.0V
- DC to 20-MHz operating frequency @ 3.3V
- Clock divide by 2, 1X, or 2X
- Fully static CMOS design with low-power standby modes
- 160-Pin QFP package

GENERAL DESCRIPTION

The Z80S188 is a general-purpose integrated microprocessor. It includes the Z8S180 processor, four 8-bit I/O ports, two multiprotocol serial channels, four Counter Timer channels, an enhanced MMU that addresses up to 8 MB of memory, two DMA channels, a Watch-Dog Timer, two enhanced UARTs, two Programmable Reload Timers, a CSI/O channel, 4 KB of on-chip RAM, and 1 KB of on-chip ROM. It is packaged in a 160-pin QFP.

32 Bits of General-Purpose I/O. These four 8-bit ports, designated A through D, are functionally identical to two Z80 PIO devices. Several programmable modes of operation are featured, including input and output handshaking, or data direction, and interrupt control at the bit/pin level.

Two Multiprotocol Serial Channels. These channels are functionally identical to a Z80 SIO, and support Asynchronous, isochronous, Classic Synchronous, and HDLC/SDLC protocols.

Four Counter/Timer Circuits. These CTCs are functionally identical to a Z80 CTC device. Each channel provides counter or timer functionality, an 8-bit down-counter with /16 or /256 timer-prescaler, a clock or trigger input, a 0 count/timeout output, and interrupt capability.

Two 16-Bit Timers. These timers are 16-bit down-counters with auto-reload and interrupt capability. One channel has waveform generation capability.

Memory Management Unit (MMU). This module has been enhanced from the MMU found on other 8X18X processors. The MMU now translates addresses from the 64-KB logical memory address space used by software, to an 8 megabyte physical memory address space, with a granularity of 1KB.

Two DMA Channels. These channels operate in an 8-Mbyte linear memory address space and a 64-KB I/O space, and can transfer blocks up to 64 KB long. These channels can be used with external memory, and external or on-chip peripherals.

Two Asynchronous Serial Channel Interfaces. These ASCIs feature baud rate generators, modem control, and status.

CSI/O. Clocked serial I/O can be used for serial interfacing to memory, peripherals, and other processors.

Chip Select and Wait logic. This feature contains two memory chip select signals and one I/O chip select signal for external memory and I/O. A programmed number of wait states can be inserted for each of these three programmable address ranges.

Watch-Dog Timer. The WDT helps detect code runaway and helps minimize the negative effects thereof. A range of timeout values is available. The $\overline{\text{WDTOUT}}$ pin is driven Low if the Watch-Dog Timer counts down to 0 and can be connected to $\overline{\text{RESET}}$ or $\overline{\text{NMI}}$, or used in some other way.

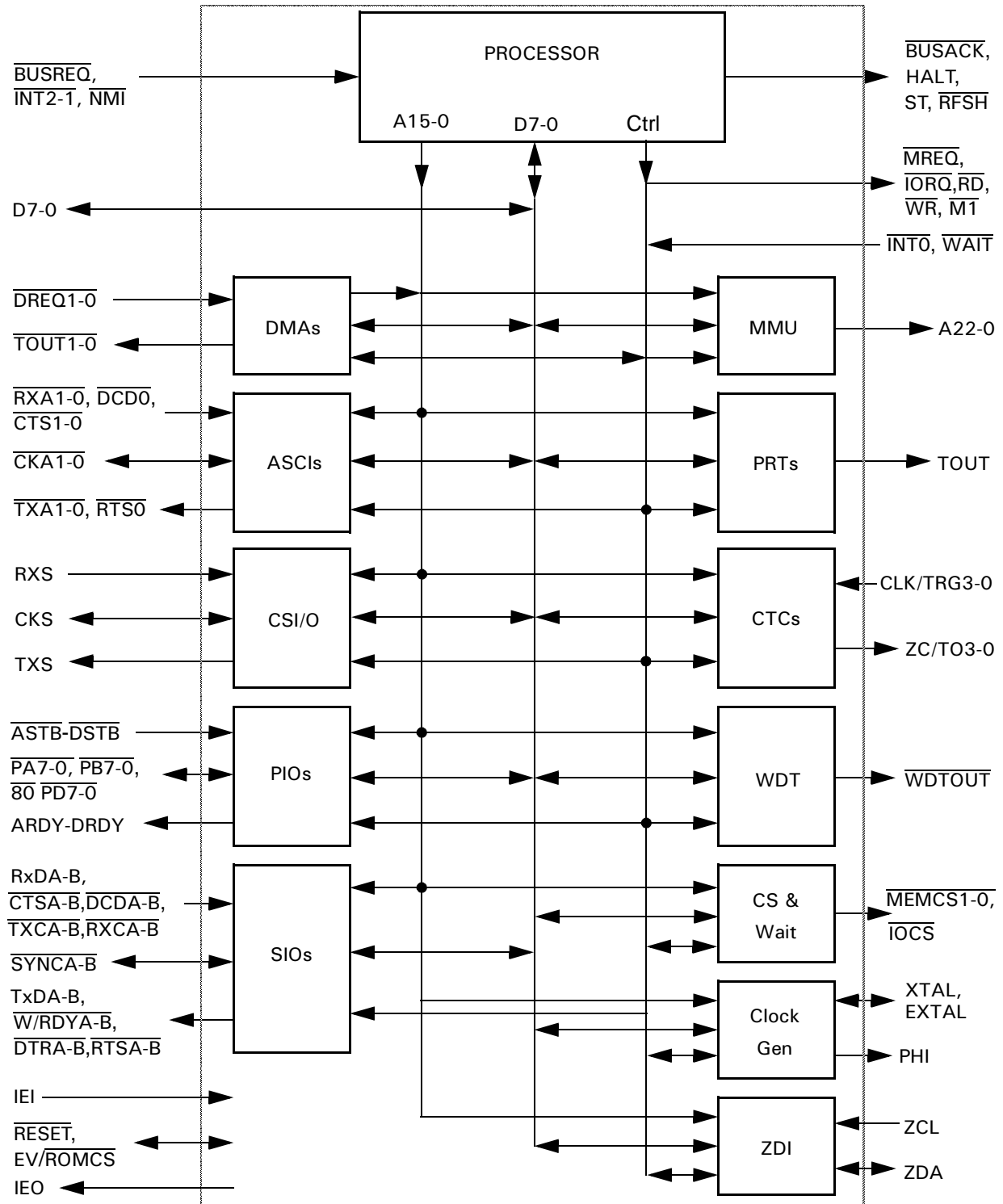
4-KB of On-Chip ROM. Optional mask-programmed security features allow restricted access via nine entry points in low memory, and no access or visibility from the external bus.

1-KB of On-chip RAM. Optional mask-programmed security feature restricts access to instructions in on-chip ROM.

ZiLOG Debug Interface. The ZDI module allows in-system debugging using the ZiLOG Developer Studio (ZDS) running on a PC, and requires a 6-pin header on the application board and a low-cost interface module.

BLOCK DIAGRAM

FIGURE 1. FUNCTIONAL BLOCK DIAGRAM



PIN DESCRIPTIONS

FIGURE 2. Z80S188 PIN DIAGRAM

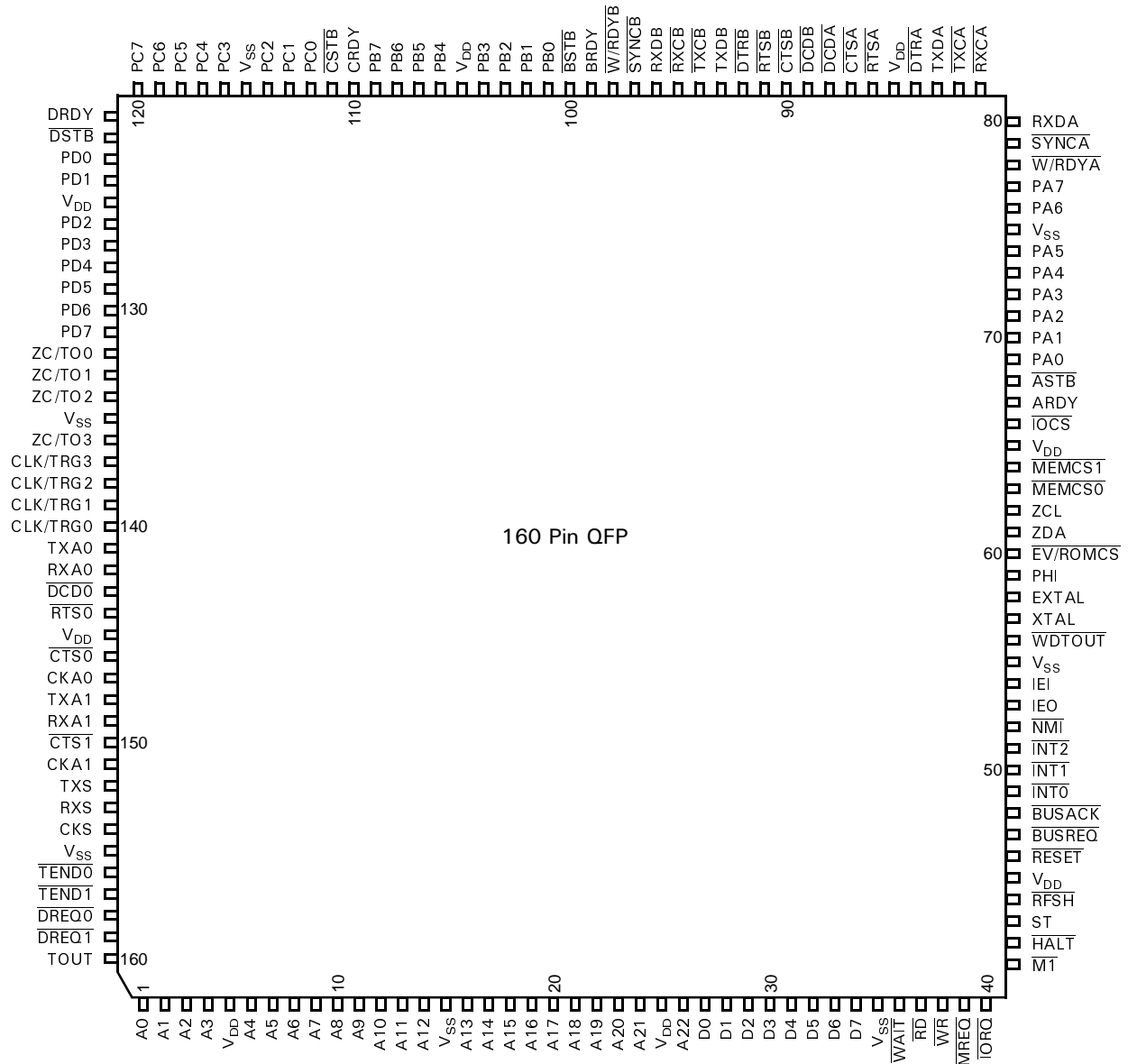


TABLE 1. PIN DESCRIPTIONS

Symbol	Pin #	Function	Type	Description
A22-0	1-4, 6-15, 17-24, 26	Address Bus	Bidirectional, 3 state	These lines select a location in memory or I/O space to be read or written. The Z80S188 does not drive these lines when $\overline{\text{RESET}}$ or $\overline{\text{BUSACK}}$ is Low.
ARDY, BRDY, CRDY, DRDY	67, 99, 110, 121	Port A-D Ready	Outputs	These pins provide Handshake signals to external device(s). Signaling on these lines depends on the operating mode of each port.
$\overline{\text{ASTB}}$, $\overline{\text{BSTB}}$, $\overline{\text{CSTB}}$, $\overline{\text{DSTB}}$	68, 100, 111, 122	Port A-D Strobes	Inputs	These pins provide Handshake signals from external device(s). Signaling on these lines depends on the operating mode of each port.
$\overline{\text{BUSACK}}$	48	Bus Acknowledge	Output, active Low	In response to a Low on $\overline{\text{BUSREQ}}$, the Z80S188 completes its current bus cycle, releases the address, data, and control signals to high-impedance state, and drives this line Low. This state persists until after the external master drives or releases $\overline{\text{BUSREQ}}$ to High.
$\overline{\text{BUSREQ}}$	47	Bus Request	Input, active Low	An external bus master can drive this line Low to make the Z80S188 release the bus. Pull up this signal if it is not used, either with a resistor or by direct connection to V_{DD} .
CKA0-1	147, 151	ASCI Clocks	Input/ outputs	These pins carry clocking to each of the ASCIs, or output clocking from the ASCIs.
CKS	154	CSI/O Clock	Input/output	This pin carries clocking to the CSI/O, or outputs clocking from the CSI/O.
CLK/TRG0-3	137-149	CTC Clock/ Triggers	Inputs	When the CTC channel associated with each of these pins is active in Counter mode, an active edge on the pin decrements the channel's downcounter. When the channel is active in Triggered Timer mode, an active edge starts the timer. In either mode, software selects whether rising or falling edges are active.
CRDY, $\overline{\text{CSTB}}$	(Described in ARDY, $\overline{\text{ASTB}}$ entries above)			
$\overline{\text{CTS0-1}}$	146, 150	ASCI Clears to Send	Inputs, active Low	These pins transmit control signals for the ASCIs. Optionally, they auto-enable the ASCII Transmitters, and can be read by software.
$\overline{\text{CTSA-B}}$	87, 90	SIO Clears to Send	Inputs, active Low	These pins transmit control signals for the SIO channels. Optionally, they auto-enable the Transmitters, and can be read by software.
D7-0	27-34	Data Bus	Bidirectional, 3 state	These lines transfer information to and from I/O and memory devices. The Z80S188 drives these lines only during Write cycles.



TABLE 1. PIN DESCRIPTIONS (CONTINUED)

Symbol	Pin #	Function	Type	Description
$\overline{\text{DCD0}}$	143	ASCI Data Carrier Detect	Input, active Low	This pin is a receive control signal for ASCIO. Optionally, this pin auto-enables the Receiver, and can be read by software.
$\overline{\text{DCDA-B}}$	88-89	SIO Data Carrier Detects	Inputs, active Low	These pins provide receive control signals for the SIO channels. Optionally, these signals auto-enable the Receivers, and can be read by software.
$\overline{\text{DRDY}}, \overline{\text{DSTB}}$	(Described in $\overline{\text{ARDY}}, \overline{\text{ASTB}}$ entries above)			
$\overline{\text{DREQ0-1}}$	158, 159	DMA Requests	Inputs	This pin is used by external peripheral(s) to request transfers from the DMA channels. These inputs can be programmed as either level- or edge- sensitive.
$\overline{\text{DTRA-B}}$	84, 92	SIO Data Terminal Ready	Outputs	General purpose outputs controlled by register bits in the SIOs.
$\overline{\text{EV/ROMCS}}$	60	Emulation mode/ ROM Chip Select	Input/output	The Z80S188 samples this pin at the rising edges of $\overline{\text{RESET}}$. If the pin is Low, the device 3-states all its outputs so that an external emulator can control the board on which it resides. In this state, no on-chip resources can be accessed or used. If the pin is High at the rising edge of $\overline{\text{RESET}}$, and it has not been mask-programmed to enable its on-chip ROM, it enables a 3-state driver. This action drives this pin Low whenever the address is in the range occupied by on-chip ROM in masked parts.
EXTAL	58	Oscillator or Clock In	Input	This pin can be connected to a crystal or to an external clock. If a crystal is used, this signal is not a logic level.
$\overline{\text{HALT}}$	42	Halt Indica- tion	Output	This pin becomes Low when the CPU executes either a HALT or SLEEP instruction. An external signal ($\overline{\text{RESET}}$ or an interrupt request) makes the processor active again.
IEI	54	Interrupt Enable In	Input	If external Z80 peripherals are connected to the Z80S188, and they have higher interrupt priority than the on-chip CTCs, PIOs, and SIO channels, this pin connects to the IEO output of the lowest-priority such device. Otherwise, this pin connects to V_{DD} .
IEO	53	Interrupt Enable Out	Output	If external Z80 peripherals are connected to the Z80S188, and they have lower interrupt priority than the on-chip CTCs, PIOs, and SIO channels, this pin connects to the IEI input of the highest-priority such device.

TABLE 1. PIN DESCRIPTIONS (CONTINUED)

Symbol	Pin #	Function	Type	Description
$\overline{\text{INT0}}$	49	Interrupt Request 0	Input/output, active Low	This signal can be driven Low in an open-drain fashion by external I/O devices. The Z80S188 drives this pin Low in an open-drain fashion, whenever any of the on-chip PIO ports, CTC channels, or SIO channels request an interrupt. The processor executes this request at the end of the current instruction cycle as long as it is enabled, and the $\overline{\text{NMI}}$ and $\overline{\text{BUSREQ}}$ signals are inactive.
$\overline{\text{INT1-2}}$	50-51	Interrupt Requests 1-2	Inputs, Active Low or Edge-Triggered	These signals are generated by external devices. The processor executes a request on one of these lines at the end of the current instruction cycle, while the $\overline{\text{NMI}}$, $\overline{\text{BUSREQ}}$, and $\overline{\text{INT0}}$ signals are inactive. The processor acknowledges one of these requests with an internal cycle, in which a fixed vector corresponding to one of the pins is used to select an interrupt service routine. These pins may be programmed for active Low level, rising- or falling-edge interrupts. The state of the external $\overline{\text{INT1}}$ and $\overline{\text{INT2}}$ pins can be read in the Interrupt Edge Register.
$\overline{\text{IOCS}}$	66	I/O Chip Select	Output, Active Low	The Z80S188 drives this output Low when software accesses an I/O address in a programmable range.
$\overline{\text{IORQ}}$	40	I/O Request	Output, Active Low, 3-State	With $\overline{\text{RD}}$ or $\overline{\text{WR}}$ Low, a Low on $\overline{\text{IORQ}}$ indicates that the processor is accessing a location in I/O space. With $\overline{\text{M1}}$ Low, a Low on $\overline{\text{IORQ}}$ indicates an interrupt acknowledge cycle. The Z80S188 does not drive this line during reset, nor during bus acknowledge cycles.
$\overline{\text{M1}}$	41	Machine cycle 1	Output, Active Low	$\overline{\text{MREQ}}$ and $\overline{\text{M1}}$ both Low indicate that the current cycle is the Op Code Fetch cycle of an instruction execution. $\overline{\text{IORQ}}$ and $\overline{\text{M1}}$ both Low indicate an interrupt acknowledge cycle.
$\overline{\text{MEMCS0-1}}$	63-64	Memory Chip Selects	Outputs, Active Low	A Low on one of these lines indicates that the current bus cycle is in memory space, and the physical address from the MMU is in the range programmed for the line.
$\overline{\text{MREQ}}$	39	Memory Request	Output, Active Low	A Low on this line indicates that the current bus cycle is in memory space.



TABLE 1. PIN DESCRIPTIONS (CONTINUED)

Symbol	Pin #	Function	Type	Description
$\overline{\text{NMI}}$	52	Non-Maskable Interrupt	Input, Falling-Edge Active	$\overline{\text{NMI}}$ has a higher priority than $\overline{\text{INT0}}$ and is always recognized at the end of an instruction, regardless of the state of the interrupt enable flip-flops. This signal forces processor execution to location 0066H. This input includes a Schmitt trigger to allow RC rise times.
PA7-0, PB7-0, PC7-0, PD7-0	69-74, 76-77, 101-104, 106-109, 112-114, 116-120, 123-124, 125-131	Parallel Ports	Input/ Outputs	Each of these four ports can be independently programmed for 8-bit parallel input, output, or bidirectional operation with optional external strobes, or for any mixture of input and output pins.
PHI	59	System Clock	Output	This output is the processor's master clock, and is provided for use by external logic. The frequency of this clock may be equal to, half of, or twice the crystal or input clock frequency, depending on internal register bits.
RD	37	Read Strobe	Output, Active Low, 3-State	$\overline{\text{RD}}$ Low indicates that the processor is reading data from a memory or I/O location. The Z80S188 does not drive this line while $\overline{\text{RESET}}$ or $\overline{\text{BUSACK}}$ is Low.
$\overline{\text{RESET}}$	46	Master Reset	Input/ Output, Active Low	This signal initializes the Z80S188 and other devices in the system. This input is held Low until the clock is stable, and must be Low for a minimum of three system clock cycles. $\overline{\text{RESET}}$ can be programmed as an output, to allow the Z80S188 to reset external devices. Power On Reset causes a global reset by forcing $\overline{\text{RESET}}$ Low. As an input, this pin has a Schmitt trigger receiver, to enable RC rise times.
$\overline{\text{RFSH}}$	44	Refresh	Output, Active Low, 3-state	$\overline{\text{RFSH}}$ Low indicates that the processor is performing a refresh cycle for dynamic memory. The Z80S188 does not drive this line while $\overline{\text{RESET}}$ or $\overline{\text{BUSACK}}$ is Low.
$\overline{\text{RTS0}}$	144	ASCIO Request to Send	Output	This pin is a general-purpose output controlled by a bit in ASCIO's CNTLA register.
$\overline{\text{RTSA-B}}$	86, 91	SIO Requests to Send	Outputs	These pins are general-purpose outputs controlled by a bit in each SIO channel's write register 5. In Asynchronous mode, a Low-to-High transition is delayed until the transmitter has sent all of its data.

TABLE 1. PIN DESCRIPTIONS (CONTINUED)

Symbol	Pin #	Function	Type	Description																																				
RXA0-1	142, 149	ASCI Receive Data	Inputs	ASCIs assemble serial receive data from these pins.																																				
RXCA-B	81, 95	SIO Receive Clocks	Inputs	The SIOs sample receive data on rising edges of these clocks. In ASYNC mode, software programs each SIO to divide its clocks by 1, 16, 32, or 64.																																				
RXDA-B	80, 96	SIO Receive Data	Inputs	SIOs assemble serial receive data from these pins.																																				
RXS	153	CSI/O Receive Data	Input	CSI/O assembles serial receive data from this pin.																																				
ST	43	Status	Output	With $\overline{\text{HALT}}$ and $\overline{\text{M1}}$, this pin indicates the current processor status: <table><tr><td>ST</td><td>HALT</td><td>M1</td><td>Status</td></tr><tr><td>L</td><td>L</td><td>H</td><td>HALT mode</td></tr><tr><td>L</td><td>L</td><td>H</td><td>DMA operation during HALT mode</td></tr><tr><td>L</td><td>H</td><td>L</td><td>First Op Code fetch</td></tr><tr><td>L</td><td>H</td><td>H</td><td>DMA operation, or 1st Op Code fetch with M1E=0</td></tr><tr><td>H</td><td>L</td><td>L</td><td>Does not occur</td></tr><tr><td>H</td><td>L</td><td>H</td><td>Sleep or System Stop</td></tr><tr><td>H</td><td>H</td><td>L</td><td>2nd/3rd Op Code fetch</td></tr><tr><td>H</td><td>H</td><td>H</td><td>None of the above, or 2nd/3rd Op Code fetch w/M1E=0</td></tr></table>	ST	HALT	M1	Status	L	L	H	HALT mode	L	L	H	DMA operation during HALT mode	L	H	L	First Op Code fetch	L	H	H	DMA operation, or 1st Op Code fetch with M1E=0	H	L	L	Does not occur	H	L	H	Sleep or System Stop	H	H	L	2nd/3rd Op Code fetch	H	H	H	None of the above, or 2nd/3rd Op Code fetch w/M1E=0
ST	HALT	M1	Status																																					
L	L	H	HALT mode																																					
L	L	H	DMA operation during HALT mode																																					
L	H	L	First Op Code fetch																																					
L	H	H	DMA operation, or 1st Op Code fetch with M1E=0																																					
H	L	L	Does not occur																																					
H	L	H	Sleep or System Stop																																					
H	H	L	2nd/3rd Op Code fetch																																					
H	H	H	None of the above, or 2nd/3rd Op Code fetch w/M1E=0																																					
SYNCA-B	79, 97	Synchronization	Input/Outputs	The function of these pins varies according to the operating mode of each SIO: Async: GP input, readable in RR0. External Sync: Low-active Sync detect input, delayed by 2 RxC rising edges Other Sync: Low-active Sync detect output																																				
TENDO-1	156-157	DMA Transfer End 0-1	Outputs, Active Low	These outputs become Low during the last Write cycle of each DMA buffer. External circuits can use this signal to determine when each buffer has been completed.																																				
TOUT	160	Timer Output	Output	Software can program PRT1 to toggle this pin, or force it High or Low, when it has decremented its counter to 0. The reload feature can then be used to generate arbitrary waveforms. The minimum pulse width is determined by the processor's response time.																																				
TXA0-1	141, 148	ASCI Transmit Data	Outputs	ASCIs output serial transmit data on these pins.																																				



TABLE 1. PIN DESCRIPTIONS (CONTINUED)

Symbol	Pin #	Function	Type	Description
TXCA-B	82, 94	SIO Transmit Clocks	Inputs	SIOs change transmit data on falling edges of these pins. In ASYNC mode, software can program an SIO to divide its clocks by 1, 16, 32, or 64.
TXDA-B	83, 93	SIO Transmit Data	Outputs	SIOs output serial transmit data on these pins.
TXS	152	CSI/O Transmit Data	Output	The CSI/O outputs serial transmit data on this pin.
V _{DD}	5, 25, 45, 65, 85, 105, 125, 145	Power Supply		These pins carry power to the device. They must be tied to the same voltage externally.
V _{SS}	15, 35, 55, 75, 115, 135, 155	Ground		These pins are the voltage reference for the device, and must be tied to the same voltage externally.
W/RDYA-B	78, 98	SIO Wait/Ready	Outputs	Software can program each SIO to drive these lines as a Wait signal (suitable for connection to the WAIT pin) or a DMA request signal (suitable for connection to DREQ0-1).
WAIT	36	Wait	Input	The Z80S188 samples this input during each bus cycle with an external memory or peripheral. If this line is Low, it extends the cycle by one PHI clock, until it samples this pin High, after which it concludes the cycle.
WDTOUT	56	Watch-Dog Timer Output	Open-drain Output, Active Low	The Watch-Dog Timer drives this pin Low if it is enabled and software has failed to reload its counter, so that it has counted down to 0. This pin can be connected to RESET or NMI, or used in some other way.
WR	38	Write Strobe	Output, Active Low, 3-State	WR Low indicates that the processor is writing data to a location in memory or I/O space. The Z80S188 does not drive this line while RESET or BUSACK is Low.
XTAL	57	Crystal	Input/Output	This pin provides the crystal oscillator connection. Leave this pin open if an external clock is used instead of a crystal. This pin does not carry a logic level. (See "DC Characteristics" on page 199.)
ZC/TO0-3	132-134, 136	CTC 0 Count/Timeout	Outputs, Active High	Each CTC channel drives its ZC/TO pin High for 1.5 PHI clocks, when it has decremented its counter to 0.
ZCL	91	ZDI Clock	Input	This pin is the clock for the ZiLOG Debugging Interface. This input includes a Schmitt trigger.



TABLE 1. PIN DESCRIPTIONS (CONTINUED)

Symbol	Pin #	Function	Type	Description
ZDA	90	ZDI Data	Input/ Output, Open Drain	Data for the ZiLOG Debugging Interface. This input includes a Schmitt trigger.

OPERATIONAL DESCRIPTION

This section describes, using text, tables, and figures, how the various parts of the Z80S188 operate. This description is presented from the processor “outward” to the peripherals. In the latter parts of this section, refer to the corresponding part of section 4, which describes the Z80S188’s I/O registers. Cross-reference links are included in both sections to aid such reference.

PROCESSOR DESCRIPTION

Like the original ZiLOG Z80, the Z80S188 is an 8-bit microprocessor that can also perform various 16-bit operations. In both data sizes, the processor includes an accumulator. A is the accumulator for 8-bit operations, and the HL register pair is the accumulator for 16-bit operations.

Processor Program Registers

In addition to A, there are six more 8-bit registers named B, C, D, E, H, and L, which can also be operated on as 16-bit register pairs BC, DE, and HL. The Flag register F completes the basic register bank.

Two of these basic register banks are included in all Z80 and Z180 processors. high-speed exchange between these banks can be used by a program internally, or one bank can be allocated to the mainline program and the other to interrupt service routines.

Finally, two Index registers IX and IY enable *base and displacement* addressing in memory. IX and IY are not included in the register banks on the Z80 and Z180; therefore, there is only one copy of each.

Memory Management Unit

Add a Memory Management Unit (MMU) that expands the addressing capability to 20 bits or 1 Megabyte to the 16-bit, 64-KB memory addressing capability of the Z80, Z180 processors other than the Z80S188. With the MMU, the 65K *logical* addressing space can be divided into one to three areas of programmable size and location in the 1 Megabyte *physical* memory space.

On the Z80S188, this MMU has been extended to 23 bits, or 8 MB.

I/O Space

A separate I/O space includes on-chip and off-chip peripheral devices. On the Z80, I/O space included 8-bit addresses and 256 bytes. As with memory space, all

Z180 processors feature an expanded I/O space with 16-bit addresses and 64 KB. The Z80S188 includes an extensive set of on-chip peripherals in I/O space, which can be augmented by external peripherals.

Processor Control Registers

In addition to the *data-oriented* registers described above, the Z80S188 processor includes several other control registers. Unlike the registers in I/O space that are described in section 4, these control registers have no addresses, but are used implicitly in certain processor operations.

Program Counter (PC). This 16-bit register tracks program execution by the processor, which automatically increments PC while fetching instructions. PC is stored on the stack when a CALL or RST instruction is executed, and when an interrupt or Trap occurs. PC is loaded with a new value when a JUMP, CALL, RST, or RET instruction is executed, and when an INTERRUPT, TRAP, or RESET instruction occurs. The PC then resets to 0000H.

Stack Pointer (SP). This 16-bit register is decremented by 2, and a 16-bit value is stored in memory at this updated address, when a PUSH, CALL, or RST instruction is executed, and when an interrupt or Trap occurs. A 16-bit value is fetched from memory at the address in SP, and SP is then incremented by 2, when a POP, RET, RETI, or RETN instruction is executed. SP can be stored in memory, loaded from memory or another register, or loaded with a constant/immediate value. Its value can be added to or subtracted from another register, and it can be incremented or decremented. Finally, the 16-bit value in memory, at which SP currently points, can be exchanged with the contents of a 16-bit register. The SP resets to 0000H.

Flags (F). The processor includes two sets of six Flag bits each, named Z (ZERO), CF (CARRY), S (SIGN), P/V (PARITY or OVERFLOW), HC (HALF-CARRY) and N (ADD/SUBTRACT). It automatically updates certain flags as part of executing certain instructions. Subsequent instructions can then use the flags, either as an operand (ADC, SBC, DAA), or to determine whether to perform a JUMP, CALL, or RET. The flags can be saved on the stack with a PUSH instruction, or restored from the stack with a POP. The two sets of Flag registers are paired with the two A accumulators; the current pair is toggled by the EX AF,AF/ instruction.

Interrupt High Address (I). The contents of this register are used as the 8 high-order address bits, when the processor fetches the address of an interrupt service routine from memory, because of an interrupt from the INT1 or INT2 pin, or from an on-chip peripheral. Thus, the I register points at a table of interrupt service routine addresses, that starts at a 256-boundary in the 64-KB logical address space. The I register resets to 00, and can be read or written by the dedicated instructions LD A,I and LD I,A.

R Counter (R). On the original Z80 this register contained the dynamic RAM refresh address, but on Z8018x family processors it contains a *count of executed Op Code fetch cycles*. R resets to 00, and can be read or written by the dedicated instructions LD A,R and LD R,A.



Illegal Instruction Traps

Like most processors, the defined instruction set for the 8018x family does not fully *cover* all possible sequences of binary values. The Op Code maps in the section “Op Code Map”, on page 191, include numerous blank cells. These cells represent Op Code sequences for which no operation is defined, and are commonly called *illegal instructions*.

When a Z80S188 or other 8018x processor fetches one of these sequences, it performs a TRAP operation as follows:

1. The TRAP bit in the Interrupt/Trap Control register is set to 1.
2. The UFO bit is cleared to 0 in the Interrupt/Trap Control register if the condition is detected while fetching the second byte of the instruction. If it detected the condition while fetching the third byte of the instruction, the UFO is set to 1.
3. The SP is decremented by 2 and stores the 16-bit logical address from PC, in memory at the new SP value. This address points to the last byte of the illegal Op Code sequence.
4. The PC clears and resumes execution at logical address 0000H.

Trap Handling. The code at logical address 0000H starts by storing the value of SP in memory, and then setting SP to an area of memory dedicated to its stack. This step is optional.

In all cases, the trap handling routine stores as many registers among AF, BC, DE, HL, IX, and IY as it uses in subsequent instructions, by pushing them onto the stack. A general-purpose routine stores all of these registers, and those in the alternate set as well, plus I and the state of the Interrupt Enable flag.

Next, the code must distinguish among four cases that can bring execution to location 0000: RESET, TRAP, an RST 0 instruction, or a program error like a JUMP to a null pointer. The code can detect a TRAP by reading the Interrupt/Trap Control register (ITC) and checking bit 7 (TRAP). If the value of bit 7 is 1, a TRAP occurred.

Next, the trap-handling code clears the TRAP bit by setting it to 0. The code then fetches the PC value stored on the stack, and examines bit 6 of the ITC (UFO). If UFO is 0, the PC value is decremented by 1 so that it points to the start of the instruction. If UFO is 1, the value is decremented by 2.

What the trap handling routine does next depends on the application and the stage of its development.

Extending the Instruction Set. Core software can use *illegal* instructions as extensions to the Z8018x instruction set. To accomplish this, the core software fetches and examines the *illegal* instruction, to determine if it is this type of extension. If so, software performs the extended operation that the instruction indicates, advances the stacked PC value over the instruction, restores the registers and performs a RET to the next instruction.

Error Message vs. Restart. Except for these extended instructions, software can either signal an illegal instruction and wait for someone to examine the situation and restart, or attempt to restart the application immediately. The former course is more common in the debugging/development stages of an application. The latter may be more appropriate in the production/deployment stage. In the latter case, software may log the event for future readout, to a storage medium or just in memory.

SECURITY FEATURES AND ROM OPTIONS

Two types of standard Z80S188s may be made available. On a *romless* part, on-chip RAM is accessible to code in any memory. On-chip ROM is disabled and can be emulated by a memory connected to the $\overline{\text{EV/ROMCS}}$ pin. On a *monitor* part, on-chip RAM is accessible to code in any memory, and on-chip ROM contains a standard debug monitor.

Customers submitting a custom ROM code for on-chip ROM can select among several ROM options that enhance the security of their products and applications. These options are described in the following sections.

Both standard parts have ROM and RAM Protection disabled and ZDI enabled.

ROM/ROMless

This option can be selected as *enable internal ROM* by any customer submitting a ROM code.

ROM Protect

At the initial Op Code fetch of each instruction, the Z80S188 determines whether the instruction is in on-chip ROM. If this ROM option is selected, the transition from executing in other memory, to executing in on-chip ROM can occur only if the first on-chip ROM address is 0000, 0008, 0010, 0018, 0020, 0028, 0030, 0038, or 0066H. If entry to on-chip ROM is attempted at any other address, an exception occurs, the PC is stacked, and the ROMEE bit in the Interrupt/Trap Control register is set. Execution proceeds from location 0000 in a manner similar to an illegal instruction trap.

NOTE: When this option is selected, interrupt service routines cannot start in on-chip ROM.

RAM Protect

If this ROM option is selected:

1. On-chip RAM can only be read and written by instructions in on-chip ROM, including: not by a DMA channel, nor by an off-chip master in $\overline{\text{BUSACK}}$ state nor EV state. No exception occurs if code located elsewhere attempts to access on-chip RAM, but no reading or writing occurs. During production device testing, the functionality of on-chip RAM must be verified by a routine in the user's ROM code. Details of how the tester calls this routine will be specified at a later date.



2. On-chip ROM can only be read by instructions in on-chip ROM, unless a special pad is contacted by a wafer-level tester probe.
3. Data is blocked from appearing on the D7-0 pins for cycles in in-chip ROM or on-chip RAM.
4. Execution in on-chip RAM is not allowed. If an Op Code fetch is attempted in on-chip RAM, an exception occurs, the PC is stacked, and the RAMEE bit in the Interrupt/Trap Control register is set. Execution proceeds from location 0000 in a manner similar to an illegal instruction trap.

ZDI Enable

For secure applications, disable this option. The ZiLOG Debug Interface (ZDI) is a built-in interface that can be used with ZiLOG's Developers' Studio and potentially with third-party products, for low-cost emulation facilities. See section "ZiLOG Debug Interface", on page 225, for a description of how to include a connector on any application board to enable debugging via the ZDI.

INTERRUPTS

ZiLOG Z80 and Z180 processors have a rich legacy of sophisticated interrupt capabilities. Because it includes both Z80- and Z80180-compatible peripherals, the Z80S188 includes aspects of both families' interrupt characteristics.

Interrupt Resources in the Z80S188

IEF1 and IEF2. These bits are internal to the processor and can only be affected and manipulated by certain specific events:

- A Reset clears both IEF1 and IEF2.
- An EI instruction sets both IEF1 and IEF2.
- A DI instruction clears both IEF1 and IEF2.
- An NMI sequence copies IEF1 to IEF2, then clears IEF1.
- A maskable interrupt clears both IEF1 and IEF2.
- An LD A,I or LD A,R instruction copies IEF2 to the P/V Flag.
- An RETN instruction copies IEF2 to IEF1.

When IEF1 is 1, $\overline{\text{RESET}}$ and $\overline{\text{BUSREQ}}$ are both High, and no falling edge has occurred on $\overline{\text{NMI}}$, the Z80S188 checks for maskable interrupt requests from external pins and on-chip peripherals, as it completes each instruction, or each instruction iteration for HALT, the block I/O instructions, block move instructions, and block scan instructions.

The I Register. The Z80S188 uses the contents of this register as A15-8 of the logical address for fetching interrupt service routine addresses from memory, in response to interrupt requests on $\overline{\text{INT0}}$, $\overline{\text{INT1}}$, and $\overline{\text{INT2}}$, and from internal peripherals.

“Interrupt Registers”, on page 114, describes the other registers associated with interrupts:

The IL Register. The Z80S188 uses the three Most Significant (MS) bits of this register as A7-5 of the logical address for fetching interrupt service routine addresses from memory, in response to interrupt requests on $\overline{\text{INT1}}$ and $\overline{\text{INT2}}$, and from the ASCIs, PRTs, DMAs, and CSI/O.

The Interrupt/Trap Control Register (ITC). The three Least Significant (LS) bits of the ITC are individual ENABLE bits for the $\overline{\text{INT2}}$, $\overline{\text{INT1}}$, and $\overline{\text{INT0}}$ pins. They reset to 001 respectively, so that requests on $\overline{\text{INT0}}$ can be enabled by an EI instruction after Reset, as is the case on the Z80. Other bits in this register identify whether a TRAP or security exception has occurred.

Non-Maskable Interrupt (NMI)

The Z80S188 latches falling edges on the $\overline{\text{NMI}}$ pin. A falling edge clears the DME bit in the DMA Status register (DSTAT), thereby disabling the on-chip DMA channels. Only a Low on $\overline{\text{RESET}}$ or on $\overline{\text{BUSREQ}}$ take precedence over $\overline{\text{NMI}}$. Unless Reset or BUSREQ is Low, the Z80S188 checks for a falling edge on $\overline{\text{NMI}}$ as it completes each instruction (each instruction iteration of HALT, the block I/O instructions, block move instructions, and block scan instructions), and performs an NMI sequence if a falling edge has occurred.

An NMI sequence includes four steps:

1. The Z80S188 copies the state of the IEF1 bit to IEF2.
2. The IEF1 is cleared to prevent maskable interrupts.
3. The SP decrements by 2, and stores the logical address in the PC in memory at the new address in SP. Typically, this new address is the address of the instruction the processor would have executed next, if no interrupt had occurred. If the processor was stopped by HALT or SLP, it is the address of the next instruction. For an incomplete block transfer, block scan, or block I/O instruction, it is the address of the instruction.
4. The value 0066H loads into PC, and execution resumes from that logical address.

NMI Handling. NMI routines fall into two categories, based on whether or not the external hardware that drives NMI can produce another falling edge on the pin, before the routine completes its execution and returns to the interrupted process. Debug monitors, that may display the state of the interrupt process, inherently fall into the *repeated edge* category.

Single Edge Guaranteed. An NMI routine in this category is much like any other interrupt service routine. The initial option is provided of storing the contents of the SP in memory and loading the SP with the address of a memory area that is dedicated for its stack. In any case, it store as many of the registers as it uses during its execution.



Repeated Edge Possible. An NMI routine in this category starts with PUSH AF, then loads A from a dedicated location in memory that indicates whether the interrupted process was, in fact, the NMI routine. If this location indicates that it was, the routine immediately performs a POP AF and then an RETN instruction to return to its former execution.

If the *in* NMI location is clear, software sets it to 1. At this point, if the NMI routine takes either of the following actions:

- Performs a DI instruction in a *save the registers* routine that it shares with other means of entry, or
- Displays the I register or the interrupt-enable state of the interrupted process, and enables a user/programmer to change these (a debug monitor),

then an LD A,I instruction and another PUSH AF instruction is performed. This action stores the I register at the address in SP plus one, and the interrupt enabled state (IEF2) in the P/V flag and bit 2 of the address in SP. If the NMI routine uses a common *save the registers* subroutine that it shares with other entry points, the save subroutine can perform a DI instruction to prevent its being interrupted by maskable interrupts.

The NMI routine can now store SP in a dedicated location in memory, and load SP with the address of its dedicated stack area.

In any case, the NMI routine perform a PUSH instruction to as many other registers as it uses. A debug monitor typically performs PUSH instructions to all registers in both banks, so that they display.

Re-enabling the DMA channels. Fairly early in an NMI service routine in an application using the DMA channels, software reads the DSTAT register and re-enables any DMA operation that was in progress, as described in section “NMI and DME”, on page 61.

Exiting the NMI routine. Upon completion of its processing, an NMI routine restores the registers. If the routine used its own stack area, it then restores the SP value of the interrupt process. If the routine set an *in* NMI memory location on the way in, this location is cleared.

NMI routines that did not save the I register and IEF2 state at the start finish with an POP AF and a RETN instruction, which copies the state of IEF2 back into IEF1, restoring the interrupt enable state of the interrupted process.

NMI routines that saved I and IEF2 at the start, finish with a POP AF instruction for the saved I register and IEF2 bit, a LD I,A, followed by a JP V to a POP AF, EI, RET sequence. If the JP does not occur, it is followed with POP AF, DI instructions if one was not performed at the front end, and RET.

INT0

The on-chip PIOs, SIO, and CTC logically OR their interrupt requests with external requests on the INT0 pin. The Z80S188 drives INT0 Low in an open-drain fashion whenever any of these peripherals requests an interrupt.

INT0 Modes. The Z80S188 can handle interrupts requested by the on-chip PIOs, SIO, CTC, or an external device on the INT0 pin, in either of two ways called mode 1 or 2. The processor itself has another mode called mode 0, but this mode cannot be used with the PIOs, SIO, or CTC. For completeness, this section describes all three modes.

The special instructions IM 0, IM 1, and IM 2 select among these three modes. RESET selects mode 0.

ZiLOG daisy-chainable peripherals, including the on-chip PIOs, SIO, CTC, are designed for use in mode 2, although they can also be used in mode 1.

Interrupt Acknowledge Daisy Chaining. Z80 peripherals, including the on-chip PIOs, SIO, CTC, use daisy chain signalling to decide amongst themselves, during an INT0 interrupt acknowledge cycle, which of them responds to the cycle by driving an 8-bit value onto the D7-0 data bus. Some other ZiLOG peripherals, including the SCC and USC serial controller families, can also be used in such daisy chains.

Such a daisy chain is illustrated in Figure 3. It implements a relatively fixed prioritization of interrupts among the devices in the chain. The highest-priority device among those requesting on INT0 always has its IEI pin High. Its IEO pin is connected to the IEI pin of the next-lower-priority INT0 device, and so on through all of the INT0 devices. The IEO output of the lowest-priority INT0 device is not used.

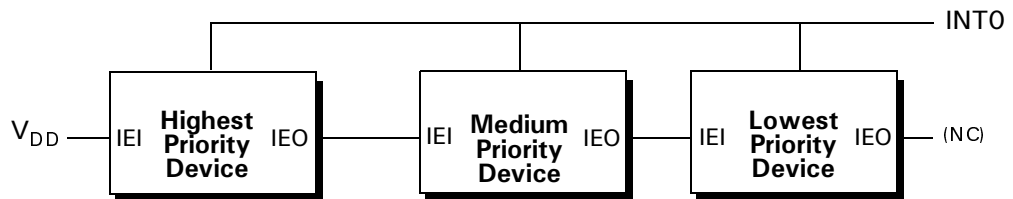


FIGURE 3. AN INTERRUPT ACKNOWLEDGE DAISY CHAIN

The Z80S188 has one IEI pin and one IEO pin, and the on-chip PIOs, SIO, and CTC are always consecutive in the daisy chain, although their relative priority can be programmed using bits 3-0 of the Interrupt Priority Register, which is illustrated on page 117.

**TABLE 2. RELATIVE INTERRUPT PRIORITY OF PIOs, SIO, AND CTC**

INTPR 3-0	Highest Priority	2nd Highest	2nd Lowest	Lowest
0000	CTC	SIO	PIO AB	PIO CD
0001	SIO	CTC	PIO AB	PIO CD
0010	CTC	PIO AB	PIO CD	SIO
0011	PIO AB	PIO CD	SIO	CTC
0100	PIO AB	PIO CD	CTC	SIO
0101	SIO	PIO AB	PIO CD	CTC
011x	Reserved			
1000	CTC	PIO AB	SIO	PIO CD
1001	SIO	PIO AB	CTC	PIO CD
1010	PIO AB	CTC	PIO CD	SIO
1011	PIO AB	SIO	PIO CD	CTC
1100	PIO AB	CTC	SIO	PIO CD
1101	PIO AB	SIO	CTC	PIO CD
111x	Reserved			

If any external peripherals have higher priority than the on-chip PIOs, SIO, and CTC, the IEO pin of the lowest-priority among such device(s) is connected or routed to the Z80S188's IEI pin. If any external peripherals have lower priority than the on-chip PIOs, SIO, and CTC, the Z80S188's IEO pin is connected or routed to the IEI pin of the highest-priority among such devices.

$\overline{\text{INT0}}$ Processor Response. The Z80S188 performs an INT0 interrupt sequence at the end of an instruction (each instruction iteration for HALT, the block I/O instructions, block move instructions, and block scan instructions), if all of the following are true:

- $\overline{\text{INT0}}$ is Low,
- Bit 0 of the Interrupt/Trap Control register is 1 to enable $\overline{\text{INT0}}$,
- The IEF1 bit is 1, to enable interrupts in general,
- RESET and BUSREQ are both High, and
- A negative edge on NMI has not been detected.

When all of these conditions occur simultaneously, the Z80S188 responds as follows:

1. The IEF1 and IEF2 flags clear to prevent further interrupts.
2. M1 goes Low. This signal makes all ZiLOG daisy-chainable peripherals, including the on-chip PIOs, SIO, and CTC, halt whether or not they are requesting an interrupt. This action allows the interrupt acknowledge daisy chain to settle and select which Z80 peripheral responds to the cycle.
3. Several clock cycles pass to allow the daisy chain to settle.

4. IORQ goes Low. Simultaneous Lows on M1 and IORQ indicate an INT0 interrupt acknowledge cycle. In response to this condition, the highest-priority ZiLOG peripheral that requests an interrupt places an 8-bit value on the D7-0 data bus.
5. WAIT is sampled, and the Z80S188 waits until WAIT is High.
6. The cycle is terminated when M1 becomes High, then IORQ becomes High.

While all INT0 acknowledge cycles follow this general pattern, they differ as to what, if anything, the processor does with the data on D7-0, and what it does after the acknowledge cycle. These actions depend on the most recently executed IM instruction, if any, as described in the next three sections.

INT0 Mode 0. If no IM instruction was executed since Reset, or if the most recently executed IM instruction was IM 0, the Z80S188 completes an INT0 sequence as illustrated in Figure 4:

7. The Z80S188 samples D7-0 and interprets the value as an instruction Op Code. In this mode, the vector registers of all ZiLOG daisy-chainable peripherals must be programmed to provide one of the Reset Op Codes C7, CF, D7, DF, E7, EF, F7, or FFH

NOTE: The Z80S188 does not automatically stack the contents of the program counter during an INT0 Mode 0 interrupt sequence. The only other Op Code that a peripheral can return, assuming the interrupted process is to be restarted is a CALL instruction DCH. Intel 808x-family interrupt controllers can return a 3-byte CALL instruction, but ZiLOG peripherals cannot.

8. If the Op Code is CALL, the processor fetches two more bytes to complete the instruction.
9. Given that the Op Code is CALL or RST, the processor decrements SP by 2, and stores the contents of PC in memory at the new address in SP. Typically, this is the address of the instruction the processor would have executed next, if no interrupt had occurred. If the processor was stopped by HALT or SLP, this address is the address of the next instruction. For an incomplete block transfer, block scan, or block I/O instruction, the address is the address of the instruction.
10. If the Op Code was RST, the processor resumes execution at logical address 0000, 0008, . . . , or 0038H. If the Op Code was CALL, it resumes at the logical address fetched in step 8.

In mode 0, each peripheral connected to INT0 must have a register, the contents of which is returned on D7-0 when it detects M1 and IORQ Low, an interrupt is requested, and the IEI pin is High. Software programs each such register with one of the RST Op Codes C7, CF, D7, . . . , FFH.

NOTE: The PIO and CTC included in the Z80S188 can only return vectors having bit 0 equal to 0. Since all of the RST Op Codes have bit 0 equal to 1, if interrupts are needed from the PIO or CTC, mode 0 cannot be used.

If a peripheral has a feature whereby it can replace the low-order bits of this value with a code reflecting its status, this feature must be turned off for mode 0 operation.

If the number of devices that can interrupt on INT0 is reasonable, each device can have its own RST instruction, which improves interrupt response time by eliminating the need for the interrupt service routine to poll multiple devices.

If multiple devices are required to share a RST instruction, the interrupt service routine polls these devices in the same priority order that they are arranged on the IEI–IEO daisy chain. This action is necessary because a ZiLOG peripheral sets the IUS bit when it detects M1 and IORQ Low, and it is requesting an interrupt, and the IEI pin is High. To insure proper operation of the daisy chain in the future, the polling process must lead to servicing the device that performed this process, and then clearing the IUS bit either explicitly, or for a Z80 peripheral, by concluding the ISR with a RETI instruction.

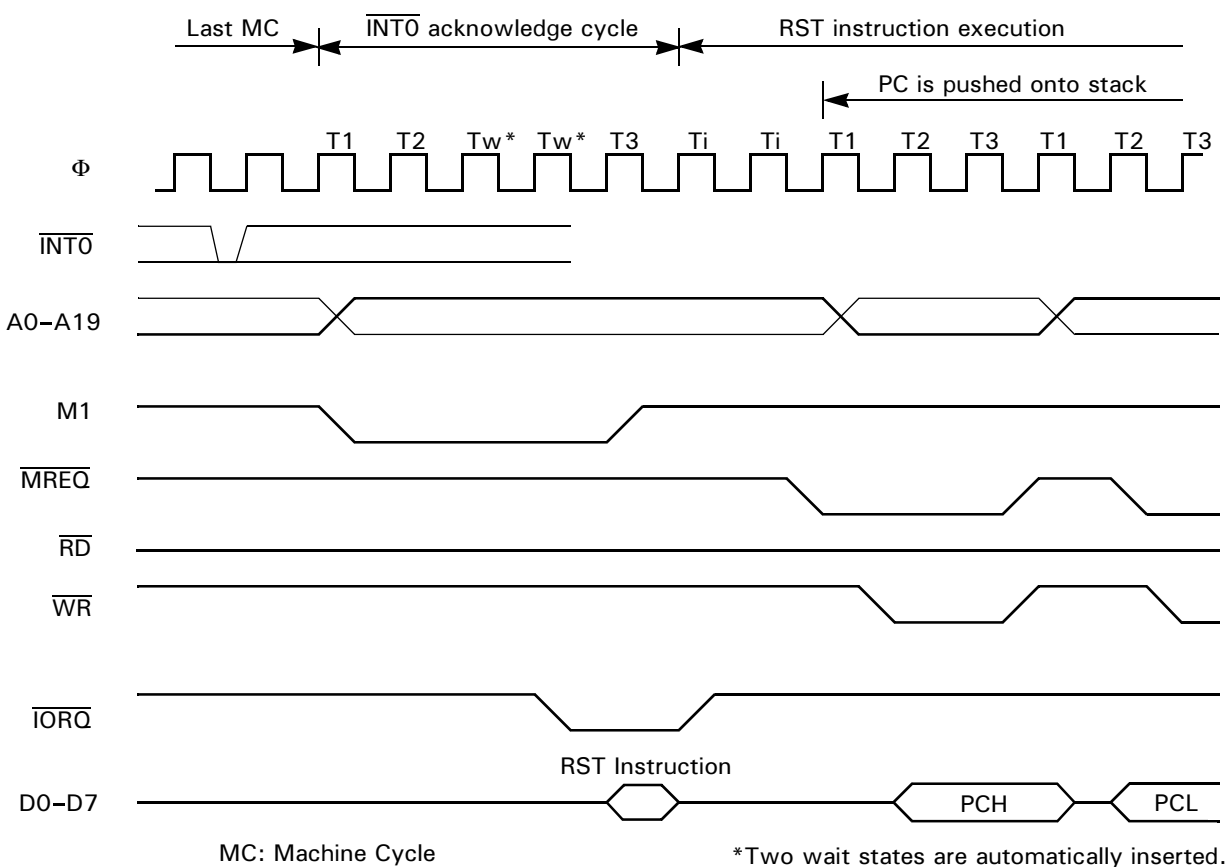


FIGURE 4. $\overline{\text{INT0}}$ MODE 0 TIMING

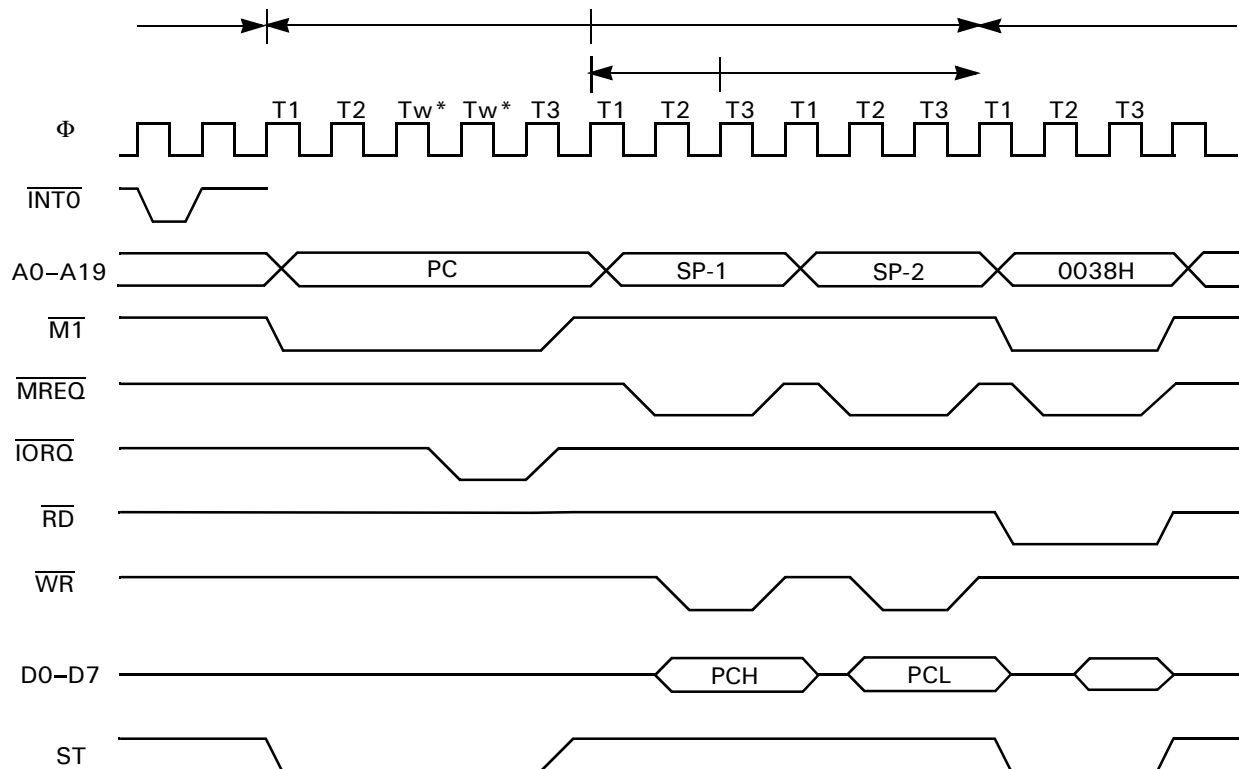
$\overline{\text{INT0}}$ Mode 1. The following steps occur after step 6 of the section “ $\overline{\text{INT0}}$ Processor Response”, on page 19. If the most recently executed IM instruction was IM 1, the Z80S188 completes an INT0 sequence as illustrated in Figure 5:

1. The data on D7–0 is ignored. Actually, the device proceeds as in mode 0, but captured FFH which is RST 38.

2. SP is decremented by 2, and stores the contents of PC in memory at the new logical address in SP. Typically, this address is the address of the instruction the processor would have executed next, if no interrupt had occurred. If the processor was stopped by HALT or SLP, this address is the address of the next instruction. For an incomplete block transfer, block scan, or block I/O instruction, this address is the address of the instruction.
3. 0038H loads into PC, and resumes instruction execution from that logical address.

In mode 1, the interrupt service routine is required to poll all of the devices connected to INT0, to detect which one generated the interrupt. If any ZiLOG peripherals, including the on-chip PIOs, SIO, or CTC, can request an interrupt, this polling must be done in the same priority order that the devices are arranged on the IEI–IEO daisy chain. This poll is required because a ZiLOG peripheral sets the IUS bit when it detects M1 and IORQ Low, an interrupt is requested, and the IEI pin is High, regardless of the processor’s IM status.

To insure proper operation of the daisy chain in the future, the polling process must lead to servicing the device that set the IUS bit, and clearing that bit either explicitly, or for a Z80 peripheral by concluding the ISR, with a RETI instruction. Probably the best way to insure this is by actually polling the IUS bits, for devices that allow them to be read.



*Two wait states are automatically inserted.

FIGURE 5. INTO MODE 1 TIMING



INT0 Mode 2. The following steps occur after step 6 of the section “INT0 Processor Response”, on page 19. If the most recently executed IM instruction was IM 2, the Z80S188 completes an INT0 sequence as illustrated in Figure 6:

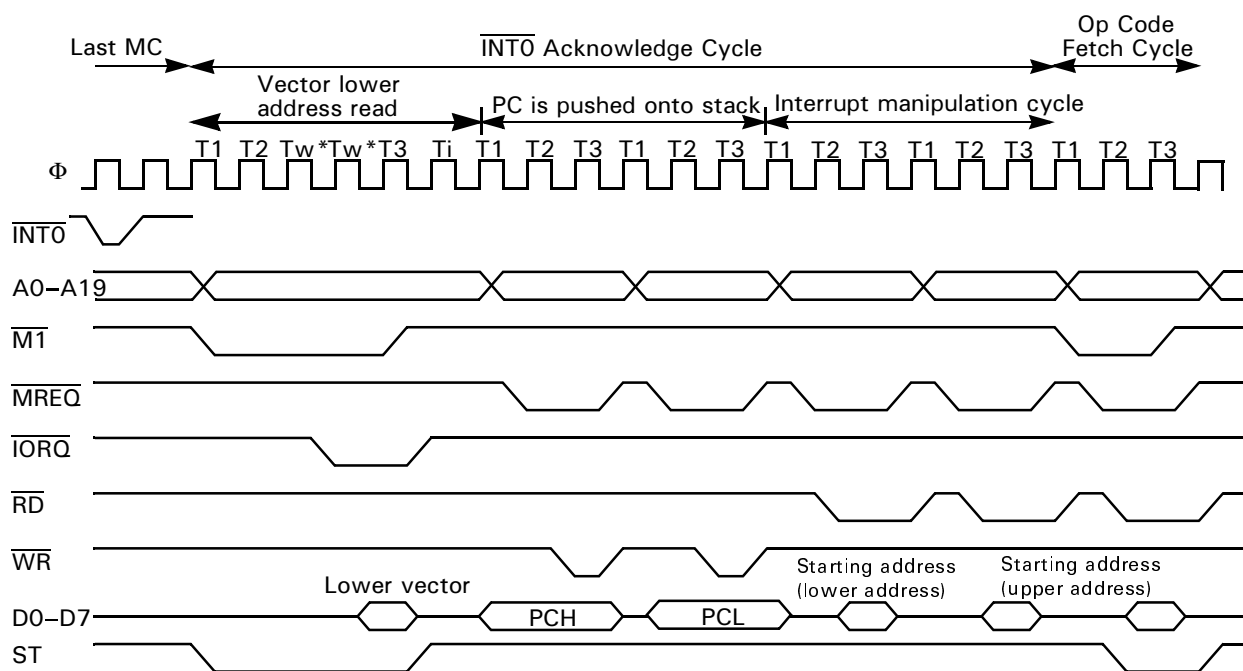
1. The IEF1 and IEF2 flags clear to prevent further interrupts.
2. M1 goes Low. This signal makes all ZiLOG daisy-chainable peripherals, including the on-chip PIOs, SIO, and CTC, halt whether or not they are requesting an interrupt. This action allows the interrupt acknowledge daisy chain to settle and select which Z80 peripheral responds to the cycle.
3. Several clock cycles pass to allow the daisy chain to settle.
4. IORQ goes Low. Simultaneous Lows on M1 and IORQ indicate an INT0 interrupt acknowledge cycle. In response to this condition, the highest-priority ZiLOG peripheral that requests an interrupt places an 8-bit value on the D7-0 data bus.
5. WAIT is sampled, and the Z80S188 waits until WAIT is High.
6. The cycle is terminated when M1 becomes High, then IORQ becomes High.
7. The IEF1 and IEF2 flags clear to prevent further interrupts.
8. M1 goes Low. This signal makes all ZiLOG daisy-chainable peripherals, including the on-chip PIOs, SIO, and CTC, halt whether or not they are requesting an interrupt. This action allows the interrupt acknowledge daisy chain to settle and select which Z80 peripheral responds to the cycle.
9. Several clock cycles pass to allow the daisy chain to settle.
10. IORQ goes Low. Simultaneous Lows on M1 and IORQ indicate an INT0 interrupt acknowledge cycle. In response to this condition, the highest-priority ZiLOG peripheral that requests an interrupt places an 8-bit value on the D7-0 data bus.
11. WAIT is sampled, and the Z80S188 waits until WAIT is High.
12. The cycle is terminated when M1 becomes High, then IORQ becomes High.
13. The data is captured from D7–0. D0 of this byte must be Low/0 for correct operation.
14. SP decrements by 2, and stores the contents of PC in memory at the new logical address in SP. Typically, this address is the address of the instruction the processor would have executed next, if no interrupt had occurred. If the processor was stopped by HALT or SLP, this address is the address of the next instruction. For an incomplete block transfer, block scan, or block I/O instruction, this address is the address of the instruction.
15. The contents of the I register are placed on A15–8, and the value captured in step 7 on A7–0, and fetches the LS byte of an interrupt service routine address from memory at that address.

16. A0 goes High/1, and fetches the MS byte of the interrupt service routine address from memory at that address.

17. Execution resumes at the logical address fetched in steps 9-10.

In mode 2, each peripheral connected to INT0 requires an Interrupt Vector Register, the contents of which is returned when M1 and IORQ Low are detected, an interrupt is requested, and the IEI pin is High. Software can program any of these registers with any even binary value.

If a peripheral has a feature whereby it can replace the low-order bits of this value with a code reflecting its status, this feature can be enabled in mode 2, in which case the peripheral *occupies more than one slot* in the interrupt vector table. Such *status affects vector* or *vector includes status* features can improve interrupt response time, by reducing the amount of status-polling that the interrupt service routine must do, to identify the exact cause of the interrupt.



*Two wait states are automatically inserted.

Interrupt Handling. Any interrupt service routine has the initial option of saving the contents of SP in memory, and loading SP with the address of a memory area that is dedicated to its stack. Most interrupt service routines do not function in this manner.

If the application includes a mechanism for allowing nested interrupts, the ISR can begin as specified by that mechanism, leading to an IE instruction that allows the ISR to be interrupted by other interrupts. Most Z80S188 applications do not function in this manner.

The ISR reads status registers from each of the devices that can request the interrupt, to identify the exact causes of the interrupt. The ISR then processes this device(s) according to their design, and the application design.

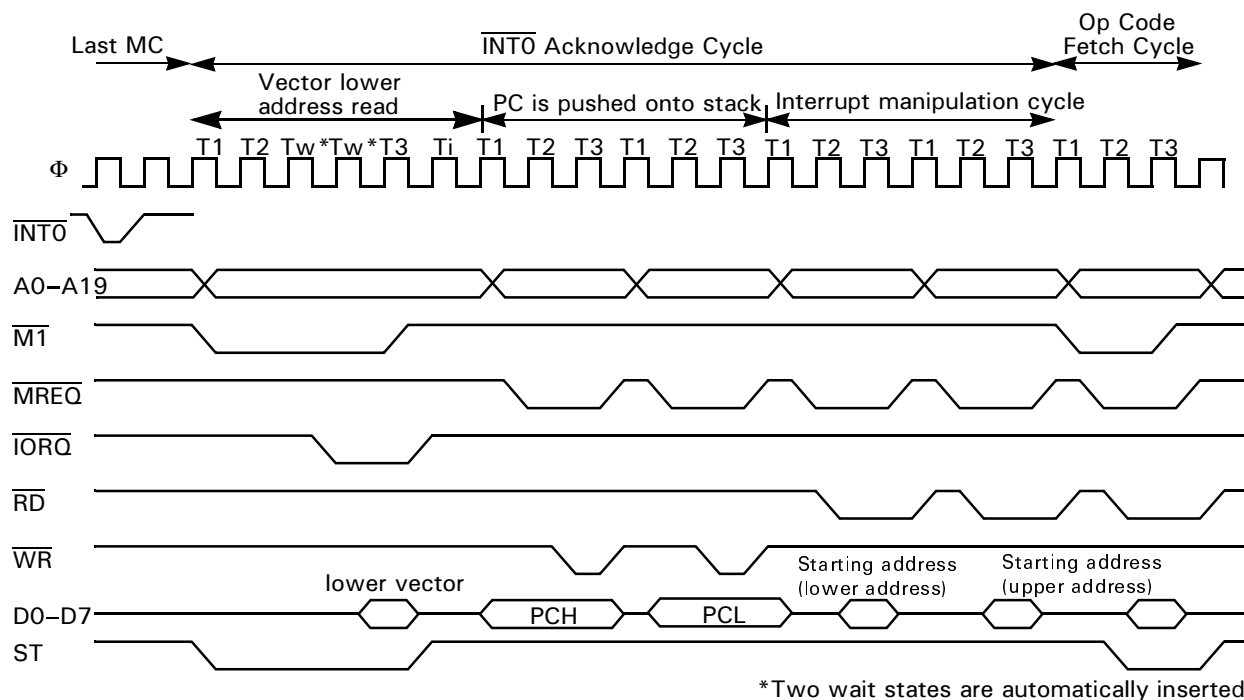


FIGURE 6. $\overline{\text{INT0}}$ MODE 2 TIMING

Many ISRs read data from the interrupting devices, or write data to them. In addition or alternatively, the ISRs may write commands to, or registers in the device, to modify its mode, status, or operation in the future.

When all of the work of the ISR is complete, if nested interrupts are allowed, the ISR terminates as specified by the nesting mechanism. If not, an ISR concludes with an EI instruction followed by RET or RETI.

Interrupt service routines for Z80 peripherals, including the on-chip PIOs, SIO, or CTC, terminate with a RETI instruction. As described in the next section, Z80 peripherals monitor the bus for these instructions, and clears the internal Interrupt Under Service (IUS) status if one is detected. The IEI status indicates the ISRs are the highest-priority device currently under service.

ISRs for non-Z80 peripherals must clear the device's IUS bit explicitly. These devices terminate with RET rather than RETI for two reasons:

- If nested interrupts are allowed, and the interrupted process was an ISR for a Z80 peripheral, RETI clears that device's IUS bit, inappropriately.
- RET is both shorter and faster than RETI, and features the same function as for a non-Z80 peripheral.

RETI Instructions

An interrupt service routine for a Z80 peripheral, including the on-chip PIOs, SIO, and CTC, terminates with an RETI instruction. To the processor, RETI features the same functionality as RET, but Z80 peripherals monitor the D7–0 lines, M1, and RD, and detect when the processor is fetching an RETI.

If a Z80 peripheral detects an RETI, has its INTERRUPT UNDER SERVICE (IUS) bit set, and its IEI pin is High, the interrupt service routine for this peripheral is terminating. The peripheral clears the IUS bit. This action allows subsequent interrupts from this device, or from lower-priority device on the daisy chain.

The original Z80 processor took four clock cycles to fetch an Op Code, and some Z80 peripherals built this timing into their logic for detecting RETI instructions. If the Z80S188 is fetching instructions from a zero-wait-state memory, it fetches Op Codes using three-clock cycles, which can interfere with RETI detection by external Z80 peripherals.

The Operating Mode Control Register, described on page 113, includes several bits that affect the Z80S188's compatibility with external Z80 peripherals.

Bit 7 (M1E). If this bit is 1, as it is after a Reset, the Z80S188 drives M1 Low when it fetches instruction Op Codes, as well as during INT0 and NMI acknowledge cycles. Use this setting if the only Z80 peripherals in the system are the on-chip PIOs, SIOs, and CTCs, and/or if all instructions are fetched from memory with one or more wait states.

NOTES:

1. The on-chip PIOs, SIO, and CTC can detect RETI instructions that are fetched using three-clock bus cycles.
2. Older members of the Z8018x family refetched an RETI instruction in this mode, which corrupt the daisy chain by clearing two IUS bits. The Z80S188, like other recent members of the Z8018x family, does not function in this manner.

If software writes a 0 to M1E, the Z80S188 does not drive M1 Low while fetching instruction Op Codes. If an RETI instruction is detected, the device refetches the RETI instruction, driving M1 Low and using at least four clocks per bus cycle. Use this setting if there are external Z80 peripherals *and* zero-wait-state instruction memory in the system. Figure 7 illustrates operation of RETI when M1E is 0.

Bit 6 (M1TE). If software writes a 0 to this bit, the Z80S188 drives M1 Low as it fetches the next Op Code, regardless of the state of M1E. This capability is needed after enabling interrupts on a PIO (external or onchip) with M1E 0, because the PIO does not actually enable the interrupt request until it detects M1 Low.

M1TE resets to 1, always reads as 1, and is cleared to 1 after the next Op Code is fetched.

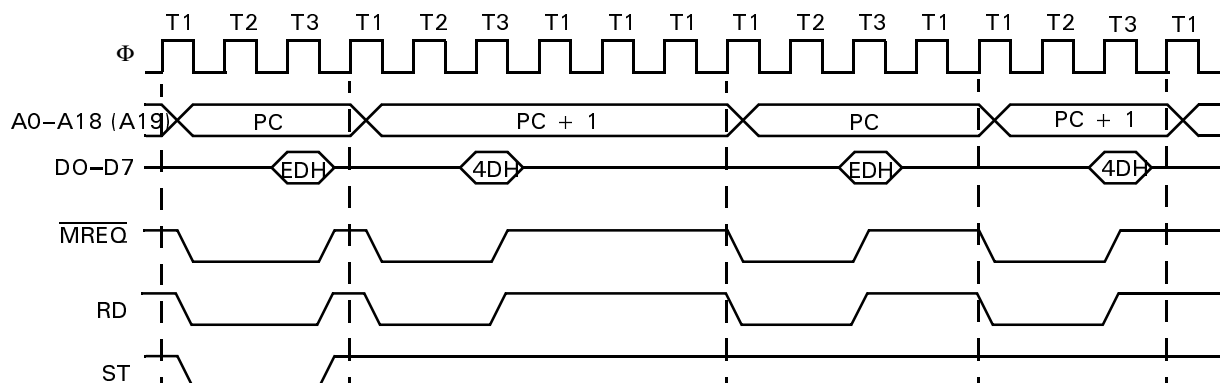


FIGURE 7. RETI INSTRUCTION WITH M1E = 0

INT1 and INT2

The Z80S188 performs an INT1 or INT2 interrupt sequence at the end of an instruction (each instruction iteration for HALT, the block I/O instructions, block move instructions, and block scan instructions), if all of the following are true:

- INT1 or INT2 is Low,
- Bit 2 or 1 of the Interrupt/Trap Control register (ITC) is 1 to enable this pin (if both pins are enabled and Low, INT1 takes precedence over INT2)
- The IEF1 bit is 1, to enable interrupts in general
- INT0 is High or bit 0 of the ITC is 0,
- RESET and BUSREQ are both High, and
- A negative edge on NMI has not been detected

When all of these conditions occur simultaneously, the Z80S188 responds as follows:

1. The IEF1 and IEF2 pins are cleared to prevent further interrupts.
2. SP decrements by 2, and stores the contents of PC in memory at the new address in SP. Typically, this address is the address of the instruction the processor would have executed next, if no interrupt had occurred. If the processor was stopped by HALT or SLP, this address is the address of the next instruction. For an incomplete block transfer, block scan, or block I/O instruction, this address is the address of the instruction.
3. A logical memory address is formed using the contents of the I register as A15–8, the three LS bits of the IL register as A7–5, and 0 as A4–0 for INT1 or 2 in A4–0 for INT2.
4. A 16-bit logical address is fetched from memory at that logical address, loaded into PC, and resumes instruction execution from there.

INT1-2 Handling. All of the considerations noted in the “Interrupt Handling”, on page 24 for INT0, also apply to ISRs for INT1 and INT2. If there is more than one device that can drive INT1 or INT2 Low, the interrupt service routine for that pin must poll all the devices connected to that pin, to determine which of them caused the interrupt.

ASCI, PRT, DMA, CSI/O, and ZDI Interrupts

The Z80S188 performs an interrupt sequence for one of these devices, at the end of an instruction (each instruction iteration for HALT, the block I/O instructions, block move instructions, and block scan instructions), if all of the following are true:

- An interrupting condition in the device has occurred
- That condition is interrupt-enabled in the device’s registers
- The IEF1 bit is 1, to enable interrupts in general
- No higher-priority internal device is requesting an interrupt (see Table 3 for the relative priorities of the devices)
- Neither INT1 nor INT2 is enabled and Low
- INT0 is High or bit 0 of the ITC is 0
- RESET and BUSREQ are both High
- A negative edge on NMI has not been detected

When all of these conditions occur simultaneously, the Z80S188 responds as follows:

1. The IEF1 and IEF2 flags are cleared to prevent further interrupts.
2. SP decrements by 2, and stores the contents of PC in memory at the new address in SP. This address is the address of the instruction the processor would have executed next, if no interrupt had occurred. If the processor was stopped by HALT or SLP, this address is the address of the next instruction. For an incomplete block transfer, block scan, or block I/O instruction, this address is the address of the instruction.
3. A logical memory address is formed using the contents of the I register as A15–8, the three LS bits of the IL register as A7–5, and the value corresponding to the interrupting device (see Table 3) as A4–0.
4. A 16-bit logical address is fetched from memory at that logical address, loaded into PC, and resumes instruction execution from there.

On-Chip Interrupt Handling. The only difference between handling an ASCI, PRT, DMA, CSI/O, or ZDI interrupt, and the considerations noted in “Interrupt Handling”, on page 24 for INT0, are that the ISR for an on-chip device never needs to differentiate among several devices connected to an INT pin.

**TABLE 3. INTERRUPT OFFSETS AND PRIORITIES**

Device	Priority	A4-0 Offset
INT1 pin	highest	0
INT2 pin		2
PRT0		4
PRT1		6
DMA0		8
DMA1		10 = 0AH
CSI/O		12 = 0CH
ASCI0		14 = 0EH
ASCI1		16 = 10H
ZiLOG Debug Interface (ZDI)	lowest	18 = 12H

NOTE: devices are ordered identically with respect to interrupt priority and offset value

MEMORY (ROM AND RAM)

The Z80S188 includes a 64-KB logical memory space in which software operates, and an 8-MB physical memory address space in which on-chip and external memory reside. The Memory Management Unit (MMU) translates 16-bit logical addresses to 23-bit physical addresses dynamically, as part of each memory access.

Memory Structure

On the Z80S188, memory is divided into five categories:

- 4-KB of on-chip ROM, or an external EPROM connected to $\overline{\text{ROMCS}}$
- 1-KB of on-chip RAM
- External memory connected to $\overline{\text{MEMCS0}}$,
- External memory connected to $\overline{\text{MEMCS1}}$
- External memory using off-chip decoding

Table 29 describes the On-chip Memory Control Register (OCMCR), which controls whether on-chip RAM, and either on-chip ROM on a masked part, or the $\overline{\text{ROMCS}}$ output on an unmasked part, are enabled.

If on-chip ROM (or the $\overline{\text{ROMCS}}$ output) is enabled, physical addresses 000000–000FFFFH become Low. If on-chip RAM is enabled, it occupies addresses 00FC00–0FFFFFH or 7FFC00–7FFFFFFH, depending on bit 6 of the OCMCR.

Reset enables on-chip RAM, and on-chip ROM or the $\overline{\text{ROMCS}}$ output at 00FC00–00FFFFFH.

Tables 42 through 45 describe the Memory Chip Select High and Low Registers for the $\overline{\text{MEMCS0}}$ and $\overline{\text{MEMCS1}}$ pins. If the Size/Enable field of a pin's Low Register are 0000H, the pin remains High, and any memory connected to this pin

is disabled. For $\overline{\text{MEMCS0}}$, this field resets to 1001H and the comparison bits reset to all 0s, so that $\overline{\text{MEMCS0}}$ is Low for addresses 001000–007FFFH. For $\overline{\text{MEMCS1}}$, the Size/Enable field resets to 0001H, so that $\overline{\text{MEMCS1}}$ is Low for addresses 008000–00FB00H and 010000–7FFFFFH.

For each pin, software can select any number of high-order address lines, from A22 through A13, to be compared for equality against a programmed value. Each pin can become Low for any block of contiguous addresses between 8 KB and 4 MB in length, starting at a multiple of that length. Alternatively, software can select the entire 8-MB address range for a pin.

If the ranges for $\overline{\text{MEMCS0}}$ and $\overline{\text{MEMCS1}}$ overlap, $\overline{\text{MEMCS0}}$ takes precedence in the region of overlap.

When programming any of these registers, software must not disable or change the mapping of the addresses where the affected code sequence is located.

Addressing Modes

This term traditionally means how an instruction can specify a memory address. For the Z80S188 these include:

Relative Addressing. JR and DJNZ instructions include a signed 8-bit displacement that specifies a range of addresses -126 to +129 from the Op Code, to which program control can be transferred.

Direct Addressing. In this mode, instructions include a 16-bit logical address.

Register Indirect Addressing. The address is taken from one of the register pairs BC, DE or HL.

Indexed Addressing. In this mode, instructions include an 8-bit signed displacement from the address in an index register IX or IY.

Other contexts in which memory is accessed include instruction fetching, interrupts, DMA operations, cycles generated by external masters while BUSACK is Low, and possibly DRAM Refresh cycles.

Memory Management Unit (MMU)

The MMU translates the 16-bit addresses used by software, called *logical* addresses, into 20- or 23-bit *physical* addresses, as part of all memory accesses performed by the processor. It has no effect on accesses performed by the DMA channels, which include 23-bit address registers. It also has no effect on addresses in I/O space, which always have A22-16 all 0.

The MMU resets to a state in which it has no effect on addresses in processor cycles, passing A15-0 through without change and keeping A22-16 all 0. If an application requires 64 KB of memory or less, the MMU is not necessary.

Even when the MMU has been programmed to do active address transaction, it passes A11-0 (or A9-0 depending on its operating mode) from the logical to the physical address without change. In other words, it manages memory in 4-KB (or 1-KB) byte blocks.



The Z80S188's MMU can operate in either of two modes: Classic and Extended. Reset selects Classic mode, in which the MMU registers at I/O addresses 0038–003AH have the same functionality as on the Z80180 and other 8018x family members (introduced before the Z80S188). This mode is limited to a 1-MB address space, and manages memory in 4-KB blocks.

By executing three specific I/O operations in succession:

1. IN from 003BH
2. IN from 003DH
3. OUT to 003CH

software selects the EXTENDED mode of MMU operation.

Extended mode expands the physical memory address space to 8 MB, and reduces the unit of memory management to 1-KB blocks. The I/O registers at 0038-003H operate differently, and are augmented by additional registers at 003B-003H

NOTES:

1. The Out operation (step 3, previously) does not actually write the register at 00CH.
2. Changing from Classic to Extended mode does not change the memory mapping in effect. Instead, the contents of registers 0038–003AH are rearranged to reflect the new mode.
3. Software selects Extended mode as part of device initialization, before interrupts are enabled. When interrupts are enabled, the software guarantees that the three I/O instructions are consecutive by performing them between DI and EI instructions.
4. Software can change back from Extended to Classic mode by using the sequence IN from 003BH, IN from 003DH, IN from 003CH.

“MMU Registers”, on page 117, describes the registers associated with the MMU.

Classic Mode Operation. In Classic mode, the MMU compares bits 15–12 of each logical address to two 4-bit fields in its Common Base Address Register (CBAR), in an unsigned manner.

- If bits 15–12 of a logical address are less than the value in bits 3-0 of the CBAR, the MMU considers the address to be in Common Area 0. For these addresses, the MMU passes bits 15–12 to the A15-12 pins unchanged, and drives A22-16 all 0s.
- If bits 15–12 of a logical address are greater than or equal to the value in bits 3-0 of the CBAR, but are less than the value in bits 7–4 of the CBAR, the MMU considers the address to be in the Bank Area. For these addresses, the MMU adds the value in its 8-bit Bank Base Register (BBR) to bits 15–12 of the logical address, and outputs the 8-bit sum on A19-12 with A22-20 all 0s.
- If bits 15–12 of a logical address are greater than or equal to the value in bits 7-4 of the CBAR, it considers the address to be in Common Area 1. For these addresses, the MMU adds the value in its 8-bit Common Base Register (CBR) to bits

15–12 of the logical address, and outputs the 8-bit sum on A19-12 with A22-20 all 0s.

NOTE: In Classic mode, software must not program the value in bits 7–4 of the CBAR to be less than the value in bits 3–0 of the CBAR.

Extended Mode Operation. In Extended mode, the MMU compares bits 15–10 of each logical address to bits 7–2 of its Bank Area Register (BAR) and Common Area Register (CAR), in an unsigned manner.

- If bits 15–10 of a logical address are less than the value in bits 7-2 of the BAR, the MMU considers the address to be in Common Area 0. For these addresses, the MMU passes bits 15–10 to the A15-10 pins unchanged, and drives A22-16 all 0s.
- If bits 15–10 of a logical address are greater than or equal to the value in bits 7–2 of the BAR, but are less than the value in bits 7–2 of the CAR, the MMU considers the address to be in the Bank Area. For these addresses, the MMU adds the 13-bit value in its Bank Base Registers High and Low (BBRH and BBRL) to bits 15–10 of the logical address, and outputs the 13-bit sum on A22-10.
- If bits 15–10 of a logical address are greater than or equal to the value in bits 7-2 of the CAR, it considers the address to be in Common Area 1. For these addresses, the MMU adds the 13-bit value in its Common Base Registers High and Low (CBRH and CBRL) to bits 15–10 of the logical address, and outputs the 13-bit sum on A22-10.

NOTE: In Classic mode, the value in bits 7-2 of the CAR must not be programmed to be less than the value in bits 7–2 of the BAR.

MMU Configurations. In the general case, the MMU divides the 64-KB logical memory space into three parts, with Common Area 0 always located at the start of the physical address space, and the Bank Area and Common Area 1 relocatable to other parts of the physical address space, by the values in the Bank Base Register and Common Base Register, respectively.

Certain combinations of values in the CBAR in Classic mode [or BAR and CAR in Extended mode] result in the logical address space being divided into fewer active areas:

If the CBAR [or BAR and CAR] contains all 0s, all logical addresses fall into Common Area 1, and are relocated to a contiguous 64-KB area starting at the address in the CBR times 4096 [or the value in CBRH6-0 and CBRL7-2, times 1024].

If CBAR3-0 are 0 but CBAR7-4 are non-0 [or bits 7–2 of the BAR are 0 but bits 7–2 of the CAR are non-0], the Bank Area and Common Area 1 are active. Logical addresses less than $(\text{CBAR7-4}) * 4096$, or $(\text{CAR7-2}) * 1024$ are relocated by the Bank Base Register, while other addresses are related by the Common Base Register.



If BAR7-2 and CAR7-2 are equal and not 0, Common Area 0 and Common Area 1 are active. Logical addresses less than $(\text{CBAR3}-0) * 4096$, or $(\text{BAR7}-2) * 1024$ are not relocated, and map to the start of physical memory. Other addresses are relocated by the Command Base register.

The MMU After Reset. Since the MMU resets to Classic mode and the CBAR resets to 11110000, logical addresses 0000–FFFFH are in the Bank Area and F000–FFFFH are in Common Area 1. But since the BBR and CBR both reset to 0, the MMU passes all logical addresses through without change, with A22-16 all 0s.

On-Chip ROM or $\overline{\text{ROMCS}}$ Device

Bit 5 in the On-Chip Memory Control Register (page 114) controls whether physical addresses 00000–00FFFH access external memory, vs. on-chip ROM on a masked part, or memory connected to the $\overline{\text{ROMCS}}$ pin on an unmasked part. If this bit is 1, as it is after a Reset, on-chip ROM or $\overline{\text{ROMCS}}$ is enabled.

On-Chip RAM

Bits 7–6 in the On-Chip Memory Control register (page 114) control processor access to on-chip RAM.

If bit 7 is 0, on-chip RAM is disabled.

If bits 7–6 are 10, as they are after a reset, on-chip RAM does not decode A22-16 of physical addresses, and responds to all physical addresses having A15-11 all 1, which we might call addresses xxF800H through xxFFFFH .

If bits 6–5 are 11, on-chip RAM responds to physical addresses with A22-11 all 1: addresses $7\text{FF800}-7\text{FFFFFFH}$.

External Memory Using $\overline{\text{MEMCS0-1}}$

Tables 42-45 on pages 122-125 describe the registers for the Memory Chip Select and Wait block. For each pin there is a high and a low register.

Size/Enable Field. Bits 5–2 of each Low Register are the Size/Enable field. Table 4 describes the possible values of this field.

In all cases, if the address ranges for $\overline{\text{MEMCS0}}$ and $\overline{\text{MEMCS1}}$ overlap, $\overline{\text{MEMCS0}}$ is Low and $\overline{\text{MEMCS1}}$ is High for addresses in the region of overlap.

TABLE 4. SIZE/ENABLE VALUES FOR $\overline{\text{MEMCS0-1}}$

Size/Enable	State of MEMCS Pin
0000	High (disabled)
0001	Low (enabled, except MEMCS1 when MEMCS0 is Low)
0010	Low when A22 = high reg bit 7 (4 MB partition)
0011	Low when A22-21 = high reg bits 7-6 (2 MB partition)
0100	Low when A22-20 = high reg bits 7-5 (1 MB partition)
0101	Low when A22-19 = high reg bits 7-4 (512K partition)
0110	Low when A22-18 = high reg bits 7-3 (256K partition)
0111	Low when A22-17 = high reg bits 7-2 (128K partition)
1000	Low when A22-16 = high reg bits 7-1 (64K partition)
1001	Low when A22-15 = high reg bits 7-0 (32K partition)
1010	Low when A22-14 = high reg + bit 7 of low reg (16K partition)
1011	Low when A22-13 = high reg + bits 7-6 of low reg (8K partition)
11xx	Reserved, do not program

Wait States Field. Bits 1-0 of each low register control how many Wait states the Z80S188 generates for memory accesses in which that pin goes Low, as described in Table 5. These bits reset to 11 (three Wait states).

NOTE: OR instructions are performed on these wait states with the number selected by the Central Wait State Generator described on page 35, and wait states generated by the external WAIT pin. That is, the cycle completes only after all three sources have permitted completion.

TABLE 5. $\overline{\text{MEMCS0-1}}$ WAIT STATES

Low Reg Bits 1-0	Wait States	Min Clocks/Cycle
00	0	3
01	1	4
10	2	5
11	3	6

External Memory with External Decoding

If there are more than two types of external memory (three, counting a ROMCS device on an unmasked part), external address decoding logic is necessary to decode separate selection for two or more of the categories. There are three methods of designing this logic:

Further Decoding of $\overline{\text{MEMCS0-1}}$. This method uses the Low state of MEMCS0 or MEMCS1 to qualify decoding of the highest address lines not used in generating the MEMCS output. RD and WR can be used directly as Low-active Output



Enables and Write Enable controls on the memory. This method is the simplest, but is a few nanoseconds slower than the following methods.

Full Decoding Qualified by $\overline{\text{MEMRQ}}$. This method uses the Low state of $\overline{\text{MEMRQ}}$ to qualify decoding of as many address lines as necessary, from A22 on down. RD and WR can be used directly as Low-active Output Enable and Write Enable controls on the memory.

Full Decoding, Strokes Qualified by $\overline{\text{MEMRQ}}$. Using this method, logic decodes chip selection from the address lines only, but performs AND instructions (positive-logic OR function) on the Low state of $\overline{\text{MEMRQ}}$ with Lows on RD and WR. This action produces Output Enable and Write Enable controls for the memory.

Depending on the memory device timing, either of these choices may yield the highest performance, or the greatest timing margin. With either of these two methods, software ensures that any memory connected to MEMCS0-1 is never selected for the same read cycle as the externally-decoded memory.

Central Wait State Generator

The Z80S188 includes a register that controls automatic insertion of wait states into all memory and I/O accesses. The output of this generator performs logical OR instructions (Low-active OR, positive-logic AND) with wait contributions from other on-chip wait state logic and the external WAIT pin. The result is that for a given address, the number of wait states taken is the largest number among these facilities and any external wait-state generator.

The DMA/Wait Control register (DCNTL) is described on page 141, and is present on all 8018x family members. The DCNTL is located in the DMA register address range, but its wait states apply to accesses by the processor as well as those generated by the DMA channels.

Bits 7-6 of DCNTL select the number of wait states for all memory accesses, as described below:

DCNTL7-6	Wait States	Min PHI Clocks/Cycle for Memory
00	0	3
01	1	4
10	2	5
11	3	6

DRAM Refresh

ZiLOG's Z80 and Z8018x families have traditionally included dynamic RAM refresh logic. This logic is identical on all Z8018x devices including the Z80S188.

The Refresh Control Register (RCR), described on page 113, controls the DRAM refresh feature. When bit 7 is 1, as it is after a reset, the part generates DRAM refresh cycles. Z80S188 applications, which do not include dynamic memory, write an all-0 byte to the RCR as part of initialization. This action disables DRAM

refreshing, and saves the bus bandwidth that would otherwise be taken by refresh cycles.

If bits 7-6 are 11, as they are after a reset, refresh cycles are three clocks long, while if they are 10, refresh cycles are two clocks long. Figure 8 illustrates the timing of a 3-clock refresh cycle; a two-clock cycle eliminates the middle clock cycle (the one labelled T_{RW}).

If bit 7 is 1, bits 1-0 of the RCR control how often refresh cycles occur, as described in Table 6.

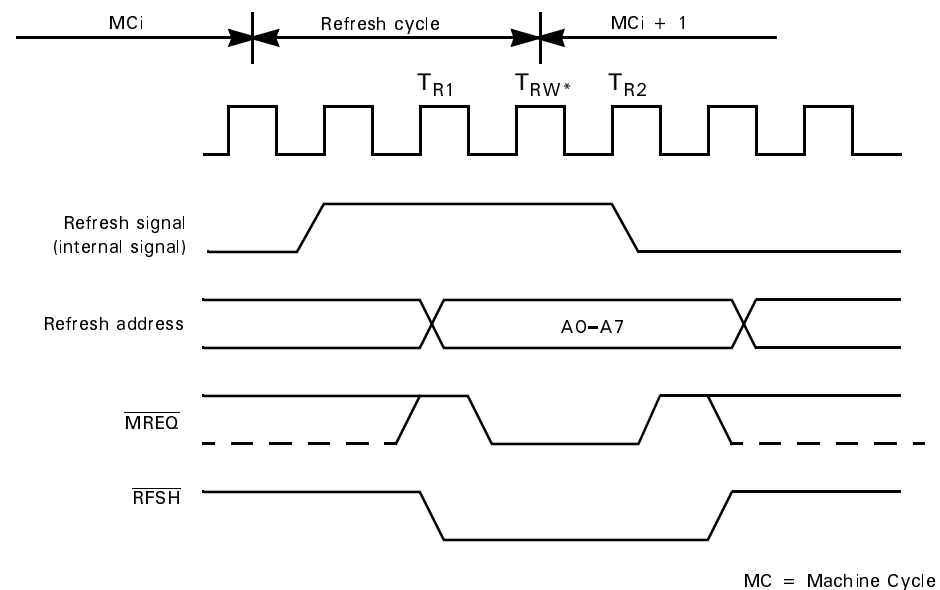


FIGURE 8. 3-CLOCK REFRESH CYCLE

NOTE: * If three refresh cycles are specified, T_{RW} is inserted. Otherwise, T_{RW} is not inserted.

TABLE 6. REFRESH CYCLE CONTROL

RCR1-0 Refresh Cycle Frequency	
00	10 PHI clocks
01	20 PHI clocks
10	40 PHI clocks
11	80 PHI clocks

INPUT/OUTPUT

The Z80S188 includes an I/O space that is distinct from memory space. This I/O space is accessed by means of IN and OUT instructions rather than LD, PUSH, POP, and other instructions that access memory space. The MMU passes addresses in I/O space through without change; such addresses always have A22-16 all 0.



I/O Instructions

The original Z80 featured a 256-byte I/O space. The following instructions are specific to this 256-byte I/O space, and must only be used to access I/O devices that do not decode A15-8.

On the Z80S188, bit 7 of the Interrupt Priority Register (INTPR) controls whether the SIO, PIO, CTC, WDT, Chip Select and Wait registers, and the Interrupt Priority register itself, decode A15-8 all 0, or ignore these lines. If this AllPageIO bit is 1, the following instructions can be used for these registers.

In no case can these instructions be used for the other registers on the port. The 80180 registers are:

OUT (port),A
 IND
 INDR
 INI
 INIR
 OTDR
 OTIR
 OUTD
 OUTI

The following instructions ensure that A15-8 are all 0, and can be used to access any of the Z80S188's on-chip I/O registers, as well as external devices that decode A15-8 as all 0:

IN0 r,(port)
 OUT0 (port),r
 OTDM
 OTDMR
 OTIM OTIMR

The following instructions drive A15-0 from the BC register pair, and can be used to access the full 64-KB I/O space:

IN r,(C)
 OUT (C),r

The following instruction can be used to access the entire 64-KB register space, but only by first loading the MS 8 bits of the address into A. This step is not necessary for devices that do not decode A15-8, including the SIO, PIO, CTC, WDT, CHIP SELECT and WAIT registers when the AllPageIO bit is 1.

IN A,(port)

Relocating the 80180 registers

The "Registers Summary", on page 108 describes how the Z80S188's I/O registers are divided into *80180 registers* and *Z80S188-specific registers*. The latter registers are always located in the range 00D0–00FCH. After a reset, the 80180 registers are located in the range 0000–003HF, but bits 7-6 of the I/O Control

register (page 114) allow software to relocate the 80180 registers to higher addresses:

IOCR 7-6 180 Register Addresses

00	0000-003F
01	0040-007F
10	0080-00BF
11	Reserved

This facility was included to ease porting of Z80 applications to the Z8018x family, but use it with care, as certain tools may assume that the 80180 registers are located in the 0000–003FH range. These tools must be reconfigured to allow for relocated 80180 registers.

I/O Chip Select

The Z80S188 includes one I/O CHIP SELECT pin (IOCS), controlled by the I/O Chip Select high and low registers (IOCSH, IOCSL), which are described on page 126.

Bit 4 of the low register controls whether or not the decode logic matches A15-8 equal to the high register.

Bits 3-2 of the low register are a Size field, and control how the logic matches Bits 7-5 of the low register against A7-5, as described in Table 7.

TABLE 7. A7-5 DECODING FOR $\overline{\text{IOCS}}$ PIN

IOCSL Bits 3-2	A7-5 Matching
00	A7-5 ignored
01	A7 = bit 7 of low register
10	A7-6 = bits 7-6 of low register
11	A7-5 = bits 7-5 of low register

For I/O cycles that drive IOCS Low, bits 1-0 of the low register select how many wait states the I/O Chip select logic add to such cycles. These bits can be considered to add 0-3 Wait states to a base cycle of 4 clocks, or to add 1-4 Wait states to a basic 3-clock cycle. In either case:

IOCSL1-0 Min PHI Clocks/Cycle for $\overline{\text{IOCS}}$ range

00	4
01	5
10	6
11	7

Central I/O Waits

As noted in a previous section, bits 5-4 of the DMA/Wait Control Register can be used to insert wait states into I/O cycles with the Z80S188-*specific registers* at addresses 00D0–00FCH, and into I/O cycles with external devices. These bits add 0-3 Wait states to a base cycle of 4 clocks, or add 1-4 Wait states to a basic 3-clock cycle:

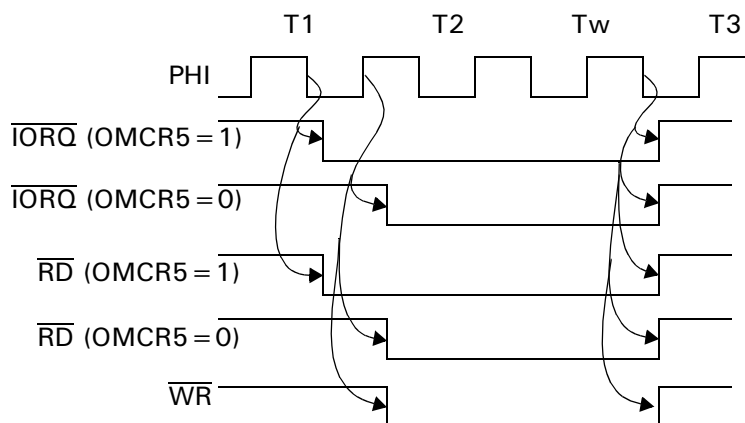
DCNTL5-4 Min PHI Clocks/Cycle for all *non-180* I/O

00	4
01	5
10	6
11	7

 $\overline{\text{IORQ}}$ and $\overline{\text{RD}}$ Timing

Bit 5 in the Operating Mode Control Register (OMCR, described on page 113) controls the timing of the $\overline{\text{IORQ}}$ signal for accesses to *non-180* I/O registers, and the timing of the $\overline{\text{RD}}$ signal when software reads from an *non-180* I/O register.

If this bit is 1, as it is after a reset, the Z80S188 drives $\overline{\text{IORQ}}$ (and $\overline{\text{RD}}$ if applicable) Low from the falling edge of PHI in the T1 clock cycle, which is compatible with the Hitachi 64180. If this bit is 0, the Z80S188 drives $\overline{\text{IORQ}}$ (and $\overline{\text{RD}}$ if applicable) Low one-half clock cycle later, from the rising edge of PHI at the start of T2, which is compatible with the ZiLOG Z80. Both cases are illustrated in Figure .

**FIGURE 9. I/O CYCLE TIMING**

CLOCKING

The Z80S188 can be clocked in either of two ways:

1. By an external TTL- or CMOS-level clock on the EXTAL pin
2. By a crystal connected to its XTAL and EXTAL pins

For an external clock, the signal must be free of overshoot or ringing, must make continuous, monotonic, and rapid transitions in both directions, and must meet the minimum High and Low times specified in “AC Characteristics”.

Multiply by 2 Option

Regardless of whether EXTAL is connected to a crystal or an external clock signal, bit 7 of the Clock Multiplier register (CMR, described on page 111) controls whether or not the Z80S188 multiplies the frequency of EXTAL by two.

If CMR bit 7 is 1, the part multiplies the frequency by two. If this bit is 0, as it is after a reset, the part does not perform the multiplication.

This feature can be used with crystals (or external clocks) up to 16.67 MHz. This feature may allow use of a lower-cost crystal, and eliminates the need for an LC tank circuit, described in “Circuits”, on page 40.

Divide by 2 Option

Bit 7 of the CPU Control Register (CCR, described on page 112) controls whether the Z80S188 uses the signal from the clock multiplier directly as PHI, or whether it divides the signal by two to obtain PHI.

If CCR bit 7 is 0, as it is after a reset, the part divides the signal from the clock multiplier by 2. This mode insulates the part against an asymmetric waveform. Software can write a 1 to CCR bit 7 as part of device initialization, to use the selected signal directly.

If an external clock is connected to EXTAL, and neither the *2 nor /2 option is used, the waveform on EXTAL must meet the minimum High and Low times specified in “AC Characteristics”, on page 201.

Circuits

When using a crystal connected to XTAL and EXTAL, locate it as close as possible to the pins, and minimize the trace lengths among the crystal, pins, and the two capacitors illustrated in Figure 10, which illustrates the connection of a fundamental mode crystal up to and including 20 MHz. C1 and C2 are 20-30 pF, a typical value for which is 22 pF.

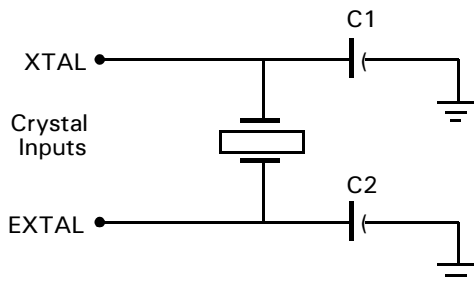


FIGURE 10. FUNDAMENTAL MODE CRYSTAL CIRCUIT ≤ 20 MHz

For frequencies above 20 MHz, use a third-overtone crystal and include a C-L tank circuit to filter the fundamental frequency, as illustrated in Figure 11. Again, it is essential to minimize trace lengths by locating all of the components as close as possible to the XTAL and EXTAL pins.

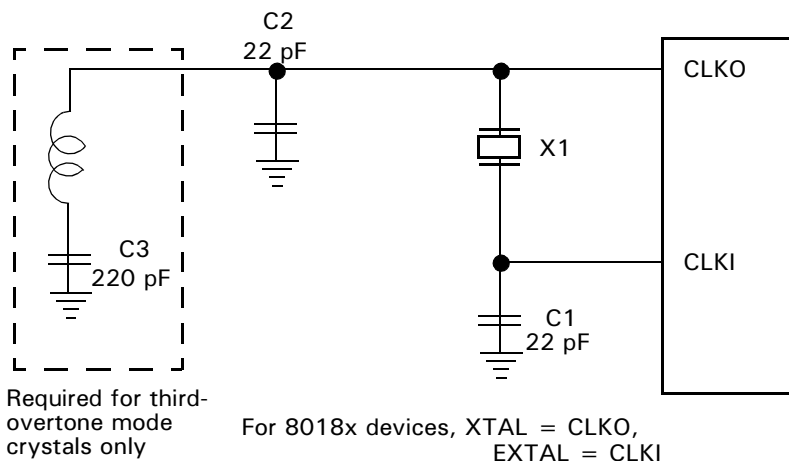


FIGURE 11. THIRD-OVERTONE CRYSTAL > 20 MHz

Crystal Specifications

For fundamental mode crystals up to 20 MHz:

Fundamental, parallel type (AT cut recommended)

Load capacitance: $C_L = C_1 = C_2 = 20\text{-}30$ pF (22 pF typical)

Equivalent Resistance $R_S \leq 60$ ohms

$C_{IN} = C_{OUT} = 15\text{-}22$ pF

Reduced Oscillator Drive Option

Bit 6 in the Clock Multiplier Register, described on page 111, controls the drive (gain) of the oscillator. When this bit is 0, as it is after a reset, the Z80S188 drives a crystal connected to XTAL and EXTAL in an energetic manner, to guarantee that oscillation always starts. This drive is suitable for traditional crystals packaged in

HC-49-type packages, but may be too powerful for crystals packaged for miniaturized applications like PCMCIA.

To reduce the drive/gain of the oscillator, software writes a 1 to bit 6 of the Clock Control Register as part of initialization. This action reduces the drive to about 25% of Normal mode, and also reduces the maximum oscillator frequency from 33 to 20 MHz.

RESET CONDITIONS

How Reset affects each of the registers in I/O space is described in the section, “I/O Registers”, on page 108. Among processor registers, the following registers and state bits are cleared to 0: PC, SP, I, IEF1, IEF2, R, and F. The following are not changed by Reset: A, B, C, D, E, H, L, IX, and IY.

The Z80S188 resets itself at power-up. When power is applied internally, the Z80S188 detects the power rising. After the oscillator starts, the Power On Reset circuitry holds the Z80S188 in reset for 2^{16} clock cycles, driving $\overline{\text{RESET}}$ low to provide a reset to external peripherals.

Another possible source of reset is the Watch-Dog Timer (WDT). See section “Watch-Dog Timer”, on page 62, for more about the WDT.

POWER MANAGEMENT

As on other members of the 8018x family, the Z80S188’s main power saving modes are controlled by the Standby and Idle/Quick bits in the CPU Control Register (page 112), the IOSTOP bit in the I/O Control Register (page 114), and the execution of SLP and HALT instructions. Section “Low-Power Modes”, next, describes the low-power modes.

Low-Power Modes

The IOSTOP bit in the I/O Control Register (page 114) controls operation of the ASCIs, PRTs and CSI/O. When this bit is 0, these peripherals operate normally. When this bit is 1, they are disabled, reducing power use.

The Standby and Idle/Quick bits in the CPU Control Register (page 112) control the actions of the Z80S188 when it executes an SLP instruction. If the application uses the XTAL/EXTAL oscillator and Standby is 1, a SLP instruction stops the oscillator, leading to the lowest power consumption of any mode. This action requires time to restart the oscillator in response to Reset, an interrupt request, or a bus request.

When Standby is 1, the Idle/Quick bit controls how many PHI clocks the Z80S188 waits after re-enabling the oscillator, before restarting operation, with 0 selecting 2^{17} (128K) clocks, and 1 selecting 64 clocks.

When Standby is 0, the oscillator runs for the duration of the SLP instruction, but clocking is blocked to most of the Z80S188. In this case, the Idle/Quick bit controls whether the oscillator output is driven onto the PHI pin, with a 1 in Idle/Quick disabling clocking on PHI.



When Standby is 1, the BREXT bit, bit 5 in the CPU Control Register (page 112), controls whether the Z80S188 restarts the oscillator in response to a Low on the BUSREQ pin, with a 1 enabling these responses.

The interaction of these various bits and states is detailed in Table 8 below, including the conditions that cause the Z80S188 to leave each low-power mode and resume normal operation.

TABLE 8. LOW-POWER MODES

Instruction	Standby	Idle/Quick	I/OSTOP	Mode: Operation
Other than HALT Other than SLP	X	X	0	Normal: The processor fetches instructions and executes them, possibly sharing the bus with on-chip DMAs and external masters. On-chip peripherals operate under software control.
Other than HALT Other than SLP	X	X	1	I/O Stop: The processor, MMU, DMAs, and external masters operate normally, but the ASCIs, PRTs, and CSIO are disabled to reduce power consumption. The only thing these devices can do is to generate an interrupt combinatorially. Software can switch the Z80S188 between NORMAL and I/O STOP mode as appropriate
HALT	X	X	0	Halt: The Processor continually fetches the Op Code following the HALT, but does not execute it, possibly sharing the bus with on-chip DMAs and external masters. HALT mode reverts to NORMAL mode in case of a Low on $\overline{\text{RESET}}$ or $\overline{\text{NMI}}$, a Low on $\overline{\text{INT0}}$ if enabled, an interrupt request from an ASCII, PRT, CSIO, or DMA, or a request on $\overline{\text{INT2-1}}$ if such are enabled.
HALT	X	X	1	Halt and I/O Stop: Similar to HALT mode except that the ASCIs, PRTs, and CSI/O are disabled.
SLP	0	0	0	Sleep: Clocking is blocked to the processor, DMAs, and DRAM refresh logic, so that these stop generating bus activity. The bus can be granted to external masters. I/O can operate except for DMA. SLEEP mode reverts to NORMAL mode under the same conditions as for HALT mode, except that DMAs cannot interrupt.
SLP	0	0	1	System Stop: the oscillator keeps running and generating PHI, but clocking is blocked to most of the chip. Bus granting can occur. SYSTEM STOP mode can revert to NORMAL mode in case of a Low on $\overline{\text{RESET}}$ or $\overline{\text{NMI}}$, a Low on $\overline{\text{INT2-0}}$ to the extent these are enabled, or an enabled interrupt request from an on-chip peripheral that can generate one combinatorially.
SLP	0	1	1	Idle: The oscillator runs, but PHI is blocked, as is clocking to most of the chip. Bus granting can occur if the BREXT bit (CCR5) is 1. IDLE mode can revert to NORMAL mode under the same circumstances as from SYSTEM STOP mode.

TABLE 8. LOW-POWER MODES

Instruction	Standby	Idle/Quick	IOSTOP	Mode: Operation
SLP	1	0	1	Standby: The XTAL/EXTAL oscillator is stopped. This mode does not apply to applications that use LFX TAL/ LFX TAL. Bus granting can occur if the BREXT bit (CCR5) is 1, in which case the device reactivates the oscillator, wait for 2 ¹⁷ (128K) clocks, grant the bus, and deactivate the oscillator again after the bus request is negated. STANDBY mode can revert to NORMAL mode under the same circumstances as from SYSTEM STOP mode. If one of these stimuli occur while the Z80S188 is waiting for 128K clocks before responding to an enabled bus request, or while the bus is granted, it re-enters NORMAL mode when the bus request is negated. Otherwise, the Z80S188 reactivates the oscillator and wait for 2 ¹⁷ (128K) clocks before commencing normal operation.
SLP	1	1	1	Standby with Quick Recovery: Similar to STANDBY mode, except that the Z80S188 waits only 64 clocks after enabling the oscillator, before granting the bus or resuming normal operation.

In SLEEP, SYSTEM STOP, IDLE, or either STANDBY mode, if the Z80S188 leaves the mode because of an NMI or an enabled interrupt with the IEF1 flag 1, it resumes operation by performing the interrupt, with the return address being the instruction after the SLP. If the device exits the low-power mode because of an individually-enabled interrupt request, but the IEF1 bit is 0, the Z80S188 resumes by executing the instruction after the SLP.

PARALLEL I/O (PIOs)

The Z80S188 includes the equivalent of two Z80 PIO devices, which provide four ports called Port A through Port D. Each port includes 8 data pins named PA7-0, PB7-0, PC7-0, or PD7-0, an input Strobe ASTB, BSTB, CSTB, or DSTB, and an output Ready signal ARDY, BRDY, CRDY, or DRDY.

PIO Registers

Two consecutive addresses in I/O space are associated with each port. As for the other Z80 peripherals on the Z80S188, bit 7 of the Interrupt Priority Register controls whether A15-8 are decoded as all 0 for these addresses, as for the *80180 peripherals*, or whether A15-8 are ignored so that Z80-compatible I/O instructions can be used to access the PIOs. The following table shows only bits 7-0 of each PIO address.

**TABLE 9. PIO ADDRESSES**

A7-0	Function
DCH	Port A Data
DDH	Port A Control
DEH	Port B Data
DFH	Port B Control
D4H	Port C Data
D5H	Port C Control
D6H	Port D Data
D7H	Port D Control

An OUT instruction to the lower (Data) address for each port sets output data, while an IN instruction from the lower (Data) address reads input data. The Z80S188 decodes the less significant bits of data written to the higher (Control) address for each port, thus categorizing such output into the following *words*:

TABLE 10. OUTPUT CONTROL WORDS

D7-0	Output Category
xxxxxxx0	Interrupt Vector Word
exxx0011	Interrupt Disable Word
eahm0111	Interrupt Control Word
mmxx1111	Mode Control Word

Two other kinds of *output words* can contain any data, and are implicitly the target of an OUT instruction to the Control address following certain other Words.

After a Mode Control Word where bits 7-6 are 11 is written to the Control address, the next OUT to the Control address is implicitly an I/O Register Control Word, in which 1s identify input pins and 0s identify outputs.

After an Interrupt Control Word where bit 4 is 1 is written to the Control address, the next OUT to the Control address is implicitly a Mask Control Word, in which 0s identify pins that can cause an interrupt.

Modes of Operation

Software can set each port into one of four modes, by writing a Mode Control Word to the port's Control address. Each mode is numbered according to the binary value of bits 7-6 of the Mode Control Word, and is also named.

Output mode/mode 0. In this mode, all 8 port pins are outputs. The XRDY pin goes High when software or a DMA channel writes data to the port's Data register. A Low pulse on the XSTB pin indicates when external logic has acquired the data. A rising edge on XSTB makes the Z80S188 drive the XRDY line back Low, and can be programmed to cause an interrupt request from the port.

NOTE: Because PIO ports do not provide a status register, the only alternative to enabling interrupts for a port operating in mode 0, 1, or 2, is to connect the port's XRDY line to a port pin of another port, that is operating in BIT CONTROL mode.

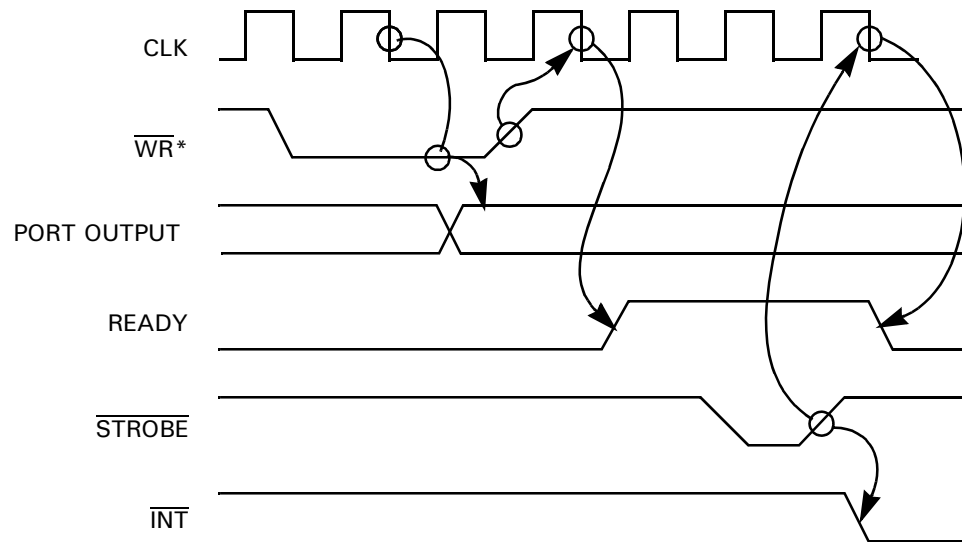


FIGURE 12. MODE 0 OUTPUT TIMING

Input mode/mode 1. In this mode, all 8 port pins are inputs. Software starts a sequence of transfers by doing a dummy read from the port's Data address, which drives XRDY High as a data request to external logic. That logic drives XSTB Low to signify that it has provided data on the port pins. The Low level of XSTB opens the Data register latches, and the rising edge latches the data, makes the Z80S188 drive XRDY Low, and can be programmed to cause an interrupt request from the port. When this interrupt occurs, or when software detects XRDY Low in another port, it can read the captured data from the port's Data address, which again drives XRDY High.

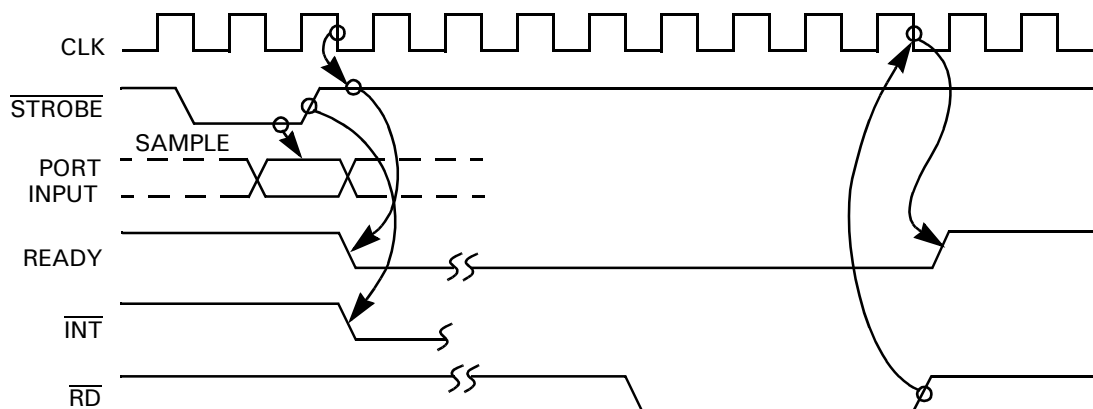


FIGURE 13. MODE 1 INPUT TIMING

Bidirectional mode/mode 2. This mode is only available on ports A and C because it uses the BRDY and BSTB pins in conjunction with port A, or uses the DRDY and DSTB pins in conjunction with port C. When this mode is selected for port A or C, the only mode that can be used on port B or D (respectively) is BIT CONTROL mode, mode 3.

To simplify the language, this mode is described for port A. ARDY and ASTB are used for output handshaking, as in mode 0, while BRDY and BSTB are used for input handshaking, as in mode 1.

If interrupts are enabled for port A, its interrupt vector is returned when ASTB goes High for an output handshake. If interrupts are enabled for port B, its interrupt vector is returned when BSTB goes High for an input handshake, or if a mode 3 interrupt condition is programmed for port B, and it occurs on the PB7-0 lines. This ambiguity can be avoided by programming port B's Mask Control Word with all ones, to disable mode 3 interrupts.

Bidirectionality is achieved by having the Z80S188 drive output data onto the PA7-0 (or PC7-0) pins in this mode, only when external logic drives ASTB (or CSTB) Low. Obviously, external logic must not drive any of the PA7-0 (or PC7-0) pins at the same time that it drives ASTB (or CSTB) Low.

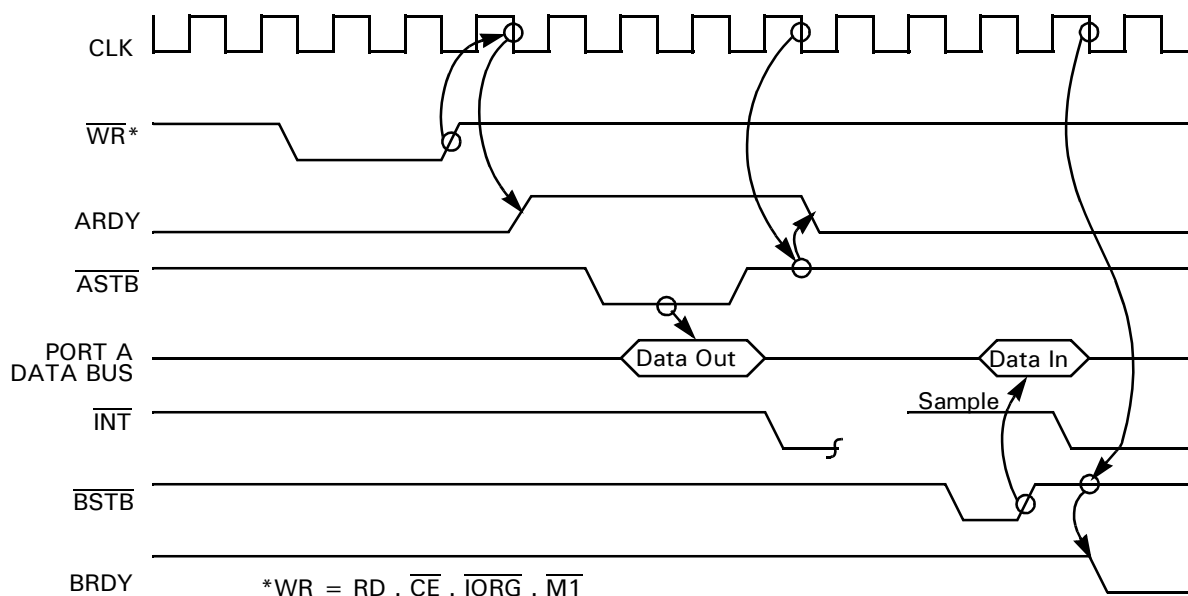


FIGURE 14. MODE 2 BIDIRECTIONAL TIMING

Bit Control mode/mode 3. In this mode, the port pins can be any mixture of inputs and outputs. After writing 11 to bits 7-6 in a Mode Control Word, software next writes an I/O Register Control Word to the port's Data address, containing a 1 for each port pin that is an input, and a 0 for each port pin that the Z80S188 drives as an output. When this value is written, the Z80S188 immediately begins driving pins corresponding to 0, and releases drive on pins corresponding to 1.

In this mode, the port's READY and STROBE pins are not used in conjunction with this port. Instead, the port can interrupt based on conditions on its port pins, as controlled by bits 7-5 of an Interrupt Control Word and a subsequent Mask Control Word. After writing a 1 to bit 4 of an Interrupt Control Word, software next writes a Mask Control Word to the port's Control address, containing a 0 for each port that is active for mode 3 interrupt, and a 1 for each pin that is ignored.

Pins with a corresponding 1 (input) in the I/O Register Control word, and a 0 in the Mask Control Word, are active with respect to mode 3 interrupts.

If bit 5 in a port's Interrupt Control Word is written as 1, the pins are active High, while if bit 5 is 0 they are active Low.

If bit 6 of the Interrupt Control Word is written as 1, all of the active pins in the port must be in the active state simultaneously to cause a mode 3 interrupt. If bit 6 is 0, *any* of the pins that are in an active state can cause an interrupt.

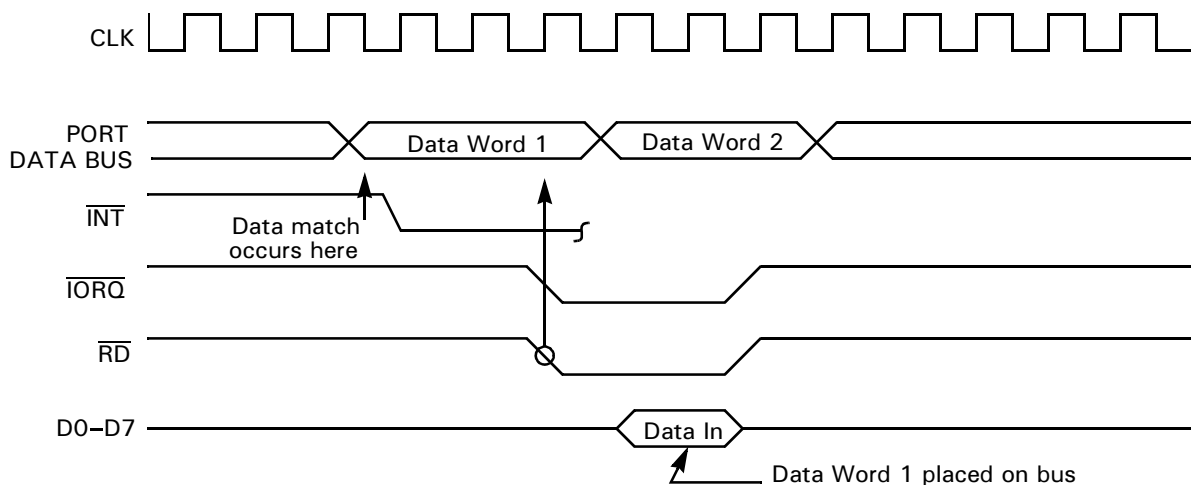


FIGURE 15. MODE 3 BIT CONTROL TIMING, BIT MODE READ

PIO Interrupts

The PIO, SIO, and CTC modules perform logical OR instructions on their interrupt requests with those from any external peripherals that may be connected to the INT0 pin. These external peripherals drive INT0 Low in an open-collector or open-drain fashion. The Z80S188 drives INT0 Low in an open-drain fashion, when any of the PIOs, SIO, or CTC request an interrupt.

Each PIO port can request an interrupt, and includes state bits called Interrupt Pending (IP) and Interrupt Under Service (IUS) and its own 8-bit interrupt vector.

The PIOs cannot be used in interrupt mode 0. They can be used in interrupt mode 1, but since the functionality of this mode is so limited, all following descriptions of PIO interrupt operation assume that software has performed an IM 2 instruction to place the processor in interrupt mode 2.

Interrupt Priority Daisy Chaining. Because more than one port, channel, or interrupt type can request an interrupt at the same time, a mechanism is needed to



select the order in which their requests are serviced. Z80 peripherals and certain other ZiLOG devices use a daisy chain to control the relative priority of interrupt requests among multiple devices and interrupt types within devices.

As described in “Interrupt Acknowledge Daisy Chaining”, on page 18, each PIO port includes an Interrupt Enable In (IEI) pin, from which it receives permission to interrupt from higher-priority devices, and an Interrupt Enable Out (IEO) pin, on which it grants permission to interrupt, to lower-priority devices.

The daisy-chain order and priority within each Z80-device-equivalent module is fixed. For the PIOs, port A has the highest priority, followed by ports B, C, and D.

The PIOs, SIO, and CTC in the Z80S188 are always consecutive on the daisy chain, but the relative priority among PIO AB, PIO CD, SIO, and CTC is programmable in the Interrupt Priority register, with the sole restriction that PIO AB always has higher priority than PIO CD.

Please see “Interrupt Acknowledge Daisy Chaining”, on page 18 for more information about interrupt daisy-chaining.

PIO Software Sequences

The following sections present typical software sequences for both polled and interrupt-driven operation in each of the four operating modes. These procedures can be applied to any channel with the following restrictions:

- BIDIRECTIONAL mode 2 can only be used on channels A and C.
- If channel A or C is used in mode 2, then channel B or D respectively can only be used in BIT CONTROL mode 3.

Polled Operation in Output Mode 1.

1. Connect the XRDY output to a pin of a port that is in Bit Control Mode 3, or to some other general purpose input such as DCD0 or CTS0, for which bit 6 or 5 (respectively) of ASCI0's Extension Control register is 1.
2. Program the register(s) associated with that pin to ensure that it is an input. For a PIO port, write a CFH to that port's Control address to select mode 3, followed by an I/O Register Control Word that includes a 1 for the pin in question.
3. Write a 03H to its Control address, if this port may have been used in another mode since Reset, to ensure that interrupts from the port are disabled.
4. Write a 0HF to this port's Control address, to select Output Mode 0.
5. Write the first (or next) data byte to be sent via the port to the port's Data address when it is available. This action sets the XRDY pin High.
6. Read, periodically, the register containing the bit corresponding to the input pin that is connected to the XRDY pin, and test the bit. When a Low appears on the pin, indicating that the destination device has responded with a rising edge on the XSTB pin, return to step 5.

Interrupt-Driven Operation in Output Mode 0.

1. Disable interrupts (DI), if necessary.
2. Execute as many of the following instructions as have not already been implemented since Reset:

```
IM 2
LD A,inttab/256      ;inttab=start of interrupt table
LD I,A
LD HL,outisr         ;outisr = start of ISR
LD (inttab+ourvec),HL ;ourvec = our vector
```

NOTE: The value of `inttab` is a multiple of 256; that is, bits 7-0 of its value is 00.

3. Write the even value called `ourvec` in the previous step, to the port's Control address. Bit 0 of this value must be 0.
4. Write 0FH to this port's Control address, to select Output Mode 0.
5. Write 87H to this port's Control address, to enable interrupts.
6. Clear an *output byte count* in memory.
7. Enable interrupts (EI)
8. As each output data byte becomes available:
 - a. Disable interrupts (DI).
 - b. Check the output byte count in memory. If it is 0, write the character to the port's Data address, otherwise store it where the interrupt service routine can find it.
 - c. Increment the output byte count in memory.
 - d. Enable interrupts (EI).
9. When control comes to `outisr`:
 - a. Save as many registers as the ISR might use (worst case), using `PUSH`, `EX AF, AF'`, or `EXX` instructions.
 - b. Decrement the output byte count. If it is still 1, fetch the next character and output it to the port's Data address.
 - c. Conclude the interrupt service routine by restoring the saved register values, then executing `EI` followed by `RETI`. The port decodes the `RETI` instruction and re-enables interrupts from itself and from lower-priority devices.



Polled Operation in Input Mode 1.

1. Connect the XRDY output to a pin of a port that is in Bit Control Mode 3, or to some other general purpose input such as DCD0 or CTS0, for which bit 6 or 5 (respectively) of ASCIO's Extension Control Register is 1.
2. Program the register(s) associated with that pin to ensure that it is an input. For a PIO port, write a CFH to that port's Control address to select mode 3, followed by an I/O Register Control Word that includes a 1 for the pin in question.
3. Write a 03H to its Control address, to ensure that interrupts from the port are disabled, if this port may have been used in another mode since Reset.
4. Write a 4FH to this port's Control address, to select Input Mode 1.
5. Read the port's Data address, to make the XRDY pin High.
6. Read the register containing the bit corresponding to the input pin that is connected to the XRDY pin periodically, and test the bit.
7. Read the port's Data address and process the byte when it shows a Low on the pin, indicating that the external device has responded with a data byte and a rising edge on the XSTB pin.

Interrupt-Driven Operation in Input Mode 1.

1. If necessary, disable interrupts (502Z90502)
2. Execute as many of the following instructions that have not been implemented since Reset:

```
IM    2
LD    A,inttab/256      ; inttab = start of interrupt table
LD    I,A
LD    HL,inisr          ; inisr = start of ISR
LD    (inttab+ourvec),HL ; ourvec = our vector
```

NOTE: The value of `inttab` must be a multiple of 256, that is, bits 7-0 of its value must be 00.

3. Write the even value called `ourvec` in the previous step, to the port's Control address. Bit 0 of this value must be 0.
4. Write 4FH to this port's Control address, to select Input Mode 1.
5. Write 87H to this port's Control address, to enable interrupts.
6. Read the port's Data address, to make the XRDY pin High.
7. Re-enable interrupts (EI).
8. Save as many registers as the ISR might use (worst case), using `PUSH`, `EX AF,AF'`, or `EXX` instructions, when control comes to `inisr`.
9. Read the port's Data address and process the character.
10. Conclude the interrupt service routine by restoring the saved register values, then executing `EI` followed by a `RETI` instruction. The port decodes the `RETI`

instruction and re-enables interrupts from itself and from lower-priority devices.

Polled Operation in Bidirectional Mode 2. This mode can only be used on ports A and C.

1. Connect both the ARDY and BRDY pins, or CRDY and DRDY pins, to pins of a port that is in Bit Control Mode 3, or to some other general purpose inputs such as DCD0 and CTS0, with bits 6 and 5 of ASCI0's Extension Control Register containing the value 11. (Since port B or D must be in Bit Control Mode 3 in order to use Bidirectional Mode 2 on port A or C respectively, two port B or D pins are natural choices.)
2. Program the register(s) associated with these two pins to ensure that they are inputs. For a PIO port, write a CFH to that port's Control address to select mode 3, followed by an I/O Register Control Word that includes 1s for the pins in question.
3. If the port may have been used in another mode since Reset, write 03H to the Control addresses of both port A and B (or C and D), to ensure that interrupts are disabled.
4. Write CFH to the port B (or D) Control address, if necessary, to select Bit Control Mode 3, followed by an I/O Control Word to the port B (or D) Control address, containing a 1 for each input pin and a 0 for each output.
5. Write 8FH to the Port A (or C) Control address, to select Bidirectional Mode 2.
6. Read the Port A (or C) Data address, to make the BRDY (or DRDY) pin High.
7. Clear an Output Started flag in memory.
8. Write the first output byte available to the Port A (or C) Data address, to make the ARDY (or CRDY) pin High. Also, set the Output Started flag at this time.
9. Read the register(s) containing the bits corresponding to the ARDY and BRDY (or CRDY and DRDY) lines periodically.
10. Write the Output Started flag to the Port A Data address if it is set and ARDY is Low, indicating that the external device has acknowledged the last output, and another output byte is available.
11. Read BRDY from the Port A Data address, and process it if BRDY is Low, indicating that external logic has provided an input character.

Interrupt-Driven Operation in Bidirectional Mode 2. This mode can only be used on ports A and C.

1. If necessary, disable interrupts (DI).
2. Execute as many of the following instructions that have not been implemented since Reset:



```

IM    2
LD    A,inttab/256      ; inttab = start of interrupt table
LD    I,A
LD    HL,outisr         ; outisr = start of output ISR
LD    (inttab+outvec),HL ; outvec = output vector
LD    HL,inisr          ; inisr = start of input ISR
LD    (inttab+invec),HL ; invec = input vector

```

NOTE: The value of `inttab` must be a multiple of 256, that is, bits 7-0 of its value must be 00.

3. Write the even value called `outvec` in the previous step, to the port A (or C) Control address, and the even value called `invec` to the port B (or D) Control address. Bit 0 of both values must be 0.
4. Write `CFH` to the port B (or D) Control address, if necessary, to select Bit Control Mode 3, followed by an I/O Control Word to the port B (or D) Control address, containing a 1 for each input pin and a 0 for each output.
5. Write `8FH` to the Port A (or C) Control address, to select Bidirectional Mode 2.
6. Write `97H` to the Port B (or D) Control address, to enable input interrupts and indicate that a mask word follows, then write `FFH` to the Port B (or D) Control address, to disable all port B pins from causing an interrupt.
7. Write `87H` to the Port A (or C) Control address, to enable output interrupts.
8. Read the Port A (or C) Data address, to make the `BRDY` (or `DRDY`) pin High.
9. Clear an Output Data Count in memory to 0.
10. Enable interrupts (`EI`).
11. When each output byte becomes available:
 - a. Disable interrupts (`DI`)
 - b. Check the output data count in memory. If it is 0, write the character to the Port A (or C) Data address, to make the `ARDY` (or `CRDY`) pin High. Otherwise, store the character in memory where the interrupt service routine can find it.
 - c. Increment the output data count.
 - d. Enable interrupts (`EI`)
12. If control comes to `outisr`:
 - a. Save as many registers that the interrupt service routine may use (worst case), using `PUSH, EX AF,AF'`, or `EXX` instructions.
 - b. Decrement the output data count in memory. If it is still non-zero, retrieve the next output character and output it to the Port A (or C) Data address.
 - c. Restore the saved registers, then execute an `EI` and `RETI`. The port decodes the `RETI` instruction and re-enables interrupts from itself and from lower-priority devices.
13. If control comes to `inisr`:

- a. Save as many registers as the interrupt service routine may use (worst case), using PUSH, EX AF, AF', or EXX instructions.
- b. Read the Port A (or C) Data address to obtain the byte from the external device, and process it.
- c. Restore the saved registers, then execute an EI and RETI. The port decodes the RETI instruction and re-enables interrupts from itself and from lower-priority devices.

Other Bidirectional Operating Modes. Software can be written for other Bidirectional possibilities than those described above. For example, output can be handled in an interrupt-driven fashion while input can be handled by polling, or operate in the opposite direction. Also, when using input interrupts, port B or D input pins can be left unmasked to cause interrupts, which use the same vector as input interrupts.

Polled Operation in Bit Control Mode 3.

1. Write 03H to this port's Control address, if this port may have been used in another mode since Reset, to ensure that interrupts from the port are disabled.
2. Write CFH to the port's Control address, to select Bit Control Mode 3.
3. Write an I/O Control Word to the port's Control address, containing a 1 for each input pin and a 0 for each output pin.
4. Read the port's Data address whenever the state of inputs is needed. For output bits this action returns the data last written to the Data address.
5. Write the new states to the port's data address whenever the state of outputs needs to change. Data written to input bits is ignored.

Interrupt-Driven Operation in Bit Control Mode 3.

1. If necessary, disable interrupts (DI)
2. Execute as many of the following instructions as have not already been implemented since Reset:

```
IM 2
LD A,inttab/256      ; inttab = start of interrupt table
LD I,A
LD HL,ourisr        ; ourisr = start of ISR
LD (inttab+ourvec),HL ; ourvec = our vector
```

NOTE: The value of inttab must be a multiple of 256, that is, bits 7-0 of its value must be 00.

3. Write the even value called ourvec in the previous step, to the port's Control address. Bit 0 of this value must be 0.
4. Write CFH to the port's Control address, to select Bit Control Mode 3.
5. Write an I/O Control Word to the port's Control address, containing a 1 for each input pin and a 0 for each output pin. If all pins are outputs, the port does



not interrupt. See the preceding section, “Polled Operation in Bit Control Mode 3”.

6. Write a value of form `1ah10111` to the port’s Control address. This enables interrupts from the port and indicates that a mask value follows. The *h* bit is 1 if a High on the selected port pin(s) is the active state, or 0 if a Low on the pin(s) is the active state. The *a* bit matters only if the following mask word contains 0s for more than one input pin. In this case a 0 in the *a* bit makes the port request an interrupt when *any* of the selected pins are in the selected active state, and a 1 in *a* makes the port request an interrupt when *all* of the selected pins are in the active state. The former is called an *OR function* and the latter an *AND function*.
7. Write a mask value to the port’s Control address. The port ignores bits in this value corresponding to output pins. For input pins, a 0 enables the pin to cause an interrupt as described above, while a 1 prevents the pin from causing an interrupt. If no input pins are enabled by a corresponding 0, the port does not interrupt: See the preceding section “Polled Operation in Bit Control Mode 3”.
8. Enable interrupts (EI).
9. Whenever the state of inputs is needed, read the port’s Data address. For output bits this action returns the data last written to the Data address.
10. Write the new states to the port’s data address whenever the state of outputs needs to change. Data written to input pins is ignored.
11. Monitor the input pin(s) selected by 0s in step 6 for the condition specified by the *h* and *a* bits. When this logical state goes from *false* to *true*, the port requests an interrupt.
12. When control comes to `ourisr`:
 - a. Software saves as many registers as it might use (worst-case), using `PUSH, EX AF, AF'`, or `EXX` instructions.
 - b. Software may need to read the port’s Data address, or other registers, to gather more data about current conditions.
 - c. The interrupt service routine’s (ISR’s) response to the interrupt is application-dependent. No action is necessary to *clear the interrupt*. The *a* and *h* bits, or the mask, may be changed for the next interrupt.
 - d. The ISR concludes by restoring the saved registers, followed by `EI` and `RETI` instructions. The port decodes the `RETI` instruction and re-enables interrupts from itself and from lower-priority devices.

PIOs and Reset

Reset configures the PIO ports as follows:

1. Each port is placed in Input Mode 1. This disables output drive on all port pins.
2. `XRDY` outputs are forced Low.
3. Interrupts are disabled.

4. Output registers for modes 0, 2, and 3 are cleared to 0s, which correspond to Lows.
5. Mode 3 mask registers are cleared to 0s to disable interrupts for all pins.

DMA CHANNELS

The Z80S188 includes two DMA channels called DMA0 and DMA1. Both channels can transfer data between memory and a peripheral in I/O space. In addition, DMA0 can perform memory to memory block transfers, and transfers between memory and memory-mapped I/O devices.

Both DMA channels are of the *flowthrough* type, in which each byte transferred requires two bus cycles, one to read the *source* and the second to write the *destination*. Because of this technique, neither memory nor peripherals differentiate the bus cycles performed by the DMA channels, because they are identical to bus cycles from the processor.

DMA transfer can occur as fast as 6 clocks/byte. At 33 MHz, the speed is up to 5.5 MB/second. Destination/output devices typically require edge-sensitive request mode, in which case the maximum rate is 9 clocks/byte, or 3.67 MB/second at 33 MHz. “DMA Registers”, on page 131, describes the registers associated with the DMA channels.

DMA Basics

DMA0 has two 23-bit address registers and a 16-bit byte count, while DMA1 has one 23-bit memory address registers and a 16-bit I/O address register and byte count. For DMA0 the address registers are called the Source and Destination Address Registers (SAR and DAR). DMA1’s registers and called the Memory and I/O Address Registers (MAR and IAR).

Each address register is divided into three Z80S188 I/O registers, called L (LOW), H (High), and B. When the DMA channel is operating, A7–0 is driven from the L register and A15–8 from the H register. For memory addresses (always for MAR) the channel drives A22–16 from the LS 7 bits of the B register. For I/O addresses (always for IAR) the channel uses the LS three bits of the B register to select the source of the DMA Request signal that controls data transfer.

Each byte count register is divided into two Z80S188 I/O registers, called L (Low) and H (High).

After programming a channel’s address and byte count registers, software starts the channel by setting its Enable bit in the DSTAT register (DE0 or DE1). As a DMA channel transfers each byte, the byte counter register decrements. When a channel has decremented the byte count to 0, it becomes inactive by clearing the Enable bit.

Software can select whether a channel increments or decrements a memory address as it transfers each byte. DMA0 also has an option to keep a memory address fixed. To select this fixed address option for the source or destination, DMA0 must be a memory-mapped I/O device that provides a DMA Request signal on the DREQ0 pin.

DMA Requests

An external peripheral, that needs a DMA channel to transfer data for it, must provide a DMA request signal to the $\overline{\text{DREQ0}}$ pin for DMA0, and/or to the $\overline{\text{DREQ1}}$ pin for DMA1. A DMA request can be connected directly to one of these pins if only one external peripheral is serviced by that DMA channel, or via external selection logic if more than one external device is to use the DMA channel.

Bits corresponding to address bits 18-16 of the register containing the I/O address of a peripheral, select between the external DREQ pin and internal peripherals as the source of each DMA channel's request. For a memory-mapped peripheral, that is, a source or destination of a DMA0 memory-to-memory operation that's programmed to use a fixed address, the $\overline{\text{DREQ0}}$ pin is always used as the REQUEST signal.

The DMA request signal indicates when an input or source peripheral has a byte to be transferred to memory, or when an output or destination peripheral needs a byte from memory.

Edge- vs. Level-Sensitive Requests

DMA requests can be programmed to be Low-level sensitive or falling-edge sensitive. For an output/destination peripheral, the timing requirements on the DMA Request signal dictate falling-edge mode. An input/source peripheral can use either an edge- or level-sensitive DMA Request.

Figure 16 illustrates the timing of a level-sensitive DMA Request. DMA operation is triggered when the Z80S188 samples the DMA request line Low. In the figure below, the Z80S188 samples the Request line again, at the rising PHI edge that begins the second-last clock cycle of the machine cycle that writes the byte to the destination. If the Request line is Low at that time, as it is in the rightmost down-arrow below, DMA operation continues for another byte. If the Request is sampled High, as at the leftmost down-arrow below, the current DMA channel relinquishes use of the bus (to the processor, the other DMA channel, or an external master) after completing the write cycle.

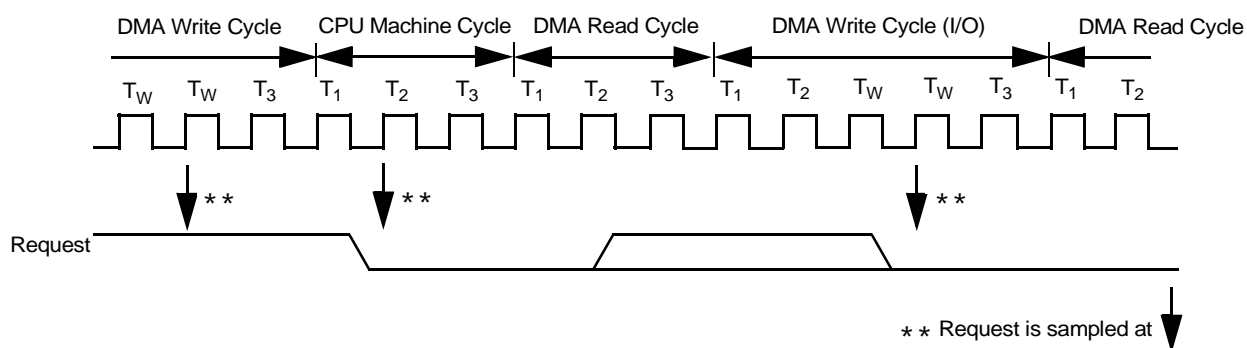


FIGURE 16. PROCESSOR/DMA OPERATION WITH LEVEL-SENSE REQUEST

Figure 17 illustrates the timing of an edge-sensitive request. At the first down-arrow, the DMA channel writes a byte to the destination. A new falling edge has not occurred by the second-last rising PHI edge of the cycle, so the bus is relinquished to the processor. At the same sampling point in the processor cycle, a new falling edge has occurred on the Request line. The DMA reads and then writes a byte. At the same point in its write cycle, a new falling edge has not occurred, and bus control is returned to the processor, and the DMA channel does not operate again until the Request line has gone High and then Low again, some time past the right edge of the figure below.

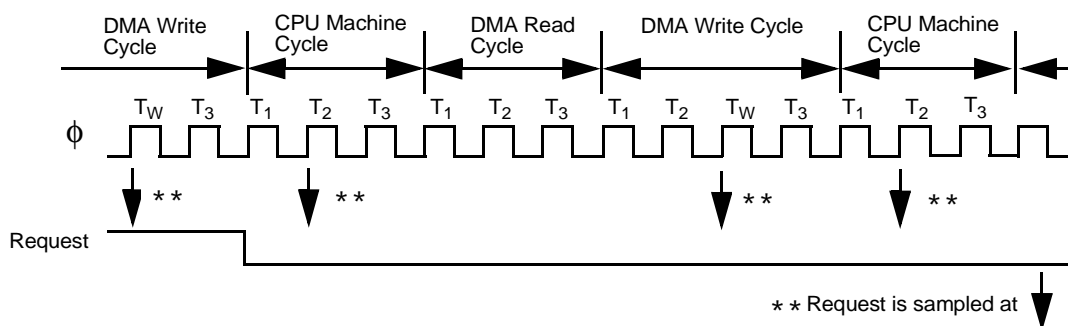


FIGURE 17. PROCESSOR/DMA OPERATION WITH EDGE-SENSE REQUEST

Memory-to-Memory Modes

In a DMA0 memory-to-memory operation, in which both the source and destination are programmed for address incrementing or decrementing, there is no peripheral to supply a request signal to control the transfer. In this case, software can select between two modes of operation by programming MMOD, bit 1 of the DMA Mode register.

If MMOD is 0, the processor and DMA channel alternate bus cycles until the DMA has completed the block transfer and decremented its byte count to 0. This mode is called CYCLE STEAL mode.

If MMOD is 1, the DMA channel does continuous cycles until it completes the block transfer, and the processor can do nothing during this time. This mode is called BURST mode.

DMA Interrupts

Software can enable interrupts from each DMA channel, which then requests an interrupt when it has decremented its byte count to 0. When the processor acknowledges such an interrupt, the interrupt service routine address is fetched from memory at (I : IL : 8) for DMA0 or (I : IL : 10) for DMA1.

If the interrupt service routine does not have another block of data for the DMA channel to transfer, it prevents further interrupts by clearing the interrupt enable bit (DSTAT bit 2 for DMA0, bit 3 for DMA1) before it re-enables interrupts with an EI instruction. If the ISR programs the DMA channel for another transfer, inter-



rupts can be re-enabled with an EI instruction, after the DMA is restarted by writing DCNTL.

Setting Up a DMA Transfer

Write the Address Registers. For DMA0, this transfer includes SAR0L, SAR0H, SAR0B, DAR0L, DAR0H, and DAR0B. If the source is in I/O space, write SAR0B with a code to select the source of the DMA Request for DMA0, as described in Table 11:

TABLE 11. SAR0B VALUE FOR A SOURCE IN I/O SPACE

SAR0B Bits 2-0	DMA Request Source
000	$\overline{\text{DREQ0}}$ pin
001	ASCIO RDRF
010	ASCI1 RDRF
011	Reserved, do not program
100	SIO A Rx
101	SIO B Rx
110	PIO A In
111	PIO B In

If the DMA destination is in I/O space, write DAR0B with a code to select the source of the DMA Request for DMA0, as described in Table 12:

TABLE 12. DAR0B VALUE FOR A DESTINATION IN I/O SPACE

DAR0B Bits 2-0	DMA Request Source
000	$\overline{\text{DREQ0}}$ pin
001	ASCIO TDRE
010	ASCI1 TDRE
011	Reserved, do not program
100	SIO A Tx
101	SIO B Tx
110	PIO A Out
111	PIO B Out

For DMA1, software must write MAR1L, MAR1H, MAR1B, IAR1L, IAR1H, and IAR1B. Write IAR1B with a code to select (with the DIM1 bit in the DCNTL register) the source of the DMA Request for DMA1, as described in Table 13:

TABLE 13. IAR1B VALUE

IAR1B Bits 2-0	DIM1	DMA Request Source
000	X	$\overline{\text{DREQ1}}$ pin
001	0 (mem→I/O)	ASCIO TDRE
	1 (I/O→mem)	ASCIO RDRF
010	0 (mem→I/O)	ASC11 TDRE
	1 (I/O→mem)	ASC11 RDRF
011	X	Reserved, do not program
100	0 (mem→I/O)	SIO A Tx
	1 (I/O→mem)	SIO A Rx
101	0 (mem→I/O)	SIO B Tx
	1 (I/O→mem)	SIO B Rx
110	0 (mem→I/O)	PIO A Out
	1 (I/O→mem)	PIO A In
111	0 (mem→I/O)	PIO B Out
	1 (I/O→mem)	PIO B In

Write the Byte Count Registers. Write the less-significant byte to BCR0L or BCR1L, and the MS byte to BCR0H or BCR1H. An all-0 value makes the DMA transfer 65,536 bytes.

For DMA0, write the DMODE Register. Bits 3-2 select the operating mode for the source, as described in Table 14. Bits 5-4 select the operating mode for the destination, as described in Table 15. For memory-to-memory block transfers, bit 1 (MMOD) selects between Cycle Steal and Burst mode, as described in “Memory-to-Memory Modes”, on page 58.

TABLE 14. DMA0 SOURCE MODE

DMODE 3-2	Mode
00	Increment Memory Address
01	Decrement Memory Address
10	Fixed Memory Address (request on $\overline{\text{DREQ0}}$ pin)
11	Fixed I/O Address

TABLE 15. DMA0 DESTINATION MODE

DMODE 5-4	Mode
00	Increment Memory Address
01	Decrement Memory Address
10	Fixed Memory Address (request on $\overline{\text{DREQ0}}$ pin)
11	Fixed I/O Address

Write the DCNTL Register. Typically, software reads this register, modifies the bits for the current DMA channel, and writes back the result. Bits 7-4 select the number of waits to insert for Memory and I/O accesses, as described in “Central Wait State Generator”, on page 35. For DMA0, bit 2 selects edge- vs. level-sensitivity on the DMA Request, as described in an earlier section. For DMA1, bit 3 selects between edge- and level-sensitivity, and bits 1-0 select the operating mode, as described in Table 16:

TABLE 16. DMA1 OPERATING MODE

DCNTL 1-0	Mode
00	Increment Memory Address → Fixed I/O Address
01	Decrement Memory Address → Fixed I/O Address
10	Fixed I/O Address → Increment Memory Address
11	Fixed I/O Address → Decrement Memory Address

Write the DSTAT Register to Enable the DMA Channel. Software reads this register, modifies the bits noted below, and writes back the result. For DMA0, write 110 to bits 6-4, keep bit 3 unchanged, and write a 1 to bit 2 if DMA0 interrupts when it has decremented the byte count to 0, or a 0 to bit 2 if DMA0 does not interrupt.

For DMA1, write a 1 to bit 7, 01 to bits 5-4, write a 1 to bit 3 if the user requires DMA1 to interrupt when it has decremented the byte count to 0, or a 0 to bit 3 if not, and keep bit 2 unchanged.

NMI and DME

When software writes to DSTAT to start either DMA channel as described above, this action also sets the DMA Master Enable (DME) bit that can be read as bit 0 in DSTAT. A 1 in this bit enables operation by either or both DMA channels.

To guarantee that a Non-Maskable Interrupt (NMI) is handled promptly, the Z80S188 clears DME to suspend DMA operation, when $\overline{\text{NMI}}$ Low is detected.

As soon as possible, the NMI service routine reads DSTAT. For each DMA channel, if the DE bit (DSTAT7 or 6) is 1, and the associated device (if any) has not overrun or underrun, the service routine clears that channel's DWE bit (DSTAT5 or 4) to 0. If either DWE bit is 0, the result is written back to DSTAT. This action sets DME again and re-enables DMA operation.

DMA Channel Completion

While a DMA channel is operating, software can stop it by reading the DSTAT register, clearing bits 6 and 4 for DMA0, or 7 and 5 for DMA1, and writing the result back to DSTAT.

Otherwise, if software enabled the channel to interrupt, when the channel decrements the byte count to 0, an interrupt is requested.

If software does not enable the DMA channel to interrupt, it polls the Enable bit in DSTAT (bit 6 for DMA0, bit 7 for DMA1) to determine when the DMA channel finishes transferring the current block of data. In some applications, software can use status or an interrupt from the associated peripheral device, to determine completion of the block transfer.

Handling DMA Interrupts

When the conditions noted in “On-Chip Interrupt Handling”, on page 28 are met with respect to a DMA interrupt request, the processor fetches the interrupt service routine address from memory at (I : IL : 8) for DMA0 or (I : IL : 10) for DMA1. The service routine, as a minimum:

1. Reads the DSTAT register to determine that the DE bit for the DMA channel corresponding to the service routine entry point, has gone from 1 to 0. If a common routine is used for both DMAs, the service routine determines which DE bit has been cleared, or both.
2. Reprograms the channel's registers if there is more for a completed DMA channel to process, and restart it, as described above.
3. Clears the DIE bit for the channel, in the DSTAT register, to prevent another interrupt for the same DMA completion, if there is no more data to process.
4. The ISR ends with an EI and a RET instruction, to return to the interrupted process.

WATCH-DOG TIMER

The Watchdog Timer (WDT) can be used to protect against unreliable software, power line faults that put the processor into unusual states, and other processes harmful to the device.

When the WDT is enabled, software must reload it periodically to prevent it from driving the WDTOUT output Low. WDTOUT can be connected to RESET, to $\overline{\text{NMI}}$, or to external logic. The time period, within which software must reload the WDT to prevent a Low on WDTOUT, is programmable among 2^{16} , 2^{18} , 2^{20} , or 2^{22} system clocks.

The registers in the WDT are described in “Watch-Dog Timer Registers”, on page 141. Several provisions of the WDT are intended to enhance its integrity against runaway execution. The WDT can be reloaded by writing the specific value 4EH to the WDT Command register. This register can only be disabled by writing a 0 to bit 7 of the WDT Master register, then writing the value B1H to the WDT Command register.

COUNTER/TIMER CHANNELS (CTCs)

The Z80S188 includes the equivalent of one Z80 CTC device: four Counter/Timer Channels numbered 0 through 3. Each channel includes a readable 8-bit down counter, a prescaler that can divide the channel's input clock by 16 or 256, a Clock/Trigger input (CLK/TRG0-3), and a Zero Count/Timeout output (ZC/TO0-3).



Each channel can operate in a Counter mode, in which the CLK/TRG pin provides its down-count clock, or in TIMER mode, in which the down-count clock is the Z80S188's master PHI clock divided by 16 or 256, and in which the CLK/TRG pin can optionally provide a trigger to start down-counting.

In both modes, when down-counting reaches 0, the channel produces a pulse on its ZC/TO output, and reloads the down counter from its Time Constant register.

CTC Addresses and Registers

Each CTC channel has one 8-bit address in I/O space. Like the other Z80 peripherals in the Z80S188, CTC I/O decoding can include A15–8 all 0 as for 80180 register decoding, or can ignore A15–8, depending on the AllPageIO bit in the Interrupt Priority register. The latter choice allows use of Z80 I/O instructions for legacy-code compatibility.

TABLE 17. CTC I/O ADDRESSES

Channel	I/O Address
0	D0H
1	D1H
2	D2H
3	D3H

“Counter/Timer (CTC) Registers”, on page 142, describes the use of the CTC addresses.

Reading a CTC channel's address always yields the contents of its down counter.

Except when a channel is expecting a Time Constant as described below, writing an even value (having a 0 in bit 0) to channel 0's address sets bits 7-3 of the interrupt vector value for all four channels, while writing an odd value (having a 1 in bit 0) to any channel loads its Channel Control register. If bit 2 written to a Channel Control register is 1, the next write to that channel's address loads the channel's Time Constant register.

Channel Control Register. Bit 7 of this register must be 1 to enable an interrupt from the channel when its down counter reaches 0, or 0 if an interrupt is not required.

A 1 in bit 6 selects Counter mode, in which the down-counter is decremented by the selected edge on the CLK/TRG input, while a 0 in bit 6 selects Timer mode, in which the down-counter is decremented by the PHI clock divided by 16 or 256, and in which the selected edge on CLK/TRG can optionally enable the channel to start down-counting.

In Timer mode only, a 1 in bit 5 conditions the prescaler to divide PHI by 256, while a 0 in bit 5 divides PHI by 16. In Counter mode, bit 5 has no significance.

In Counter mode, a 1 in bit 4 selects rising edges on CLK/TRG to decrement the down-counter, while a 0 selects falling edges. In Timer mode with a 1 in bit 3, a 1

in bit 4 selects a rising edge on CLK/TRG to start down-counting, while a 0 selects a falling edge on CLK/TRG. In Timer mode with a 0 in bit 3, bit 4 has no significance.

In Timer mode, a 1 in bit 3 conditions the channel to wait for the edge selected by bit 4, following the rising edge of machine cycle T2 after the one loading the Time Constant. A 0 in bit 3 starts the prescaler counting at the rising edge of T2. In Counter mode, bit 3 is insignificant.

A 1 written to bit 2 conditions a channel to load the next byte written to the channel's address into the Time Constant register. Writing a 0 to bit 2 keeps the channel decoding bit 0 to select between the Channel Control register and the Interrupt Vector register. If a channel is already operating, and software writes 10 to bits 2-1, the subsequently-written Time Constant cannot take effect until the next time the channel counts down to 0.

A 1 written to bit 1 stops the channel, if it was running, and resets it. If software writes 11 to bits 2-1, the channel is re-enabled for operation when software writes the new Time Constant. Writing 0 to bit 1 of a running channel, allows the channel to continue running.

Bit 0 must be 1 to identify a byte for the Channel Control register. Writing a byte to channel 0, with a 0 in bit 0, writes bits 7-3 of the value into the Interrupt Vector register that applies to all four channels.

Time Constant Register. After software writes a 1 to bit 2 of a Channel Control register, the next byte written to that channel's address is loaded into the channel's Time Constant register. If the channel is not already running, the value is also loaded into the down-counter, and the channel is enabled to count down from that value. (In Timer mode with a 1 in bit 3 of the CCR, actual down-counting is delayed until the channel senses the selected edge on CLK/TRG.)

A 0 time constant forces a channel count down from 256.

Interrupt Vector Register. If software writes a value with a 0 in bit 0 to channel 0's address, bits 7-3 of the value are captured in this register. Subsequently, when any CTC channel interrupts the processor (assuming the processor is in INT0 interrupt mode 2), the CTC returns these bits as bits 7-3 of the Interrupt Vector, with the number of the interrupting channel as bits 2-1 and a 0 in bit 0.

Reading the Down Counter. Reading any channel's address yields the current contents of its down-counter. The software can remember the status of each channel.

CTC Interrupts

The CTC, PIO, and SIO modules perform logical OR instructions on interrupt requests using those from any external peripherals that may be connected to the INT0 pin. These external peripherals drive INT0 Low in an open-collector or open-drain fashion. The Z80S188 drives INT0 Low in an open-drain fashion when any of the CTC or SIO channels or PIO ports are requesting an interrupt.



Each CTC channel can request an interrupt, and includes state bits called Interrupt Pending (IP) and Interrupt Under Service (IUS). The 4 CTC channels share a 5-bit base interrupt vector. A CTC channel that is the highest priority requesting device during an interrupt acknowledge cycle returns the channel number in bit 2-1 of the interrupt vector.

Interrupt Priority Daisy Chaining. Since more than one port, channel, or interrupt type can request an interrupt at the same time, a mechanism is needed to select the order in which their requests are serviced. Z80 peripherals and certain other ZiLOG devices use a daisy chain to control the relative priority of interrupt requests among multiple devices and interrupt types within devices.

As described in “Interrupt Acknowledge Daisy Chaining”, on page 18, each CTC channel includes an Interrupt Enable In (IEI) pin, from which permission is received to interrupt from higher-priority devices, and an Interrupt Enable Out (IEO) pin, on which permission is received to interrupt to lower-priority devices.

The daisy-chain order and priority within each Z80-device-equivalent module is fixed. For the CTC, channel 0 has the highest priority and channel 3 the lowest.

The CTC, PIOs, and SIO in the Z80S188 are always consecutive on the daisy chain, but the relative priority among them is programmable in the Interrupt Priority register.

See “Interrupt Acknowledge Daisy Chaining”, on page 18 for more information about interrupt daisy-chaining.

CTC Software Sequences

The following sections describe polled and interrupt-driven operations in both Counter and Timer modes. These operations apply equally to any of the four channels.

Polled Operation in Counter Mode.

1. Write a 47H to the channel's address if falling edges on the channel's CLK/TRG n pin decrement the counter, or 57H if rising edges decrement the counter.
2. Write the value from which the counter counts down, to the channel's address. A 0 value forces the channel count down from 256. This value is loaded into the counter and the channel's TIME CONSTANT register. Unless software reloads a new TIME CONSTANT value in step 5, this is also the value from which the counter restarts, after it has counted down to 0.
3. Enable the channel to count down by 1 whenever the selected edge occurs on its CLK/TRG n pin. Software can read the channel's address at any time, to determine the counter value.
4. When a channel's counter contains 01, and the selected edge occurs on its CLK/TRG n pin, instead of going to 00, the counter is reloaded with the value currently in the channel's Time Constant register. This event is accompanied by a High pulse on the channel's ZC/TON pin. Unless software is monitoring the counter very closely, so that it is guaranteed to read every value of the counter, the most reliable method for software to detect this event is to compare

successive values that have read from the channel's address. When a value is larger than its predecessor, the counter has counted down to 0 and reloaded. (Software cannot detect reloads if the Time Constant value is 01.)

5. To load a new value into the Time Constant register while the channel is running, software first writes a 45H or 55H to the channel's address (depending on the same edge selection as in step 1), and then writes the new Time Constant value to the channel's address.

Interrupt-Driven Operation in Counter Mode.

1. Disable interrupts (DI), if necessary.
2. Execute as many of the following instructions that have not been implemented since Reset:

```
IM 2
LD A,inttab/256      ; inttab = start of interrupt table
LD I,A
LD HL,ctcNisr        ; ctcNisr = start of ISR
LD (inttab+ctcNvec),HL ; ctcNvec = our vector
```

NOTE: The value of `inttab` must be a multiple of 256, that is, bits 7-0 of its value must be 00.

3. Bit 0 of the value called `ctcNvec` in the previous step must be 0, and bits 2-1 must be equal to this channel's number. Unless this step has been done previously, write this value to channel 0's address (D0H).
4. Write a C7H to this channel's address if falling edges on the channel's CLK/TRGn pin decrements the counter, or D7H if rising edges decrement the counter.
5. Write the value from which the counter counts down, to the channel's address. A 0 value makes the channel count down from 256. This value is loaded into the counter and the channel's Time Constant register. Unless software reloads a new Time Constant value in step 6, this is also the value from which the counter restarts, after it has counted down to 0.
6. The channel is enabled to count down from the value written in the previous step, whenever the selected edge occurs on its CLK/TRGn pin. If the counter restarts from a different value when it reaches 0, software immediately writes a C5H or D5H to the channel's address (depending on the same edge selection as in step 4), and then writes the new Time Constant value to the channel's address.
7. Software reads the counter's value from the channel's address at any time.
8. When a channel's counter contains 01, and the selected edge occurs on its CLK/TRGn pin, instead of containing the value 00, the counter reloads with the value currently in the channel's Time Constant register. This event is accompanied by a High pulse on the channel's ZC/TOn pin, and the channel requests an interrupt.



9. When the processor acknowledges the interrupt, the CTC automatically supplies the channel's number in bits 2-1 of the interrupt vector, so that control comes to `ctcNisr`. At that time:
 - a. Software saves as many registers as it may use (worst-case), using `PUSH`, `EX AF, AF'`, or `EXX` instructions.
 - b. If the counter must be reloaded with a different value the next time it reaches 0, the interrupt service routine (ISR) writes a `C5H` or `D5H` to the channel's address (depending on the same edge selection as in step 4), and then writes the new Time Constant value to the channel's address.
 - c. Other actions by the ISR are application-dependent.
 - d. The ISR concludes by restoring the saved registers, followed by `EI` and `RETI` instructions. The channel decodes the `RETI` instruction and re-enables interrupts from itself and from lower-priority devices.

Polled Operation in Timer Mode.

1. Write a value of form `00pet111B` to the channel's address. A 0 in the `p` bit conditions the channel to decrement its counter once every 16 PHI clocks, while a 1 in bit `p` (`20H`) conditions the channel to decrement the counter once every 256 PHI clocks. A 0 in the `t` bit conditions the timer to start as soon as the following time constant is written, while a 1 in bit `t` forces the timer to wait for the time constant, then the edge selected by the `e` bit, on its `CLK/TRGn` pin. If `t` is 1 and `e` is 0 (`08H`), the channel waits for a falling edge on `CLK/TRGn` before starting the timer. If `t` and `e` are both 1 (`18H`), the channel waits for a rising edge on `CLK/TRGn` before starting the timer.
2. Write the value from which the timer counts down, to the channel's address. This value is loaded into the counter and the channel's Time Constant register. Unless software reloads a new Time Constant value, this is also the value from which the timer restarts, after it has counted down to 0.
3. Software can read the channel's address at any time, to determine the counter value.
4. When a channel's counter is decremented to 00, it is reloaded with the value currently in the channel's Time Constant register. This event is accompanied by a High pulse on the channel's `ZC/TOn` pin. Unless software is monitoring the counter very closely, so that every value of the counter is read, the most reliable way for software to detect this event is to compare successive values that were read from the channel's address. When a value is larger than its predecessor, the counter has counted down to 0 and then reloaded. (Software cannot detect reloads if the Time Constant value is 01.)
5. To load a new value into the Time Constant register while the channel is running, software waits for any programmed `CLK/TRG` wait to finish, as evidenced by the counter value being decremented from its initial value. The software writes `00pet101B` to the channel's address, and then writes the new Time Constant value to the channel's address.

Interrupt-Driven Operation in Timer Mode.

1. Disable interrupts (DI), if necessary.
2. Execute as many of the following instructions that have not been implemented since Reset:

```
IM 2
LD A,inttab/256      ; inttab = start of interrupt table
LD I,A
LD HL,ctcNisr        ; ctcNisr = start of ISR
LD (inttab+ctcNvec),HL ; ctcNvec = our vector
```

NOTE: The value of *inttab* must be a multiple of 256, that is, bits 7-0 of its value must be 00.

3. Write the value of bit 0 to channel 0's address (D0H) unless this step has been implemented previously. Bit 0 of the value called *ctcNvec* in the previous step must be 0, and bits 2-1 must be equal to this channel's number.
4. Write a value of form 10pet111B to the channel's address. A 0 in the p bit conditions the channel to decrement its counter once every 16 PHI clocks, while a 1 in p (20H) conditions the channel to decrement the counter once every 256 PHI clocks. A 0 in the t bit conditions the timer to start as soon as the following time constant is written. A 1 in t makes the timer wait for the time constant, followed by the edge selected by the e bit, on its CLK/TRGn pin. If t is 1 and e is 0 (08H), the channel waits for a falling edge on CLK/TRGn before starting the timer. If t and e are both 1 (18H), the channel waits for a rising edge on CLK/TRGn before starting the timer.
5. Write the value from which the timer counts down, to the channel's address. This value is loaded into the counter and the channel's Time Constant register. Unless software reloads a new Time Constant value, this value is also the value from which the timer restarts, after it has counted down to 0.
6. If the counter restarts from a different value when it reaches 0, software waits for any programmed CLK/TRG wait to finish, as evidenced by the counter value being decremented from its initial value. Software writes 10pe0101B to the channel's address, and then writes the new Time Constant value to the channel's address.
7. Software can read the channel's address at any time, to determine the counter value.
8. When a channel's counter is decremented to 00, the counter is reloaded with the value currently in the channel's Time Constant register. This event is accompanied by a High pulse on the channel's ZC/TOn pin, and the channel requests an interrupt.
9. When the processor acknowledges the interrupt, the CTC automatically supplies the channel's number in bits 2-1 of the interrupt vector, so that control comes to *ctcNisr*. At that time:
 - a. Software saves as many registers as it might use (worst-case), using PUSH, EX AF, AF', or EXX instructions.



- b. If the timer restarts from a different value the next time it reaches 0, the interrupt service routine (ISR) writes `10p0101B` to the channel's address, and then writes the new Time Constant value to the channel's address.
- c. Other actions by the ISR are application-dependent.
- d. The ISR concludes by restoring the saved registers, followed by `EI` and `RETI` instructions. The channel decodes the `RETI` instruction and re-enables interrupts from itself and from lower-priority devices.

PROGRAMMABLE RELOAD TIMERS (PRTs)

Each PRT is a 16-bit down-counter that can be read dynamically, with a reload value that can be dynamically programmed. Each PRT can optionally interrupt the processor when it counts down to 0 and reloads.

The PRTs on most other 8018x family members count down at the fixed frequency of $\text{PHI}/20$, but the Z80S188 includes a flexible prescaler that can count down each PRT at any power-of-two divisor between PHI and $\text{PHI}/16384$. The prescalers reset to an 80180-compatible countdown rate of $\text{PHI}/20$.

Software can program PRT1 to do waveform generation on its TOUT output, under control of the TOC 1-0 bits in the Timer Control Register. This capability is enhanced by the flexible prescaler.

“Programmable Reload Timer (PRT) Registers”, on page 146, shows the PRT registers. Reset clears both Timer Downcount Enable bits (`TDE1`, `TDE0`) in the Timer Control register to 0, which inhibits the PRTs from operating.

Starting a PRT

Before starting a PRT, software writes the Timer Prescale register to select the downcounter frequency, which can be $\text{PHI}/20$ as on the 80180, or any power-of-two divisor between PHI and $\text{PHI}/16384$.

Next, software programs the PRT's initial down-count value to its Timer Data registers (`TMDR0L` and `TMDR0H`, or `TMDR1L` and `TMDR1H`), and write its second (and possibly constant) down-count value to its Timer Reload registers (`RLDR0L` and `RLDR0H`, or `RLDR1L` and `RLDR1H`).

Then software reads the Timer Control register (`TCR`), set the appropriate Timer Downcount Enable bit (`TDE0` or `TDE1`), set or clear the corresponding Timer Interrupt Enable bit (`TIE0` or `TIE1`) depending on whether an interrupt is desired when the count is decremented to 0, and write the result value back to the `TCR`.

The read-modify-write procedure, described above, ensures that starting one PRT does not affect the operation of the other. Applications that only use one PRT can write only the desired value to the `TCR`.

Stopping a PRT

Software can stop a running PRT at any time, by reading the `TCR`, clearing its `TDE` bit, and writing the result value back to the `TDR`. The software may do this

because the PRT's counting is no longer necessary, or before rewriting its RLDR value as described below.

PRT Operation

While a PRT is running, the down-counter is decremented at the frequency selected in the Timer Prescale Register, which resets to PHI/20. When the count down reaches 0, the PRT automatically reloads its TMDR from its RLDR, and sets its TIF bit in the TCR. If the TIE bit in the TCR is 1, an interrupt is requested.

Software can clear a TIF bit after reading it as 1 in the TCR, by thereafter reading either half of that PRT's TMDR. However, a TMDR read, without first reading a TIF bit as 1 in the TCR, does not clear the PRT's TIF.

Software can read the down-counter, from TMDR0L and TMDR0H or TMDR1L and TMDR1H, at any time without worrying about the timer counting between the two 8-bit IN0 instructions, as long as it reads the L register first. Reading the L register captures the 8 MS bits of the down-counter in a separate 8-bit latch, from which the value is read when software reads the H register.

Writing an RLDR. Software can write a new reload value, to RLDR0L and RLDR0H or RLDR1L and RLDR1H, while the PRT is running, but there is no hardware safeguard against the down-counter decrementing to 0 between the two 8-bit OUT0 instructions necessary to write the new reload value, and consequently loading an incorrect value.

If software writes a new reload value in response to a PRT interrupt or to detecting a TIF bit 1 in the TCR, and count values are always large enough to prevent this type of problem, software can automatically write the RLDR. Otherwise, software performs the following tasks:

1. Reads the Timer Control Register (TCR)
2. Clears the TDE bit,
3. Writes the result back to the TCR,
4. Writes the RLDR (L and H, in either order)
5. Writes the value from step 1 (with the TDE bit 1) back to the TCR.

Handling PRT interrupts

When the conditions noted in "On-Chip Interrupt Handling", on page 28 are met with respect to an interrupt request from a PRT, the processor reads the address of the interrupt service routine from memory at address (I : IL : 4) for PRT0, or (I : IL : 6) for PRT1. The PRT ISR performs the following actions:

1. Saves as many registers of the interrupted process as it may use, by means of PUSH, EX AF, AF', and/or EXX instructions.
2. Reads the TCR, verify that its TIF bit is 1, and then reads the PRT's TMDR register to clear its TIF bit. This process prevents another interrupt during the same 0 count.



3. Changes the RLDR value for the down count sequence after the one that is currently in progress, as described above.

A PRT interrupt involves time-periodic functions in service to the overall application.

4. Restores the saved registers when the PRT ISR is complete, then returns to the interrupt process using EI and RET instructions.

If both PRTs are active and both are started and stopped, sometimes at interrupt level and sometimes at mainline level, mainline code that reads, modifies, and writes the TCR protects against conflicts with an ISR for the other PRT, surrounding the read-modify-write (or steps 1-5 in “Writing an RLDR”, on page 70) with DI and EI instructions.

PRTs and Reset

Both prescalers reset to PHI/20, both TMDRs and both RLDRs reset to FFFFH, and the TCR resets to all 0s, which inhibits PRT operation until a TDE bit is set.

SERIAL I/O CHANNELS (SIOs)

The Z80S188 includes the equivalent of one Z80-SIO device: two multiprotocol, full-duplex serial channels called A and B. Each channel can handle Asynchronous, Classic Synchronous, or HDLC/SDLC communications, providing a variety of options in each mode.

Each channel provides pins for Tx and Rx Data, Tx and Rx Clock inputs, Request to Send and Data Terminal Ready outputs, Clear to Send and Data Carrier Detect inputs, a Sync input or output, and a Wait/Ready output.

SIO Addresses

Each channel has a Data address and a Control address in I/O space.

As for the other Z80 peripherals on the Z80S188, bit 7 of the Interrupt Priority register controls whether A15–8 are decoded as all 0 for these addresses, as for the 80180 peripherals, or whether A15–8 are ignored so that Z80-compatible I/O instructions can be used to access the SIO channels. The following table describes only bits 7-0 of each SIO address.

TABLE 18. SIO ADDRESSES

A7-0	Function
D8H	Channel A Data
D9H	Channel A Control
DAH	Channel B Data
DBH	Channel B Control

Writing to a channel’s Data address provides a byte of data to be transmitted, while reading from a channel’s Data address fetches a byte of received data.

Read or write the Data address only when status information read from the Control address indicates that received data is available for reading, or that the Transmitter is ready for a data byte.

Alternatively, if a transmitter or receiver is programmed to operate with DMA channel 0 or 1, internal signalling controls when the DMA channel reads or writes the Data address.

The Control address for an SIO channel uses indirect addressing to access several write-only and read-only registers, as described in the following table

TABLE 19. WRITE AND READ REGISTERS PER CHANNEL

	Write Registers	Read Registers
Channel A	WR0–1, WR3–7	RR0–1
Channel B	WR0–7	RR0–2

Indirect addressing works as follows. After Reset, reading the Control address for an SIO channel returns the contents of Read Register 0, and writing to the Control address writes Write Register 0.

Write Register 0 includes three fields. Bits 7-6 allow software to issue one of three separate commands affecting frame/message structure, with 00 signifying No Operation. Bits 5-3 allow software to issue one of seven other commands, with 000 signifying No Operation. Bits 2-0 are an indirect register number that allows a software to select a different Write or Read register for its next access to the Control register, with 000 keeping the next access referring to Read or Write register 0.

Writing an all-0 byte to WR0 affects nothing. At the other extreme, one write to WR0 can issue two separate commands and select another register for the next access to the Control address.

When software writes a non-0 value to bits 2-0 of WR0, and then reads or writes the register selected by that value, the SIO channel clears bits 2-0 of WR0 back to 000 immediately thereafter, so that the next access to the CONTROL address again accesses Write register 0 or Read register 0.

Accesses to the Data register have no effect on this *alternating access* mechanism in the Control register.

SIO Write Registers

Rather than describing registers as bit 7 of WR0 through bit 0 of WR7, then RR0-RR2, this section describes registers, and fields within them, in a top-down order that makes the SIO easier to learn. This order is also the order in which software writes the Write Registers during channel initialization.

“Serial I/O (SIO) Registers”, on page 151, presents the SIO registers and fields in Classic numerical order.

WR4. This register sets the basic modes of the overall channel.



WR4 bits 3-2 are the most basic mode selection in an SIO channel. If these bits are 00, the channel operates in a Synchronous mode. Otherwise, the channel operates in an Asynchronous mode, and the field indicates the (minimum) number of STOP bits that the transmitter sends between characters.

WR4 Bits 3-2 Basic Mode

00	Synchronous
01	Async, Tx min 1 stop bit
10	Async, Tx min 1.5 stop bits
11	Async, Tx min 2 stop bits

If WR4 bits 3-2 are 00, then WR4 bits 5-4 select the Synchronous protocol:

WR4 Bits 5-4 Synchronous mode

00	Classic with 8-bit Sync (monosync)
01	Classic with 16-bit Sync (Bisync)
10	HDLC/SDLC
11	External Sync

If WR4 bits 3-2 are 00 to select a Synchronous mode, WR4 bits 7-6 must be 00 to select a 1X clock. If WR4 bits 3-2 are non-0 to select Async operation, WR4 bits 7-6 may be any of the following values:

WR4 Bits 7-6 Clock Division

00	Bit rate = \overline{RxC} and \overline{TxC} pin frequency (Sync or isochronous)
01	Bit rate = \overline{RxC} and \overline{TxC} pin frequency / 16
10	Bit rate = \overline{RxC} and \overline{TxC} pin frequency / 32
11	Bit rate = \overline{RxC} and \overline{TxC} pin frequency / 64

The combination of Asynchronous format on TxD and RxD, and a 1X clock, is sometimes called Isochronous mode. In this mode, data on RxD must be Synchronized with the clock on RXC as in Synchronous modes, so it is not appropriate to call this an Asynchronous mode on the bit level. However, characters can still occur irregularly/asynchronously, because start and stop bits frame each character.

WR4 bits 1-0 control whether the transmitter generates and sends, and the receiver expects and checks, an additional parity bit in each character, after the number of bits specified in WR3 (for Rx) or WR5 (for Tx):

WR4 Bits 1-0 Synchronous mode

x0	No parity
01	Even parity

WR4 Bits 5-4	Synchronous mode
11	Odd parity

Even parity means that the total number of one bits in each character, including the parity bit, is even. On the receive side, if parity is used with less than 8 bits/character, the parity bit is included in data that software or a DMA channel reads from the channel's Data address.

NOTE: Parity can be used in either Asynchronous or Synchronous modes, but it is more common in Asynchronous applications.

WR5. Most of the bits in WR5 control the Transmitter.

A 1 in WR5 bit 4 enables the transmitter to send data, when software or a DMA channel writes data to the channel's Data address. A 0 in WR5 bit 4 disables the Transmitter.

WR5 bits 6-5 control how many data bits the Transmitter sends in each character:

WR5 Bits 6-5	Transmit Data Bits per Character
00	5 or Less
01	6
10	7
11	8

This number does not include any start, parity, or stop bits.

If this field is less than 11 (to select less than 8 bits per character), the Transmitter sends the less significant bits of each character that software or a DMA channel writes to the channel's Data address.

If WR5 bits 6-5 are 00 (to select 5 or less bits per character), each byte written to the channel's Data address must have one of the formats described in the following table. The Transmitter sends the 1–5 least significant bits of each character, based on the contents of the more significant bits of each byte:

Tx Character	Number of Bits Sent
000DDDDD	5 Bits
1000DDDD	4 Bits
11000DDD	3 Bits
111000DD	2 Bits
1111000D	1 Bit

Writing a 1 to WR5 bit 4 immediately forces the TXD pin Low to send a break condition, even if the transmitter is currently sending a character. Writing a 0 to



WR5 bit 4 releases this forcing condition, typically letting the pin return to High/1/Mark.

WR5 bit 0 controls whether the transmitter includes transmitted characters in its CRC calculation. In Classic Synchronous applications in which not all Tx characters are included in the CRC, software sets this bit to 1 for a character to be included, or 0 for a character to be excluded, just before writing each character to the Data address.

WR5 bit 2 selects the CRC polynomial for both transmitting and receiving. It must be 0 for HDLC/SDLC mode. For Classic Synchronous mode it can be either:

WR5 Bit 2	CRC polynomial
0	$X^{16}+X^{12}+X^5+1$ (CCITT)
1	$X^{16}+X^{15}+X^2+1$ (CRC-16)

WR5 bits 7 and 1 control the DTR and RTS pins respectively. In both cases, a 1 makes the pin Low and a 0 makes it High. The only exception to complete software control is that in Async mode, if software changes the RTS bit from 1 to 0 while data is still being sent, the pin does not become High until all data previously written to the channel's Data address has been sent.

WR3. The bits in WR3 control the receiver.

A 1 in WR3 bit 0 enables the receiver, a 0 disables the receiver. Set this bit only after all other receive parameters have been set and the receiver is completely initialized.

WR3 bits 7-6 control the number of data bits the receiver captures in each character, not including any start, parity, or stop bits:

WR3 Bits 7-6	Receive Data Bits per Character
00	5
01	6
10	7
11	8

WR3 bit 5 controls whether the CTS and DCD pins auto-enable the Transmitter and Receiver, respectively. If bit 5 is 0, the pins can be read in Read Register 0, but have no effect on the hardware.

In Synchronous modes, writing a 1 to WR3 bit 4 sets the Sync/Hunt bit in Read Register 0, forcing the receiver into Hunt mode, in which it searches for a Sync character or Flag.

WR3 bit 3 controls whether the receiver includes received characters in its CRC checking. In HDLC/SDLC mode, software sets this bit to 1 during initialization, because all characters in every frame are covered by the CRC. In other Synchron-

nous modes, software must set this bit to the appropriate state for each received character, before the next character is transferred from the receive shift register to the receive FIFO, or within 8-bit times of when each character goes into the FIFO.

This procedure is possible because, in Synchronous modes other than HDLC/SDLC, the receiver includes an extra 8 bits of shift register between the receive shift register (from which each character is transferred to the Rx FIFO) and the receive CRC checker. This action allows 8-bit times for software to read each character, decide if it must be included in the CRC, and set or clear WR3 bit 3 accordingly.

In HDLC/SDLC mode, WR3 bit 2 controls whether the receiver checks an address at the start of each frame. If bit 2 is 1, the receiver ignores any frame in which the first 8 bits do not match either the contents of WR6 or the global address FFH. If this bit is 0, the receiver receives all frames.

In Synchronous modes other than HDLC/SDLC, WR3 bit 1 controls whether the receiver places Sync characters in the Rx FIFO (if this bit is 0) or strips them from the data stream (if this bit is 1).

NOTE: Because this option does not affect whether received Sync characters are included in CRC checking, software must clear this bit when it receives the first (non-Sync) character of each message, and thereafter check for embedded Syncs and clear WR3 bit 3 for such characters.

WR1. This register controls a channel's interrupts and WAIT/READY pin.

WR1 bits 7-5 control the WAIT/READY pin. A 1 in bit 7 enables the pin. Bit 6 selects between the Ready function (1) and the Wait function (0). Bit 5 selects whether the pin is based on the transmitter (0) or receiver (1). Treating all three bits as a field:

WR1 Bits 7-5	Function of Wait/Ready Pin
00x	Not driven
01x	High
100	Low = Tx not full; High = Tx full
101	Low = Rx data available; High = Rx empty
110	Low = write to Data address, Tx full; Not Driven = no write or not full
111	low = read from Data address, Rx empty; Not Driven = no read or not empty

NOTE: This field does not have to be programmed with any particular value in order to use a receiver or transmitter with a DMA channel.

WR1 bit 2 in Channel B controls whether the SIO modifies the interrupt vector that it returns during an interrupt acknowledge cycle, to identify the highest-priority cause of the interrupt. If this bit is 0, the SIO always returns the interrupt vector as software wrote it to Write Register 2 in channel B. If this bit is 1, the SIO



modifies bits 3-1 of the vector it returns, to identify the highest-priority cause of the interrupt (higher values have higher priority):

Vector Bits 3-1	Highest-Priority Interrupt
000	Channel B Tx Buffer Empty
001	Channel B External/Status Change
010	Channel B Rx Character Available
011	Channel B Special Receive Condition
100	Channel A Tx Buffer Empty
101	Channel A External/Status Change
110	Channel A Rx Character Available
111	Channel A Special Receive Condition

NOTE: WR1 bit 2 in channel A has no function.

WR1 bits 4-3 control receive interrupts:

WR1 Bits 4-3	Receive interrupt mode
00	Disabled (never)
01	On first character
10	All characters; parity error is a Special Receive Condition
11	All characters; parity error is not a Special Receive Condition

If parity checking is enabled in WR4, WR bit 2 is 1, and this field is 10, an interrupt for a received character with a parity error is written to the Special Receive Condition ISR, while interrupts for parity-correct characters are written to the Rx Character Available ISR.

A 1 in WR1 bit 1 enables an interrupt whenever the Transmitter buffer is empty.

A 1 in WR1 bit 0 enables External/Status Change interrupts. “Handling External/Status Interrupts”, on page 93, describes how to handle such interrupts.

Commands in WR0. In addition to the indirect register address in bits 2-0, WR0 contains two command fields. Commands are values written to these fields, that are not themselves latched like most other register bits, but rather serve to change the state of the SIO channel at the end of the write operation.

WR0 bits 5-3 can contain the following commands:

WR0 Bits 5-3	Command Name and Description
000	Null code. Use this value when writing WR0 to set an indirect address and/or issue a command in bits 7-6.
001	Send Abort. In HDLC mode, this command makes the transmitter send an Abort sequence, consisting of 8 to 13 1s.
010	Reset External/Status Interrupts. After an External/Status interrupt, the status bits in RR0 are latched. This command unlatches them and re-enables such interrupts. Latching the status bits captures short pulses, so that software has a chance to detect how these bits have changed.
011	Channel Reset. This command resets the channel state like a Low on $\overline{\text{RESET}}$. Writing this command to channel A also resets the interrupt prioritization logic. Allow four extra PHI clocks after issuing this command, before writing to the channel again.
100	Enable Interrupt on Next Rx Character. If WR3 bits are 01 to select Interrupt On First Character mode for the receiver, this command re-enables an interrupt for the next character received.
101	Reset Transmitter Interrupt Pending. If transmitter interrupts are enabled by WR1 bit 1, interrupts occur when the Tx buffer register becomes empty. Software can issue this command to prevent further Tx interrupts, until software writes a new Tx character to the channel's Data address, or until the CRC has been sent.
110	Error Reset. The Parity and Overrun bits in RR1 are cumulative/latched. This command reset these bits.
111	Return From Interrupt. Writing this command to channel A has the same effect as executing an RETI instruction: it clears the Interrupt Under Service (IUS) status of the highest priority SIO module that had IUS set (if any). This enables lower-priority interrupts that had been inhibited by the IUS condition.



WR0 bits 7-6 may contain the following commands:

WR0 Bits 7-6	Command Name and Description
00	Null code. Use this value when writing to WR0 to set an indirect address and/or issue a command in bits 5-3.
01	Reset Rx CRC Checker. This command is needed only in Classic Synchronous mode, because in HDLC/SDLC mode CRC checking is fully automatic. In Classic Synchronous mode, software issues this command before the first frame, and after it has checked the CRC correctness of a frame.
10	Reset Tx CRC Generator. In HDLC/SDLC mode, this command is needed only during device initialization. In Classic Synchronous mode, software issues this command before writing the first character of a new frame to the channel's data address.
11	Reset Tx Underrun/EOM Latch. Issuing this command clears an internal state in the transmitter called the Underrun/EOM bit, which is used only in Synchronous modes. This bit is set by RESET and when the transmitter sends the end of a Synchronous frame/message. It controls what the transmitter does when it runs out of data, a Transmit Underrun condition.

In Classic Synchronous mode, if the bit is 1 the Tx sends a Sync character, while if the bit is 0 it sends the accumulated CRC following by one or more Sync character(s). In HDLC/SDLC mode, if the bit is 1 the Tx sends one or more Flag(s) when an underrun occurs. If the bit is 0, it sends the accumulated CRC followed by one or more Flag(s).

Historically, SIO documentation has urged HDLC/SDLC software to issue this command and clear the bit, as soon as possible after it presents the first character of the frame to the Transmitter. In this case, the transmitter always sends a CRC. If software detects that the underrun occurred at an inappropriate point, software can override the sending of the CRC by issuing a Send Abort command. This sequence is typical when an underrun occurs within a frame.

Interrupt Vector in Channel B WR2. Data written to this register are returned during an interrupt acknowledge cycle for any interrupt from either channel. If channel B's WR1 bit 2 is 0, all eight bits are returned as written. If this bit is 1, bits 7-4 and 0 are returned as written, and bits 3-1 reflect the highest priority channel and interrupt type being requested, as described above for WR1 bit 2.

WR6. The contents of this register depends on the channel mode:

- Async: not used

- Monosync or External Sync: Tx Sync character
- Bisync: First 8 bits of common Tx/Rx Sync pattern
- HDLC/SDLC: Address match character if WR2 bit 2 is 1

WR7. The contents of this register depends on the channel mode:

- Async or External Sync: not used
- Monosync: Rx Sync character
- Bisync: Last 8 bits of common Tx/Rx Sync pattern
- HDLC/SDLC: must be 01111110 (Flag pattern)

SIO Read registers

RR0. This register includes the most basic channel status.

RR0 bit 0 is 1 if there is at least one received character available to be read from the channel's data address.

In channel A, RR0 bit 1 is 1 if there are any interrupt condition(s) in either channel.

RR0 bit 2 is 1 if there is permission to write a transmit character to the channel's data address.

NOTES:

1. Bits 7-3 of RR0 are latched as a group by the channel, whenever any of bits 7 or 5-3 change, or when bit 6 is 1. This latching prevents transient conditions from being lost before software can detect them. To clear this latching and read the real-time status of these bits, write a `Reset External Status Interrupts` command to WR0, as described above.
2. In the following descriptions, the phrase *the last time status was latched or unlatched* is defined as the more recent of the following states
 - a. The first time any of bits 7 or 5-3 changed, or bit 6 was set, since `Reset` or since the last `Reset External/Status Interrupts` command was written to WR0
 - b. When the last `Reset External/Status Interrupts` command was written to WR0

RR0 bit 3 describes the state of the channel's DCD pin (0 = High, 1 = Low) the last time status was latched or unlatched.

RR0 bit 4 describes the state of the SYNC pin the last time status was latched or unlatched. Behind the latch, this pin is an input or output according to the mode:



Mode	Behind the Latch, RRO Bit 4:
Async	Reflects the state of the channel's $\overline{\text{SYNC}}$ pin (0 is High, 1 is Low)
External Sync	Reflects the state of the channel's $\overline{\text{SYNC}}$ pin (0 is High, 1 is Low). After software first enables the receiver, or after it writes an Enter Hunt mode command to WR0, the receiver is in Hunt mode. A falling edge on $\overline{\text{SYNC}}$ clears Hunt mode and enables character assembly, as well as setting this bit and causing an External/Status interrupt if it is enabled. However, a rising edge on this pin does not set Hunt mode nor disable character assembly. It only clears this bit, and causes another External/Status interrupt if it is enabled. Software responds to this bit (interrupt) by writing an Enter Hunt Mode command to WR0.
Classic Sync	Clears when the receiver is disabled, and when software writes an Enter Hunt Mode command to WR0. This bit is set when the hardware detects a (8- or 16-bit) Sync pattern. The channel's $\overline{\text{SYNC}}$ pin is an output controlled by this bit (0 is High, 1 is Low).
HDLC/SDLC	Clears when the receiver is disabled, and when software writes an Enter Hunt Mode command to WR0. This bit is set when the hardware detects a Flag pattern (01111110). The channel's $\overline{\text{SYNC}}$ pin is an output controlled by this bit (0 = High, 1 = Low).

RR0 bit 5 describes the state of the channel's CTS pin (0=High, 1=Low) the last time status was latched or unlatched.

RR0 bit 6 displays the Transmit Underrun status the last time status was latched or unlatched. Behind the latch, Transmit Underrun status is set by Reset, and when an Underrun occurs in a Synchronous mode, that is, when software or a DMA channel has not written a character to the channel's Data address by the time one is needed, inside a frame or message. Transmit Underrun status is cleared only by writing the Reset Transmit Underrun command to WR0, as described above.

NOTE: In Synchronous modes the transmitter behaves differently when an Underrun occurs, depending on whether software has cleared the Transmit Underrun status. If so, it sends the CRC before sending the Sync or Flag. If not, it sends a Sync or Flag.

RR0 bit 7 displays the Break detection status in Async mode, or Abort detection status in HDLC/SDLC mode, the last time status was latched or unlatched. Behind the latch, Break detection is set by an all-0character with a Framing Error, and cleared when the channel's RXD pin returns to 1. Break detection can also be set when Abort detection is set by seven consecutive 1s inside a frame, and is cleared when a 0 is received.

RR1. This register contains Special Receive Condition status and the HDLC/SDLC Residue value.

In HDLC/SDLC mode, a 1 in RR1 bit 7 indicates that a closing Flag has been received. It can be cleared by writing an ERROR RESET command to WR0, and is automatically cleared when the first character of the next frame is received.

A 1 in RR1 bit 6 indicates a Framing Error in Async mode or a CRC error in other modes. Framing Error accompanies the character in which the error was detected. As CRC error, this bit is meaningful only when bit 7 is 1. In either meaning, the bit is not latched and is updated after a character is read and the next character has been received.

A 1 in RR1 bit 5 indicates a Receive Overrun condition. Overrun occurs if the Receive FIFO contains 3 characters and another character has been assembled in the receive shift register. This error bit accompanies the character that overwrote the previous character, which was lost. But even if the character is read, this bit is latched, and maintains the value of 1 until software writes an Error Reset command to WR0. If receive data interrupts are enabled and Status Affects Vector operation is enabled because WR1 bit 2 is 1, the channel returns the Special Receive Condition vector for a character with an Overrun error.

RR1 bit 4 indicates a Parity Error. It is 1 if parity checking is enabled because WR4 bit 0 is 1, and the character parity did not match the type selected by WR4 bit 1. This bit is latched, and stays 1 until software writes an Error Reset command to WR0.

For HDLC/SDLC reception, when RR1 bit 7 is 1, RR1 bits 3-1 indicate the number of data bits in the last few characters presented by the receiver for the frame. Figures 18 through illustrate the relationship between values of this field and the contents of these characters, for 8 through 5 bits/character respectively.

In Async modes, RR1 bit 0 is set when all transmit data written to the channel's data address has been sent. It is always 1 in Synchronous modes.

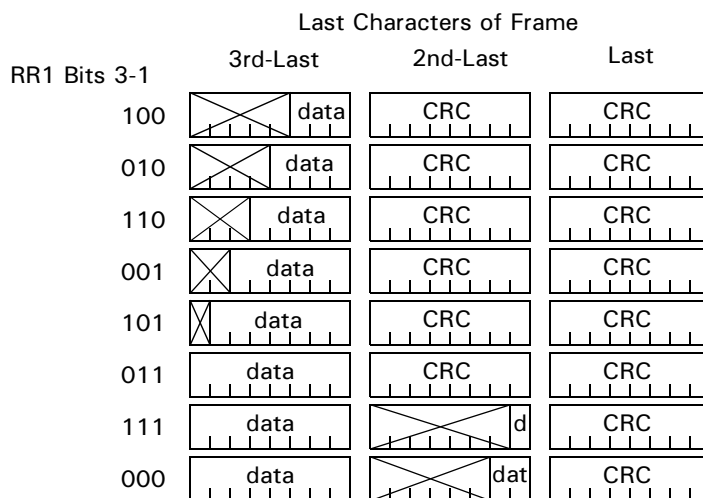


FIGURE 18. RESIDUE VALUES VS. LAST CHARACTERS OF FRAME (8 BITS/CHAR)

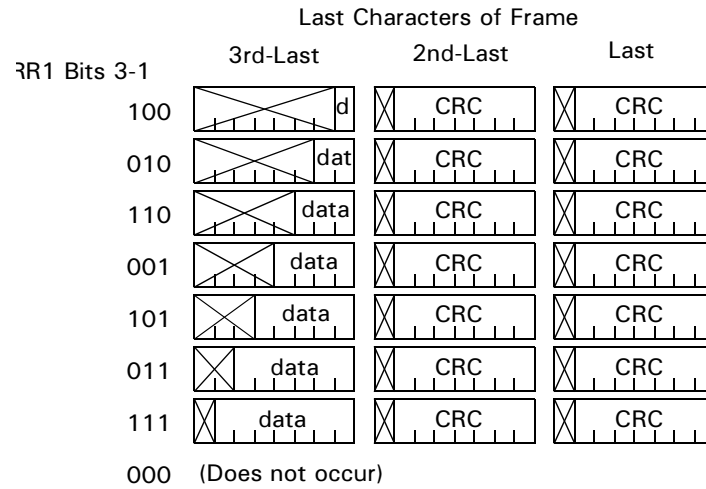


FIGURE 19. RESIDUE VALUES VS. LAST CHARACTERS OF FRAME (7 BITS/CHAR)

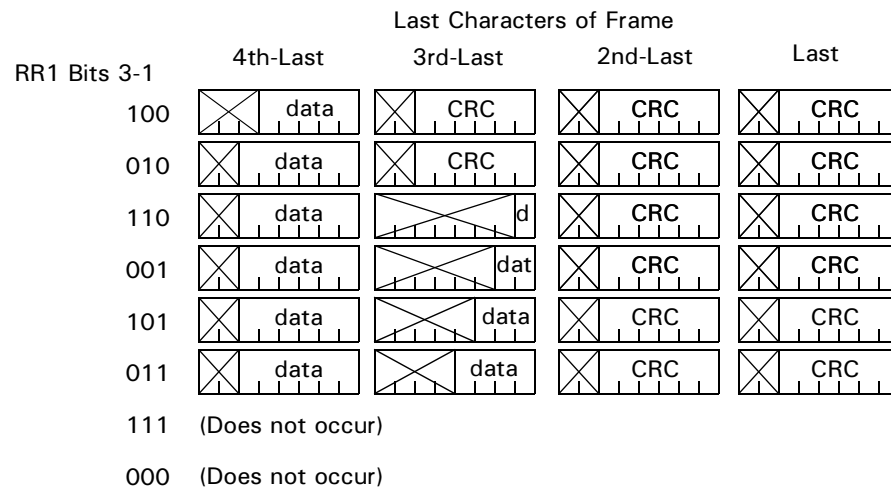


FIGURE 20. RESIDUE VALUES VS. LAST CHARACTERS OF FRAME (6 BITS/CHAR)

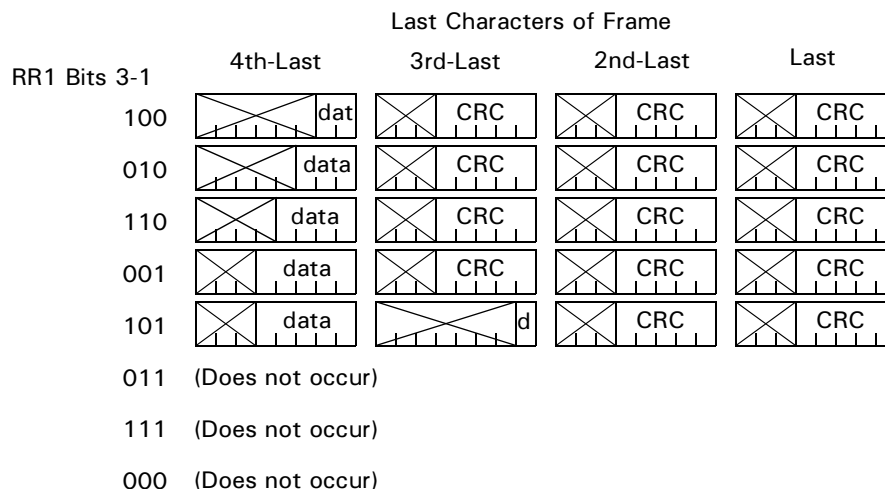


FIGURE 21. RESIDUE VALUES VS. LAST CHARACTERS OF FRAME (5 BITS/CHAR)

RR2. This register can be read only from channel B, and returns the interrupt vector that an interrupt acknowledge cycle returns at the same moment. If channel B's WR1 bit 2 is 1 to select Status Affects Vector operation, bits 7-4 and 0 are returned as software write them to WR2, while bits 3-1 contain a code for the highest-priority condition for which an interrupt is currently pending, or 011 if no interrupts are currently pending. Table 20 describes the codes. If WR1 bit 2 is 0, reading RR2 returns exactly what the software last wrote to WR2.

TABLE 20. STATUS AFFECTS VECTOR CODES IN RR2 BITS 3-1

RR2 Bits 3-1 (if WR1 Bit 2 = 1) Highest Priority Pending Interrupt	
000	Channel B Transmit Buffer Empty (lowest priority)
001	Channel B External/Status Change
010	Channel B Receive Character Available
011	Channel B Special Receive Condition (or no interrupt pending)
100	Channel A Transmit Buffer Empty
101	Channel A External/Status Change
110	Channel A Receive Character Available
111	Channel A Special Receive Condition (highest priority)

SIO Interrupts

The SIO, CTC, and PIO modules logically OR their interrupt requests with those from any external peripherals that are connected to the INT0 pin. Three external peripherals must drive $\overline{\text{INT0}}$ Low in an open-collector or open-drain fashion. The Z80S188 drives INT0 Low in an open-drain fashion, when any of the SIO or CTC channels or PIO ports are requesting an interrupt.



Each SIO channel includes three modules that can request an interrupt: the Receiver, Transmitter, and External/Status conditions. Each of these 6 *interrupt types* include state bits called Interrupt Pending (IP) and Interrupt Under Service (IUS). All six types share the same base interrupt vector, but if bit 2 of WR1 is 1, the SIO returns a code identifying the highest priority SIO type requesting an interrupt in bits 3-1 of the interrupt vector.

Interrupt Priority Daisy Chaining. Because more than one channel, port, or interrupt type can request an interrupt at the same time, a mechanism is needed to select the order in which their requests are serviced. Z80 peripherals and certain other ZiLOG devices use a daisy chain to control the relative priority of interrupt requests among multiple devices and interrupt types within devices.

As described in “Interrupt Acknowledge Daisy Chaining”, on page 18, each SIO interrupt type has an Interrupt Enable In (IEI) pin, from which it receives permission to interrupt from higher-priority devices, and an Interrupt Enable Out (IEO) pin, on which it grants permission to interrupt, to lower-priority devices.

The daisy-chain order and priority within each Z80-device-equivalent module is fixed. For the SIO, Channel A’s Receiver has the highest priority, followed by Transmitter A, External/Status A, Receiver B, Transmitter B, and External/Status B.

The SIO, CTC, and PIOs in the Z80S188 are always consecutive on the daisy chain, but the relative priority among them is programmable in the Interrupt Priority register.

Please see “Interrupt Acknowledge Daisy Chaining”, on page 18 for more information about interrupt daisy-chaining.

Initializing for Async Operation

1. Write a Channel Reset command, 08H, to WR0
2. Write the desired interrupt vector to channel B’s WR2, if the application enables any interrupts from either SIO channel, and previous initialization has not yet been implemented.
3. Write binary cc00sspp to WR4:
 - a. cc are typically 01 for /16 clocking, but can be 00 for 1X isochronous, 10 for /32, or 11 for /64 clocking. In isochronous operation, data on Rx/D need not be synchronous to the clock on Rx/C, as in Sync modes.
 - b. ss can be 01 to send 1 Stop bit, 10 for 1.5, or 11 to send 2 Stop bits.
 - c. pp are typically 00, but can be 10 for odd parity generation and checking, or 11 for even parity.

4. Write binary `nne00001` to WR3:
 - a. `nn` is typically 11 to assemble 8 data bits per character, but can be 10 for 7, 01 for 6, and 00 for 5 bits/character. This number does not include the parity bit if the previous step enabled parity.
 - b. `e` is typically 0, but can be 1 if the signal on the DCD pin auto-enables the receiver, and the signal on CTS *autoenables* the transmitter.
5. Write binary `dnn010r0` to WR5:
 - a. `d` controls the DTR pin, 0 for High, 1 for Low.
 - b. `nn` is typically 11 to send 8 bits/character, but can be 10 for 7, 01 for 6, and 00 for 5 bits/character. This number does not include the parity bit if parity is enabled in WR4.
 - c. `r` controls the RTS pin, 0 for High, 1 for Low.
6. Write binary `wwiiste` to WR1:
 - a. `www` are typically 000; otherwise they select the function of the WAIT/RDY pin as described on page 76.
 - b. `ii` are typically 00 for polled reception or 11 for interrupt-driven reception. 01 selects a receive interrupt on the first received character. 10 selects interrupt-driven reception in which a character with a parity error is handled via the Special Receive condition ISR.
 - c. `s` is typically 1 to select status affects vector operation, in which the SIO returns one of 8 consecutive vectors depending on the highest priority interrupting condition. If `s` is 0, the SIO returns the vector in WR2 for any interrupt on either channel.
 - d. A 1 in `t` enables transmit interrupts
 - e. A 1 in `e` enables external/status interrupts

Initializing for Classic Sync Operation

1. Write a Channel Reset command, 08H, to WR0.
2. Write the desired interrupt vector to channel B's WR2 if the application enables any interrupts from either SIO channel, and previous initialization has not yet been implemented.
3. Write binary `00mm00pp` to WR4:
 - a. `mm` are 00 to select an 8-bit Sync pattern (monosync), 01 to select a 16-bit Sync pattern (Bisync), or 11 to select external Sync detection.
 - b. `pp` are typically 00, but can be 10 for odd parity generation and checking, or 11 for even parity.
4. Write binary `dnn00cr0` to WR5:
 - a. `d` controls the DTR pin, 0 for High, 1 for Low.
 - b. `nn` is typically 11 to send 8 bits/character, but can be 10 for 7, 01 for 6, and 00 for 5 bits/character. This number does not include the parity bit if parity is enabled in WR4.



- c. *c* is 0 to use the CRC/CCITT polynomial, 1 for CRC-16.
 - d. *r* controls the RTS pin, 0 for High, 1 for Low.
5. Write binary *nne10010* to WR3:
 - a. *nn* is typically 11 to assemble 8 data bits per character, but can be 10 for 7, 01 for 6, and 00 for 5 bits/character. This number does not include the parity bit if parity is enabled in WR4.
 - b. *e* is typically 0, but can be 1 if the DCD pin auto-enables the receiver, and the signal on CTS *autoenables* the transmitter.
 6. Write the Sync character to WR6 for monosync operation. For Bisync, write the first Sync character to WR6 and the second Sync character to WR7.
 7. Write binary *wwiiste* to WR1:
 - a. *www* are typically 000; otherwise they select the function of the WAIT/RDY pin as described on page 76.
 - b. *ii* are typically 00 for polled reception or 11 for interrupt-driven reception. 01 selects a receive interrupt on the first received character. 10 selects interrupt-driven operation in which a character with a parity error is handled via the Special Receive condition ISR.
 - c. *s* is typically 1 to select *status affects vector* operation, in which the SIO returns one of 8 consecutive vectors depending on the highest priority interrupting condition. If *s* is 0, the SIO returns the vector in WR2 for any interrupt on either channel.
 - d. A 1 in *t* enables transmit interrupts.
 - e. A 1 in *e* enables external/status interrupts.
 8. Write the same value as in step 4 to WR5, but with bit 3 (08H) set to enable the transmitter. Write the value from step 5 to WR3, but with bit 0 set to enable the receiver.
 9. Write a Reset Rx CRC Checker command to WR0.

Initializing for HDLC/SDLC Operation

1. Write a Channel Reset command, 08H, to WR0
2. Write the desired interrupt vector to channel B's WR2 if the application enables any interrupts from either SIO channel, and previous initialization has not yet been implemented.
3. Write binary 001000*pp* to WR4. *pp* are typically 00, but can be 10 for odd parity generation and checking, or 11 for even parity.

4. Write binary `dnn000r1` to WR5:
 - a. `d` controls the DTR pin, 0 for High, 1 for Low.
 - b. `nn` is typically 11 to send 8 bits/character, but can be 10 for 7, 01 for 6, and 00 for 5 bits/character. This number does not include the parity bit if parity is enabled in WR4.
 - c. `r` controls the RTS pin, 0 for High, 1 for Low.
5. Write binary `nne10a00` to WR3:
 - a. `nn` is typically 11 to assemble 8 data bits per character, but can be 10 for 7, 01 for 6, and 00 for 5 bits/character. This number does not include the parity bit if parity is enabled in WR4.
 - b. `e` is typically 0, but can be 1 if the DCD pin auto-enables the receiver, and CTS autoenables the transmitter.
 - c. `a` is 1 if the channel matches the first character of each received frame against the character in WR6, and ignores non-matching frames.
6. Write the match character to WR6.
7. Write 7EH (the Flag pattern) to WR7.
8. Write binary `wwiiste` to WR1:
 - a. `www` are typically 000; otherwise, they select the function of the WAIT/RDY pin as described on page 76.
 - b. `ii` are typically 00 for polled reception or 11 for interrupt-driven reception. 01 selects a receive interrupt on the first received character. 10 selects interrupt-driven operation in which a character with a parity error is handled via the Special Receive condition ISR.
 - c. `s` is typically 1 to select *status affects vector* operation, in which the SIO returns one of 8 consecutive vectors depending on the highest priority interrupting condition. If `s` is 0, the SIO returns the vector in WR2 for any interrupt on either channel.
 - d. A 1 in `t` enables transmit interrupts
 - e. A 1 in `e` enables external/status interrupts
9. Write the same value as in step 4 to WR5, except with bit 3 (08H) set to enable the transmitter. Write the value from step 5 to WR3, except with bit 0 set to enable the receiver.
10. Write a Reset Tx CRC Generator command (80H) to WR0.

Polled Transmission

1. If there is data to send, software reads RR0 from the channel's Control address periodically, and proceeds to the next step if bit 2, Tx Empty, is 1.
2. In HDLC/SDLC mode, if frames have a required format and the next character is not the first one of a frame, read RR0 and check the Tx Underrun/EOM bit. If it is 1, an underrun condition has occurred inside of a frame. Write a Send Abort command to WR0, and return to step 1.



3. In CLASSIC SYNC mode only, if the next character is the first included in the CRC for the message, write a Reset Tx CRC Generator command to WR0.
4. In CLASSIC SYNC mode only, determine if the next character is included in the CRC. Write WR5 (if necessary) with bit 0 set to 1 for inclusion or 0 to exclude.
5. Fetch the next character from memory and write it to the channel's data address.
6. In HDLC mode only, if the character just written is the first one of a new frame, write a Reset Tx Underrun/EOM command to WR0.
7. In CLASSIC SYNC mode only, if messages have a *required format* and the character just written was the *last* one of a message, or if messages can be of *any length* and serve only to convey a byte stream with its own internal structure, and the character just written was the *first* one of a message, write a Reset Tx Underrun/EOM command to WR0 and set an internal flag indicating that this operation was performed.
8. Return to step 1.

Polled Reception

1. Read RR0 from the channel's Control address periodically. Proceed to the next step if either bit 0 or bit 7, Rx Available or Break/Abort, is 1.
2. In Async or HDLC/SDLC modes only, if RR0 bit 7, Break/Abort, is 1, write a Reset External/Status command to WR0 to unlatch the status in RR0, then read RR0 periodically until RR0 bit 7 returns to 0, which indicates that the Break or Abort sequence is over. At that time, write another Reset External/Status command to WR0 to unlatch the RR0 status again.

In HDLC/SDLC mode, an Abort sequence while a frame is in progress indicates that software discards that frame, then returns to step 1 to wait for the first character of the next frame.

In Async mode on a full duplex link, a Break traditionally indicates that the station receiving it stops sending. If a Break follows a character that contained a Framing Error, the Break indicates that sender started the Break while that character was being sent. When a BREAK is over, bit 0 of RR0, Rx Available, is set, and software reads the extraneous all-0 character from the Data address, then discards it, and returns to step 1.

3. For Rx Available without Break/Abort, software reads the status associated with the character from RR1, then saves it, and reads the character from the channel's Data address.
4. If parity checking is enabled, handling of a character with a parity error is application- and protocol-dependent, except that software must write an Error Reset command to WR0 to clear the parity error bit.

5. If the RR1 status shows an overrun condition, handling of preceding data is application- and protocol-dependent. One character was lost before the character flagged with the overrun status; others may have been lost subsequently. Software must write an **Error Reset** command to WR0 to clear the overrun bit. In HDLC/SDLC and CLASSIC SYNC modes, software typically discards/ignores the frame/message in progress, then writes a value to WR3 that includes a 1 in bit 4, which forces the receiver into HUNT mode, then returns to step 1.
6. In ASYNC mode, software checks the Framing Error bit in RR1, but the handling of a character with a framing error is application-dependent.
7. Except for certain kinds of error characters in certain protocols, software stores the received character in memory, advances the buffer address, and if the current buffer is filled, advances the pointer to the next buffer.
8. In CLASSIC SYNC mode, if the Strip Sync bit was set in WR3, software rewrites WR3 to clear this bit.
9. In CLASSIC SYNC mode with CRC checking, software examines the character to determine if it is included in CRC checking, and if necessary, write to WR3 so that bit 3 is 1 to include or 0 to exclude the character. (Sync characters within a message are excluded.)
10. In Classic Sync mode, software must examine whether each character is a message-terminator. When a message-terminator is detected, if CRC checking is used and the Rx character length is less than 8 bits, software rewrites WR3 for 8 bits/character, at least for the duration of the CRC.
 - a. For CRC checking, software waits for four more Rx Available flags (RR0 bit 0). The first two flags are associated with the CRC characters themselves; the third is typically a *Pad* or *Sync* character. These three characters can be read and discarded. When the fourth Rx Available flag is set, software reads RR1 and checks bit 6 to determine the CRC correctness of the message.
 - b. Software writes a **Reset Rx CRC Checker** command to WR0. Depending on the protocol, software reads the next character and examines it for a Sync character or the start of a new frame. Software may then discard the character, exclude it from the Rx CRC, and write to WR3 to force the receiver back into Hunt mode for a new Sync sequence.
11. In HDLC/SDLC mode, software checks bit 7 of RR1 (EOF) to detect when the last character of the CRC was read. When RR1 bit 7 is 1, the accompanying bit 6 indicates CRC correctness, and the accompanying bits 3-1 indicate the frame length.
12. Unless indicated otherwise in a step above, after processing each character, the software returns to step 1.

Interrupt-Driven Transmission

The following steps assume the processor is in Interrupt mode 2 and that bit 2 of WR1 is 1 to select Status Affects Vector mode. Otherwise, when an interrupt



occurs, software must read RR0 and possibly RR1 of each SIO channel, to determine the cause of each interrupt.

1. When the SIO returns an interrupt vector, resulting in execution of an SIO Tx Interrupt Service Routine (ISR), the SIO saves as many registers that it may use (worst case), using PUSH or EX AF, AF' and EXX instructions.
2. Next, the Tx ISR retrieves the transmit context. If this interrupt follows CRC transmission in a Synchronous mode, and another frame or message is not to follow immediately, proceed to step 4.
3. Otherwise, the ISR proceeds as in steps 2-7 of "Polled Transmission", on page 88. If the transmit context was not the end of a frame or message in a Synchronous mode, or if there is more data to send in Async mode, proceed to step 5.
4. Write a Reset Transmitter Interrupt Pending command to WR0. If a CRC is sent in Synchronous modes, the next Tx interrupt cannot occur until after the CRC has been sent. Otherwise software must write the next Tx character to the channel's Data address at mainline level, before another Tx interrupt can occur.
5. The ISR concludes by restoring the saved registers and executing EI and RETI instructions. The transmitter detects the RETI and re-enables interrupts from itself and lower-priority devices.

Interrupt-Driven Reception

The following steps assume the processor is in Interrupt mode 2 and that bit 2 of WR1 is 1 to select Status Affects Vector mode. Otherwise, when an interrupt occurs, software must read RR0 and possibly RR1 of each SIO channel, to determine the cause of each interrupt.

1. When the SIO returns an interrupt vector, resulting in execution of an SIO Rx Interrupt Service Routine (ISR), it saves as many registers that it may use (worst case), using PUSH or EX AF, AF' and EXX instructions.
2. The next received character is read and stored in memory. The buffer pointer advances, and if the current buffer is full, the pointer advances to the next buffer.
3. In CLASSIC SYNC mode, proceed as in steps 8-10 of "Polled Reception", on page 89.
4. Because all exceptional receive conditions, including EOM in HDLC mode, result in External/Status interrupts or Special Receive interrupts rather than normal Rx interrupts, the ISR concludes by restoring the saved registers and executing EI and RETI instructions. The receiver detects the RETI and re-enables interrupts from itself and lower-priority devices.

DMA Transmission

This section assumes that in Synchronous modes, each frame or message to be sent occupies a single contiguous buffer in memory. Otherwise, software processing is more complex.

NOTE: CLASSIC SYNC transmission can only be handled by DMA if all of the characters in a DMA buffer are included in the CRC. Otherwise the message is sent using polling or interrupt-driven techniques.

1. Ensure that WR1 bit 1 is 0 to disable Tx interrupts.
2. In CLASSIC SYNC mode only, write a Reset Tx CRC Generator command to WR0, and ensure that WR5 bit 0 is 1 to include all characters in the CRC (see note above).
3. Two of the four SIO transmitters and receivers can be handled by the Z80S188's DMA channels at one time. If these assignments are not fixed or were not set up during device initialization:
 - a. Write 100 to DAR0B bits 2-0, and 11 to DMODE bits 5-4, to handle SIO channel A Tx using DMA0.
 - b. Write 101 to DAR0B bits 2-0, and 11 to DMODE bits 5-4, to handle SIO channel B Tx using DMA0.
 - c. Write 100 to IAR1B bits 2-0, and 0 to DCNTL bit 1, to handle SIO channel A Tx using DMA1.
 - d. Write 101 to IAR1B bits 2-0, and 0 to DCNTL bit 1, to handle SIO channel B Tx using DMA1.
4. Load DAR0L,H or IAR1L,H with 00D8H for channel A or 00DAH for channel B. Program the other DMA registers as described in "Setting Up a DMA Transfer", on page 59. When the DMA channel is enabled and the SIO Tx requests data, the DMA begins providing data to the SIO Tx.
5. In Sync modes, if the Tx is auto-enabled, wait to be sure that the transfer had begun.
6. In Sync modes, write a Reset Tx Underrun/EOM command to WR0.
7. When the DMA channel interrupts or when software polls bit 7 or 6 of the DSTAT register as 0, all the data has been transferred to the SIO transmitter.

DMA Reception

In general, CLASSIC SYNC reception cannot be handled by DMA because some received characters (such as embedded Syncs) must be excluded from CRC checking, and each received character must be checked for message termination.

Only very straightforward CLASSIC SYNC reception, in which message lengths are fixed and all characters in every message are included in CRC checking, can be handled by DMA.

1. Ensure that WR1 bits 4-3 are 00 to disable Rx interrupts. WR1 bit 0 can be 1 to enable External/Status interrupts, or 0 if software polls RR0 and RR1.



2. Two of the four SIO transmitters and receivers can be handled by the Z80S188's DMA channels at one time. If these assignments are not fixed or were not set up during device initialization:
 - a. Write 100 to SAR0B bits 2-0, and 11 to DMODE bits 3-2, to handle SIO channel A Rx using DMA0.
 - b. Write 101 to SAR0B bits 2-0, and 11 to DMODE bits 3-2, to handle SIO channel B Rx using DMA0.
 - c. Write 100 to IAR1B bits 2-0, and 1 to DCNTL bit 1, to handle SIO channel A Rx using DMA1.
 - d. Write 101 to IAR1B bits 2-0, and 1 to DCNTL bit 1, to handle SIO channel B Rx using DMA1.
3. In Classic Sync mode only, write a Reset Rx CRC Generator command to WR0, ensure that WR3 bit 3 is 1 to include all characters in the CRC and set WR3 bit 1 to strip initial Syncs.
4. Load SAR0L, H or IAR1L, H with 00DH for channel A or 00DAH for channel B. In Classic Sync mode, load the BCR with the fixed message length plus 3, to allow for the CRC and the subsequent *Pad* or *Sync* character. Program the other DMA registers as described in "Setting Up a DMA Transfer", on page 59. When the DMA channel is enabled and the SIO Rx receives a character, the DMA begins storing SIO Rx data.
5. If External/Status interrupts are not enabled, software must periodically poll RR0 and RR1 to watch for conditions like BREAK, ABORT, DCD change, and EOF.
6. When the DMA channel interrupts or when software detects bit 7 or 6 of the DSTAT register as 0, the DMA has filled the buffer with received data. If more Rx data is expected, software returns to step 3 or 4 to set up a new DMA buffer.

Handling External/Status Interrupts

The following steps assume the processor is in interrupt mode 2 and that bit 2 of WR1 is 1 to select *status affects vector* mode. Otherwise, when an interrupt occurs, software reads RR0 and possibly RR1 of each SIO channel, to determine the cause of each interrupt.

1. When the SIO returns an interrupt vector resulting in execution of an SIO External/Status Interrupt Service Routine (ISR), the SIO saves as many registers that it may use, using PUSH or EX AF,AF' and EXX instructions.
2. Software then retrieves the SIO context, including the last value read from RR0.
3. Next, software reads RR0, save this value, and then performs an XOR instruction on the current RR0 value and the previous value. Differences (1s) among bits 7-3 of this result identify the reason(s) for the interrupt. A 1-to-0 transition of bit 6 does not produce an interrupt.

4. Before or after handling these events, software writes a Reset External/Status Interrupts command to WR0. This action unlatches bits 7–3 of RR0, and allows future External/Status interrupts.
5. The ISR concludes by restoring the saved register values, followed by EI and RETI instructions. The External/Status logic detects the RETI and re-enables interrupts from itself and from lower-priority devices.

ASYNC SERIAL COMMUNICATIONS INTERFACES (ASCIs)

The ASCIs are asynchronous full-duplex UARTs with the following features:

- 7- or 8-Bit data
- Odd, even, or no parity
- 1 or 2 Tx stop bits
- Checking for parity, framing, and overrun errors
- Break generation and detection
- A choice of two baud rate generators
- Rx and Tx interrupts
- Input or output clocking on CKA0-1
- DCD and CTS on ASCI0
- Operation with the on-chip DMA channels, and
- A multiprocessor mode with an extra bit designating *address* vs. *data* characters.

The registers associated with the ASCIs are described in section “Async Serial Communications Interface (ASCI) Registers”, on page 164. Control registers A and B (CNTLA0, CNTLA1, CNTLB0, and CNTLB1) and the Extension Control registers (ASEXT0, ASEXT1) are typically written once each, to configure an ASCI. The Status registers (STAT0 and STAT1) indicate the current state of each ASCI’s Receiver and Transmitter. Under control of this status, software can write bytes to be transmitted to the Transmit Data Registers (TDR0 and TDR1), and can read received bytes from the Receiver Data Registers (RDR0 and RDR1).

Basic Clocking

Each ASCI uses the same basic clock for both transmitting and receiving. If bits 2–0 of an ASCI’s CNTLB register are 111, as they are after a Reset, the basic clock is taken from the CKA0 or CKA1 pin, and neither Baud Rate Generator is used.

To use the *old* BRG, which is compatible with the original ZiLOG Z80180, software must:

1. Clear bit 3 (BRG mode) of the Extension Control register to 0, to select the old BRG.
2. Write bit 5 (PS) and bits 2–0 (SS) in the CNTLB register to select what value the *old* BRG divides the PHI clock by, to obtain the *basic clock* for this ASCI,



as indicated in Table 21. This basic clock is then driven onto the CKA0 or CKA1 pin.

TABLE 21. OLD BRG DIVISION FACTORS

PS (CNTLB5)	SS (CNTLB2-0)	Basic Clock is PHI Divided By
0	000	10
	001	20
	010	40
	011	80
	100	160
	101	320
	110	640
	111	No clock
1	000	30
	001	60
	010	120
	011	240
	100	480
	101	960
	110	1920
	111	No clock

To use the new BRG, which is compatible with the ZiLOG SCC family, software must:

1. Set bit 3 (BRG mode) of the Extension Control register to 1, to select the new BRG.
2. Write a 16-bit binary value to the Time Constant Low and High registers. This value is the factor by which PHI is divided to produce the basic clock, divided by two, minus two. The new BRG derives the basic clock as:

$$\text{basic clock} = \text{PHI} / 2 (\text{TC} + 2)$$

This basic clock is then driven onto the CKA0 or CKA1 pin.

Clock Mode. If bit 4 (X1 clock) in an ASCI's Extension Control Register is 1, the basic clock on CKA0 or CKA1 is used directly as a 1X *isochronous* bit clock, which must be synchronized to the data on RXA0 or RXA1. This mode can be used in either of two ways:

- With CKA0 or CKA1 as an input, bearing a clock from the remote transmitter, that is synchronous to the data on this ASCI's RXA pin.
- With CKA0 or CKA1 as an output, bearing the clock to the remote transmitter, which uses it to output data on this ASCI's RXA pin.

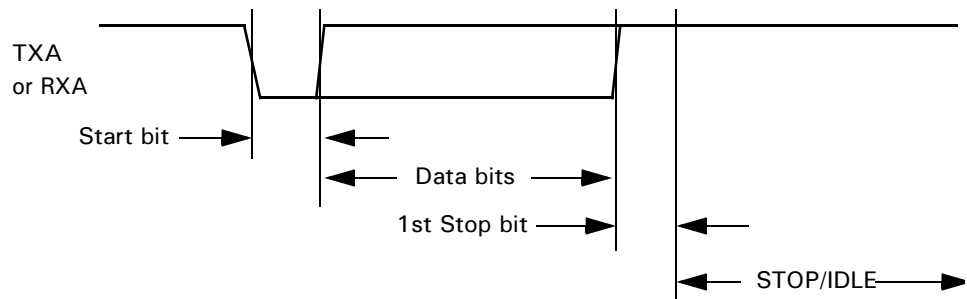
If bit 4 (X1 clock) in an ASCI's Extension Control Register is 0, and the DR bit (bit 3) in the CNTLB register is 0, the ASCI divides its basic clock by 16 to obtain

the serial bit rate. If X1 clock is 0 and DR is 1, the ASCI divides its basic clock by 64.

Async Transmission

Figure 22 shows a single asynchronous character on the TXA or RXA pin. When a Transmitter is disabled, or when it has completed sending any and all characters that software or a DMA channel has provided, it maintains a High level on the TXD pin. This state is also called 1 or Mark.

FIGURE 22. AN ASYNCHRONOUS CHARACTER



When software or a DMA channel provides a character to an idle ASCI transmitter, it

1. Drives TXA low to start a start bit, on the next falling edge of the basic clock, and maintains TXA Low for 1, 16, or 64 basic clocks depending on the X1 clock and DR bits.
2. Switches TXA to the state of the least significant bit (bit 0) of the character, and holds that for 1, 16, or 64 clocks, and so on through the most significant bit (bit 6 or 7) of the character, and for a parity or MP bit if one of these is enabled.
3. Switches TXA to High (1, Mark) again for the *first Stop bit*, and maintains it High for at least 1, 16, or 64 clocks, plus another 1, 16, or 64 clocks if bit 0 of its CNTLA register is 1 to select *send two Stop bits*.

The Tx character is now complete. If software or a DMA channel has provided another character to send, the transmitter starts another start bit as described above. Otherwise, it keeps TXA High until a character is provided.

Async Reception

The Receiver has a more complex task. When it is first enabled, or after a character has been received:

1. The receiver samples the RXA pin on the rising edge of each basic clock is sampled. If RXA is High it remains in the same state of waiting for a start bit. If it samples RXA Low, and the X1 clock bit is 0, the receiver counts off half a bit time, 8 basic clocks if DR is 0, or 32 basic clocks if DR is 1. The receiver samples RXA again.



2. If RXA is now High, the receiver rejects the transient Low state on RXA as not representing a Start bit, and returns to step 1. If the X1 clock bit is 1, the receiver skips this step.
3. The receiver then counts off the number of basic clocks in a bit, 1, 16, or 64 depending on the X1 clock and DR bits, and samples RXA for the least significant data bit (bit 0). This operation continues for each More Significant Data bit (through bit 6 or 7), and optionally for a Parity or MP bit if these are enabled.
4. Finally, the receiver counts off 1, 16, or 64 more basic clocks and samples the first Stop bit. When X1 clock is 0, if the receiver's basic clock is close to the clock that the transmitter used to send the character, (near the middle of the first Stop bit), and if there is no noise or other error on the line, the receiver samples the Stop bit as High/1/Mark.

If X1 clock is 0, and the receiver's and transmitter's clocks were sufficiently different, or if there is noise or contention on the line, the receiver samples a Low/0/Space. This latter situation is called a *framing error*; when it occurs, the receiver sets an error bit that accompanies the character through the receiver FIFO, and sets the FE bit in the STAT register when the character becomes the oldest in the FIFO.

The received character is now complete, and the receiver returns to sample the line for a new Start bit, as described above.

Clocking Summary

The *Sampling Rate* can be 1, 16, or 64 depending on the X1 bit in the ASEXT register and the DR bit in the CNTLA register:

X1	DR	Sampling Rate
0	0	16
0	1	64
1	0	1
1	1	Reserved, do not program.

The following notations apply to the equations below, where:

bits/second is the serial rate

f_{PHI} is the system clock frequency

PS is the value written to bit 5 of CNTLB (0 or 1)

^ indicates exponentiation (2 to the power)

SS2-0 is the binary value of bits 2-0 of CNTLB (0 thru 6)

DR is bit 3 of CNTLB (0 or 1)

TC is the 16-bit value in the ASTCL and ASTCH registers

If the SS2–0 bits in the CNTLA register are 111, the CKA pin is a clock input, which is divided by the sampling rate by the receiver and transmitter:

$$\text{bits/second} = f_{CKAin} / \text{Sampling Rate}$$

If the SS2–0 bits are not 111, and the BRG Mode bit in the ASEXT register is 0, the *old* baud rate generator divides PHI as follows for serial clocking.

$$\text{bits/second} = f_{PHI} / ((10 + 20 * PS) * 2^{SS2-0} * \text{Sampling Rate})$$

where PS selects between a *prescaler* of 10 and 30, and 2^{SS2-0} is a power of two between 1 and 64. The CKA pin outputs the clock before the final division by the Sampling Rate:

$$f_{CKAout} = f_{PHI} / ((10 + 20 * PS) * 2^{SS2-0})$$

If the SS2–0 bits are not 111, and the BRG mode bit is 1, the new baud rate generator divides PHI for serial clocking as on the ZiLOG SCC family:

$$\text{bits/second} = f_{PHI} / (2 * (TC + 2) * \text{Sampling Rate})$$

Where TC is the 16-bit value programmed into the TC high and low registers. The CKA pin outputs the clock before the final division by the Sampling Rate:

$$f_{CKAout} = f_{PHI} / (2 * (TC + 2))$$

To find the TC value for a particular serial bit rate:

$$TC = (f_{PHI} / (2 * \text{bits/second} * \text{Sampling Rate})) - 2$$

Options

7 or 8 Data Bits. If bit 2 of the CNTLA register is 0, the transmitter sends, and the receiver accumulates, 7 data bits per character. If CNTLA2 is 1, the transmitter sends, and the receiver accumulates, 8 data bits per character.

Parity. If bit 1 of the CNTLA register is 1, the transmitter accumulates and sends a Parity bit after the data bits, and the receiver samples and checks such a bit. If CNTLA1 is 1, a 1 in bit 4 of the CNTLB register selects odd parity and a 0 selects even parity. Odd parity defines a correct character that contains an odd number of 1 bits, through and including the Parity bit, while even parity defines a correct character that contains an even number of 1 bits.

If the receiver samples the Parity bit in the incorrect state for a character, an error bit is set that accompanies the character through the receiver FIFO, and sets the PE bit in the STAT register when the character becomes the oldest one in the FIFO.

Transmit Stop Bits. If bit 0 of CNTLA is 0, the transmitter sends a minimum of one Stop bit between characters. If CNTLA is 1, it sends at least two Stop bits between characters. Selecting two stop bits has been known to work around timing mismatches between a transmitter and a receiver.

NOTE: The ASCI receivers on the Z80S188 check only one *Stop* bit, regardless of bit 0 of CNTLA. This is also true of ASCI receivers on other current 8018x family



members, as well as other UARTs, but on the original 80180, the ASCIs actually checked two *Stop* bits if CNTLA0 was 1.

Status

Break Conditions. Break conditions date back to the early days of Async communications using Teletypewriters, which were functionally half-duplex in that text could only flow in one direction at a time. If the operator of a receiving TTY) had a problem, or something to say, and wanted to interrupt the data from the other machine, there was a Break key that could be pressed. This drove the line to 0/Space state for several character times, which served to turn a light on at the other machine and stopped its paper tape reader if it was in use.

A Break is still defined as at least two character times of consecutive 0s on the line. A receiver detects this as one or more all-0 character(s) with Framing error(s).

Software can make a Z80S188 ASCI send a Break by writing a 1 to bit 0 of the Extension Control register. Software can ensure that any characters previously written to the Transmit Data register are sent (by monitoring bit 0 of the Extension Control register), before it writes a 1 to the Tx Break bit. The duration of a transmitted Break is completely under software control—the Break is terminated by writing a 0 to ASEXTEXT bit 0.

While receiving an all-0 character with a Framing error is definitive evidence of a Break, bit 1 of the Extension Control register provides more specific break detection. When the receiver detects an all-0 character with a framing error, a Break status bit is set that accompanies the character through the receiver FIFO, and sets bit 1 in the ASEXTEXT register when the character becomes the oldest in the FIFO.

When this procedure occurs, the receiver does not assemble any further characters until the RXA line returns to High/1/Mark, signalling the end of the Break condition.

Rx Overrun. The ASCIs in the Z80S188 have 4-character Rx FIFOs between their RX SHIFT registers and the Receive data registers that software or a DMA channel can read. If the receiver finishes receiving a character when there are already 4 characters in the FIFO, an overrun status bit is set that accompanies the *preceding* character through the FIFO, and the OVRN bit (bit 6 in the STAT register) is set when that character becomes the oldest in the FIFO.

The receiver discards the character that triggers the overrun condition, *and subsequent characters*, until the last good character has come to the top of the FIFO so that OVRN is set. Software then writes a 0 to the EFR bit to clear it.

Receive Status. There are 4 receive status bits in the STAT register and one in the Extension Control register. RDRF, bit 7 of the STAT register, is set to 1 whenever there is at least one received character in the Rx FIFO. This bit is cleared when software or a DMA channel has read all characters out of the Rx FIFO, by Reset, in I/O STOP mode, and on ASCI0, when the $\overline{\text{DCD0}}$ pin is *autoenabled* and is High.

FE, PE, and OVRN in the STAT register are set to 1 when a character with a framing error or parity error, or the last character before an Rx Overrun, comes to the top of the receive FIFO. Similarly, bit 1 in the Extension Control register is set to 1 when a break character comes to the top of the FIFO.

Any of these bits stay 1 even if the character associated with the condition is read out of the receive FIFO, that is, they *latch* an error or exception condition. All four of these bits are cleared by reading CNTLA, clearing bit 3 (Error Flag Reset, EFR) to 0, and writing the result back to CNTLA. This action also allows the receiver to put subsequent characters into the receive FIFO after an Overrun, and if the receiver is being handled by a DMA channel, allows the channel to service the receiver once again.

Modem Control/Status. ASCII modem control or status signals on the Z80S188 are $\overline{\text{DCD}}$ and $\overline{\text{CTS}}$. The state of the $\overline{\text{DCD}}$ pin can be read as bit 2 of the STAT0 register, and that of $\overline{\text{CTS}}$ in bit 5 of CNTLB0. Both bits read as 1 if the pin is High/Inactive.

Bit 6 in Extension Control register 0 controls whether $\overline{\text{DCD}}$ automatically enables and disables the receiver, while bit 5 controls whether $\overline{\text{CTS}}$ automatically enables and disables the transmitter. In each case, if one of these ASEXT bits is 0, as it is after a Reset, then a Low on the pin allows the module to operate. A High disables the module. If an ASEXT bit is 1, software can still read the state of the corresponding pin, but this state does not affect the hardware.

When ASEXT0 bit 6 is 0, a High on $\overline{\text{DCD}}$ forces the status bits RDRF, PE, FE, OVRN, and RxBreak to 0. If $\overline{\text{DCD}}$ goes Low thereafter, the next read of the STAT register still indicates a 1 in bit 2 (DCD0). Subsequent reads of STAT indicate current status.

When ASEXT0 bit 5 is 0, a High on $\overline{\text{CTS}}$ clears the TDRE bit in STAT. This prevents software or a DMA channel from putting further Tx data into TDR, but as many as three characters (one from the Tx shift register, two from the 2-stage Tx FIFO) can still come out on TDA0 after $\overline{\text{CTS}}$ goes High.

DMA Operation

On the Z80S188, data can be transferred to or from either ASCII by either DMA channel. Setting up for such operation is covered in the following ASCII and DMA topics:

- “Starting a Transmitter”(below)
- “Starting a Receiver”, on page 101
- “Handling ASCII Interrupts”, on page 102
- “Setting Up a DMA Transfer”, on page 59.

A DMA channel used with an ASCII transmitter must be programmed to use the appropriate TDRE flag as its DMA request, and must be set up for *edge-sensitive DMA request*.

A DMA channel used with an ASCII receiver must be programmed to use the appropriate RDRF flag as its DMA request, and can be used in either edge- or level-sensitive mode. In this application, software must set both RIE in the STAT



register, and bit 7 of the Extension Control register to \bar{v} , to enable ASCI receive interrupts in case of errors, but not for each received character.

NOTE: The signal that an ASCI receiver provides as a Request to a DMA channel, is not simply RDRF but rather *RDRF and not PE and not FE and not OVRN and not RxBreak*. DMA operation is suspended if one of these errors or exceptional conditions occurs, so that software can determine the point in the receive data stream at which the error or condition occurred.

Programming Techniques

Starting a Transmitter. Software can enable a transmitter at the same time as its associated receiver, or separately, as follows:

1. Write the CNTLB register to set up the clocking and basic options.
2. Write the Extension Control register, if necessary, to select the X1 and BRG modes and, on ASCI0, the mode of the $\overline{\text{CTS0}}$ pin.
3. Set bit 0 (TIE) in the STAT register to 1 if Transmit Interrupts are desired, otherwise, clear it to 0.
4. Write the CNTLA register with a 1 in bit 5 to enable the transmitter, as well as other desired mode settings.

The TDRE flag is set immediately. If Transmit Interrupts were enabled in step 3, an interrupt occurs immediately.

NOTE: If a DMA channel provides data to the transmitter, it can be set up whenever transmit data is available, including selecting TDRE as its DMA Request. Edge-sensitivity is required on the DMA request for transmit/output devices. If TDRE is set before the DMA channel is started, device software must set up the DMA to not include the first Tx character. It then writes the first character to the TDR, to *prime* the ASCI-DMA handshake.

Starting a Receiver. Software can enable a receiver at the same time as its associated transmitter, or separately, as follows:

1. Write the CNTLB register to set up the clocking and basic options.
2. Write the Extension Control register, if necessary, to select the X1 and BRG modes and, on ASCI0, the mode of the $\overline{\text{DCD0}}$ pin. If the receiver is to be handled by a DMA channel, set bit 7 of the Extension Control register to block the RDRF flag from requesting a receive interrupt.
3. Set bit 3 (RIE) in the STAT register to 1 if receive interrupts are desired, else clear it to 0.
4. Set up the receiver, if the receiver is to be handled by a DMA channel, including selecting RDRF as the DMA Request signal.
5. Write the CNTLA register with a 1 in bit 6 to enable the receiver, as well as other desired mode settings.
6. RDRF is set when there is data in the Rx FIFO, and if RDRF interrupts are enabled, an interrupt occurs at that time.

Polled Transmission. When a transmitter is started without interrupts enabled, software must:

1. Read the STAT register (preferably more often than once per character time) until bit 1 (TDRE) set to 1 is detected.
2. Write the next character to be transmitted to the TDR. If there is more data to send, return to step 1.

Polled Reception. Once it has started a receiver without interrupt enabled, software must:

1. Read the STAT register, at least once per character time, until bit 7 (RDRF) set to 1 is detected.
2. Read a received character from the RDR.
3. Process that character, then return to step 1.

Handling ASCI Interrupts. As noted above, software can output to an ASCI in interrupt-driven mode by setting bit 0 (TIE) in the STAT register, and/or can input from an ASCI in interrupt-driven mode by setting bit 3 (RIE) in STAT. If any of following are true, an ASCI requests an interrupt from the processor.

- TIE and TDRE are both 1,
- RIE is 1 and any of OVRN, PE, FE, or RxBreak (ASEXT bit 1) are 1,
- RIE is 1, bit 7 of the Extension Control register is 0, and RDRF is 1, or
- For ASCI0, RIE is 1, bit 6 in the Extension Control register is 0, and the $\overline{\text{DCD}}$ pin is High.

When the conditions listed in “On-Chip Interrupt Handling”, on page 28 are met with respect to an ASCI’s interrupt request, the processor fetches the address of the Interrupt Service Routine (ISR) from (I : IL : 14) for ASCI0, or from (I : IL : 16) for ASCI1.

An ASCI ISR that handles both kinds of interrupts must:

1. Save as many registers as it may use, by means of PUSH, EX AF,AF’, and/or EXX instructions.
2. Read the STAT register for this ASCI.
3. If TIE and TDRE in STAT are both 1:
 - a. If Another Transmit character is available, write it to the TDR
 - b. If Not, clear the TIE bit until another Tx character is available.
4. If RIE in STAT is 1 and any of OVRN, PE, FE, or RxBreak (ASEXT bit 1) are 1:
 - a. If bit 7 of the Extension Control register is 1, indicating that a DMA channel is handling receive data, read the DSTAT register, clear the DE and DWE bits for that DMA channel, and write the result back to DSTAT.
 - b. Read the CNTLA register, clear bit 3 (EFR) and write the result back to CNTLA,



- c. Read the associated character from the RDR,
 - d. For PE or FE conditions, applications can choose to discard the character, replace it with a standard *error character*, or simply handle it like other characters,
 - e. For OVRN (without any other error), most applications process this *last good* character, as in step 5. An application may then post an *overrun occurred here* notification in the received data stream, or otherwise deal with the situation.
 - f. For Break, most applications discard the all-0 character. An application may post a *break occurred here* notification in the received data stream. The receiver does not assemble more all-0 characters, but waits for RDA to go High before searching for a new Start bit.
5. Read the next received character from the RDR, if no errors were found in step 4, but RIE is 1, bit 7 of the Extension Control register is 0, and RDRF is 1. Process that character, of which the simplest case is storing it at the next memory location in a buffer.
 6. If for ASCI0, RIE is 1, bit 6 in the Extension Control register is 0, and bit 2 of STAT is 1, carrier has been lost, and the ASCI receiver is inactive until it returns. In this case, software may either clear RIE to 0, or set bit 6 in the Extension Control register, to prevent further interrupts because DCD0 is High.
 7. Software can now read STAT again, and return to step 3 if either RIE and RDRF are both 1, or TIE and TDRE are both 1. (This routine saves interrupt overhead.)
 8. If the ISR disabled the receive DMA channel in step 4a, the ISR restarts the channel.
 9. Finally, the ISR restores the saved registers, and return to the interrupted process by means of EI and RET instructions.

Multiprocessor Mode

In this mode, the transmitter sends, and the receiver expects, an extra bit between the data and stop bits, that differentiates *address* from *data* characters. Other manufacturers' devices have a similar mode called *9-bit mode*. To enable this mode for both the transmitter and receiver, software sets bit 6 of the CNTLB register to 1 as part of initializing the ASCI.

NOTE: MULTIPROCESSOR mode cannot be used with parity generation and checking.

In Multiprocessor mode, data is grouped into *frames* or *messages*, each preceded by an *address* character, which differs from *data* characters in that the *extra* bit is 1.

To send a frame in MULTIPROCESSOR mode, software must:

1. Wait, if necessary, for the TDRE bit (bit 1 of the STAT register) to become 1, indicating that software can write a new Tx character to the TDR. The next two steps can be done in an ASCI Interrupt Service Routine.
2. Read the CNTLB register, set bits 7–6 to 11, and write the result back to CNTLB. This routine sets up the transmitter to send the first character of the frame with the extra bit 1.
3. Write the address value for the intended destination device to the TDR.
4. For each data character in the frame, again wait, if necessary for the TDRE bit (bit 1 of the STAT register) to be 1. The following two steps can be implemented in an ASCI ISR.
5. Read CNTLB, clear bit 7, and write the result back. This routine causes the transmitter to send the next character with the extra bit 0.
6. Write the next data character to the TDR. If there are more characters in the frame, return to step 4.

Assuming that bit 6 of CNTLB was set to 1 during ASCI initialization, to receive a frame in Multiprocessor mode software must:

1. Read CNTLA, set bit 7, and write the result back to CNTLA. This routine causes the receiver to ignore *data* characters, that have a 0 in their extra bit.
2. Wait for the RDRF bit in the STAT register to be 1. The following steps can be done in an ASCI interrupt service routine.
3. Read CNTLA and check that bit 3 is 1, verifying that the next available character is indeed an address character. If not, read the RDR, discard the data character, and return to step 2.
4. Read the RDR and check if the value obtained indicates a frame destined for this processor. (Some schemes use a unique address for each node, plus *broadcast* and/or *group* addresses.) If not, discard the character and return to step 2. The hardware ignores the data characters in the frame.
5. Store the address character in memory (this action is optional). The required actions may vary depending on the address.
6. Read CNTLA, clear bit 7, and write the result back to CNTLA. This routine causes the receiver assemble the following data characters.
7. Wait for the RDRF bit in the STAT register to be 1 for each data character in the frame. The following steps can be done in an ASCI interrupt service routine.
8. Read the data character from the RDR and store it in memory. If frames are not fixed-length, software determines the frame length from its content. If the frame is not complete, return to step 7.
9. Check for checking or validation information in the frame. Most protocols specify that a receiver ignores a frame that does not pass checking/validation.
10. Process the frame based on its content. This routine is application-dependent.



CLOCKED SERIAL INPUT/OUTPUT MODULE (CSI/O)

The CSI/O allows synchronous communication with serial memories, peripherals, and other processors that include compatible interfaces. It is a half-duplex interface that can send or receive 8-bit bytes, but not both simultaneously.

The CSI/O includes separate receive and transmit data pins, RXS and TXS, plus a clock pin CKS that can be either an input or an output. In either direction, CKS is gated so that it switches only during active data characters on RXS and TXS.

The bit rate can be as much as $\text{PHI}/20$ for an internally generated clock, and even faster with an externally-generated clock.

The CSI/O Control (CNTR) and Data (TRDR) registers are described in section “Clocked Serial I/O (CSI/O) Registers”, on page 174.

Clock Selection

After Reset, bits 2–0 of the CNTR are 111, which conditions the CKS pin to be an input. If this Z80S188 is to provide the clock to the other station(s), write bits 2–0 with one of the other values described in Table 22, to select by what factor the CSI/O divides PHI to produce the clock it drives onto CKS.

TABLE 22. CSI/O CLOCK RATE SELECTION

CNTR 2-0	CKS Bit rate
000	$\text{PHI}/20$
001	$\text{PHI}/40$
010	$\text{PHI}/80$
011	$\text{PHI}/160$
100	$\text{PHI}/320$
101	$\text{PHI}/640$
110	$\text{PHI}/1280$
111	input from external source

NOTE: Wait at least 1 bit time after the transmitter clears its TE bit, or the receiver clears its RE bit, before changing the baud rate.

Operation

The clock signal on CKS and the data signals on TXS and RXS have the same basic relationship, regardless of whether this Z80S188 is driving or receiving the clock on CKS, and regardless of whether it is sending on TXS or receiving on RXS. Figure 23 illustrates this relationship, along with the operation of the flags in the CNTR.

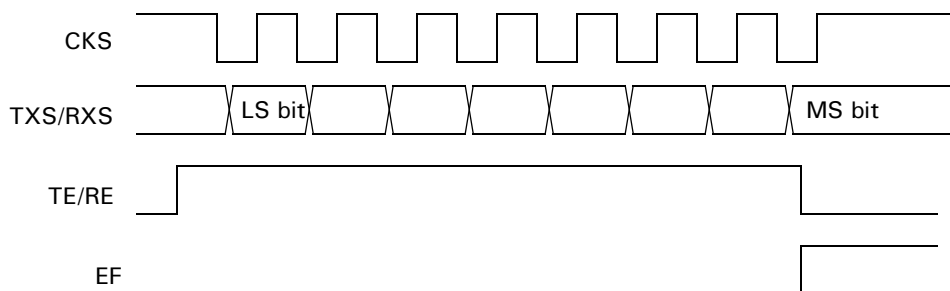


FIGURE 23. CSI/O OPERATION

CKS is High between bytes. The CSI/O operates as follows in the four possible cases of clock and data sourcing:

Output Clock, Output Data. After software writes a byte to be transmitted to the TRDR and sets the TE bit in the CNTR, the CSI/O drives CKS Low, and shortly thereafter drives the LS bit of the byte onto TXS. Thereafter it toggles CKS at the selected clock rate, driving each next-more-significant bit shortly after each falling edge on TXS, until it has driven the MS bit onto TXS. Thereafter the CSI/O clears the TE bit and sets the EF bit in the CNTR. Driving CKS High thereafter, completes the operation for this byte.

Input Clock, Output Data. Software must write the byte to be transmitted to the TRDR, and then set the TE bit in the CNTR, before the external clock source drives CKS Low for the first bit of the byte. The CSI/O waits for falling edges on CKS, and after each such edge it drives a bit of the character onto TXS, starting with the LS bit. After driving the MS bit onto TxS, the CSI/O clears the TE bit and sets the EF bit in the CNTR.

Output Clock, Input Data. After software sets the RE bit in the TRDR, the CSI/O drives CKS Low, and thereafter toggles CKS at the selected clock rate, for a total of 8 falling and 8 rising edges. The CSI/O samples one bit of the byte starting with the LS bit at each rising edge on CKS. sampled into the TRDR, starting with the LS bit. After sampling the MS bit, the CSI/O clears the RE bit and sets the EF bit in the CNTR.

Input Clock, Input Data. Software must set the RE bit in the CNTR, before the external clock source drives CKS Low for the first bit of the byte. After each rising edge on CKS, one bit of the byte is sampled starting with the LS bit. After sampling the MS bit, the CSI/O clears the RE bit and sets the EF bit in the CNTR.

Transmitting a Byte. Both the TE and RE flags in the CNTR must be 0 before software can send a(nother) byte. At that point, software must:

1. Write the (next) byte to be transmitted into the TRDR,
2. Write a 1 to the TE bit (bit 4) in the CNTR, with the desired clock control value in bits 2–0. If no interrupt is desired when the byte has been sent, write a 0 in



bit 6 (EIE) of this register. Otherwise, write a 1 and set a Transmit flag for the interrupt service routine.

3. If the transmit flag is cleared:
 - a. Read the TRDR.
 - b. Process the byte.

Receiving a Byte. Both the TE and RE flags in the CNTR must be 0 before software can condition the CSI/O to receive a(nother) byte. At this point software must:

1. Write a 1 to the RE bit (bit 5) in the CNTR, with the desired clock control value in bits 2–0. For polled operation write a 0 in bit 6 (EIE) of this value, else write a 1 and clear the Transmit flag for the Interrupt Service Routine.
2. For polled operation, read the CNTR, periodically or in a tight loop, until RE is 0 and EF is 1. For interrupt-driven operation, the following step is processed in the interrupt service routine, as described in the next topic.
3. For polled operation, read the TRDR to acquire the byte and clear the EF flag. If the sending station contains more data, return to step 1.

Cancelling Transmission or Reception. Software can cancel byte transmission or reception in progress, by writing 0s to the TE and RE bits. Avoid this process when Z80S188 is sourcing CKS, because the remote station's hardware may hang in mid-byte. For operation with an external clock, software could perform this action after a time-out period has expired, indicating that the remote station has nothing more to send or cannot accept further data.

Handling CSI/O Interrupts

If software sets bit 6 (EIE) of the CNTR at the start of an transmit or receive operation, then when the CSI/O completes the operation and sets the EF bit, the CSI/O requests an interrupt from the processor. If the conditions listed in “On-Chip Interrupt Handling”, on page 28 are met with respect to this request, the processor responds by fetching the address of the CSI/O interrupt service routine (ISR) from memory at address (I : IL : 12). This ISR must:

1. Save as many registers as it may use, by means of PUSH, EX AF, AF', and/or EXX instructions.
2. Read the CNTR. If the EF bit is 0, the ISR may log this *unknown interrupt* before restoring the registers and returning to the interrupted process with EI and RET instructions.
3. If the XMIT flag is cleared:
 - a. Read the TRDR to acquire the byte and clear the EF flag.
 - b. Process the byte.
 - c. If further reception is necessary, write a 1 to RE as in step 1 of “Receiving a Byte” above.

- d. If there is data to be sent, write the first byte to the TRDR (this action clears the EF flag) and then set TE in the CNTR, as in steps 1-2 of the “Transmitting a Byte” section. Also, set the XMIT flag for the next interrupt.
4. If the XMIT flag is set:
 - a. Write the next byte to the TRDR (this clears the EF flag).
 - b. Set TE in the CNTR, as in steps 1-2 of “Transmitting a Byte” above.
5. If there is data to be received, perform a dummy read of TRDR to clear the EF flag, then write a 1 to RE as in step 1 of the section, “Receiving a Byte”. Also, clear the XMIT flag.
6. If there is no other actions to be taken, perform a dummy read of TRDR to clear the EF flag.
7. Restore the saved registers and return to the interrupted process using EI and RET instructions.

I/O REGISTERS

The registers that are integral to the processor and the programming model are described in section “Processor Description”, on page 11. This chapter describes the registers in I/O space that control the operation of the overall device and its on-chip peripherals.

REGISTERS SUMMARY

I/O registers on the Z80S188 are divided into two classes, the 80180 registers, and other on-chip registers.

The 80180 registers:

- Are located at addresses between 0000H and 003FH.
- Can be relocated to 0040–007FH or 0080–00BFH if desired.
- Must be accessed using IN0, IN r, (C), OUT0, OTDM(R), OTIM(R), or OUT (C), r instructions.
- Require three clocks per I/O bus cycle.

Other on-chip registers:

- If bit 7 of the Interrupt Priority Register is 1, are located at I/O addresses between xxD0 and xxFFH, and can be accessed using IN, OUT, IND(R), INI(R), OUTD, OUTI, OTDR, or OTIR instructions.
- If bit 7 of the Interrupt Priority Register is 0, are located at I/O addresses between 00D0 and 00FFH, and can be accessed using IN0, IN r, (C), OUT0, OTDM(R), OTIM(R), or OUT (C), r instructions.
- Require 4 clocks per I/O bus cycle.

The following table includes all on-chip registers in both classes. I/O addresses not described are not used.



REGISTER NAME	ADDR (Hex)	REGISTER NAME	ADDR (Hex)
ASCI0 Control Register A	00	ASCI1 Control Register A	01
ASCI0 Control Register B	02	ASCI1 Control Register B	03
ASCI0 Status Register	04	ASCI1 Status Register	05
ASCI0 Tx Data Register	06	ASCI1 Tx Data Register	07
ASCI0 Rx Data Register	08	ASCI1 Rx Data Register	09
CSI/O Control Register	0A	CSI/O Data Register	0B
PRT0 Timer Data Register Low	0C	PRT0 Timer Data Register High	0D
PRT0 Timer Reload Register Low	0E	PRT0 Timer Reload Register High	0F
Timer Control Register	10	Timer Prescale Register	11
ASCI0 Extension Control Reg	12	ASCI1 Extension Control Reg	13
PRT1 Timer Data Register Low	14	PRT1 Timer Data Register High	15
PRT1 Timer Reload Register Low	16	PRT1 Timer Reload Register High	17
Free Running Counter	18		
ASCI0 Time Constant Low	1A	ASCI0 Time Constant High	1B
ASCI1 Time Constant Low	1C	ASCI1 Time Constant High	1D
Clock Multiplier Register	1E	CPU Control Register	1F
DMA0 Source Addr Register L	20	DMA0 Source Addr Register H	21
DMA0 Source Addr Register B	22	DMA0 Dest Addr Register L	23
DMA0 Dest Addr Register H	24	DMA0 Dest Addr Register B	25
DMA0 Byte Count Register L	26	DMA0 Byte Count Register H	27
DMA1 Memory Addr Register L	28	DMA1 Memory Addr Register H	29
DMA1 Memory Addr Register B	2A	DMA1 I/O Addr Register L	2B
DMA1 I/O Addr Register H	2C	DMA1 I/O Addr Register B	2D
DMA1 Byte Count Register L	2E	DMA1 Byte Count Register H	2F
DMA Status Register	30	DMA Mode Register	31
DMA/Wait Control Register	32	Interrupt Vector Low Register	33
Interrupt/Trap Control Register	34		
Refresh Control Register	36		
MMU Common Base Register OR MMU Common Base Register Low	38	MMU Bank Base Register OR MMU Common Base Register High	39
MMU Common/Bank Area Reg OR MMU Band Area Register	3A	MMU Common Area Register	3B
MMU Bank Base Register Low	3C	MMU Bank Base Register High	3D
Operating Mode Control Register	3E	I/O Control Register	3F
CTC Channel 0	D0	CTC Channel 1	D1
CTC Channel 2	D2	CTC Channel 3	D3
PIO C Data Register	D4	PIO C Command Register	D5

REGISTER NAME	ADDR (Hex)	REGISTER NAME	ADDR (Hex)
PIO D Data Register	D6	PIO D Command Register	D7
SIO A Data Register	D8	SIO A Control Register	D9
SIO B Data Register	DA	SIO B Control Register	DB
PIO A Data Register	DC	PIO A Command Register	4D
PIO B Data Register	DE	PIO B Command Register	DF
WDT Master Register	F0	WDT Command Register	F1
Interrupt Priority Register	F4		
MEMCS0 Low Register	F6	MEMCS0 High Register	F7
MEMCS1 Low Register 0	F8	MEMCS1 High Register	F9
IOCS Low Register	FA	IOCS High Register	FB
On-Chip Memory Control Register	FC		

NOTE: The Z80S188 decodes I/O addresses F2, F3, F5, and FD-FFH as on-chip, and these addresses cannot be used for external devices.

BASIC DEVICE REGISTERS

TABLE 23. FREE-RUNNING COUNTER (0018H)

Bit	7	6	5	4	3	2	1	0
Bit/Field	Count							
R/W	R							
Reset	1	1	1	1	1	1	1	1

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Count	R		This value is counted down by 1 every 10 PHI clocks, including during I/O Stop mode.

**TABLE 24. CLOCK MULTIPLIER REGISTER (001EH) CMR**

Bit	7	6	5	4	3	2	1	0
Bit/Field	2X Clock	low Noise XTAL	Reserved					
R/W	R/W	R/W	?					
Reset	0	0	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7	2X Clock	R/W	1	If this bit is 1, the oscillator or external clock is multiplied by 2 to produce PHI. This option may be selected independently of the 1X Clock option controlled by bit 7 of the CPU Control Register. If this bit is 1 and bit 7 of the CCR is 0, PHI operates on the same frequency as when this bit is 0 and bit 7 of the CCR is 1.
6	Low Noise XTAL	R/W	1	If this bit is 1, the oscillator is operated in a low-noise mode, in which the gain is reduced and the output drive is reduced to about 30% of normal operation. This mode can be used for PCMCIA applications, in which the crystal would otherwise be driven too energetically. This mode limits the crystal frequency to 20 MHz at $V_{DD} = 4.5V$ and 10 MHz at $V_{DD} = 3.0V$.

TABLE 25. CPU CONTROL REGISTER (001FH) CCR

Bit	7	6	5	4	3	2	1	0
Bit/Field	X1 Clock	Stand-by	BREXT	LNPHI	Idle/Quick	LNIO	LNCTL	LNA/D
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7	X1 Clock	R/W	0 1	The crystal frequency is divided by 2. The crystal frequency is used directly.
6,3	Standby, Idle/Quick	R/W	00 01 10 11	SLP inst enters Sleep or System Stop mode IOSTOP + SLP enters Idle mode IOSTOP + SLP enters Standby mode IOSTOP + SLP enters Quick Recovery Standby mode
5	BREXT	R/W	1	Z80S188 honors Bus Requests in Standby mode
4	LNPHI	R/W	1	PHI Low-Noise mode: 25% of normal drive
2	LNIO	R/W	1	The following pins have 25% of normal drive: ARDY, BRDY, CKA1-0, CKS, CRDY, DRDY, DTRA-B, IEO, PA7-0, PB7-0, PC7-0, PD7-0, RTS0, RTSA-B, SYNCA-B, TEND1-0, TOUT, TXA1-0, TxDA-B, TXS, ZC/TO3-0
1	LNCTL	R/W	1	The following pins have 25% of normal drive: BUSACK, HALT, IOCS, IORQ, M1, MEMCS1-0, MREQ, RFSH, ROMCS, ST, RD, WR
0	LNA/D	R/W	1	A22-0/D7-0 Low-Noise mode: 25% of normal drive

**TABLE 26. REFRESH CONTROL REGISTER (0036H) RCR**

Bit	7	6	5	4	3	2	1	0
Bit/Field	REFE	REFW	Reserved				Cycle	
R/W	R/W	R/W	?				R/W	
Reset	1	1	X	X	X	X	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7	REFE	R/W	0	The Z80S188 does not Refresh
			1	The Z80S188 generates Refresh cycles
6	REFW	R/W	0	2-clock Refresh cycles
			1	3-clock Refresh cycles
1-0	Cycle	R/W	00	Refresh cycle every 10 PHI clocks
			01	Refresh cycle every 20 PHI clocks
			10	Refresh cycle every 40 PHI clocks
			11	Refresh cycle every 80 PHI clocks

TABLE 27. OPERATING MODE CONTROL REGISTER (003EH) OMCR

Bit	7	6	5	4	3	2	1	0
Bit/Field	M1E	M1TE	IOC	Reserved				
R/W	R/W	W	R/W	?				
Reset	1	1	1	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7	M1E	R/W	0	$\overline{M1}$ is asserted Low during refetch of RETI instruction and $\overline{INT0}$ acknowledge cycle (Z80 peripheral compatible).
			1	$\overline{M1}$ is asserted Low in all Op Code fetches, $\overline{INT0}$ acknowledge cycles, and 1st cycle of \overline{NMI} acknowledge. The Z80S188 does not refetch RETI instructions.
6	M1TE	W	0	After a 0 is written to this bit, the next Op Code fetch asserts $\overline{M1}$ Low. (Automatically returns to 1 state.)
			1	M1E governs operation of $\overline{M1}$
5	IOC	R/W	0	In I/O cycles \overline{IORQ} , and \overline{RD} if applicable, are driven Low from the rising edge at the start of the T2 cycle (Z80 compatible)
			1	In I/O cycles \overline{IORQ} , and \overline{RD} if applicable, are driven Low from the falling edge in the T1 cycle (64180 compatible)

TABLE 28. I/O CONTROL REGISTER (003FH) IOCR

Bit	7	6	5	4	3	2	1	0
Bit/Field	IOA		IOSTP	Reserved				
R/W	R/W		R/W	?				
Reset	0	0	0	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-6	IOA	R/W	00	180 registers are located at 0000-003F
			01	180 registers are located at 0040-007F
			10	180 registers are located at 0080-00BF
			11	Do not program this value.
5	IOSTP	R/W	0	Normal operation
			1	The ASCIs, PRTs, and CSIO are disabled.

TABLE 29. ON-CHIP MEMORY CONTROL REGISTER (xxFC OR 00FCH) OCMCR

Bit	7	6	5	4	3	2	1	0
Bit/Field	OC RAM Enable	OC RAM Hi Phy	OC ROM Enable	Reserved				
R/W	R/W	R/W	R/W	?				
Reset	1	0	1	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7	OC RAM Enable	R/W	0	On-chip RAM is not accessible.
			1	On-chip RAM is enabled.
6	OC RAM Hi Phy	R/W	0	On-chip RAM is located at xxFC00-xxFFFFH.
			1	On-chip RAM is located at 7FFC00-7FFFFFH.
5	OC ROM Enable	R/W	0	On-chip ROM and $\overline{\text{ROMCS}}$ are disabled.
			1	Masked part: on-chip ROM is enabled at 000000-000FFFFH. Unmasked part: $\overline{\text{ROMCS}}$ becomes Low for 000000-000FFFFH.

INTERRUPT REGISTERS

See section “Interrupts”, on page 15, for more about these registers.

TABLE 30. INTERRUPT VECTOR LOW REGISTER (0033H) IL

Bit	7	6	5	4	3	2	1	0
Bit/Field	IL7-5			Reserved				

**TABLE 30. INTERRUPT VECTOR LOW REGISTER (0033H) IL**

R/W	R/W			?				
Reset	0	0	0	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-5	IL7-5	R/W		The processor uses these bits as A7-5 (along with the contents of the I register as A15-8) when fetching an interrupt service routine address for INT1-2, ASCIO-1, PRT0-1, DMA0-1, CSI/O, or the ZDI.

TABLE 31. INTERRUPT/TRAP CONTROL REGISTER (0034H) ITC

Bit	7	6	5	4	3	2	1	0
Bit/Field	Trap	UFO	ROM EE	RAM EE	Reserved	INT2 en	INT1 en	INT0 en
R/W	RWOC	R	RWOC	RWOC	?	R/W	R/W	R/W
Reset	0	X	0	0	X	0	0	1

NOTE: R = Read W = Write X = Indeterminate RWOC = Readable; Writing 0 Clears

Bit Position	Bit/Field	R/W	Value	Description
7	Trap	RWOC	1	Illegal Instruction Trap. Writing a 0 to this bit clears it; writing a 1 has no effect.
6	UFO	R	0 1	Inst started at stacked PC - 1 Inst started at stacked PC - 2
5	ROME	RWOC	1	In a masked ROM part with the ROM Entry Protect option masked as 1, this bit is set if a ROM Entry Exception occurs. Writing a 0 to this bit clears it; writing a 1 has no effect.
4	RAMEE	RWOC	1	In a masked ROM part with the RAM Protect option masked as 1, this bit is set if a RAM Execution Exception occurs. Writing a 0 to this bit clears it; writing a 1 has no effect.
2	INT2 en	R/W		$\overline{\text{INT2}}$ interrupt enable
1	INT1 en	R/W		$\overline{\text{INT1}}$ interrupt enable
0	INT0 en	R/W		$\overline{\text{INT0}}$ interrupt enable

TABLE 32. INTERRUPT PRIORITY REGISTER (xxF4 OR 00F4H) INTPR

Bit	7	6	5	4	3	2	1	0
Bit/Field	All Page IO	SIOA 32	SIOB 32	Reserved	Z80 Peripheral Priority			
R/W	R/W	R/W	R/W	?	R/W			
Reset	1	0	0	X	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7	All Page IO	R/W	0	SIO, PIO, CTC, WDT, Chip Select and Wait blocks decode A15-8 for all 0, like the "180 registers".
			1	These blocks ignore A15-8, as on the Z80.
6-5	SIOA32, SIOB32	R/W	0	SIO channel generates, checks 16 bit CRC
			1	SIO channel generates, checks 32 bit CRC
3-0	Z80 Peripheral Priority	R/W		These bits control the relative interrupt priority of the SIO, CTC, and PIOs: CTC, SIO, PIO AB, PIO CD (lowest) 0000 SIO, CTC, PIO AB, PIO CD (lowest) 0001 CTC, PIO AB, PIO CD, SIO (lowest) 0010 PIO AB, PIO CD, SIO, CTC (lowest) 0011 PIO AB, PIO CD, CTC, SIO (lowest) 0100 SIO, PIO AB, PIO CD, CTC (lowest) 0101 CTC, PIO AB, SIO, PIO CD (lowest) 1000 SIO, PIO AB, CTC, PIO CD (lowest) 1001 PIO AB, CTC, PIO CD, SIO (lowest) 1010 PIO AB, SIO, PIO CD, CTC (lowest) 1011 PIO AB, CTC, SIO, PIO CD (lowest) 1100 PIO AB, SIO, CTC, PIO CD (lowest) 1101

MMU REGISTERS

See section "Memory Management Unit (MMU)", on page 30, for more about these registers.

TABLE 33. COMMON BASE REGISTER (0038H IN CLASSIC MODE) CBR

Bit	7	6	5	4	3	2	1	0
Bit/Field	Base of Common Area 1							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Common 1 Area Base	R/W		If the comparison of bits 15-12 of a logical address indicates that the address is in Common Area 1, this value (shifted left 12 bits, times 4096) is added to the logical address to form the physical address.

TABLE 34. BANK BASE REGISTER (0039H IN CLASSIC MODE) BBR

Bit	7	6	5	4	3	2	1	0
Bit/Field	Base of Bank Area							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Bank Area Base	R/W		If the comparison of bits 15-12 of a logical address indicates that the address is in the Bank Area, this value (shifted left 12 bits, times 4096) is added to the logical address to form the physical address.

**TABLE 35. COMMON/BANK AREA REGISTER (003AH IN CLASSIC MODE) CBAR**

Bit	7	6	5	4	3	2	1	0
Bit/Field	Bank/Common 1 Boundary				Common 0/Bank Boundary			
R/W	R/W				R/W			
Reset	1	1	1	1	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-4	Bank/Common 1 Boundary	R/W		If bits 15-12 of a logical address are greater than or equal to this value, the address is in Common Area 1.
3-0	Common 0/Bank Boundary	R/W		If bits 15-12 of a logical address are less than this value, the address is in Common Area 0.

NOTE: In Classic mode, if bits 3-0 of this reg \leq bits 15-12 of a logical address $<$ bits 7-4 of this reg, the address is in the Bank Area. Do not program this register so that bits 3-0 $>$ bits 7-4. All comparisons are unsigned.

TABLE 36. COMMON BASE REGISTER LOW (0038H IN EXTENDED MODE) CBRL

Bit	7	6	5	4	3	2	1	0
Bit/Field	Common Area 1 Base address, bits 15-10						Reserved	
R/W	R/W						X	
Reset	0	0	0	0	0	0	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-2	Common Area 1 Base Address 15-10	R/W		In Extended mode, if the comparison of bits 15-10 of a logical address indicates that the address is in Common Area 1, this value is added to bits 15-10 of the logical address to form bits 15-10 of the physical address.

TABLE 37. COMMON BASE REGISTER HIGH (0039H IN EXTENDED MODE) CBRH

Bit	7	6	5	4	3	2	1	0
Bit/Field	Reserved	Common Area 1 Base address, bits 22-16						
R/W	X	R/W						
Reset	X	0	0	0	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-2	Common Area 1 Base Address 22-16	R/W		In Extended mode, if the comparison of bits 15-10 of a logical address indicates that the address is in Common Area 1, this value, plus the carry from the addition of bits 15-10, is used as bits 22-16 of the physical address.

TABLE 38. BANK AREA REGISTER (003AH IN EXTENDED MODE) BAR

Bit	7	6	5	4	3	2	1	0
Bit/Field	Common 0/Bank Area Boundary, bits 15-10						Reserved	
R/W	R/W						X	
Reset	0	0	0	0	0	0	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-2	Common 0/Bank Area Boundary 15-10	R/W		In Extended mode, if bits 15-10 of a logical address are less than this value, the address is in Common Area 0.

**TABLE 39. COMMON AREA REGISTER (003BH IN EXTENDED MODE) CAR**

Bit	7	6	5	4	3	2	1	0
Bit/Field	Bank/Common 1 Area Boundary, bits 15-10						Reserved	
R/W	R/W						X	
Reset	0	0	0	0	0	0	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-2	Common 0/ Bank Area Boundary 15-10	R/W		In Extended mode, if bits 15-10 of a logical address are greater than or equal to this value, the address is in Common Area 1.

NOTE: In Extended mode, if bits 7-2 of the BAR \leq bits 15-12 of a logical address $<$ bits 7-2 of this reg, the address is in the Bank Area. Do not program these registers so that bits 7-2 of the BAR $>$ bits 7-2 of this register. All comparisons are unsigned.

TABLE 40. BANK BASE REGISTER LOW (003CH IN EXTENDED MODE) BBRL

Bit	7	6	5	4	3	2	1	0
Bit/Field	Bank Area Base address, bits 15-10						Reserved	
R/W	R/W						X	
Reset	0	0	0	0	0	0	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-2	Bank Area Base Address 15- 10	R/W		In Extended mode, if the comparison of bits 15-10 of a logical address indicates that the address is in the Bank Area, this value is added to bits 15-10 of the logical address to form bits 15-10 of the physical address.

TABLE 41. BANK BASE REGISTER HIGH (003DH IN EXTENDED MODE) BBRH

Bit	7	6	5	4	3	2	1	0
Bit/Field	Reserved	Bank Area Base address, bits 22-16						
R/W	X	R/W						
Reset	X	0	0	0	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-2	Bank Area Base Address 22-16	R/W		In Extended mode, if the comparison of bits 15-10 of a logical address indicates that the address is in Common Area 1, this value, plus the carry from the addition of bits 15-10, is used as bits 22-16 of the physical address.

CHIP SELECT AND WAIT REGISTERS

See pages 33 and 38 for more detail about these registers.

TABLE 42. MEMORY CHIP SELECT 0 LOW REGISTER (xxF6 OR 00FH6) MCS0L

Bit	7	6	5	4	3	2	1	0
Bit/Field	CA14-13		Size/Enable				Wait States	
R/W	R/W		R/W				R/W	
Reset	0	0	1	0	0	1	1	1

NOTE: R = Read W = Write X = Indeterminate



Bit Position	Bit/Field	R/W	Value	Description
7-6	Comparison Address 14-13	R/W		The Size/Enable field controls whether these bits are compared for equality to A14-13.
5-2	Size/Enable	R/W	0000	$\overline{\text{MEMCS0}}$ is disabled (remains High).
			0001	$\overline{\text{MEMCS0}}$ Low for all of memory
			0010	$\overline{\text{MEMCS0}}$ Low when A22==CA22
			0011	$\overline{\text{MEMCS0}}$ Low when A22-21==CA22-21
			0100	$\overline{\text{MEMCS0}}$ Low when A22-20==CA22-20
			0101	$\overline{\text{MEMCS0}}$ Low when A22-19==CA22-19
			0110	$\overline{\text{MEMCS0}}$ Low when A22-18==CA22-18
			0111	$\overline{\text{MEMCS0}}$ Low when A22-17==CA22-17
			1000	$\overline{\text{MEMCS0}}$ Low when A22-16==CA22-16
			1001	$\overline{\text{MEMCS0}}$ Low when A22-15==CA22-15
			1010	$\overline{\text{MEMCS0}}$ Low when A22-14==CA22-14
			1011	$\overline{\text{MEMCS0}}$ Low when A22-13==CA22-13
			11xx	Reserved; do not program these values
1-0	Wait States	R/W	00	No wait states added in $\overline{\text{MEMCS0}}$ range
				At least 1 wait state in $\overline{\text{MEMCS0}}$ range
			01	At least 2 wait states in $\overline{\text{MEMCS0}}$ range
				At least 3 wait states in $\overline{\text{MEMCS0}}$ range
			10	
			11	

TABLE 43. MEMORY CHIP SELECT 0 HIGH REGISTER (xxF7 OR 00F7H) MCS0H

Bit	7	6	5	4	3	2	1	0
Bit/Field	Comparison Address 22-15							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Comparison Address 22-15	R/W		The Size/Enable field in MCS0L controls how many of these bits are compared for equality to A22-15.

TABLE 44. MEMORY CHIP SELECT 1 LOW REGISTER (xxF8 OR 00F8H) MCS1L

Bit	7	6	5	4	3	2	1	0
Bit/Field	CA14-13		Size/Enable				Wait States	
R/W	R/W		R/W				R/W	
Reset	0	0	0	0	0	1	1	1

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-6	Comparison Address 14-13	R/W		The Size/Enable field controls whether these bits are compared for equality to A14-13.
5-2	Size/Enable	R/W	0000	$\overline{\text{MEMCS1}}$ is disabled (remains High)
			0001	$\overline{\text{MEMCS1}}$ Low for all memory (see note)
			0010	$\overline{\text{MEMCS1}}$ Low when A22==CA22
			0011	$\overline{\text{MEMCS1}}$ Low when A22-21==CA22-21
			0100	$\overline{\text{MEMCS1}}$ Low when A22-20==CA22-20
			0101	$\overline{\text{MEMCS1}}$ Low when A22-19==CA22-19
			0110	$\overline{\text{MEMCS1}}$ Low when A22-18==CA22-18
			0111	$\overline{\text{MEMCS1}}$ Low when A22-17==CA22-17
			1000	$\overline{\text{MEMCS1}}$ Low when A22-16==CA22-16
			1001	$\overline{\text{MEMCS1}}$ Low when A22-15==CA22-15
			1010	$\overline{\text{MEMCS1}}$ Low when A22-14==CA22-14
			1011	$\overline{\text{MEMCS1}}$ Low when A22-13==CA22-13
			11xx	Reserved; do not program these values
1-0	Wait States	R/W	00	No wait states added in $\overline{\text{MEMCS1}}$ range
			01	At least 1 wait state in $\overline{\text{MEMCS1}}$ range
			10	At least 2 wait states in $\overline{\text{MEMCS1}}$ range
			11	At least 3 wait states in $\overline{\text{MEMCS1}}$ range

**Bit**

Position	Bit/Field	R/W	Value	Description
----------	-----------	-----	-------	-------------

NOTE: If the ranges for $\overline{\text{MEMCS0}}$ and $\overline{\text{MEMCS1}}$ overlap, $\overline{\text{MEMCS0}}$ goes Low and $\overline{\text{MEMCS1}}$ remains High for addresses in the overlapped region.

TABLE 45. MEMORY CHIP SELECT 1 HIGH REGISTER (xxF9 OR 00F9H) MCS1H

Bit	7	6	5	4	3	2	1	0
Bit/Field	Comparison Address 22-15							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit

Position	Bit/Field	R/W	Value	Description
----------	-----------	-----	-------	-------------

7-0	Comparison Address 22-15	R/W		The Size/Enable field in MCS1L controls how many of these bits are compared for equality to A22-15.
-----	--------------------------	-----	--	---

TABLE 46. I/O CHIP SELECT LOW REGISTER (xxFA OR 00FAH) IOCSL

Bit	7	6	5	4	3	2	1	0
Bit/Field	CA7-5			Comp Hi	Size		Wait States	
R/W	R/W			R/W	R/W		R/W	
Reset	0	0	0	0	0	0	1	1

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-5	Comparison Address 7-5	R/W		The Size field controls how many of these bits are compared for equality to A7-5.
4	CompHi	R/W	0 1	$\overline{\text{IOCS}}$ decoding ignores A15-8 $\overline{\text{IOCS}}$ Low requires A15-8 = CA15-8
3-2	Size	R/W	00 01 10 11	$\overline{\text{IOCS}}$ does not decode A7-5 (see note) $\overline{\text{IOCS}}$ Low requires A7 = CA7 $\overline{\text{IOCS}}$ Low requires A7-6 = CA7-6 $\overline{\text{IOCS}}$ Low requires A7-5 = CA7-5
1-0	Wait States	R/W	00 01 10 11	Min 4 clocks/cycle in $\overline{\text{IOCS}}$ range Min 5 clocks/cycle in $\overline{\text{IOCS}}$ range Min 6 clocks/cycle in $\overline{\text{IOCS}}$ range Min 7 clocks/cycle in $\overline{\text{IOCS}}$ range

NOTE: $\overline{\text{IOCS}}$ always remains High for on-chip I/O addresses.

TABLE 47. I/O CHIP SELECT HIGH REGISTER (xxFB OR 00FBH) IOCSH

Bit	7	6	5	4	3	2	1	0
Bit/Field	Comparison Address 15-8							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Comparison Address 15-8	R/W		The CompHi bit in IOCSL controls whether these bits are compared for equality to A15-8.



I/O PORT (PIO) REGISTERS

See section “Parallel I/O (PIOs)”, on page 44, for more about these registers.

TABLE 48. PORT A DATA (xxDC OR 00DCH) PAD

Bit	7	6	5	4	3	2	1	0
Bit/Field	Port A Data							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Port A Data	R/W		Writing to this address sets output data for the PA7-0 pins, and in mode 0 or 2 makes ARDY High. Reading from this address returns input data from the PA7-0 pins, and in mode 1 or 2 makes ARDY or BRDY (respectively) High.

TABLE 49. PORT A CONTROL (xxDD OR 00DDH) PAC

Bit	7	6	5	4	3	2	1	0
Bit/Field	Port A Control							
R/W	WO							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Port A Control	WO		The port decodes the LS bits of data written to this address, to select among the following control words:
			vvvvvvv0	Interrupt Vector Word. The port returns this value as an interrupt vector.
			exxx0011	Interrupt Disable Word. Bit 7 enables (1) or disables (0) port interrupts.
			eahm0111	Interrupt Control Word. Bit 7 enables (1) or disables (0) port interrupts.
				(Continued)

Bit Position	Bit/Field	R/W	Value	Description
7-0 (cont.)	Port A Control	WO		In mode 3 only, bit 5 controls whether pins selected by 1s in the mask interrupt when High (1) or Low (0), bit 6 controls whether interrupt occurs when all (1) or any (0) of these pins is in the active state, and a 1 in bit 4 indicates that the next value written to this address is an interrupt mask. Mode Control Word. Bits 7-6 select the port's operating mode: 00 Output 01 Input 10 Bidirectional 11 Bit Control. Setting this value also makes the port capture the next value written to this address, as an I/O Register Control Word, in which 1s identify inputs and 0s identify outputs.
			mmxx1111	

TABLE 50. PORT B DATA (xxDE OR 00DEH) PBD

Bit	7	6	5	4	3	2	1	0
Bit/Field	Port B Data							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Port B Data	R/W		Writing to this address sets output data for the PB7-0 pins, and in mode 0 makes BRDY High. Reading from this address returns input data from the PB7-0 pins, and in mode 1 makes BRDY High.

**TABLE 51. PORT B CONTROL (xxDF or 00DFH) PBC**

Bit	7	6	5	4	3	2	1	0
Bit/Field	Port B Control							
R/W	WO							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Port B Control	WO		The port decodes the LS bits of data written to this address, to select among the following control words:
			vvvvvvv0	Interrupt Vector Word. The port returns this value as an interrupt vector.
			exxx0011	Interrupt Disable Word. Bit 7 enables (1) or disables (0) port interrupts.
			eahm0111	Interrupt Control Word. Bit 7 enables (1) or disables (0) port interrupts. In mode 3 only, bit 5 controls whether pins selected by 1s in the mask interrupt when High (1) or Low (0), bit 6 controls whether interrupt occurs when all (1) or any (0) of these pins is in the active state, and a 1 in bit 4 indicates that the next value written to this address is an interrupt mask.
			mmxx1111	Mode Control Word. Bits 7-6 select the port's operating mode: 00 Output 01 Input 10 Do not program 11 Bit Control.
				Setting this value also makes the port capture the next value written to this address, as an I/O Register Control Word, in which 1s identify inputs and 0s identify outputs.

TABLE 52. PORT C DATA (xxD4 OR 00D4H) PCD

Bit	7	6	5	4	3	2	1	0
Bit/Field	Port C Data							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

NOTE: This address operates as described for Port A Data above, except that it is associated with the PC7-0 and CRDY pins, and for mode 2 the DRDY pin.

TABLE 53. PORT C CONTROL (xxD5 OR 00D5H) PCC

Bit	7	6	5	4	3	2	1	0
Bit/Field	Port C Control							
R/W	WO							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate WO = Write Only

NOTE: This address operates as described for Port A Control above, except that it is associated with the PC7-0 and CRDY pins, and for mode 2 the DRDY pin.

TABLE 54. PORT D DATA (xxD6 OR 00D6H) PDD

Bit	7	6	5	4	3	2	1	0
Bit/Field	Port D Data							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

NOTE: This address operates as described for Port B Data above, except that it is associated with the PD7-0 pins, and in mode 0 or 1 the DRDY pin.

TABLE 55. PORT D CONTROL (xxD7 OR 00D7H) PDC

Bit	7	6	5	4	3	2	1	0
Bit/Field	Port D Control							
R/W	WO							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate WO = Write Only

NOTE: This address operates as described for Port B Control above, except that it is associated with the PD7-0 pins, and in mode 0 or 1 the DRDY pin



DMA REGISTERS

See section “DMA Channels”, on page 56, for more about these registers.

TABLE 56. DMA0 SOURCE ADDRESS REGISTER LOW (0020H) SAR0L

Bit	7	6	5	4	3	2	1	0
Bit/Field	LS byte of DMA0 Source Address							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	DMA0 Source Address LS byte	R/W		LS byte of the Source Address for DMA channel 0.

TABLE 57. DMA0 SOURCE ADDRESS REGISTER HIGH (0021H) SAR0H

Bit	7	6	5	4	3	2	1	0
Bit/Field	Middle byte of DMA0 Source Address							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	DMA0 Source Address middle byte	R/W		Bits 15-8 of the Source Address for DMA channel 0.

TABLE 58. DMA0 SOURCE ADDRESS REGISTER B (0022H) SAR0B

Bit	7	6	5	4	3	2	1	0
Bit/Field	Reserved	DMA0 Source Address 22-16, OR						
		Not used				DMA Request Select		
R/W	N/A	R/W						
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
6-0	DMA0 Source Address 22- 16, OR DMA Request Select	R/W		<p>If the Source Mode field in the DMODE register is 00-10, this field contains A22-16 of the Source memory address</p> <p>If the Source Mode field in the DMODE register is 11, indicating an I/O source, bits 2-0 select which source device handshake line controls data transfer, as follows:</p> <p>DREQ0 pin</p> <p>000 ASCIO RDRF</p> <p>001 ASCI1 RDRF</p> <p>010 Reserved, do not program</p> <p>011 SIO A Rx</p> <p>100 SIO B Rx</p> <p>101 PIO A In</p> <p>110 PIO B In</p> <p>111</p>

TABLE 59. DMA0 DESTINATION ADDRESS REGISTER LOW (0023H) DAR0L

Bit	7	6	5	4	3	2	1	0
Bit/Field	LS byte of DMA0 Destination Address							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	DMA0 Destination Address LS byte	R/W		LS byte of the Destination Address for DMA channel 0.

**TABLE 60. DMA0 DESTINATION ADDRESS REGISTER HIGH (0024H) DAR0H**

Bit	7	6	5	4	3	2	1	0
Bit/Field	Middle byte of DMA0 Destination Address							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	DMA0 Destination Address middle byte	R/W		Bits 15-8 of the Destination Address for DMA channel 0.

TABLE 61. DMA0 DESTINATION ADDRESS REGISTER B (0025H) DAR0B

Bit	7	6	5	4	3	2	1	0
Bit/Field	Reserved	DMA1 Destination Address 22-16, OR						
		Not used				DMA Request Select		
R/W	N/A	R/W						
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
6-0	DMA0 Destination Address 22-16, OR	R/W		If the Destination Mode field in the DMODE register is 00-10, this field contains A22-16 of the Destination memory address
	DMA Request Select			If the Destination Mode field in the DMODE register is 11, indicating an I/O destination, bits 2-0 select which destination device handshake line controls data transfer, as follows: DREQ0 pin
			000	ASCIO TDRE
			001	ASCI1 TDRE
			010	Reserved, do not program
			011	SIO A Tx
			100	SIO B Tx
			101	PIO A Out
			110	PIO B Out
			111	

TABLE 62. DMA0 BYTE COUNT REGISTER LOW (0026H) BCR0L

Bit	7	6	5	4	3	2	1	0
Bit/Field	LS byte of DMA0 Byte Count							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	DMA0 Byte Count LS byte	R/W		LS byte of the Byte Count for DMA channel 0.

TABLE 63. DMA0 BYTE COUNT REGISTER HIGH (0027H) BCR0H

Bit	7	6	5	4	3	2	1	0
Bit/Field	MS byte of DMA0 Byte Count							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	DMA0 Byte Count MS byte	R/W		MS byte of the Byte Count for DMA channel 0.

TABLE 64. DMA1 MEMORY ADDRESS REGISTER LOW (0028H) MAR1L

Bit	7	6	5	4	3	2	1	0
Bit/Field	DMA1 Memory Address bits 7-0							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	DMA1 Memory Address LS byte	R/W		Bits 7-0 of the Memory Address for DMA channel 1.

**TABLE 65. DMA1 MEMORY ADDRESS REGISTER HIGH (0029H) MAR1H**

Bit	7	6	5	4	3	2	1	0
Bit/Field	Middle byte of DMA1 Memory Address							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	DMA1 Memory Address middle byte	R/W		Bits 15-8 of the Memory Address for DMA channel 1.

TABLE 66. DMA1 MEMORY ADDRESS REGISTER B (002AH) MAR1B

Bit	7	6	5	4	3	2	1	0
Bit/Field	Reserved	DMA1 Memory Address 22-16						
R/W	N/A	R/W						
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
3-0	DMA1 Memory Address 19-16	R/W		Bits 22-16 of the DMA1 Memory address.

TABLE 67. DMA1 I/O ADDRESS REGISTER LOW (002BH) IAR1L

Bit	7	6	5	4	3	2	1	0
Bit/Field	Bits 7-0 of DMA1 I/O Address							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	DMA1 I/O Address LS byte	R/W		Bits 7-0 of the I/O Address for DMA channel 1.

TABLE 68. DMA1 I/O ADDRESS REGISTER HIGH (002CH) IAR1H

Bit	7	6	5	4	3	2	1	0
Bit/Field	Bits 15-8 of DMA1 I/O Address							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	DMA1 I/O Address MS byte	R/W		Bits 15-8 of the I/O Address for DMA channel 1.

TABLE 69. DMA1 I/O ADDRESS REGISTER B (002DH) IAR1B

Bit	7	6	5	4	3	2	1	0
Bit/Field	AltE	AltC	Reserved			DMA1 I/O Handshake Select		
R/W	R/W	R/W	N/A			R/W		
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7	AltE	R/W	1	<p>This bit must be set only when both DMA channels are programmed for the same I/O source or I/O destination. In this case, a <i>channel end</i> condition (byte count is 0) on channel 0 sets bit 6 (AltC), which then enables channel 1's request and blocks channel 0's.</p> <p>Similarly, a channel end condition on channel 1 clears bit 6 (AltC), which then enables channel 0's request and blocks channel 1's. In order to use this feature with external requests, the request from the device must be routed or connected to both the DREQ0 and DREQ1 pins.</p>



Bit Position	Bit/Field	R/W	Value	Description																																
6	AltC	R/W		<p>If bit 7 (AltE) is 0, this bit has no effect. When bit 7 (AltE) is 1 and this bit is 0, the Request signal selected by bits 2-0 is not presented to channel 1, but channel 0's Request operates normally.</p> <p>When AltE is 1 and this bit is 1, the Request selected by SAR18-16 or DAR18-16 is not presented to channel 0, but channel 1's request operates normally.</p> <p>This bit can be written by software, to select which channel operates first, but this process must be done only when both channels are stopped (both DE1 and DE0 are 0).</p>																																
2-0	DMA1 I/O Handshake Select	R/W		<p>If bit DIM1 in the DCNTL register is 1, indicating an I/O source, these bits select which source handshake signal controls the transfer, as follows:</p> <p><u>DREQ1</u> pin</p> <table><tr><td>000</td><td>ASCIO RDRF</td></tr><tr><td>001</td><td>ASCII RDRF</td></tr><tr><td>010</td><td>Reserved, do not program</td></tr><tr><td>011</td><td>SIO A Rx</td></tr><tr><td>100</td><td>SIO B Rx</td></tr><tr><td>101</td><td>PIO A In</td></tr><tr><td>110</td><td>PIO B In</td></tr><tr><td>111</td><td></td></tr></table> <p>If DIM1 is 0, indicating an I/O destination, these bits select which destination handshake signal controls the transfer, as follows:</p> <p><u>DREQ1</u> pin</p> <table><tr><td>000</td><td>ASCIO TDRE</td></tr><tr><td>001</td><td>ASCII TDRE</td></tr><tr><td>010</td><td>Reserved, do not program</td></tr><tr><td>011</td><td>SIO A Tx</td></tr><tr><td>100</td><td>SIO B Tx</td></tr><tr><td>101</td><td>PIO A Out</td></tr><tr><td>110</td><td>PIO B Out</td></tr><tr><td>111</td><td></td></tr></table>	000	ASCIO RDRF	001	ASCII RDRF	010	Reserved, do not program	011	SIO A Rx	100	SIO B Rx	101	PIO A In	110	PIO B In	111		000	ASCIO TDRE	001	ASCII TDRE	010	Reserved, do not program	011	SIO A Tx	100	SIO B Tx	101	PIO A Out	110	PIO B Out	111	
000	ASCIO RDRF																																			
001	ASCII RDRF																																			
010	Reserved, do not program																																			
011	SIO A Rx																																			
100	SIO B Rx																																			
101	PIO A In																																			
110	PIO B In																																			
111																																				
000	ASCIO TDRE																																			
001	ASCII TDRE																																			
010	Reserved, do not program																																			
011	SIO A Tx																																			
100	SIO B Tx																																			
101	PIO A Out																																			
110	PIO B Out																																			
111																																				

TABLE 70. DMA1 BYTE COUNT REGISTER LOW (002EH) BCR1L

Bit	7	6	5	4	3	2	1	0
Bit/Field	LS byte of DMA1 Byte Count							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	DMA1 Byte Count LS byte	R/W		LS byte of the Byte Count for DMA channel 1.

TABLE 71. DMA1 BYTE COUNT REGISTER HIGH (002FH) BCR1H

Bit	7	6	5	4	3	2	1	0
Bit/Field	MS byte of DMA1 Byte Count							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	DMA1 Byte Count MS byte	R/W		MS byte of the Byte Count for DMA channel 1.

**TABLE 72. DMA STATUS REGISTER (0030H) DSTAT**

Bit	7	6	5	4	3	2	1	0
Bit/Field	DE1	DE0	DWE1	DWE0	DIE1	DIE0	Reserved	DME
R/W	R/W	R/W	W	W	R/W	R/W	N/A	R/W
Reset	0	0	X	X	0	0	X	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7	DE1	R/W		DMA channel 1 enable. This bit can only be written when DWE1 is 0 in a write operation. DMA channel 1 clears this bit when it counts its byte count down to 0.
6	DE0	R/W		DMA channel 0 enable. This bit can only be written when DWE0 is 0 in a write operation. DMA channel 0 clears this bit when it counts its byte count down to 0.
5-4	DWE1-0	W	0 1	Writing a 0 to one of these bits makes the Z80S188 capture the value of the corresponding DE bit. Writing a 1 to one of these bits does not affect the state of the corresponding DMA channel. These bits always read as 11.
3-2	DIE1-0	R/W		Interrupt enable for DMA channels 1-0. If one of these bits is 1, that DMA channel interrupts when it decrements its byte count to 0.
0	DME	R	0 1	Operations of both DMA channels are disabled. Reset and a Non-Maskable interrupt both clear this bit. Operation of a DMA channel that has its DE bit 1 is enabled. This bit is set when a 1 is written to a DE bit, and a 0 is written to the corresponding DWE bit in the same write operation.

TABLE 73. DMA MODE REGISTER (0031H) DMODE

Bit	7	6	5	4	3	2	1	0
Bit/Field	Reserved		DMA0 Dest Mode		DMA0 Source Mode		MMOD	Reserved
R/W	?		R/W		R/W		R/W	?
Reset	X	X	0	0	0	0	0	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
5-4	DMA0 Dest Mode	R/W	00 Memory write, address increment 01 Memory (or memory mapped I/O) write, fixed address 10 I/O write, fixed address 11	This field controls operation of the destination side of DMA channel 0: Memory write, address increment Memory write, address decrement Memory (or memory mapped I/O) write, fixed address I/O write, fixed address
3-2	DMA0 Source Mode	R/W	00 Memory read, address increment 01 Memory read, address decrement 10 Memory (or memory mapped I/O) read, fixed address 11 I/O read, fixed address	This field controls operation of the source side of DMA channel 0: Memory read, address increment Memory read, address decrement Memory (or memory mapped I/O) read, fixed address I/O read, fixed address
1	MMOD	R/W	0 1	When the Source and Dest Mode fields above are both 0x, indicating memory to memory operation, no device request (for example, $\overline{\text{DREQ0}}$) controls data transfer on DMA channel 0. In this case, this bit selects between two modes: 0 Cycle Steal mode: the DMA(s) and processor alternate bus cycles. 1 Burst mode: DMA0 uses the bus continuously to complete the block transfer.

**TABLE 74. DMA/WAIT CONTROL REGISTER (0032H) DCNTL**

Bit	7	6	5	4	3	2	1	0
Bit/Field	Memory Waits		I/O Waits		Request Sense 1-0		DMA1 Mode	
R/W	R/W		R/W		R/W		R/W	
Reset	1	1	1	1	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-6	Memory Waits	R/W		This field controls how many wait states are injected into DMA and processor memory cycles:
			00	None
			01	One
			10	Two
			11	Three
5-4	I/O Waits	R/W		This field controls how many wait states are injected into DMA and processor I/O cycles:
			00	None
			01	One
			10	Two
			11	Three
3-2	Request Sense 1-0	R/W		Each of these bits controls how the corresponding DMA channel samples its Request signal (except when DMA 0's Source and Dest Mode fields are both 0x)
			0	Level sense: the DMA samples its Request again during the 2nd cycle for each byte
			1	Edge sense: another falling edge is needed on the Request line before the DMA channel transfers another byte See section "Edge- vs. Level-Sensitive Requests", on page 57, for timing of both cases.
1-0	DMA1 Mode	R/W		This field controls the direction of both transfer and address stepping on DMA channel 1:
			00	Incrementing Memory addrs to I/O
			01	Decrementing Memory addrs to I/O
			10	I/O to incrementing Memory addrs
			11	I/O to decrementing Memory addrs

WATCH-DOG TIMER REGISTERS

See section "Watch-Dog Timer", on page 62, for more about these registers.

TABLE 75. WATCH-DOG TIMER MASTER REGISTER (xxF0 OR 00F0H) WDTMR

Bit	7	6	5	4	3	2	1	0
Bit/Field	Enable	Period Select		Reserved				
R/W	R/W	R/W		X				
Reset	1	0	0	1	1	0	1	1

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7	WDT Enable	R/W	0	Writing B1H to the WDT Command register disables the WDT
			1	The WDT cannot be disabled.
6-5	Period Select	R/W		The field selects how long software can leave the Watch-Dog Timer unattended, before it drives $\overline{\text{WDTOUT}}$
				Low:
			00	2^{16} (65,536) PHI clocks
			01	2^{18} (262,144) PHI clocks
			10	2^{20} (1,048,576) PHI clocks
			11	2^{22} (4,194,304) PHI clocks

TABLE 76. WATCH-DOG TIMER COMMAND REGISTER (xxF1 OR 00F1H) WDTCR

Bit	7	6	5	4	3	2	1	0
Bit/Field	WDT Command							
R/W	W							
Reset	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	WDT Command	W		Software can write the following values to this register, to affect the status of the WDT and Z80S188:
				Reloads/Restarts WDT
			4EH	Disables WDT if WDTMR bit 7 is 0
			B1H	

COUNTER/TIMER (CTC) REGISTERS

See section “Counter/Timer Channels (CTCs)”, on page 62, for more about these registers.

**TABLE 77. CTC0 (xxD0 or 00D0H) CTC0**

Bit	7	6	5	4	3	2	1	0
Bit/Field	Read: value of CTC0 down-counter Write: Interrupt Vector or Channel Control or Time Constant							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	CTC0	R/W	vvvvvxx0	Reading this address returns the value of channel 0's down-counter. Writing an even value sets bits 7-3 of the interrupt vector for all four channels.
			ecputfr1	Writing an odd value loads a Channel Control word, in which: Bit 7 enables (1) or disables (0) interrupts from this channel. Bit 6 selects Counter (1) or Timer (0) mode. In Timer mode, bit 5 selects whether the prescaler divides PHI by 256 (1) or 16 (0). Bit 4 selects rising (1) or falling (0) edges on CLK/TRG0 to decrement the down-counter in Counter mode, or to trigger down-counting in Timer mode with bit 3 set. In Timer mode, bit 3 controls whether the channel starts down-counting immediately after software loads the Time Constant (0), or waits for the edge selected by bit 4, on CLK/TRG0, before counting down (1). Writing a 1 to bit 2 indicates that the channel loads the next byte written to this address, into its Time Constant register. If the channel is stopped, or bit 1 is 1, the value is also loaded into its down-counter. Writing a 1 to bit 1 stops the channel if it has been running, and resets it in any case. If bits 2-1 are 11, the channel is re-enabled after software writes the new Time Constant.

**TABLE 78. CTC1 (xxD1 OR 00D1H) CTC1**

Bit	7	6	5	4	3	2	1	0
Bit/Field	Read: value of CTC1 down-counter Write: Channel Control or Time Constant							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate**NOTE:** This address operates as described for CTC0 above, except:

1. Software cannot write an interrupt vector to this address, and
2. The pin associated with CTC1 is CLK/TRG1

TABLE 79. CTC2 (xxD2 OR 00D2H) CTC2

Bit	7	6	5	4	3	2	1	0
Bit/Field	Read: value of CTC2 down-counter Write: Channel Control or Time Constant							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate**NOTE:** This address operates as described for CTC0 above, except:

1. Software cannot write an interrupt vector to this address, and
2. The pin associated with CTC2 is CLK/TRG2

TABLE 80. CTC3 (xxD3 OR 00D3H) CTC3

Bit	7	6	5	4	3	2	1	0
Bit/Field	Read: value of CTC3 down-counter Write: Channel Control or Time Constant							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate**NOTE:** This address operates as described for CTC0 above, except:

1. Software cannot write an interrupt vector to this address, and
2. The pin associated with CTC3 is CLK/TRG3

PROGRAMMABLE RELOAD TIMER (PRT) REGISTERS

See section “Programmable Reload Timers (PRTs)”, on page 69, for more about these registers.

TABLE 81. PRT0 TIMER DATA REGISTER LOW (000CH) TMDROL

Bit	7	6	5	4	3	2	1	0
Bit/Field	LS Byte of PRT0 Counter							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	LS Byte of PRT0 Counter	R/W		The LS 8 bits of PRT0's down-counter.

TABLE 82. PRT0 TIMER DATA REGISTER HIGH (000DH) TMDROH

Bit	7	6	5	4	3	2	1	0
Bit/Field	MS Byte of PRT0 Counter							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	MS Byte of PRT0 Counter	R/W		The MS 8 bits of PRT0's down-counter.

TABLE 83. PRT0 RELOAD REGISTER LOW (000EH) RLDR0L

Bit	7	6	5	4	3	2	1	0
Bit/Field	LS Byte of PRT0 Reload Value							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

NOTE: R = Read W = Write X = Indeterminate



Bit Position	Bit/Field	R/W	Value	Description
7-0	LS Byte of PRT0 Reload Value	R/W		The LS 8 bits of the value that is loaded into PRT0's down-counter, when it is decremented to 0.

TABLE 84. PRT0 RELOAD REGISTER HIGH (000FH) RLDR0H

Bit	7	6	5	4	3	2	1	0
Bit/Field	MS Byte of PRT0 Reload Value							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	MS Byte of PRT0 Reload Value	R/W		The MS 8 bits of the value that is loaded into PRT0's down-counter, when it is decremented to 0.

TABLE 85. TIMER CONTROL REGISTER (0010H) TCR

Bit	7	6	5	4	3	2	1	0
Bit/Field	TIF1	TIF0	TIE1	TIE0	TOC		TDE1	TDE0
R/W	R	R	R/W	R/W	?		R/W	R/W
Reset	0	0	0	0	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-6	TIF1,0	R		One of these status bits is set when a PRT decrements its down-counter to 0. It is cleared when software has read this register and either byte of the TMDR.
5-4	TIE1,0	R/W		If one of these bits is 1, the PRT requests an interrupt when its down-counter has counted down to 0, and it has set its TIF bit.



Bit Position	Bit/Field	R/W	Value	Description
3-2	TOC	R/W		This field controls what happens on the TOUT pin when PRT1 counts down to 0:
			00	No change
			01	TOUT is toggled
			10	TOUT goes Low
			11	TOUT goes High
1-0	TDE1,0	R/W	0	The corresponding PRT is stopped.
			1	The corresponding PRT is running.

**TABLE 86. TIMER PRESCALE REGISTER (0011H) TPR**

Bit	7	6	5	4	3	2	1	0
Bit/Field	PRT1 Prescale Select				PRT0 Prescale Select			
R/W	R/W				R/W			
Reset	1	1	1	1	1	1	1	1

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-4, 3-0	PRT1-0 Prescale Select	R/W		These fields select the down counter clock for PRT1 and PRT0 respectively:
			0000	PHI
			0001	PHI/2
			0010	PHI/4
			0011	PHI/8
			0100	PHI/16
			0101	PHI/32
			0100	PHI/64
			0111	PHI/128
			1000	PHI/256
			1001	PHI/512
			1010	PHI/1024
			1011	PHI/2048
			1100	PHI/4096
			1101	PHI/8192
			1110	PHI/16384
			1111	PHI/20 (Default after Reset)

TABLE 87. PRT1 TIMER DATA REGISTER LOW (0014H) TMDR1L

Bit	7	6	5	4	3	2	1	0
Bit/Field	LS Byte of PRT1 Counter							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	LS Byte of PRT1 Counter	R/W		The LS 8 bits of PRT1's down-counter.

TABLE 88. PRT1 TIMER DATA REGISTER HIGH (0015H) TMDR1H

Bit	7	6	5	4	3	2	1	0
Bit/Field	MS Byte of PRT1 Counter							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	MS Byte of PRT1 Counter	R/W		The MS 8 bits of PRT1's down-counter.

TABLE 89. PRT1 RELOAD REGISTER LOW (0016H) RLDR1L

Bit	7	6	5	4	3	2	1	0
Bit/Field	LS Byte of PRT1 Reload Value							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	LS Byte of PRT1 Reload Value	R/W		The LS 8 bits of the value that is loaded into PRT1's down-counter, when it is decremented to 0.

TABLE 90. PRT1 RELOAD REGISTER HIGH (0017H) RLDR1H

Bit	7	6	5	4	3	2	1	0
Bit/Field	MS Byte of PRT1 Reload Value							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	MS Byte of PRT1 Reload Value	R/W		The MS 8 bits of the value that is loaded into PRT1's down-counter, when it is decremented to 0.



SERIAL I/O (SIO) REGISTERS

See section “Serial I/O Channels (SIOs)”, on page 71, for more about these registers.

TABLE 91. SIO A DATA (xxD8 OR 00D8H) SIOAD

Bit	7	6	5	4	3	2	1	0
Bit/Field	Write: Tx character, Read: Rx character							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Data	R/W		A character to be transmitted can be written to this address, only when bit 2 of RR0 is 1. If bits 6-5 of WR5 specify less than 8 bits/character, only the LS bits of the value written is sent. This address is read only when bit 0 of RR0 is 1, in which case it returns the oldest available received character.

TABLE 92. SIO A CONTROL (xxD9 OR 00D9H) SIOAC

Bit	7	6	5	4	3	2	1	0
Bit/Field	Write: WR0,1,3-7 per WR0 bits 2-0, Read: RR0-1 per WR0 bits 2-0							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	WR or RR	R/W		Writing to this address loads the Write Register selected by bits 2-0 of WR0, while reading from this address returns the contents of the Read Register selected by bits 2-0 of WR0. Bits 2-0 of WR0 are cleared to 000 by reset and after any access to this address except a write to WR0, so that every other access to this address is to WR0 or RR0. The following tables describe the contents of the Write and Read Registers.

TABLE 93. SIO WR0 (WRITE TO 0xD9, 0x0D9, 0xDB, OR 0x0DBH WITH WR0 2-0 = 000)

Bit	7	6	5	4	3	2	1	0
Bit/Field	CRC/EOM command		Command			Register address		
R/W	W		W			W		
Reset	0	0	0	0	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-6	CRC/EOM command	W	00	Null command
			01	Reset Rx CRC Checker
			10	Reset Tx CRC Generator
			11	Reset Tx Underrun/EOM latch
5-3	Command	W	000	Null command
			001	Send Abort
			010	Reset External/Status Interrupts
			011	Channel Reset
			100	Enable Interrupt on Next Rx Character
			101	Reset Tx Interrupt Pending
			110	Error Reset
			111	Return From Interrupt (software RETI)
2-0	Register Address	W	000	WR0 or RR0
			001	WR1 or RR1
			010	WR2 or RR2 (both in channel B only)
			011	WR3 (Rx Control)
			100	WR4 (Channel Control)
			101	WR5 (Tx Control)
			110	WR6 (Sync or Address)
			111	WR7 (Sync or Flag)

NOTE: Bits 2-0 are cleared to 000 after any access to this address other than a write to WR0, so that the next access is to WR0 or RR0.



TABLE 94. SIO WR1 (WRITE TO xxD9, 00D9, xxDB, OR 00DBH WITH WR0 2-0 = 001)

Bit	7	6	5	4	3	2	1	0
Bit/Field	WAIT/RDY function			Rx interrupt mode		Status Affects Vector	Tx IE	External /Status IE
R/W	W			W		W	W	W
Reset	0	0	0	0	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-5	WAIT/RDY function	W	00x 01x 100 101 110 111	WAIT/RDY pin is not driven WAIT/RDY pin is High L = Tx not full, H = Tx full L = Rx data available, H = Rx empty L during write to Data address = Tx full, not driven = no write or Tx not full L during read f/Data address = Rx empty, not driven = no read or Rx available
4-3	Rx interrupt mode	W	00 01 10 11	Rx interrupts disabled On first character All characters, Parity Error is Special All characters, Parity Error not Special
2	Status Affects Vector (channel B only)	W	0 1	Channel returns value in ch B RR2 for all interrupts Channel returns bits 7-4 and 0 from ch B RR2, bits 3-1 indicate highest priority interrupt.
1	Tx IE	W		Transmit Interrupt enable
0	External/Status IE	W		External/Status Interrupt enable

TABLE 95. SIO WR3 (WRITE TO `xxD9`, `00D9`, `xxDB`, OR `00DBH` WITH `WR0 2-0 = 011`)

Bit	7	6	5	4	3	2	1	0
Bit/Field	Rx Bits/Character		Auto Enables	Hunt Mode	Rx CRC	HDLC Addr	Strip Sync	Rx Enable
R/W	W		W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-6	Rx Bits/Character	W	00	5 data bits/character
			01	6 data bits/character
			10	7 data bits/character
			11	8 data bits/character
5	Auto Enables	W	0	State of $\overline{\text{DCD}}$ and $\overline{\text{CTS}}$ pins can be read in RR0, but do not affect the channel.
			1	$\overline{\text{DCD}}$ Low enables the Receiver, and $\overline{\text{CTS}}$ Low enables the Transmitter.
4	Hunt Mode	W		In Synchronous modes, writing a 1 to this bit sets bit 4 in RR0 and forces the Receiver into Hunt mode, in which it searches for the Sync or Flag pattern in WR7 (and WR6 in Bisync mode).
3	Rx CRC	W		In Classic Sync reception, software can set or clear this bit to include (1) or exclude (0) each received character from the CRC checking for the message. In these modes, the receiver includes an extra 8-bit delay between when each character is placed in the Rx FIFO, and when it would start being shifted into the Rx CRC checker. This means that software has up to 8 bit times, from when each character is placed in the Rx FIFO, to set or clear this bit for the character. In HDLC/SDLC mode this bit is 1 because all characters in each frame are included in the CRC.
2	HDLC address	W		If this bit is 1 in HDLC/SDLC mode, the receiver matches the first character of each received frame against the character in WR6, and discards/ignores frames that do not match.



Bit Position	Bit/Field	R/W	Value	Description
1	Strip Sync	W		If this bit is 1 in Classic Sync modes, Sync characters are not placed in the Rx FIFO. This bit is 1 only at the start of a message, because software needs to detect embedded Syncs so that it can exclude them from the CRC.
0	Rx Enable	W		A 1 in this bit enables the receiver. This bit must set only after all other Rx control bits have been initialized, which means that WR3 needs to be written twice during device initialization.

TABLE 96. SIO WR4 (WRITE TO `xxD9`, `00D9`, `xxDB`, OR `00DBH` WITH `WR0 2-0 = 100`)

Bit	7	6	5	4	3	2	1	0
Bit/Field	Clock Mode		Sync Mode		Channel mode		Parity Sense	Parity Enable
R/W	W		W			W		
Reset	0	0	0	0	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-6	Clock Mode	W	00 01 10 11	X1 clock (Sync or isochronous) /16 clock (Async) /32 clock (Async) /64 clock (Async)
5-4	Sync Mode	W	00 01 10 11	When bits 3-2 are 00, this field selects the Synchronous protocol: 8-bit Sync (monosync) 16-bit Sync (Bisync) HDLC/SDLC External Sync
3-2	Channel Mode	W	00 01 10 11	Synchronous mode Async w/ 1 Stop bit Async w/ 1.5 Stop bits Async s/ 2 Stop bits
1	Parity Sense	W		If bit 0 is 1, this bit selects between even (1) or odd (0) parity checking and generation.

Bit Position	Bit/Field	R/W	Value	Description
0	Parity Enable	W		A 1 in this bit makes the transmitter send, and the receiver sample and check, a parity bit in each character, after the number of data bits selected in WR5 and WR3 respectively.

TABLE 97. SIO WR5 (WRITE TO 00D9, 00DB, OR 00DBH w/WR0 2-0 = 101)

Bit	7	6	5	4	3	2	1	0
Bit/Field	DTR	Tx bits/character		Send Break	Tx Enable	CRC Select	RTS	Tx CRC
R/W	W	W		W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7	DTR	W		This bit controls the channel's $\overline{\text{DTR}}$ pin, 0 for negation/high, and 1 for assertion/low.
6-5	Tx bits/character	W	00 01 10 11	5 or less bits/Tx character 6 bits/Tx character 7 bits/Tx character 8 bits/Tx character
4	Send Break	W		A 1 in this bit forces the TxD pin Low (space, 0) to send a Break sequence in Async or isochronous mode. A 0 puts TxD under the control of the transmitter.
3	Tx Enable	W		A 1 in this bit enables the Transmitter. This bit must be set only after all other Tx parameters have been set up, which means that WR5 must be written twice during device initialization.
2	CRC Select	W	0 1	Tx and Rx CRC use CCITT polynomial (required in HDLC/SDLC mode) Tx and Rx CRC use CRC-16 polynomial.
1	RTS	W		This bit controls the channel's $\overline{\text{RTS}}$ pin, 0 for negation/high, and 1 for assertion/low.



Bit Position	Bit/Field	R/W	Value	Description
0	Tx CRC	W		In Classic Sync transmission, software can set this bit when it detects RR0 bit 2 set to solicit a new Tx character, to include (1) or exclude (0) the next character it writes to the Data address, in CRC generation for the message. This bit must remain 1 in HDLC/SDLC mode, because all characters in each frame are included in the CRC.

TABLE 98. SIO WR6 (WRITE TO **xxD9, **00D9**, **xxDB**, OR **00DBH** WITH **WR0 2-0 = 110**)**

Bit	7	6	5	4	3	2	1	0
Bit/Field	Monosync/External: Tx Sync, Bisync: 1st 8 Sync bits, HDLC: address							
R/W	W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Sync or Address	W		In monosync or External Sync mode, this register holds the Tx Sync character. In Bisync mode, it holds the first 8 bits of the Tx/Rx Sync pattern. In HDLC/SDLC mode with WR3 bit 2 set, it holds the address byte against which the receiver matches the first character of each frame.

TABLE 99. SIO WR7 (WRITE TO 0xD9, 0x0D9, 0xDB, OR 0x0DBH WITH WR0 2-0 = 111)

Bit	7	6	5	4	3	2	1	0
Bit/Field	Monosync: Rx Sync, Bisync: 2nd 8 Sync bits, HDLC: Flag							
R/W	W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Sync or Flag	W		In monosync mode, this register holds the Rx Sync character. In Bisync mode, it holds the last 8 bits of the Tx/Rx Sync pattern. In HDLC/SDLC mode it must be written with the Flag pattern 01111110.

TABLE 100. SIO RR0 (READ F/0xD9, 0x0D9, 0xDB, OR 0x0DBH WITH WR0 2-0 = 000)

Bit	7	6	5	4	3	2	1	0
Bit/Field	Break/Abort	Tx Under run/EOM	$\overline{\text{CTS}}$	Sync/Hunt	$\overline{\text{DCD}}$	Tx Empty	Int Pendg (ch A only)	Rx Avail
R/W	R	R	R	R	R	R	R	R
Reset	0	1	N/A	N/A	N/A	1	0	0

NOTE: R = Read W = Write X = Indeterminate



Bit Position	Bit/Field	R/W	Value	Description
7	Break/Abort	R		<p>In Async or isochronous mode, the receiver sets this bit when it detects a Break sequence, namely an all-0 character with a Framing Error.</p> <p>In HDLC/SDLC mode the receiver sets this bit when it detects an Abort sequence, seven or more consecutive ones when a frame is in progress. In either case, if External/Status interrupts are enabled, setting this bit results in an interrupt. When this bit changes in either direction, software unlatches it by writing a Reset External/Status command to WRO. This action also enables the receiver to interrupt again when it clears this bit because RxD returns to 1 in ASYNC mode or 0 in HDLC mode.</p>
6	Tx Underrun/ EOM	R		<p>This bit is set by reset, and when a Tx underrun occurs. It is cleared if software writes a Reset Tx Underrun/EOM command to WRO bits 7-6.</p> <p>In Synchronous modes, if this bit is 1 when the transmitter underruns, that is, when it needs a byte to transmit but neither software nor a DMA channel has provided one, it sends a closing Sync or Flag. This process generally (65535 times out of 65536) causes a receive CRC error at the remote node. If this bit is 0 when the transmitter underruns, it sends the accumulated CRC followed by a closing Sync or Flag.</p> <p>In HDLC/SDLC mode, software can override the sending of the CRC and Flag, in case of an unintentional Underrun, by issuing a Send Abort command. When software detects a 0-to-1 transition of this bit, it unlatches the bit by writing a Reset External/Status command to WRO.</p>
5	CTS	R		<p>This bit reflects the state of the channel's $\overline{\text{CTS}}$ pin, with a 1 indicating an active/low state and 0 indicating inactive/high. When this bit changes in either direction, software unlatches it by writing a Reset External/Status command to WRO.</p>

Bit Position	Bit/Field	R/W	Value	Description
4	Sync/Hunt	R		<p>In Async or External Sync mode, this bit reflects the state of the channel's SYNC pin, with a 1 indicating an active/low state and 0 indicating inactive/high.</p> <p>In Classic Sync or HDLC/SDLC modes, this bit is set whenever the receiver is disabled, or when software writes a 1 to WR3 bit 4, and is cleared when the receiver detects a Sync or Flag pattern. When this bit changes in either direction, software unlatches it by writing a Reset External/Status command to WR0.</p>
3	DCD	R		This bit reflects the state of the channel's $\overline{\text{DCD}}$ pin, with a 1 indicating an active/low state and 0 indicating inactive/high. When this bit changes in either direction, software unlatches it by writing a Reset External/Status command to WR0.
2	Tx Empty	R		This bit is 1 when it is OK to write a transmit character to the channel's data address.
1	Interrupt Pending	R		This bit is 1 in channel A, if there are any interrupting conditions pending in either channel. This bit is always 0 in channel B.
0	Rx Available	R		This bit is 1 when there is/are one or more received characters available to be read from the channel's data address.

TABLE 101. SIO RR1 (READ F/XXD9, 00D9, XXDB, OR 00DBH WITH WR0 2-0 = 001)

Bit	7	6	5	4	3	2	1	0
Bit/Field	EOF	CRCE/FE	Rx Overrun	Parity Error	HDLC/SDLC Rx Residue			All Sent
R/W	R	R	R	R	R			R
Reset	0	0	0	0	0	1	1	0

NOTE: R = Read W = Write X = Indeterminate



Bit Position	Bit/Field	R/W	Value	Description
7	HDLC/SDLC End of Frame	R		This bit is set only in HDLC/SDLC mode, when a closing Flag is detected. It is cleared if software writes an Error Reset command to WR0, and also when the first character of the next frame arrives.
6	CRC/ Framing Error	R		<p>In Async or isochronous mode, this bit is 1 if a character is available to be read from the channel's data address, in which the receiver sampled the Stop bit as 0/space/Low.</p> <p>In HDLC/SDLC mode, a 1 in this bit indicates a CRC error when accompanied by a 1 in bit 7.</p> <p>In Classic Sync modes, when software reads a message-terminating character from the channel's data address, if the Rx character length in WR3 is less than 8 bits, WR3 is written to change it to 8 bits/character. Then software waits for 4 more Rx Available flags, the first two characters of which are the CRC.</p>
6 (cont.)				In Classic Sync modes, this bit validly reflects CRC correctness when RR0 bit 0 is 1 for the 2nd character after the last CRC character. Regardless of the mode, this bit can be cleared by writing an Error Reset command to WR0, but the bit is not latched and is updated for each received character
5	Rx Overrun	R		In any mode, the receiver sets this bit if software or a DMA channel has not read received characters from the channel's data address in a timely manner, so that another character arrives when the 3-character Rx FIFO is full. This bit is set when the character that overwrote its predecessor comes to the top of the Rx FIFO, and remains set until it is cleared by an Error Reset command written to WR0.

Bit Position	Bit/Field	R/W	Value	Description
4	Parity Error	R		If bit 0 of WR4 is 1, the receiver sets this bit when a character with parity different from the sense defined by WR4 bit 1, comes to the top of the Rx FIFO. When set, this bit remains 1 until it is cleared by an Error Reset command written to WRO.
3-1	HDLC/SDLC Rx Residue	R		In HDLC/SDLC mode, when bit 7 of this register is 1, these bits indicate the number and placement of the final data bits and CRC, in the last 3 or 4 characters of the frame. The meaning of this value depends on the number of bits per character, and is illustrated in Figures 18 through 21.
0	All Sent	R		In Async and isochronous modes, this bit is 1 when all characters written to the channel's data address have been sent.

**TABLE 102. SIO B DATA (xxDA OR 00DAH) SIOBD**

Bit	7	6	5	4	3	2	1	0
Bit/Field	Write: Tx character, Read: Rx character							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

NOTE: This address operates as described for SIO A Data above, except that it deals with transmit data to be sent on the TxDB pin, and received data from the RxDB pin.

TABLE 103. SIO B CONTROL (xxDB OR 00DBH) SIOBC

Bit	7	6	5	4	3	2	1	0
Bit/Field	Write: WR0-7 per WR0 bits 2-0, Read: RR0-2 per WR0 bits 2-0							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

NOTE: This address operates as described for SIO A Control above, providing access to the same Write and Read Registers for SIO channel B, except:

1. The Return From Interrupt command cannot be issued in channel B WR0.
2. The Status Affects Vector bit (WR1 bit 2) is only effective in channel B WR1.
3. The Interrupt Pending bit (RR0 bit 1) is always 0 in channel B RR0.
4. The following two registers are only accessible in channel B:

TABLE 104. SIO B WR2 (WRITE TO xxDB OR 00DBH WITH WR0 2-0 = 010)

Bit	7	6	5	4	3	2	1	0
Bit/Field	Interrupt Vector							
R/W	W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Interrupt Vector	W		These bits are returned as the vector during interrupt acknowledge cycles in which either SIO channel is the highest priority requesting device. If bit 2 of channel B WR1 is 1, bits 3-1 contain a code identifying the highest priority SIO interrupt pending, as described for RR2. Bit 0 must be 0 for mode 2 interrupts.

TABLE 105. SIO B RR2 (READ FROM XXDB OR 00DBH WITH WR0 2-0 = 010)

Bit	7	6	5	4	3	2	1	0
Bit/Field	Interrupt Vector				Vector or Type Code			IV 0
R/W	R				R			R
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-4	Interrupt Vector	R		These bits are identical to bits 7-4 of the latest value written to WR2.
3-1	Vector or Type Code	R		If bit 2 in channel B WR1 is 0, these bits are identical to bits 3-1 of the latest value written to WR2. If bit 2 is 1, these bits identify the highest priority interrupt pending in the SIO channels, or 001 if no interrupt is pending: 000 Channel B Transmit (lowest priority) 001 Channel B External/Status 010 Channel B Rx Character Available 011 Channel B Special Receive Condition Channel A Transmit 100 Channel A External/Status 101 Channel A Rx Character Available 110 Channel A Special Receive Condition 111
0	Vector	R		This bit is identical to bit 0 of the latest value written to WR2.

ASYNC SERIAL COMMUNICATIONS INTERFACE (ASCI) REGISTERS

See Async Serial Communications Interface (ASCI), which starts on page 94, for more detail about these registers.

**TABLE 106. ASCI0 CONTROL REGISTER A (0000H) CNTLA0**

Bit	7	6	5	4	3	2	1	0
Bit/Field	MPE	RE	TE	RTS0	MPBR/ EFR	MOD2	MOD1	MOD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	X	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7	Multi-Processor Mode Enable	R/W		If this bit and the MP bit in CNTLB are both 1, only received characters having a 1 in an additional bit between the last data bit and the stop bit, is placed in the Rx FIFO. If either the MP bit or this bit is 0, all received characters are placed in the Rx FIFO.
6	Receive Enable	R/W		A 1 in this bit enables the receiver. Writing a 0 summarily stops reception.
5	Transmit Enable	R/W		A 1 in this bit enables the transmitter. Writing a 0 summarily stops transmission.
4	RTS0	R/W		this bit controls the $\overline{\text{RTS0}}$ output.
3	MP Bit Rcv/ Error Flag Reset	R/W		Reading this bit returns the value of the "multiprocessor" bit. Read this register before reading the RDR. Writing a 0 to this bit clears the OVRN, FE, PE, and Break Detect bits. Writing a 1 has no effect.
2	MOD2	R/W	0 1	7 bit data (Tx and Rx) 8 bit data
1	MOD1	R/W	0 1	No parity (Tx and Rx) Parity generated (Tx), checked (Rx)
0	MOD0	R/W	0 1	1 Stop bit Transmitted 2 Stop bits Transmitted

TABLE 107. ASCI1 CONTROL REGISTER A (0001H) CNTLA1

Bit	7	6	5	4	3	2	1	0
Bit/Field	MPE	RE	TE	Reserved	MPBR/ EFR	MOD2	MOD1	MOD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	X	0	0	0

NOTE: R = Read W = Write X = Indeterminate

All bits in this register are as described in the previous table.

TABLE 108. ASCIO CONTROL REGISTER B (0002H) CNTLB0

Bit	7	6	5	4	3	2	1	0
Bit/Field	MPBT	MP	$\overline{\text{CTS}}$ / PS	PEO	DR	Speed Select		
R/W	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	X	0	0	0	0	1	1	1

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7	Multi-Processor Bit Tx	R/W		If the MP bit (next) is 1, this bit defines the value to send in the MP bit with the next character written to the Transmit Data Register.
6	Multi-Processor Mode	R/W		If this bit is 1, the ASCI sends, and expects to receive, an extra bit after the last data bit. This bit is 1 to identify address characters that begin frames, or 0 to identify following data characters.
5	$\overline{\text{CTS}}$ /PS	R/W		Reading this pin returns the state of the CTS pin (0 = low, 1 = high). For writing, this bit is PS. If bits 2-0 in this register are not 111 and the BRG Mode bit in the Extension Control register is 0, PS determines whether the PHI clock is "Prescaled" by 10 (for 0) or 30 (for 1) as the first stage in ASCI clocking.
4	Parity Even/Odd	R/W		If MOD1 (CNTLA bit 1) is 1, this bits selects whether parity is generated and checked as even (for 0) or odd (for 1).



Bit Position	Bit/Field	R/W	Value	Description
3	DR	R/W	0	The ASCI divides the its basic clock by 16 to obtain its bit rate.
			1	The ASCI divides the its basic clock by 64 to obtain its bit rate
2-0	Speed Select	R/W	111 (other)	Do not program this value. If the BRG0 Mode bit is 1, the output of the new BRG is the basic clock of the ASCI. If BRG0 Mode is 0, these bits determine what the output of the Prescaler is divided by, to obtain basic clock for the ASCI: Used as is
			000	/2
			001	/4
			010	/8
			011	/16
			100	/32
			101	/64
			110	In any case, the basic clock is divided by 16 or 64 to obtain the ASCI bit rate.

TABLE 109. ASCI1 CONTROL REGISTER B (0003H) CNTLB1

Bit	7	6	5	4	3	2	1	0
Bit/Field	MPBT	MP	PS	PEO	DR	Speed Select		
R/W	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	X	0	0	0	0	1	1	1

NOTE: R = Read W = Write X = Indeterminate

All bits in this register are as described in the preceding table, except that bit 5 has no function in write operations on the Z80S188 ASCI1.

TABLE 110. ASCIO STATUS REGISTER (0004H) STAT0

Bit	7	6	5	4	3	2	1	0
Bit/Field	RDRF	OVRN	PE	FE	RIE	DCD0	TDRE	TIE
R/W	R	R	R	R	R/W	R	R	R/W
Reset	0	0	0	0	0	pin	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7	RDRF	R		The Z80S188 sets this bit when a character is received. Reading the last received byte from the Rx FIFO clears this bit. See Note below.
6	OVRN	R		This bit is set when the last character received before an Overrun condition comes to the top of the Rx FIFO. It is cleared when software writes a 0 to the EFR bit in CNTLO. See Note below.
5	PE	R		This bit is set if parity is enabled, and a character with a Parity Error comes to the top of the Rx FIFO. It is cleared when software writes a 0 to the EFR bit in CNTLO. See Note below.
4	FE	R		This bit is set if a character with a Framing Error (one in which the Stop bit was sampled as 0) comes to the top of the Rx FIFO. It is cleared when software writes a 0 to the EFR bit in CNTLO. See Note below.
3	RIE	R/W		If this bit is 1, the ASCI requests an interrupt when any of the flags OVRN, PE, FE, or Break Detect is set, or if bit 7 of the Extension Control register is 0 and RDRF is set, or if bit 6 of the Extension Control register is 0 and $\overline{\text{DCD0}}$ is Low.
2	DCD0	R		This bit is 1 whenever the $\overline{\text{DCD0}}$ pin is High. When $\overline{\text{DCD0}}$ goes Low, the next read of this register returns a 1, but the next read returns a 0 if $\overline{\text{DCD0}}$ is still Low. Reset sets this bit according to the state of the pin.



Bit Position	Bit/Field	R/W	Value	Description
1	TDRE	R		This bit is cleared when software writes a character to the TDR. It is set when the character leaves the TDR for transmission, by reset, and in I/O STOP mode. It is cleared if bit 5 of the Extension Control register is 0 and CTS0 is High.
0	TIE	R/W		If this bit is 1, the ASCI requests an interrupt when TDRE is 1.

NOTE: The RDRF, OVRN, PE, and FE bits are cleared by Reset, during I/O Stop mode, and for ASCI0, if the DCD Disable bit in the Extension Control register is 0 and $\overline{DCD0}$ is High.

TABLE 111. ASCI1 STATUS REGISTER (0005H) STAT1

Bit	7	6	5	4	3	2	1	0
Bit/Field	RDRF	OVRN	PE	FE	RIE	Reserv ed	TDRE	TIE
R/W	R	R	R	R	R/W	R	R	R/W
Reset	0	0	0	0	0	pin	0	0

NOTE: R = Read W = Write X = Indeterminate

This register is as described in the previous Table, except that bit 2 has no function on the Z80S188 ASCI1.

TABLE 112. ASCI0 Tx DATA REGISTER (0006H) TDR0

Bit	7	6	5	4	3	2	1	0
Bit/Field	Character to Tx							
R/W	W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Tx character	R/W		Software can write a character to be transmitted to this register, whenever the TDRE flag in STAT0 is 1.

TABLE 113. ASCI1 Tx DATA REGISTER (0007H) TDR1

Bit	7	6	5	4	3	2	1	0
Bit/Field	Character to Tx							
R/W	W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Tx character	R/W		Software can write a character to be transmitted to this register, whenever the TDRE flag in STAT1 is 1.

TABLE 114. ASCI0 Rx DATA REGISTER (0008H) RDR0

Bit	7	6	5	4	3	2	1	0
Bit/Field	Received Character							
R/W	R							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Rx character	R/W		Whenever the RDRF flag in STAT0 is 1, software can read a received character from this register.

TABLE 115. ASCI1 Rx DATA REGISTER (0009H) RDR1

Bit	7	6	5	4	3	2	1	0
Bit/Field	Received Character							
R/W	R							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	Rx character	R/W		Whenever the RDRF flag in STAT1 is 1, software can read a received character from this register.

**TABLE 116. ASCI0 EXTENSION CONTROL REGISTER (0012H) ASEXTO**

Bit	7	6	5	4	3	2	1	0
Bit/Field	RDID	$\overline{\text{DCD0}}$ Disable	$\overline{\text{CTS0}}$ Disable	X1 Clock	BRG Mode	Start IE	Rx Break	Tx Break/ TxEnd
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
Reset	0	0	0	0	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7	Rx Data Interrupt Disable	R/W		If RIE (STAT bit 3) and this bit are both 1, the ASCI requests receive interrupts only when OVRN, PE, or FE is set. If RIE is 1 and this bit is 0, interrupts are also requested when RDRF is set (for each received character). If RIE is 0 this bit has no effect.
6	$\overline{\text{DCD0}}$ Disable	R/W	0 1	$\overline{\text{DCD0}}$ auto-enables the receiver $\overline{\text{DCD0}}$ has no effect on the receiver
5	$\overline{\text{CTS0}}$ Disable	R/W	0 1	$\overline{\text{CTS0}}$ auto-enables the transmitter $\overline{\text{CTS0}}$ has no effect on the transmitter
4	X1 Clock	R/W	0 1	The clock on the CKA0 pin is divided by 16 or 64 to obtain the ASCI bit clock. The clock on the CKA0 pin is used as the ASCI bit clock.
3	BRG Mode	R/W	0 1	The SS bits determine the factor by which the Prescaler output is divided, to obtain the ASCI's basic clock. The ASCI's basic clock comes from the new BRG.
2	Start IE	R/W		If this bit and RIE are both 1, the ASCI requests an interrupt when it detects the start of a start bit, for auto-bauding. Writing a 0 to this bit after such an interrupt occurs, clears the StartIE interrupt request.
1	RxBreak	R		This bit is 1 if the receiver has detected a Break condition, that is, if all bits in a character, including the stop bit, are 0. The all-0 character is placed in the Rx FIFO if there is space, but the receiver does not assemble any more characters until the RxA pin returns to High.

Bit Position	Bit/Field	R/W	Value	Description
0	TxBreak/ TxEnd	R/W		Writing a 1 to this bit makes the transmitter drive TXA Low to send a Break condition, until software writes a 0 to this bit. This bit reads as 0 while a character is being transmitted, but goes to 1 when the number of stop bits selected by MOD0 in CNTLA have been sent.

TABLE 117. ASCII1 EXTENSION CONTROL REGISTER (0013H) ASEXT1

Bit	7	6	5	4	3	2	1	0
Bit/Field	RDID	Reserved		X1 Clock	BRG Mode	Start IE	Rx Break	Tx Break/ TxEnd
R/W	R/W	N/A		R/W	R/W	R/W	R	R/W
Reset	0	X	X	0	0	0	0	0

NOTE: R = Read W = Write X = Indeterminate

This register is as described in the previous table, except that bits 6–5 have no function for Z80S188 ASCII1.

TABLE 118. ASCII0 TIME CONSTANT LOW (001AH) ASTCOL

Bit	7	6	5	4	3	2	1	0
Bit/Field	LS Byte of Time Constant							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	LS byte of Time Constant	R/W		The LS 8 bits of the ASCII0 Baud Rate Generator's Time Constant.

**TABLE 119. ASCI0 TIME CONSTANT HIGH (001BH) ASTC0H**

Bit	7	6	5	4	3	2	1	0
Bit/Field	MS Byte of Time Constant							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	MS Byte of Time Constant	R/W		The MS 8 bits of the ASCI0 Baud Rate Generator's Time Constant.

TABLE 120. ASCI1 TIME CONSTANT LOW (001CH) ASTC1L

Bit	7	6	5	4	3	2	1	0
Bit/Field	LS Byte of Time Constant							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	LS Byte of Time Constant	R/W		The LS 8 bits of the ASCI1 Baud Rate Generator's Time Constant.

TABLE 121. ASCI1 TIME CONSTANT HIGH (001DH) ASTC1H

Bit	7	6	5	4	3	2	1	0
Bit/Field	MS Byte of Time Constant							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0	MS Byte of Time Constant	R/W		The MS 8 bits of the ASCI1 Baud Rate Generator's Time Constant.

CLOCKED SERIAL I/O (CSI/O) REGISTERS

See the section “Clocked Serial Input/Output Module (CSI/O)”, on page 105, for more about these registers.

TABLE 122. CSI/O CONTROL REGISTER (000AH) CNTR

Bit	7	6	5	4	3	2	1	0
Bit/Field	EF	EIE	RE	TE		Speed Select (SS)		
R/W	R	R/W	R/W	R/W	N/A	R/W		
Reset	0	0	0	0	X	1	1	1

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7	End Flag (EF)	R		The CSI/O sets this bit to 1 when it completes sending or receiving a byte. It clears this bit when software reads or writes the TRDR, on Reset and during I/O Stop mode.
6	End Interrupt Enable (EIE)	R/W		If this bit is 1, the CSI/O requests an interrupt when it completes sending or receiving a byte and sets EF.
5	Receive Enable (RE)	R/W		Write a 1 to this bit to start a CSI/O receive operation. If the SS bits are 111, the CSI/O waits for 8 clock pulses on CKS, otherwise it outputs 8 clock pulses on CKS. In either case, it clocks data on RXS into the TRDR after each falling edge on CKS. After capturing the 8th bit, it clears this bit and sets EF.
4	Transmit Enable (TE)	R/W		Write a 1 to this bit to start CSI/O transmission. If the SS bits are 111, the CSI/O waits for 8 clock pulses on CKS, otherwise it outputs 8 clock pulses on CKS. In either case, it clocks data onto TXS after each falling edge on CKS. After sending 8 bits, the CSI/O clears this bit and sets EF.



Bit Position	Bit/Field	R/W	Value	Description
2-0	Speed Select (SS)	R/W		If these bits are 111, as they are after a Reset, the CSI/O takes external clocking from the CKS pin. Otherwise, it drives a clock onto CKS, that it derives from PHI as follows:
			000	PHI/20
			001	PHI/40
			010	PHI/80
			011	PHI/160
			100	PHI/320
			101	PHI/640
			110	PHI/1280

TABLE 123. CSI/O DATA REGISTER (000BH) TRDR

Bit	7	6	5	4	3	2	1	0
Bit/Field	Byte to Send (W) or Received Byte (R)							
R/W	R/W							
Reset	X	X	X	X	X	X	X	X

NOTE: R = Read W = Write X = Indeterminate

Bit Position	Bit/Field	R/W	Value	Description
7-0		R/W		Software writes a byte to this register, before setting the TE bit, allowing the CSI/O to send it. Software reads a received byte from this register, after the CSI/O sets the EF flag in response to software setting the RE bit.

INSTRUCTION SET

The Z80S188 includes the 8S180 processor, which is descended from the ZiLOG Z80. Its 8-bit data bus and 23-bit address space fit well into a wide variety of mid-range embedded processing applications, providing significantly more computing power than a microcontroller, at a fraction of the system cost of a larger microprocessor.

For details of these instructions see the Z80S188 User Manual, or the Z8S180 or Z80185 User Manuals until the Z80S188 UM is available.

CLASSES OF INSTRUCTIONS

TABLE 124. LOAD INSTRUCTIONS

Mnemonic	Operands	Instruction
LD	dst,src	Load
POP	dst	Pop
PUSH	src	Push

TABLE 125. ARITHMETIC INSTRUCTIONS

Mnemonic	Operands	Instruction
ADC	dst,src	Add with Carry
ADD	dst,src	Add
CP	A,src	Compare
CPD(R)		Block Scan, decrementing (and Repeat)
CPI(R)		Block Scan, incrementing (and Repeat)
DAA		Decimal Adjust Accumulator
DEC	dst	Decrement
INC	dst	Increment
MLT	rr	Multiply
NEG		Negate Accumulator
SBC	dst,src	Subtract with Carry
SUB	A,src	Subtract

TABLE 126. LOGICAL INSTRUCTIONS

Mnemonic	Operands	Instruction
AND	A,src	Logical AND
CPL		Complement accumulator
OR	A,src	Logical OR
TST	A,src	Test accumulator
XOR	A,src	Logical Exclusive OR

TABLE 127. EXCHANGE INSTRUCTIONS

Mnemonic	Operands	Instruction
EX	AF,AF'	Exchange Accumulator and Flags
EX	DE,HL	Exchange DE and HL
EX	(SP),rr	Exchange register and top of stack
EXX		Exchange register banks

**TABLE 128. PROGRAM CONTROL INSTRUCTIONS**

Mnemonic	Operands	Instruction
CALL	cc,dst	Conditional Call
CALL	dst	Call
DJNZ	dst	Decrement and Jump if Non-Zero
JP	cc,dst	Conditional Jump
JP	dst	Jump
JR	cc',dst	Conditional Jump Relative
JR	dst	Jump Relative
RET	cc	Conditional Return
RET		Return
RETI		Return from Interrupt
RETN		Return from Non-maskable interrupt
RST	dst	Restart

TABLE 129. BIT MANIPULATION INSTRUCTIONS

Mnemonic	Operands	Instruction
BIT	n,src	Bit test
RES	n,dst	Reset bit
SET	n,dst	Set bit

TABLE 130. BLOCK TRANSFER INSTRUCTIONS

Mnemonic	Operands	Instruction
LDD(R)		Block Move, decrementing (and Repeat)
LDI(R)		Block Move, incrementing (and Repeat)

TABLE 131. ROTATE AND SHIFT INSTRUCTIONS

Mnemonic	Operands	Instruction
RL	dst	Rotate Left
RLA		Rotate Left Accumulator
RLC	dst	Rotate Left Circular
RLCA		Rotate Left Circular Accumulator
RLD		Rotate Left Decimal
RR	dst	Rotate Right
RRA		Rotate Right Accumulator
RRC	dst	Rotate Right Circular
RRCA		Rotate Right Circular Accumulator

TABLE 131. ROTATE AND SHIFT INSTRUCTIONS (CONTINUED)

Mnemonic	Operands	Instruction
RRD		Rotate Right Decimal
SLA	dst	Shift Left
SRA	dst	Shift Right Arithmetic
SRL	dst	Shift Right Logical

TABLE 132. INPUT/OUTPUT INSTRUCTIONS

Mnemonic	Operands	Instruction
IN	A,(n)	Input to A from port n
IN	r,(C)	Input to register from port in BC
INO	r,(n)	Input to r from port n in page 0
IND(R)		Block Input, decrementing (and Repeat)
INI(R)		Block Input, incrementing (and Repeat)
OTDM(R)		Block Output, page 0, decrementing (and Repeat)
OTIM(R)		Block Output, page 0, incrementing (and Repeat)
OUT	(n),A	Output from A to port n
OUT	(C),r	Output from register to port in BC
OUTO	(n),r	Output from register to port n in page 0
OUTD (OTDR)		Block Output, decrementing (and Repeat)
OUTI (OTIR)		Block Output, incrementing (and Repeat)
TSTIO	n	Test port (0,C) under mask

TABLE 133. PROCESSOR CONTROL INSTRUCTIONS

Mnemonic	Operands	Instruction
CCF		Complement Carry Flag
DI		Disable Interrupts
EI		Enable Interrupts
HALT		Halt
IM	0/1/2	Interrupt Mode
NOP		No Operation
SCF		Set Carry Flag
SLP		Sleep

PROCESSOR FLAGS

The following Figure shows the Flags register. Bits in this register are set and cleared by certain instructions as described in the Z80S188 User Manual. Some of the Flags can be tested by conditional JR, JP, CALL, and RET instructions, and

some are used by subsequent instructions such as ADC, SBC, and DAA. The Flags can also have PUSH and POP instructions applied to them with the accumulator A.

TABLE 134. FLAG REGISTER

Bit	7	6	5	4	3	2	1	0
Name	S	Z	x	HC	x	P/V	N	CF
Reset	0	0	x	0	x	0	0	0

NOTE: X = Indeterminate

Bit/ Field	Bit Position	Description
S	7	Sign Flag
Z	6	Zero Flag
	5	Reserved
HC	4	Half-carry Flag
	3	Reserved
P/V	2	Parity or Overflow Flag
N	1	Add/Subtract Flag
CF	0	Carry Flag

CONDITION CODES

Table 135 shows the codes used in the “Flags Affected” columns of the later Instruction Summary Table, to indicate how each Flag is affected by each type of instruction.

TABLE 135. FLAG SETTINGS DEFINITIONS

Symbol	Definition
0	Cleared to 1
1	Set to 1
*	Set or cleared according to the result of the operation
–	Unaffected
X	Undefined
V	Set if Overflow or Underflow
P	Set if Parity or result is Even
NZ	Set if the count in B or BC is Non-Zero

The following table shows the condition codes that can be used in conditional JP, CALL, and RET instructions in assembly language. A subset of these codes can also be used in JR instructions, which are shorter and faster than JP's.

TABLE 136. CONDITION CODES

Mnemonic	Definition	Flag Settings	Valid in JR?
C	Carry	CF = 1	Y
NC	No Carry	CF = 0	Y
Z	Zero	Z = 1	Y
NZ	Non-Zero	Z = 0	Y
M	Minus	S = 1	N
P	Positive or 0	S = 0	N
PE	Parity Even	P/V = 1	N
PO	Parity Odd	P/V = 0	N
V	Overflow	P/V = 1	N
NV	No Overflow	P/V = 0	N



NOTATION

The following table describes other notation used in the subsequent Instruction Summary table.

TABLE 137. SYMBOLS

Symbol	Definition
(aa)	(mn), (IX \pm d), (IY \pm d), (BC), (DE), or (HL).
(BC), (DE), (HL)	The 8-bit contents of memory, at the address pointed to by a register pair
(IX \pm d), (IY \pm d)	The 8-bit content of memory at the address formed by adding the contents of the index register and the signed displacement d in the instruction.
(mn)	The 8-bit content of memory at the direct address mn
(SP)	The 16-bit contents of memory at the address pointed to by SP, and the next higher address.
\pm d	Because d is signed, it would be more correct to just write + instead. But we write \pm to emphasize that d is signed.
AF	A concatenated with F, with A as the MS byte
b	A bit number 0-7
cc	A condition code C, NC, Z, NZ, S, M, PE, PV, V, or NV
cc'	A condition code C, NC, Z, or NZ
d	An 8-bit signed displacement -128-127
ee	A 16-bit register BC, DE, HL, SP, IX, or IY
IEF1,2	The processor's two Interrupt Enable Flags. See the "Interrupts" section for more detail.
mn	A 16-bit immediate data value or direct address
n	A 8-bit immediate value or port number, 0-255H or 0-FFH
op1-op2	A range of Op Code values, that includes some of the values between the low and high values. See the Note.
PC	Program Counter
pp	A 16-bit register BC, DE, HL, SP, IX, IY, or AF
r, r'	An 8-bit register A, B, C, D, E, H, or L.
rr	A 16-bit register HL, IX, or IY.
s	an 8-bit register or memory location
SP	Stack Pointer
ss	A 16-bit register BC, DE, HL, or SP.
ss _H , ss _L	The more- and less-significant 8 bits of a register pair
tt	A 16-bit register like ss, except that the value that designates HL in the ss encoding, here means "same as the destination register HL, IX, or IY".

NOTE: The – between Op Codes (op1-op2), in the Op Codes column of the following Instruction Summary table, indicates all the binary values between the lower and upper limits inclusive, that can be formed by incrementing the set of bits that differ between the lower and upper value.

EXAMPLE: 00–C0 denotes 00, 40, 80, and C0, while 40-BF includes all the values in that range.

ASSEMBLY LANGUAGE SYNTAX

For two-operand instructions, Z80 assembly language syntax puts the *destination* operand before the *source* operand.

EXAMPLE: LD A, (1234) is a Load instruction, while LD (1234), A is a Store instruction.

Past Z80 assemblers allowed the destination operand to be omitted (implicit) if the Op Code mnemonic only allowed one destination operand, for example, AND L instead of AND A, L. Use of these short forms is discouraged because it is a cause of possible error (the programmer mistakes the implicit destination). But for legacy code, all known Z80 assemblers still accept the short form.

NOTE: The assembly language uses C ambiguously, to designate one of the 8-bit registers as well as a condition code to test the Carry flag. This processor description uses CF to designate the Carry flag, and HC to designate the Half-Carry flag (as opposed to the 8-bit register H).

INSTRUCTION SUMMARY

The following table describes each type or class of instruction, using the notation described in the preceding sections. The table is sorted by the assembly language mnemonics.

TABLE 138. INSTRUCTION SUMMARY

	Address Mode		OpCode(s) (Hex)	Flags Affected					
Instruction and Operation	dst	src		S	Z	HC	P/V	N	CF
ADC A,s A ← A + s + CF		r	88–8F	*	*	*	V	0	*
		n	CE						
		(HL)	8E						
		(IX/Y ± d)	DD/FD 8E						
ADC HL,ss HL ← HL + ss + CF			ED 4A–7A	*	*	*	V	0	*
ADD A,s A ← A + s		r	80–87	*	*	*	V	0	*
		n	C6						
		(HL)	86						
		(IX/Y ± d)	DD/FD 86						
ADD rr,tt rr ← rr + tt	HL		09–39	–	–	*	–	0	*
	IX/Y		DD/FD 09–39						
AND A,s A ← A and s		r	A0–A7	*	*	1	P	0	0
		n	E6						
		(HL)	A6						
		(IX/Y ± d)	DD/FD A6						



TABLE 138. INSTRUCTION SUMMARY (CONTINUED)

Instruction and Operation	Address Mode		OpCode(s) (Hex)	Flags Affected					
	dst	src		S	Z	HC	P/V	N	CF
BIT b,m $Z \leftarrow \text{not (bit b of m)}$		r (HL) (IX/Y \pm d)	CB 40–7F CB 46–7E DD/FD CB d 46–7E	X	*	1	X	0	–
CALL cc,mn IF cc {SP \leftarrow SP – 2 (SP) \leftarrow PC PC \leftarrow mn}			C4–FC	–	–	–	–	–	–
CALL mn SP \leftarrow SP – 2 (SP) \leftarrow PC PC \leftarrow mn			CD	–	–	–	–	–	–
CCF CF \leftarrow not CF			3F	–	–	*	–	0	*
CP A,s A – s		r n (HL) (IX/Y \pm d)	B8–BF FE BE DD/FD BE	*	*	*	V	1	*
CPD A – (HL) HL \leftarrow HL – 1 BC \leftarrow BC – 1			ED A9	*	*	*	NZ	1	–
CPDR repeat {A – (HL) HL \leftarrow HL – 1 BC \leftarrow BC – 1 } while (not Z and BC != 0)			ED B9	*	*	*	NZ	1	–
CPI A – (HL) HL \leftarrow HL + 1 BC \leftarrow BC – 1			ED A1	*	*	*	NZ	1	–
CPIR repeat {A – (HL) HL \leftarrow HL + 1 BC \leftarrow BC – 1 } while (not Z and BC != 0)			ED B1	*	*	*	NZ	1	–
CPL A \leftarrow not A			2F	–	–	1	–	1	–
DAA A \leftarrow decimal adjust (A,F)			27	*	*	*	P	–	*

TABLE 138. INSTRUCTION SUMMARY (CONTINUED)

Instruction and Operation	Address Mode		OpCode(s) (Hex)	Flags Affected					
	dst	src		S	Z	HC	P/V	N	CF
DEC ee $ee \leftarrow ee - 1$	ss		0B–3B	–	–	–	–	–	–
	IX/Y		DD/FD 2B						
DEC m $m \leftarrow m - 1$	r		05–3D	*	*	*	V	1	–
	(HL)		35						
	(IX/Y \pm d)		DD/FD 35						
DI $IEF1,2 \leftarrow 0$			F3	–	–	–	–	–	–
DJNZ d $B \leftarrow B - 1$ if $B \neq 0 \{PC \leftarrow PC \pm d\}$			10	–	–	–	–	–	–
EI $IEF1,2 \leftarrow 1$			FB	–	–	–	–	–	–
EX AF,AF' $AF \leftrightarrow AF'$			08	*	*	*	*	*	*
EX (SP),rr $(SP) \leftrightarrow rr$	HL		E3	–	–	–	–	–	–
	IX/Y		DD/FD E3						
EXX $BC \leftrightarrow BC'$ $DE \leftrightarrow DE'$ $HL \leftrightarrow HL'$			D9	–	–	–	–	–	–
HALT			76	–	–	–	–	–	–
IM n			ED 40–58	–	–	–	–	–	–
IN A,(n) $A \leftarrow (n)$			DB	–	–	–	–	–	–
IN r,(C) $r \leftarrow (BC)$			ED 40–78	*	*	0	P	0	–
IN0 r,(n) $r \leftarrow (0,n)$			ED 00–38	*	*	0	P	0	–
INC ee $ee \leftarrow ee + 1$	ss		03–33	–	–	–	–	–	–
	IX/Y		DD/FD 23						
INC m $m \leftarrow m + 1$	r		04–3C	*	*	*	V	0	–
	(HL)		34						
	IX/Y		DD/FD 34						
IND $(HL) \leftarrow (C)$ $B \leftarrow B - 1$ $HL \leftarrow HL - 1$			ED AA	X	*	X	X	1	–
INDR do $\{(HL) \leftarrow (C)$ $B \leftarrow B - 1$ $HL \leftarrow HL - 1$ $\}$ while $B \neq 0$			ED BA	X	1	X	X	1	–



TABLE 138. INSTRUCTION SUMMARY (CONTINUED)

Instruction and Operation	Address Mode		OpCode(s) (Hex)	Flags Affected					
	dst	src		S	Z	HC	P/V	N	CF
INI (HL) \leftarrow (C) B \leftarrow B - 1 HL \leftarrow HL + 1			ED A2	X	*	X	X	1	-
INIR do {(HL) \leftarrow (C) B \leftarrow B - 1 HL \leftarrow HL + 1 } while B != 0			ED B2	X	1	X	X	1	-
JP (rr) PC \leftarrow rr	(HL)		E9	-	-	-	-	-	-
	(IX/Y)		DD/FD E9						
JP cc,mn if cc {PC \leftarrow mn}			C2-FA	-	-	-	-	-	-
JP mn PC \leftarrow mn			C3	-	-	-	-	-	-
JR cc',d if cc' {PC \leftarrow PC \pm d}			10-38	-	-	-	-	-	-
JR d PC \leftarrow PC \pm d			18	-	-	-	-	-	-
LD (aa),A (aa) \leftarrow A	(BC)		02	-	-	-	-	-	-
	(DE)		12						
	(HL)		77						
	(mn)		32						
	(IX/Y \pm d)		DD/FD 77						
LD (mn),ee (mn) \leftarrow ee	HL		22	-	-	-	-	-	-
	ss		ED 43-73						
	IX/Y		DD/FD 22						
LD A,(aa) A \leftarrow (aa)	(BC)		0A	-	-	-	-	-	-
	(DE)		1A						
	(HL)		7E						
	(mn)		3A						
	(IX/Y \pm d)		DD/FD 7E						
LD A,I A \leftarrow I			ED 57	*	*	0	IEF2	0	-
LD A,R A \leftarrow R			ED 5F	*	*	0	IEF2	0	-
LD ee,mn ee \leftarrow mn	ss		01-31	-	-	-	-	-	-
	IX/Y		DD/FD 21						
LD ee,(mn) ee \leftarrow (mn)	HL		2A	-	-	-	-	-	-
	ss		ED 4B-7B						
	IX/Y		DD/FD 2A						

TABLE 138. INSTRUCTION SUMMARY (CONTINUED)

Instruction and Operation	Address Mode		OpCode(s) (Hex)	Flags Affected					
	dst	src		S	Z	HC	P/V	N	CF
LD I,A $I \leftarrow A$			ED 47	–	–	–	–	–	–
LD m,n $m \leftarrow n$	r		06–3E	–	–	–	–	–	–
	(HL)		36						
	(IX/Y \pm d)		DD/FD 36						
LD m,r $m \leftarrow r$	r'		40–7F	–	–	–	–	–	–
	(HL)		70–77						
	(IX/Y \pm d)		DD/FD 70–77						
LD R,A $R \leftarrow A$			ED 4F	–	–	–	–	–	–
LD r,s $r \leftarrow s$		r'	40–7F	–	–	–	–	–	–
		n	06–3E						
		(HL)	46–7E						
		(IX/Y \pm d)	DD/FD 46–7E						
LD SP,rr $SP \leftarrow rr$		HL	F9	–	–	–	–	–	–
		IX/Y	DD/FD F9						
LDD (DE) \leftarrow (HL) DE \leftarrow DE – 1 HL \leftarrow HL – 1 BC \leftarrow BC – 1			ED A8	–	–	0	NZ	0	–
LDDR do {(DE) \leftarrow (HL)} DE \leftarrow DE – 1 HL \leftarrow HL – 1 BC \leftarrow BC – 1 } while BC != 0			ED B8	–	–	0	0	0	–
LDI (DE) \leftarrow (HL) DE \leftarrow DE + 1 HL \leftarrow HL + 1 BC \leftarrow BC – 1			ED A0	–	–	0	NZ	0	–
LDIR do {(DE) \leftarrow (HL)} DE \leftarrow DE + 1 HL \leftarrow HL + 1 BC \leftarrow BC – 1 } while BC != 0			ED B0	–	–	0	0	0	–
MLT ss $ss \leftarrow ss_L * ss_H$			ED 4C–7C	–	–	–	–	–	–
NEG $A \leftarrow 0 - A$			ED 44	*	*	*	V	1	*



TABLE 138. INSTRUCTION SUMMARY (CONTINUED)

Instruction and Operation	Address Mode		OpCode(s) (Hex)	Flags Affected					
	dst	src		S	Z	HC	P/V	N	CF
NOP			00	–	–	–	–	–	–
OR A,s $A \leftarrow A \text{ OR } s$		r	B0–B7	*	*	0	P	0	0
		n	F6						
		(HL)	B6						
		(IX/Y \pm d)	DD/FD B6						
OTDM (O,C) \leftarrow (HL) $B \leftarrow B - 1$ $C \leftarrow C - 1$ $HL \leftarrow HL - 1$			ED 8B	*	*	*	P	*	*
OTDMR do {(O,C) \leftarrow (HL)} $B \leftarrow B - 1$ $C \leftarrow C - 1$ $HL \leftarrow HL - 1$ } while B != 0			ED 8B	0	1	0	1	*	0
OTDR do {(C) \leftarrow (HL)} $B \leftarrow B - 1$ $HL \leftarrow HL - 1$ } while B != 0			ED BB	X	1	X	X	1	–
OTIM (O,C) \leftarrow (HL) $B \leftarrow B - 1$ $C \leftarrow C + 1$ $HL \leftarrow HL + 1$			ED 83	*	*	*	P	*	*
OTIMR do {(O,C) \leftarrow (HL)} $B \leftarrow B - 1$ $C \leftarrow C + 1$ $HL \leftarrow HL + 1$ } while B != 0			ED 93	0	1	0	1	*	0
OTIR do {(C) \leftarrow (HL)} $B \leftarrow B - 1$ $HL \leftarrow HL + 1$ } while B != 0			ED B3	X	1	X	X	1	–
OUT (C),r (BC) \leftarrow r			ED 41–79	–	–	–	–	–	–
OUT (n),A (n) \leftarrow A			D3	–	–	–	–	–	–
OUT0 (n),r (O,n) \leftarrow r			ED 01–79	–	–	–	–	–	–

TABLE 138. INSTRUCTION SUMMARY (CONTINUED)

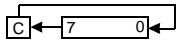
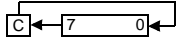
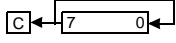
Instruction and Operation	Address Mode		OpCode(s) (Hex)	Flags Affected					
	dst	src		S	Z	HC	P/V	N	CF
OUTD (C) \leftarrow (HL) B \leftarrow B - 1 HL \leftarrow HL - 1			ED AB	X	*	X	X	1	-
OUTI (C) \leftarrow (HL) B \leftarrow B - 1 HL \leftarrow HL + 1			ED AB	X	*	X	X	1	-
POP pp pp \leftarrow (SP) SP \leftarrow SP + 2	qq IX/Y		C1-F1 DD/FD E1	(no change unless operand is AF)					
PUSH pp SP \leftarrow SP-2 (SP) \leftarrow pp		qq IX/Y	C5-F5 DD/FD E5	-	-	-	-	-	-
RES b,m m \leftarrow m and not (2 ^b)	r (HL) (IX/Y \pm d)		CB 80-BF CB 86-BE DD/FD CB d 86-BE	-	-	-	-	-	-
RET PC \leftarrow (SP) SP \leftarrow SP + 2			C9	-	-	-	-	-	-
RET cc if cc {PC \leftarrow (SP) SP \leftarrow SP + 2}			C0-F8	-	-	-	-	-	-
RETI PC \leftarrow (SP) SP \leftarrow SP + 2 + recognition by Z80 peripherals			ED 4D	-	-	-	-	-	-
RETN PC \leftarrow (SP) SP \leftarrow SP + 2 IEF1 \leftarrow IEF2			ED 45	-	-	-	-	-	-
RL m  (CF,m) \leftarrow rotL(CF,m)	r (HL) (IX/Y \pm d)		CB 10-17 CB 16 DD/FD CB d 16	*	*	0	P	0	*
RLA  (CF,A) \leftarrow rotL(CF,A)			17	-	-	0	-	0	*
RLC m  (CF,m) \leftarrow rotL(m)	r (HL) (IX/Y \pm d)		CB 00-07 CB 06 DD/FD CB d 06	*	*	0	P	0	*



TABLE 138. INSTRUCTION SUMMARY (CONTINUED)

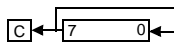
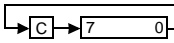
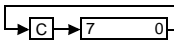
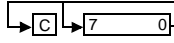
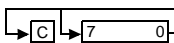
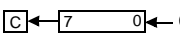
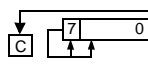
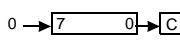
	Address Mode		OpCode(s) (Hex)	Flags Affected						
Instruction and Operation	dst	src		S	Z	HC	P/V	N	CF	
RLCA  (CF,A) ← rotL(A)			07	–	–	0	–	0	*	
RLD tmp ← A[3:0] A[3:0] ← (HL)[7:4] (HL)[7:4] ← (HL)[3:0] (HL)[3:0] ← tmp			ED 6F	*	*	0	P	0	–	
RR m  (CF,m) ← rotR(CF,m)	r		CB 18–1F	*	*	0	P	0	*	
	(HL)		CB 1E							
	(IX/Y ± d)		DD/FD CB d 1E							
RRA  (CF,A) ← rotR(CF,A)	r		1F	–	–	0	–	0	*	
RRC m  (CF,m) ← rotR(m)	r		CB 08–0F	*	*	0	P	0	*	
	(HL)		CB 0E							
	(IX/Y ± d)		DD/FD CB d 0E							
RRCA  (CF,A) ← rotR(A)			0F	–*	–	0	–	0	*	
RRD tmp ← (HL)[3:0] (HL)[3:0] ← (HL)[7:4] (HL)[7:4] ← A[3:0] A[3:0] ← tmp			ED 67	*	*	0	P	0	–	
RST p SP ← SP – 2 (SP) ← PC PC ← 0,p note: p = 0,8,10,18,...,38 ₁₆			C7–FF	*	*	0	P	0	–	
SBC A,s A ← A – s – CF	r		98–9F	*	*	*	V	1	*	
	n		DE							
	(HL)		9E							
	(IX/Y ± d)		DD/FD 9E							
SBC HL,ss HL ← HL – ss – CF	r		ED 42–72	*	*	*	V	1	*	
SCF CF ← 1			37	–	–	0	–	0	1	

TABLE 138. INSTRUCTION SUMMARY (CONTINUED)

	Address Mode		OpCode(s) (Hex)	Flags Affected						
Instruction and Operation	dst	src		S	Z	HC	P/V	N	CF	
SET b,m $m \leftarrow m \text{ or } (2^b)$	r		CB C0–FF	–	–	–	–	–	–	
	(HL)		CB C6–FE							
	(IX/Y ± d)		DD/FD CB d C6–FE							
SLA m  $(CF,m) \leftarrow m + m$	r		CB 20–27	*	*	0	P	0	*	
	(HL)		CB 26							
	(IX/Y ± d)		DD/FD CB d 26							
SLP			ED 76	–	–	–	–	–	–	
SRA m  $(m,CF) \leftarrow \text{arith_shR}(m)$	r		CB 28–2F	*	*	0	P	0	*	
	(HL)		CB 2E							
	(IX/Y ± d)		DD/FD CB d 2E							
SRL m  $(m,CF) \leftarrow \text{logic_shR}(m)$	r		CB 38–3F	0	*	0	P	0	*	
	(HL)		CB 3E							
	(IX/Y ± d)		DD/FD CB d 3E							
SUB A,s $A \leftarrow A - s$	r		90–97	*	*	*	V	1	*	
	n		D6							
	(HL)		96							
	(IX/Y ± d)		DD/FD 96							
TST A,s $A \text{ AND } s$	r		ED 04–3C	*	*	1	P	0	0	
	n		ED 64							
	(HL)		ED 34							
TSTIO n $(O,C) \text{ AND } n$			ED 34	*	*	1	P	0	0	
XOR A,s $A \leftarrow A \text{ XOR } s$	r		A8–AF	*	*	0	P	0	0	
	n		EE							
	(HL)		AE							
	(IX/Y ± d)		DD/FD AE							

OP CODE MAP

TABLE 139. OP CODE MAP (1ST OP CODE)

		LOWER NIBBLE (HEX)															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
UPPER NIBBLE (HEX)	0	NOP	LD BC,nn	LD (BC),A	INC BC	INC B	DEC B	LD B,n	RLCA	EX AF,AF'	ADD HL,BC	LD A,(BC)	DEC BC	INC C	DEC C	LD C,n	RRCA
	1	DJNZ d	LD DE,nn	LD (DE),A	INC DE	INC D	DEC D	LD D,n	RLA	JR d	ADD HL,DE	LD A,(DE)	DEC DE	INC E	DEC E	LD E,n	RRA
	2	JR NZ,d	LD HL,nn	LD (nn),HL	INC HL	INC H	DEC H	LD H,n	DAA	JR Z,d	ADD HL,HL	LD (HL),nn	DEC HL	INC L	DEC L	LD L,n	CPL
	3	JR NC,d	LD SP,nn	LD (nn),A	INC SP	INC (HL)	DEC (HL)	LD (HL),n	SCF	JR C,d	ADD HL,SP	LD A,(nn)	DEC SP	INC A	DEC A	LD A,n	CCF
	4	LD B,B	LD B,C	LD B,D	LD B,E	LD B,H	LD B,L	LD B,(HL)	LD C,A	LD C,B	LD C,C	LD C,D	LD C,E	LD C,H	LD C,L	LD C,(HL)	LD C,A
	5	LD D,B	LD D,C	LD D,D	LD D,E	LD D,H	LD D,L	LD D,(HL)	LD E,A	LD E,B	LD E,C	LD E,D	LD E,E	LD E,H	LD E,L	LD E,(HL)	LD E,A
	6	LD H,B	LD H,C	LD H,D	LD H,E	LD H,H	LD H,L	LD H,(HL)	LD L,A	LD L,B	LD L,C	LD L,D	LD L,E	LD L,H	LD L,L	LD L,(HL)	LD L,A
	7	LD (HL),B	LD (HL),C	LD (HL),D	LD (HL),E	LD (HL),H	LD (HL),L	HALT	LD (HL),A	LD A,B	LD A,C	LD A,D	LD A,E	LD A,H	LD A,L	LD A,(HL)	LD A,A
	8	ADD A,B	ADD A,C	ADD A,D	ADD A,E	ADD A,H	ADD A,L	ADD A,(HL)	ADC A,A	ADC A,B	ADC A,C	ADC A,D	ADC A,E	ADC A,H	ADC A,L	ADC A,(HL)	ADC A,A
	9	SUB A,B	SUB A,C	SUB A,D	SUB A,E	SUB A,H	SUB A,L	SUB A,(HL)	SBC A,A	SBC A,B	SBC A,C	SBC A,D	SBC A,E	SBC A,H	SBC A,L	SBC A,(HL)	SBC A,A
	A	AND A,B	AND A,C	AND A,D	AND A,E	AND A,H	AND A,L	AND A,(HL)	AND A,A	XOR A,B	XOR A,C	XOR A,D	XOR A,E	XOR A,H	XOR A,L	XOR A,(HL)	XOR A,A
	B	OR A,B	OR A,C	OR A,D	OR A,E	OR A,H	OR A,L	OR A,(HL)	OR A,A	CP A,B	CP A,C	CP A,D	CP A,E	CP A,H	CP A,L	CP A,(HL)	CP A,A
	C	RET NZ	POP BC	JP NZ,nn	JP nn	CALL NZ,nn	PUSH BC	ADD A,n	RST 0	RET Z	RET	JP Z,nn	(Table 140)	CALL Z,nn	CALL nn	ADC A,n	RST 8
	D	RET NZ	POP DE	JP NC,nn	OUT (n),A	CALL NC,nn	PUSH DE	SUB A,n	RST 10H	RET C	EXX	JP C,nn	IN A,(n)	CALL C,nn	(Table 141)	SBC A,n	RST 18H
	E	RET PO	POP HL	JP PO,nn	EX (SP),HL	CALL PO,nn	PUSH HL	AND A,n	RST 20	RET PE	JP (HL)	JP PE,nn	EX DE,HL	CALL PE,nn	(Table 142)	XOR A,n	RST 28H
	F	RET P	POP AF	JP P,nn	DI	CALL P,nn	PUSH AF	OR A,n	RST 30H	RET M	LD SP,HL	JP M,nn	EI	CALL M,nn	(Table 143)	CP A,n	RST 38H
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Notes:

n = 8-bit data

nn = 16-bit addr or data

d = signed 8-bit displacement

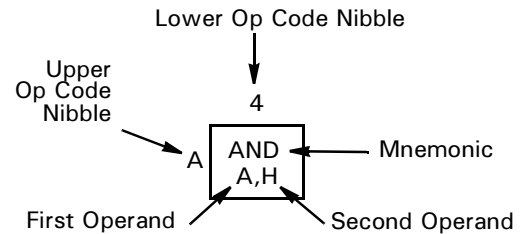


TABLE 140. OP CODE MAP (2ND OP CODE AFTER 0CBH)

		LOWER NIBBLE (HEX)															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
UPPER NIBBLE (HEX)	0	RLC B	RLC C	RLC D	RLC E	RLC H	RLC L	RLC (HL)	RLC RRCA	RRC B	RRC C	RRC D	RRC E	RRC H	RRC L	RRC (HL)	RRC A
	1	RL B	RL C	RL D	RL E	RL H	RL L	RL (HL)	RL A	RR B	RR C	RR D	RR E	RR H	RR L	RR (HL)	RR A
	2	SLA B	SLA C	SLA D	SLA E	SLA H	SLA L	SLA (HL)	SLA A	SRA B	SRA C	SRA D	SRA E	SRA H	SRA L	SRA (HL)	SRA A
	3									SRL B	SRL C	SRL D	SRL E	SRL H	SRL L	SRL (HL)	SRL A
	4	BIT 0,B	BIT 0,C	BIT 0,D	BIT 0,E	BIT 0,H	BIT 0,L	BIT 0,(HL)	BIT 0,A	BIT 1,B	BIT 1,C	BIT 1,D	BIT 1,E	BIT 1,H	BIT 1,L	BIT 1,(HL)	BIT 1,A
	5	BIT 2,B	BIT 2,C	BIT 2,D	BIT 2,E	BIT 2,H	BIT 2,L	BIT 2,(HL)	BIT 2,A	BIT 3,B	BIT 3,C	BIT 3,D	BIT 3,E	BIT 3,H	BIT 3,L	BIT 3,(HL)	BIT 3,A
	6	BIT 4,B	BIT 4,C	BIT 4,D	BIT 4,E	BIT 4,H	BIT 4,L	BIT 4,(HL)	BIT 4,A	BIT 5,B	BIT 5,C	BIT 5,D	BIT 5,E	BIT 5,H	BIT 5,L	BIT 5,(HL)	BIT 5,A
	7	BIT 6,B	BIT 6,C	BIT 6,D	BIT 6,E	BIT 6,H	BIT 6,L	BIT 6,(HL)	BIT 6,A	BIT 7,B	BIT 7,C	BIT 7,D	BIT 7,E	BIT 7,H	BIT 7,L	BIT 7,(HL)	BIT 7,A
	8	RES 0,B	RES 0,C	RES 0,D	RES 0,E	RES 0,H	RES 0,L	RES 0,(HL)	RES 0,A	RES 1,B	RES 1,C	RES 1,D	RES 1,E	RES 1,H	RES 1,L	RES 1,(HL)	RES 1,A
	9	RES 2,B	RES 2,C	RES 2,D	RES 2,E	RES 2,H	RES 2,L	RES 2,(HL)	RES 2,A	RES 3,B	RES 3,C	RES 3,D	RES 3,E	RES 3,H	RES 3,L	RES 3,(HL)	RES 3,A
	A	RES 4,B	RES 4,C	RES 4,D	RES 4,E	RES 4,H	RES 4,L	RES 4,(HL)	RES 4,A	RES 5,B	RES 5,C	RES 5,D	RES 5,E	RES 5,H	RES 5,L	RES 5,(HL)	RES 5,A
	B	RES 6,B	RES 6,C	RES 6,D	RES 6,E	RES 6,H	RES 6,L	RES 6,(HL)	RES 6,A	RES 7,B	RES 7,C	RES 7,D	RES 7,E	RES 7,H	RES 7,L	RES 7,(HL)	RES 7,A
	C	SET 0,B	SET 0,C	SET 0,D	SET 0,E	SET 0,H	SET 0,L	SET 0,(HL)	SET 0,A	SET 1,B	SET 1,C	SET 1,D	SET 1,E	SET 1,H	SET 1,L	SET 1,(HL)	SET 1,A
	D	SET 2,B	SET 2,C	SET 2,D	SET 2,E	SET 2,H	SET 2,L	SET 2,(HL)	SET 2,A	SET 3,B	SET 3,C	SET 3,D	SET 3,E	SET 3,H	SET 3,L	SET 3,(HL)	SET 3,A
	E	SET 4,B	SET 4,C	SET 4,D	SET 4,E	SET 4,H	SET 4,L	SET 4,(HL)	SET 4,A	SET 5,B	SET 5,C	SET 5,D	SET 5,E	SET 5,H	SET 5,L	SET 5,(HL)	SET 5,A
	F	SET 6,B	SET 6,C	SET 6,D	SET 6,E	SET 6,H	SET 6,L	SET 6,(HL)	SET 6,A	SET 7,B	SET 7,C	SET 7,D	SET 7,E	SET 7,H	SET 7,L	SET 7,(HL)	SET 7,A
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

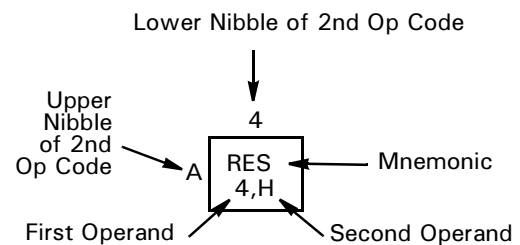


TABLE 141. OP CODE MAP (2ND OP CODE AFTER 0DDH)

		LOWER NIBBLE (HEX)															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
UPPER NIBBLE (HEX)	0										ADD IX,BC						
	1										ADD IX,DE						
	2		LD IX,nn	LD (nn),IX	INC IX						ADD IX,IX	LD IX,(nn)	DEC IX				
	3					INC (IX ± d)	DEC (IX ± d)	LD (IX ± d),n			ADD IX,SP						
	4							LD B, (IX ± d)								LD C, (IX ± d)	
	5							LD D, (IX ± d)								LD E, (IX ± d)	
	6							LD H, (IX ± d)								LD L, (IX ± d)	
	7	LD (IX ± d),B	LD (IX ± d),C	LD (IX ± d),D	LD (IX ± d),E	LD (IX ± d),H	LD (IX ± d),L		LD (IX ± d),A							LD A, (IX ± d)	
	8							ADD A, (IX ± d)								ADC A, (IX ± d)	
	9							SUB A, (IX ± d)								SBC A, (IX ± d)	
	A							AND A, (IX ± d)								XOR A, (IX ± d)	
	B							OR A, (IX ± d)								CP A, (IX ± d)	
	C												(Table 144)				
	D																
	E		POP IX		EX (SP),IX		PUSH IX				JP (IX)						
	F										LD SP,IX						
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Notes:

n = 8-bit data
 nn = 16-bit addr or data
 d = signed 8-bit displacement

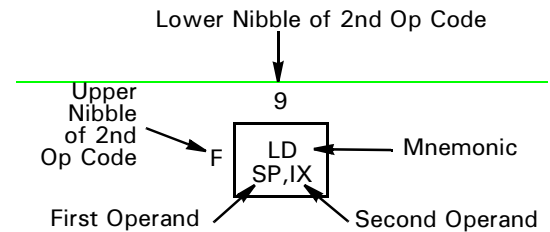


TABLE 142. OP CODE MAP (2ND OP CODE AFTER 0EDH)

		LOWER NIBBLE (HEX)															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
UPPER NIBBLE (HEX)	0	IN0 B,(n)	OUT0 (n),B			TST A,B				IN0 C,(n)	OUT0 (n),C			TST A,C			
	1	IN0 D,(n)	OUT0 (n),D			TST A,D				IN0 E,(n)	OUT0 (n),E			TST A,E			
	2	IN0 H,(n)	OUT0 (n),H			TST A,H				IN0 L,(n)	OUT0 (n),L			TST A,L			
	3	IN0 F,(n)				TST A,(HL)				IN0 A,(n)	OUT0 (n),A			TST A,A			
	4	IN B,(C)	OUT (C),B	SBC HL,BC	LD (nn),BC	NEG	RETN	IM 0	LD I,A	IN C,(C)	OUT (C),C	ADC HL,BC	LD BC,(nn)	MLT BC	RETI		LD R,A
	5	IN D,(C)	OUT (C),D	SBC HL,DE	LD (nn),DE			IM 1	LD A,I	IN E,(C)	OUT (C),E	ADC HL,DE	LD DE,(nn)	MLT DE		IM 2	LD A,R
	6	IN H,(C)	OUT (C),H	SBC HL,HL	LD (nn),HL	TST A,n			RRD	IN L,(C)	OUT (C),L	ADC HL,HL	LD HL,(nn)	MLT HL			RLD
	7	IN F,(C)		SBC HL,SP	LD (nn),SP	TST IOn		SLP		IN A,(C)	OUT (C),A	ADC HL,SP	LD SP,(nn)	MLT SP			
	8				OTIM								OTDM				
	9				OTIMR								OTDMR				
	A	LDI	CPI	INI	OUTI					LDD	CPD	IND	OUTD				
	B	LDIR	CPIR	INIR	OTIR					LDDR	CPDR	INDR	OTDR				
	C																
	D																
	E																
	F																
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

n = 8-bit data
nn = 16-bit addr or data
d = signed 8-bit displacement

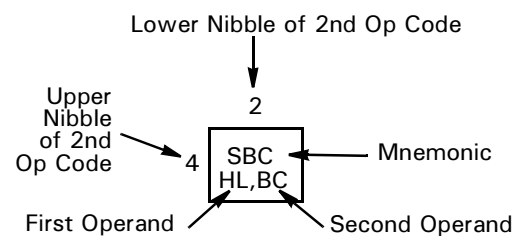


TABLE 143. OP CODE MAP (2ND OP CODE AFTER 0FDH)

		LOWER NIBBLE (HEX)															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
UPPER NIBBLE (HEX)	0										ADD IY,BC						
	1										ADD IY,DE						
	2		LD IY,nn	LD (nn),IY	INC IY						ADD IY,IY	LD IY,(nn)	DEC IY				
	3					INC (IY ± d)	DEC (IY ± d)	LD (IY ± d),n			ADD IY,SP						
	4							LD B, (IY ± d)								LD C, (IY ± d)	
	5							LD D, (IY ± d)								LD E, (IY ± d)	
	6							LD H, (IY ± d)								LD L, (IY ± d)	
	7	LD (IY ± d),B	LD (IY ± d),C	LD (IY ± d),D	LD (IY ± d),E	LD (IY ± d),H	LD (IY ± d),L		LD (IY ± d),A							LD A, (IY ± d)	
	8							ADD A, (IY ± d)								ADC A, (IY ± d)	
	9							SUB A, (IY ± d)								SBC A, (IY ± d)	
	A							AND A, (IY ± d)								XOR A, (IY ± d)	
	B							OR A, (IY ± d)								CP A, (IY ± d)	
	C												(Table 145)				
	D																
	E		POP IY		EX (SP),IY		PUSH IY				JP (IY)						
	F										LD SP,IY						
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Notes:

n = 8-bit data

nn = 16-bit addr or data

d = signed 8-bit displacement

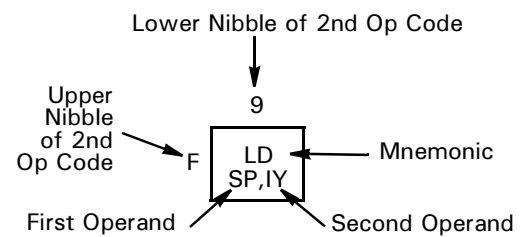


TABLE 144. OP CODE MAP (4TH BYTE, AFTER 0DDH, 0CBH, AND d)

		LOWER NIBBLE (HEX)															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
UPPER NIBBLE (HEX)	0							RLC (IX ± d)								RRC (IX ± d)	
	1							RL (IX ± d)								RR (IX ± d)	
	2							SLA (IX ± d)								SRA (IX ± d)	
	3															SRL (IX ± d)	
	4							BIT 0, (IX ± d)								BIT 1, (IX ± d)	
	5							BIT 2, (IX ± d)								BIT 3, (IX ± d)	
	6							BIT 4, (IX ± d)								BIT 5, (IX ± d)	
	7							BIT 6, (IX ± d)								BIT 7, (IX ± d)	
	8							RES 0, (IX ± d)								RES 1, (IX ± d)	
	9							RES 2, (IX ± d)								RES 3, (IX ± d)	
	A							RES 4, (IX ± d)								RES 5, (IX ± d)	
	B							RES 6, (IX ± d)								RES 7, (IX ± d)	
	C							SET 0, (IX ± d)								SET 1, (IX ± d)	
	D							SET 2, (IX ± d)								SET 3, (IX ± d)	
	E							SET 4, (IX ± d)								SET 5, (IX ± d)	
	F							SET 6, (IX ± d)								SET 7, (IX ± d)	
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Notes:

d = signed 8-bit displacement

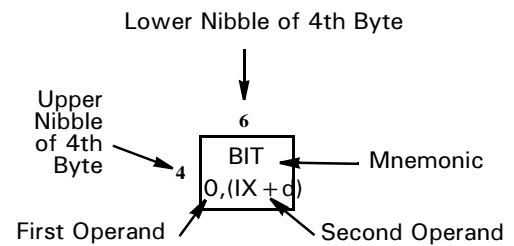
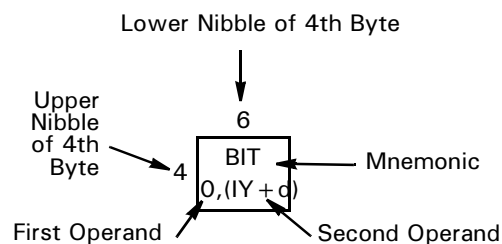


TABLE 145. OP CODE MAP (4TH BYTE, AFTER 0FDH, 0CBH, AND D)

		LOWER NIBBLE (HEX)															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
UPPER NIBBLE (HEX)	0							RLC (IY ± d)								RRC (IY ± d)	
	1							RL (IY ± d)								RR (IY ± d)	
	2							SLA (IY ± d)								SRA (IY ± d)	
	3															SRL (IY ± d)	
	4							BIT 0, (IY ± d)								BIT 1, (IY ± d)	
	5							BIT 2, (IY ± d)								BIT 3, (IY ± d)	
	6							BIT 4, (IY ± d)								BIT 5, (IY ± d)	
	7							BIT 6, (IY ± d)								BIT 7, (IY ± d)	
	8							RES 0, (IY ± d)								RES 1, (IY ± d)	
	9							RES 2, (IY ± d)								RES 3, (IY ± d)	
	A							RES 4, (IY ± d)								RES 5, (IY ± d)	
	B							RES 6, (IY ± d)								RES 7, (IY ± d)	
	C							SET 0, (IY ± d)								SET 1, (IY ± d)	
	D							SET 2, (IY ± d)								SET 3, (IY ± d)	
	E							SET 4, (IY ± d)								SET 5, (IY ± d)	
	F							SET 6, (IY ± d)								SET 7, (IY ± d)	
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Notes:

d = signed 8-bit displacement



ELECTRICAL CHARACTERISTICS

ABSOLUTE MAXIMUM RATINGS

Stresses greater than those listed below may cause permanent damage to the device. These are stress ratings only; proper 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.

TABLE 146. ABSOLUTE MAXIMUM RATINGS

Parameter	Min	Max	Units	Notes
Ambient Temperature under Bias	−40	+ 105	C	1
Storage Temperature	−65	+ 150	C	
Voltage on any Pin with Respect to V_{SS}	−0.7	+ 12	V	2
Voltage on V_{DD} Pin with Respect to V_{SS}	−0.3	+ 7	V	
Total Power Dissipation		TBD	mW	
Maximum Current out of V_{SS}		TBD	mA	
Maximum Current into V_{DD}		TBD	mA	
Maximum Current on Input and/or Inactive Output Pin	−TBD	+ TBD	μ A	
Maximum Output Current	−TBD	TBD	mA	

Notes:

1. Operating temperature is specified in DC Characteristics
2. Applies to all pins except where noted otherwise. Maximum current through a pin is specified below.

STANDARD TEST CONDITIONS

Unless otherwise noted, the DC and AC characteristics in this document are measured under standard test conditions that include the load circuit described in Figure 24. This circuit closely mimics the loading presented by active devices such as memories and peripheral devices.

All voltages are referenced to the V_{SS} pins (ground, 0V). Positive current flows into the referenced pin.

All AC parameters assume a load capacitance of 100 pF. See “Characteristic Curves” on page 224 for the effect of lesser or greater total capacitance on the timing. AC timing measurements are referenced to the high and low voltage thresholds given in the DC specifications, as indicated in Figures 25 through 40.

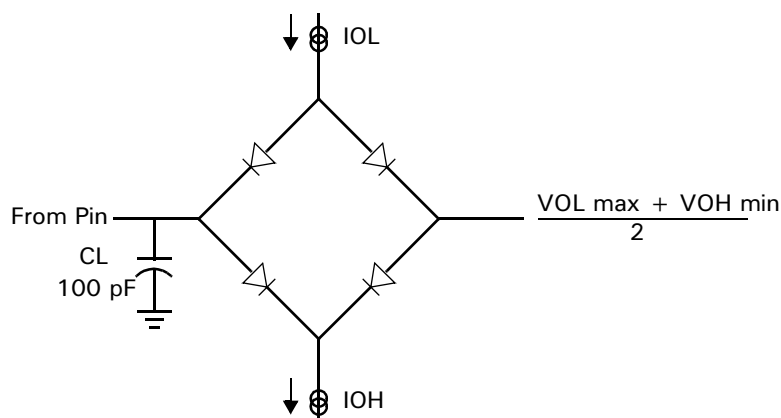


FIGURE 24. TEST LOAD CIRCUIT

DC CHARACTERISTICS

Tables 147 and 148 show the DC Characteristics of the Z80S188, for temperature ranges $T_A = 0^\circ\text{C}$ to $+70^\circ\text{C}$ and -40°C to $+85^\circ\text{C}$ respectively.

TABLE 147. DC CHARACTERISTICS, $T_A = 0^\circ\text{C}$ to $+70^\circ\text{C}$

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{IH1}	Input high Voltage (RESET, EXTAL, NMI)		$V_{DD}-0.6$		$V_{DD}+0.3$	V
V_{IH2}	Input high Voltage (Except RESET, EXTAL, NMI, CKS)		2.0		$V_{DD}+0.3$	V
V_{IH3}	Input high Voltage (CKS)		2.4		$V_{DD}+0.3$	V
V_{IL1}	Input low Voltage (RESET, EXTAL, NMI)		-0.3		0.6	V
V_{IL2}	Input low Voltage (except RESET, EXTAL, NMI)		-0.3		0.6	V
V_{OH}	Output high Voltage	$I_{OH} = -200 \mu\text{A}$	2.4			V
		$I_{OH} = -20 \mu\text{A}$	$V_{DD}-1.2$			
V_{OL}	Output low Voltage	$I_{OL} = 2.2 \text{ mA}$			0.45	V
I_{IL}	Input Leakage (All inputs except XTAL, EXTAL, LFXTAL, LFEXTAL)	$V_{IN} = 0.5 \text{ to } V_{DD}-0.5$			1.0	μA
I_{OL}	Output Leakage	$V_{IN} = 0.5 \text{ to } V_{DD}-0.5$			1.0	μA
I_{DD}	Supply Current (Normal Operation)	20 MHz (note 1)		TBD	TBD	mA
		33 MHz (note 1)		TBD	TBD	

TABLE 147. DC CHARACTERISTICS, $T_A = 0^{\circ}\text{C}$ to $+70^{\circ}\text{C}$ (CONTINUED)

Symbol	Parameter	Condition	Min	Typ	Max	Units
I_{DD1}	Standby Current (System Stop Mode)	20 MHz (note 1)		TBD	TBD	mA
		33 MHz (note 1)		TBD	TBD	
I_{DD2}	Standby Current (Standby Mode)			TBD	TBD	μA

Notes:
1. $V_{IH} > V_{DD}-1.0\text{ V}$, $V_{IL} < 0.8\text{ V}$, $V_{DD} = 5.0\text{ V}$, no outputs loaded.

TABLE 148. DC CHARACTERISTICS, $T_A = -40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{IH1}	Input High Voltage ($\overline{\text{RESET}}$, $\overline{\text{EXTAL}}$, $\overline{\text{NMI}}$)		$V_{DD}-0.6$		$V_{DD}+0.3$	V
V_{IH2}	Input High Voltage (Except $\overline{\text{RESET}}$, $\overline{\text{EXTAL}}$, $\overline{\text{NMI}}$, CKS)		2.0		$V_{DD}+0.3$	V
V_{IH3}	Input High Voltage (CKS)		2.4		$V_{DD}+0.3$	V
V_{IL1}	Input Low Voltage ($\overline{\text{RESET}}$, $\overline{\text{EXTAL}}$, $\overline{\text{NMI}}$)		-0.3		0.6	V
V_{IL2}	Input Low Voltage (Except $\overline{\text{RESET}}$, $\overline{\text{EXTAL}}$, $\overline{\text{NMI}}$)		-0.3		0.6	V
V_{OH}	Output High Voltage	$I_{OH} = -200\text{ }\mu\text{A}$	2.4			V
		$I_{OH} = -20\text{ }\mu\text{A}$	$V_{DD}-1.2$			
V_{OL}	Output Low Voltage	$I_{OL} = 2.2\text{ mA}$			0.45	V
I_{IL}	Input Leakage (All inputs except XTAL, EXTAL, LFX TAL, LFEXTAL)	$V_{IN} = 0.5\text{ to }V_{DD}-0.5$			1.0	μA
I_{OL}	Output Leakage	$V_{IN} = 0.5\text{ to }V_{DD}-0.5$			1.0	μA
I_{DD}	Supply Current (Normal Opera- tion)	20 MHz (note 1)		TBD	TBD	mA
		33 MHz (note 1)		TBD	TBD	
I_{DD1}	Standby Current (System Stop Mode)	20 MHz (note 1)		TBD	TBD	mA
		33 MHz (note 1)		TBD	TBD	
I_{DD2}	Standby Current (Standby Mode)			TBD	TBD	μA

NOTE:
1. $V_{IH} > V_{DD}-1.0\text{ V}$, $V_{IL} < 0.8\text{ V}$, $V_{DD} = 5.0\text{ V}$, no outputs loaded.



AC CHARACTERISTICS

Tables 149 and 150 give the AC Characteristics of the Z80S188, over temperature ranges $T_A = 0^\circ\text{C}$ to $+70^\circ\text{C}$ and $T_A = -40^\circ\text{C}$ to $+85^\circ\text{C}$ respectively, for the parameters illustrated in Figures 25 through 40.

TABLE 149. AC CHARACTERISTICS, $T_A = 0^\circ\text{C}$ TO $+70^\circ\text{C}$, $C_L = 100\text{ pF}$

No	Symbol	Parameter	20 MHz		33 MHz		Units
			Min	Max	Min	Max	
1	f_{OSC}	Crystal Frequency		20		33.33	MHz
2	t_{EXCYC}	External Clock Cycle Time (EXTAL)	50	DC	30	DC	ns
3	t_{CHW}	External Clock High Time (EXTAL)		15		10	ns
4	t_{CLW}	External Clock Low Time (EXTAL)		15		10	ns
5	t_{EXr}	External Clock Rise Time (EXTAL)		10		5	ns
6	t_{EXf}	External Clock Fall Time (EXTAL)		10		5	ns
7	t_{CHW}	PHI High Time	20		12		ns
8	t_{CLW}	PHI Low Time	20		12		ns
9	t_{CYC}	PHI Cycle Time	50	DC	30	DC	ns
10	t_{RES}	$\overline{\text{RESET}}$ Setup to PHI Fall ²	10		8		ns
11	t_{REH}	$\overline{\text{RESET}}$ Hold from PHI Fall ²	5		5		ns
12	t_{Rr}	$\overline{\text{RESET}}$ Fall Time ¹		50		50	ms
13	t_{RL}	$\overline{\text{RESET}}$ Low Time	6		6		t_{CYC}
14	t_{Rf}	$\overline{\text{RESET}}$ Rise Time ¹		50		50	ms
15	t_{lr}	Input Fall Time (except EXTAL, $\overline{\text{RESET}}$) ¹		50		50	ns
16	t_{lf}	Input Rise Time (except EXTAL, $\overline{\text{RESET}}$) ¹		50		50	ns
17	t_{AV}	PHI Rise to Address Valid		15		5	ns
Memory Read							
18	t_{CSV}	PHI Rise to Chip Selects Valid ³		TBS		TBS	ns
19	t_{M1L}	PHI Rise to $\overline{\text{M1}}$ Fall		15		15	ns
20	t_{STV}	PHI Fall to ST Valid ⁴		15		15	ns
21	t_{ASMR}	Address Valid to $\overline{\text{MREQ}}$ Fall	$t_{\text{CHW}}-5$		$t_{\text{CHW}}-10$		ns
22	t_{CSVMR}	Chip Selects Valid to $\overline{\text{MREQ}}$ Fall ³	TBS		TBS		ns
23	t_{STBL}	PHI Fall to $\overline{\text{MREQ}}$, ST (1st op), $\overline{\text{RD}}$ Fall		15		15	ns
24	t_{WTS}	$\overline{\text{WAIT}}$ Setup to PHI Fall ²	15		15		ns

TABLE 149. AC CHARACTERISTICS, $T_A = 0^\circ\text{C}$ TO $+70^\circ\text{C}$, $C_L = 100\text{ pF}$ (CONTINUED)

No	Symbol	Parameter	20 MHz		33 MHz		Units
			Min	Max	Min	Max	
25	t_{WTH}	$\overline{\text{WAIT}}$ Hold from PHI Fall ²	5		5		ns
26	t_{STBW}	$\overline{\text{MREQ}}$, ST (1st op), $\overline{\text{RD}}$ Width Low	$2t_{CYC}-15$		$2t_{CYC}-10$		ns
27	t_{RDS}	Read Data Setup to PHI Rise ¹	15		15		ns
28	t_{M1H}	PHI Rise to $\overline{\text{M1}}$ Rise		15		15	ns
29	t_{STBH}	PHI Fall to $\overline{\text{MREQ}}$, ST (1st op), $\overline{\text{RD}}$ Rise		15		15	ns
30	t_{RDH}	Read Data Hold from $\overline{\text{RD}}$ Rise ¹	0		0		ns
31	t_{MRHAC}	$\overline{\text{MREQ}}$ Rise to Address Change	$t_{CLW}-10$		$t_{CLW}-10$		ns
32	t_{MRCSC}	$\overline{\text{MREQ}}$ Rise to Chip Selects Change ³	TBS		TBS		ns
33	t_{MRWH}	$\overline{\text{MREQ}}$ Width High Between Any 2 Cycles	$t_{CYC}-15$		$t_{CYC}-10$		ns
Memory Write							
34	t_{WDV}	PHI Fall to Write Data Valid		20		20	ns
35	t_{WRDS}	Write Data Valid to $\overline{\text{WR}}$ Fall	$t_{CLW}-15$		$t_{CLW}-15$		ns
36	t_{WRL}	PHI Rise to $\overline{\text{WR}}$ Fall		15		15	ns
37	t_{WRPL}	$\overline{\text{WR}}$ Width Low	$t_{CYC}+$ $t_{CHW}-10$		$t_{CYC}+$ $t_{CHW}-10$		ns
38	t_{WRH}	PHI Fall to $\overline{\text{WR}}$ Rise		15		15	ns
39	t_{WDH}	$\overline{\text{WR}}$ Rise to Write Data Change	$t_{CLW}-20$		$t_{CLW}-12$		ns
40	t_{WDZ}	PHI Rise to Write Data Float		10		10	ns
I/O Read							
41	t_{AVIR}	Address Valid to $\overline{\text{IORQ}}$ Fall (/IOC = 1)	$t_{CHW}-10$		$t_{CHW}-10$		ns
42		Address Valid to $\overline{\text{IORQ}}$ Fall (/IOC = 0)	$t_{CYC}-10$		$t_{CYC}-10$		ns
43	t_{CSVIR}	Chip Selects Valid to $\overline{\text{IORQ}}$ Fall (/IOC = 1) ³	$t_{CHW}-10$		$t_{CHW}-10$		ns
44		Chip Selects Valid to $\overline{\text{IORQ}}$ Fall (/IOC = 0) ³	$t_{CYC}-10$		$t_{CYC}-10$		ns
45	t_{ISTL}	PHI Fall to $\overline{\text{IORQ}}$, $\overline{\text{RD}}$ Fall (/IOC = 1)		15		15	ns
46		PHI Rise to $\overline{\text{IORQ}}$, $\overline{\text{RD}}$ Fall (/IOC = 0)		15		15	ns
47	t_{ISTW}	$\overline{\text{IORQ}}$, $\overline{\text{RD}}$ Width Low (/IOC = 1)	$3t_{CYC}-15$		$3t_{CYC}-10$		ns
48		$\overline{\text{IORQ}}$, $\overline{\text{RD}}$ Width Low (/IOC = 0)	$2t_{CYC}+$ $t_{CHW}-15$		$2t_{CYC}+$ $t_{CHW}-10$		ns
49	t_{ISTH}	PHI Fall to $\overline{\text{IORQ}}$, $\overline{\text{RD}}$ Rise		15		15	ns
50	t_{IRAC}	$\overline{\text{IORQ}}$ Rise to Address Change	$t_{CLW}-20$		$t_{CLW}-12$		ns

TABLE 149. AC CHARACTERISTICS, $T_A = 0^{\circ}\text{C}$ TO $+70^{\circ}\text{C}$, $C_L = 100\text{ pF}$ (CONTINUED)

No	Symbol	Parameter	20 MHz		33 MHz		Units
			Min	Max	Min	Max	
51	t_{IRCSC}	$\overline{\text{IORQ}}$ Rise to Chip Selects Change ³	TBS		TBS		ns
I/O Write							
52	t_{OWPL}	$\overline{\text{WR}}$ Width Low	$2t_{\text{CYC}} + t_{\text{CHW}} - 15$		$2t_{\text{CYC}} + t_{\text{CHW}} - 10$		ns
Bus Exchange Timing							
53	t_{BRS}	$\overline{\text{BUSREQ}}$ Setup to PHI Fall ²	10		10		ns
54	t_{BRH}	$\overline{\text{BUSREQ}}$ Hold After PHI Fall ²	10		10		ns
55	t_{RLAL}	$\overline{\text{BUSREQ}}$ Low to $\overline{\text{BUSACK}}$ Low	$t_{\text{CYC}} + t_{\text{CLW}}$		$t_{\text{CYC}} + t_{\text{CLW}}$		ns
56	t_{BAL}	PHI Rise to $\overline{\text{BUSACK}}$ Fall		15		15	ns
57	t_{BZ}	PHI Rise to Bus Float		10		10	ns
58	t_{RHAH}	$\overline{\text{BUSREQ}}$ High to $\overline{\text{BUSACK}}$ High	t_{CYC}	$2t_{\text{CYC}}$	t_{CYC}	$2t_{\text{CYC}}$	ns
59	t_{BAH}	PHI Fall to $\overline{\text{BUSACK}}$ Rise		15		15	ns
60	t_{BV}	PHI Rise to Bus Valid		30		20	ns
Interrupt Timing							
61	t_{INTS}	$\overline{\text{INT0}}$ Setup to PHI Fall ²	15		15		ns
62	t_{INTH}	$\overline{\text{INT0}}$ Hold after PHI Fall ²	5		5		ns
63	t_{IELD}	IEI Low to IEO Low		10		10	ns
64	t_{IEHD}	IEI High to IEO High		TBS		TBS	ns
65	t_{IEHD}	$\overline{\text{M1}}$ Low to IEO Low ⁵		TBS		TBS	ns
66	t_{IORL}	PHI Low to $\overline{\text{IORQ}}$ Low (Int Ack Cycle)		15		15	ns
67	t_{IEIS}	IEI setup to $\overline{\text{IORQ}}$ Low (Int Ack Cycle) ¹	TBS		TBS		ns
68	t_{INTS}	D7-0 Hold from $\overline{\text{M1}}$ High (Int Ack Cycle) ¹	0		0		ns
69	t_{INTS}	$\overline{\text{INT1-2}}$ Setup to PHI Fall (Level Sense) ²	15		15		ns
70	t_{INTH}	$\overline{\text{INT1-2}}$ Hold after PHI Fall (Level Sense) ²	5		5		ns
71	t_{INTL}	$\overline{\text{INT1-2}}$ Pulse Width (Edge Sense) ¹	15		15		ns
72	t_{NMIL}	$\overline{\text{NMI}}$ Width Low ¹	35		25		ns
Refresh Timing							
73	t_{REFL}	PHI rise to RFSH Low		TBS		TBS	ns

TABLE 149. AC CHARACTERISTICS, $T_A = 0^{\circ}\text{C}$ TO $+70^{\circ}\text{C}$, $C_L = 100\text{ pF}$ (CONTINUED)

No	Symbol	Parameter	20 MHz		33 MHz		Units
			Min	Max	Min	Max	
74	t_{REFH}	PHI rise to RFSH High		TBS		TBS	ns
75	t_{MRFWL}	$\overline{\text{MREQ}}$ width Low in 2-clock refresh	$t_{\text{CYC}}-15$		$t_{\text{CYC}}-10$		ns
76	t_{MRQWL}	$\overline{\text{MREQ}}$ width Low in any other cycle	$2t_{\text{CYC}}-15$		$2t_{\text{CYC}}-10$		ns
DMA Timing							
77	t_{DROS}	$\overline{\text{DREQ0-1}}$ Setup to PHI Rise (Level Sense) ²	15		15		ns
78	t_{DROH}	$\overline{\text{DREQ0-1}}$ Hold from PHI Rise (Level Sense) ²	5		5		ns
79	t_{DRQL}	$\overline{\text{DREQ0-1}}$ Low Width (Edge Sense) ¹	t_{CYC}		t_{CYC}		ns
80	t_{DRQH}	$\overline{\text{DREQ0-1}}$ High Width (Edge Sense) ¹	t_{CYC}		t_{CYC}		ns
81	t_{TEV}	PHI Fall to $\overline{\text{TEND0-1}}$ Valid		TBS		TBS	ns
82	t_{TEW}	$\overline{\text{TEND0-1}}$ Pulse Width Low	$2t_{\text{CYC}}-15$		$2t_{\text{CYC}}-10$		ns
Halt, Sleep Timing							
83	t_{HAL}	PHI Rise to $\overline{\text{HALT}}$ Low		TBS		TBS	ns
84	t_{HAH}	PHI Rise to $\overline{\text{HALT}}$ High		TBS		TBS	ns
PRT Timing							
85	t_{CKATX}	PHI Fall to TOUT Valid		TBS		TBS	ns
ASCI Timing							
86	t_{CKATX}	CKAn fall to TXA Valid		TBS		TBS	ns
87	t_{RXSCKA}	RXAn Setup to CKAn Rise (1X Mode)	TBS		TBS		ns
88	t_{RXHCKA}	RXAn Hold From CKAn rise (1X Mode)	TBS		TBS		ns
89	f_{OLDBRG}	Max Data Rate (/16 Clock, Old BRG)		$f_{\text{PHI}}/160$		$f_{\text{PHI}}/160$	bits/s
90	f_{NEWBORG}	Max Data Rate (/16 Clock, New BRG)		$f_{\text{PHI}}/64$		$f_{\text{PHI}}/64$	bits/s
CSI/O Timing							
91	t_{STDI}	CKS Low to TXS Valid		10		10	ns
92	t_{SRSI}	RXS Valid to CKS High ¹	15		15		ns
93	t_{SRHI}	CKS High to RXS Invalid ¹	5		5		ns
WDT Timing							
94	t_{WDTL}	PHI rise to $\overline{\text{WDTOUT}}$ Low		TBS		TBS	ns
95	t_{WDTH}	PHI rise to $\overline{\text{WDTOUT}}$ High		TBS		TBS	ns

TABLE 149. AC CHARACTERISTICS, $T_A = 0^{\circ}\text{C}$ TO $+70^{\circ}\text{C}$, $C_L = 100\text{ pF}$ (CONTINUED)

			20 MHz		33 MHz		
No	Symbol	Parameter	Min	Max	Min	Max	Units
PIO Timing							
96	t _{MOPD}	$\overline{\text{IORQ}}$ High (mode 0 port write) to Px7-0 Valid		TBS		TBS	ns
97	t _{RDYH}	PHI Fall to xRDY High		TBS		TBS	ns
98	t _{STBL}	$\overline{\text{xSTB}}$ Pulse Width Low ¹	TBS		TBS		ns
99	t _{STBS}	$\overline{\text{xSTB}}$ Rise to PHI Fall ²	TBS		TBS		ns
100	t _{RDYL}	PHI Fall to xRDY Low		TBS		TBS	ns
101	t _{STBH}	$\overline{\text{xSTB}}$ Hold after PHI Fall ²	TBS		TBS		ns
102	t _{M1DS}	Px7-0 Setup to xSTB High (Mode 1, 2) ¹	TBS		TBS		ns
103	t _{M1DH}	Px7-0 Hold After xSTB High (Mode 1, 2) ¹	TBS		TBS		ns
104	t _{RDLSTL}	xRDY fall to $\overline{\text{xSTB}}$ Fall (Mode 1, 2)	0		0		ns
105	t _{SLDV}	$\overline{\text{xSTB}}$ Low to Px7-0 Valid (Mode 2 In)		TBS		TBS	ns
106	t _{SHDZ}	$\overline{\text{xSTB}}$ High to Px7-0 released (mode 2 In)		TBS		TBS	ns
107	t _{SHXDNZ}	$\overline{\text{xSTB}}$ High to Px7-0 Ext Driven (Mode 2)	t _{SHDZ}		t _{SHDZ}		ns
108	t _{PS}	Px7-0 Setup to $\overline{\text{RD}}$ Low (Mode 3)	TBS		TBS		ns
109	t _{PH}	Px7-0 Hold After $\overline{\text{RD}}$ Low (Mode 3)	TBS		TBS		ns
CTC Timing							
110	t _{CTCY}	CLK/TRGn Cycle Time	2 t _{CYC}		2 t _{CYC}		ns
111	t _{CTW}	CLK/TRGn High or Low Pulse Width	t _{CYC}		t _{CYC}		ns
112	t _{CTS}	CLK/TRGn Active Edge Setup to PHI Rise ²	TBS		TBS		ns
113	t _{CTH}	CLK/TRGn Active Edge Hold After PHI Rise ²	TBS		TBS		ns
114	t _{ZCH}	PHI rise to ZC/TOn rise		TBS		TBS	ns
115	t _{ZCL}	PHI Fall to ZC/TOn Fall		TBS		TBS	ns
SIO Timing							
116	t _{PWH}	$\overline{\text{CTS}}$, $\overline{\text{DCD}}$, $\overline{\text{SYNC}}$ Pulse Width High	TBS		TBS		ns
117	t _{PWL}	$\overline{\text{CTS}}$, $\overline{\text{DCD}}$, $\overline{\text{SYNC}}$ Pulse Width Low	TBS		TBS		ns
118	t _{TXCY}	$\overline{\text{TxC}}$ Cycle Time ¹	TBS		TBS		ns

TABLE 149. AC CHARACTERISTICS, $T_A = 0^\circ\text{C}$ TO $+70^\circ\text{C}$, $C_L = 100\text{ pF}$ (CONTINUED)

No	Symbol	Parameter	20 MHz		33 MHz		Units
			Min	Max	Min	Max	
119	t_{TXL}	$\overline{\text{TxC}}$ Width Low ¹	TBS		TBS		ns
120	t_{TXH}	$\overline{\text{TxC}}$ Width High ¹	TBS		TBS		ns
121	t_{TXD}	$\overline{\text{TxC}}$ Fall to TxD Valid		TBS		TBS	ns
122	t_{TXRDY}	$\overline{\text{TxC}}$ Fall to $\overline{\text{W/RDY}}$ (Ready) Low		TBS		TBS	ns
123	t_{RXCy}	$\overline{\text{RxC}}$ Cycle Time ¹	TBS		TBS		ns
124	t_{RXL}	$\overline{\text{RxC}}$ Width Low ¹	TBS		TBS		ns
125	t_{RXH}	$\overline{\text{RxC}}$ Width High ¹	TBS		TBS		ns
126	t_{RXDS}	RxD Valid Setup to $\overline{\text{RxC}}$ Rise (X1 Mode)	TBS		TBS		ns
127	t_{RXDS}	RxD Valid Hold From $\overline{\text{RxC}}$ Rise (X1 Mode)	TBS		TBS		ns
128	t_{RXRDY}	$\overline{\text{RxC}}$ Rise to $\overline{\text{W/RDY}}$ (Ready) Low		TBS		TBS	ns
129	t_{RXSY}	$\overline{\text{RxC}}$ Rise to $\overline{\text{SYNC}}$ (Output) Low		TBS		TBS	ns
130	t_{SYS}	$\overline{\text{SYNC}}$ (Ext Sync) Low Setup to $\overline{\text{RxC}}$ Rise ¹	-30		-30		ns

Notes:

1. These timing requirements must be met to assure correct device operation.
2. These Setup and Hold times must be met to guarantee recognition at the clock edge in question. If they are not met between a signal transition and a clock edge, the Z80S188 may or may not recognize the new state of the signal at that edge. If it doesn't, and the signal remains in the same state, the Z80S188 recognizes the new state one clock period later.
3. "Chip Selects" refers to $\overline{\text{MEMCS0}}$, $\overline{\text{MEMCS1}}$, $\overline{\text{IOCS}}$, and $\overline{\text{EV/ROMCS}}$ on a ROMless part.
4. PHI Fall to ST Valid applies between DMA and CPU bus cycles.
5. $\overline{\text{MT}}$ Low to IEO Low max applies when an on-chip device started requesting an interrupt just before $\overline{\text{MT}}$ went Low.

TABLE 150. AC CHARACTERISTICS, $T_A = -40^\circ\text{C}$ TO $+85^\circ\text{C}$, $C_L = 100\text{ pF}$

No	Symbol	Parameter	20 MHz		33 MHz		Units
			Min	Max	Min	Max	
1	f_{OSC}	Crystal Frequency		20		33.33	MHz
2	t_{EXCYC}	External Clock Cycle Time (EXTAL)	50	DC	30	DC	ns
3	t_{CHW}	External Clock High Time (EXTAL)		15		10	ns
4	t_{CLW}	External Clock Low Time (EXTAL)		15		10	ns
5	t_{EXr}	External Clock Rise Time (EXTAL)		10		5	ns
6	t_{EXf}	External Clock Fall Time (EXTAL)		10		5	ns

TABLE 150. AC CHARACTERISTICS, $T_A = -40^{\circ}\text{C}$ TO $+85^{\circ}\text{C}$, $C_L = 100\text{ pF}$ (CONTINUED)

No	Symbol	Parameter	20 MHz		33 MHz		Units
			Min	Max	Min	Max	
7	t_{CHW}	PHI High Time	20		12		ns
8	t_{CLW}	PHI Low Time	20		12		ns
9	t_{CYC}	PHI Cycle Time	50	DC	30	DC	ns
10	t_{RES}	$\overline{\text{RESET}}$ Setup to PHI Fall ²	10		8		ns
11	t_{REH}	$\overline{\text{RESET}}$ Hold from PHI Fall ²	5		5		ns
12	t_{Rr}	$\overline{\text{RESET}}$ Fall Time ¹		50		50	ms
13	t_{RL}	$\overline{\text{RESET}}$ Low Time	6		6		t_{CYC}
14	t_{Rf}	$\overline{\text{RESET}}$ Rise Time ¹		50		50	ms
15	t_{lr}	Input Fall Time (except EXTAL, $\overline{\text{RESET}}$) ¹		50		50	ns
16	t_{lf}	Input Rise Time (except EXTAL, $\overline{\text{RESET}}$) ¹		50		50	ns
17	t_{AV}	PHI Rise to Address Valid		15		5	ns
Memory Read							
18	t_{CSV}	PHI Rise to Chip Selects Valid ³		TBS		TBS	ns
19	t_{M1L}	PHI Rise to $\overline{\text{M1}}$ Fall		15		15	ns
20	t_{STV}	PHI Fall to ST Valid ⁴		15		15	ns
21	t_{ASMR}	Address Valid to $\overline{\text{MREQ}}$ Fall	$t_{CHW}-5$		$t_{CHW}-10$		ns
22	t_{CSVMR}	Chip Selects Valid to $\overline{\text{MREQ}}$ Fall ³	TBS		TBS		ns
23	t_{STBL}	PHI Fall to $\overline{\text{MREQ}}$, ST (1st op), $\overline{\text{RD}}$ Fall		15		15	ns
24	t_{WTS}	$\overline{\text{WAIT}}$ Setup to PHI Fall ²	15		15		ns
25	t_{WTH}	$\overline{\text{WAIT}}$ Hold from PHI Fall ²	5		5		ns
26	t_{STBW}	$\overline{\text{MREQ}}$, ST (1st op), $\overline{\text{RD}}$ Width Low	$2t_{CYC}-15$		$2t_{CYC}-10$		ns
27	t_{RDS}	Read Data Setup to PHI Rise ¹	15		15		ns
28	t_{M1H}	PHI Rise to $\overline{\text{M1}}$ Rise		15		15	ns
29	t_{STBH}	PHI Fall to $\overline{\text{MREQ}}$, ST (1st op), $\overline{\text{RD}}$ Rise		15		15	ns
30	t_{RDH}	Read Data Hold from $\overline{\text{RD}}$ Rise ¹	0		0		ns
31	t_{MRHAC}	$\overline{\text{MREQ}}$ Rise to Address Change	$t_{CLW}-10$		$t_{CLW}-10$		ns
32	t_{MRCSC}	$\overline{\text{MREQ}}$ Rise to Chip Selects Change ³	TBS		TBS		ns
33	t_{MRWH}	$\overline{\text{MREQ}}$ Width High Between Any 2 Cycles	$t_{CYC}-15$		$t_{CYC}-10$		ns

TABLE 150. AC CHARACTERISTICS, $T_A = -40^{\circ}\text{C}$ TO $+85^{\circ}\text{C}$, $C_L = 100\text{ pF}$ (CONTINUED)

			20 MHz		33 MHz		
No	Symbol	Parameter	Min	Max	Min	Max	Units
Memory Write							
34	t _{WDV}	PHI Fall to Write Data Valid		20		20	ns
35	t _{WRDS}	Write Data Valid to \overline{WR} Fall	t _{CLW} -15		t _{CLW} -15		ns
36	t _{WRL}	PHI Rise to \overline{WR} Fall		15		15	ns
37	t _{WRPL}	\overline{WR} Width Low	t _{CYC} + t _{CHW} -10		t _{CYC} + t _{CHW} -10		ns
38	t _{WRH}	PHI Fall to \overline{WR} Rise		15		15	ns
39	t _{WDH}	\overline{WR} Rise to Write Data Change	t _{CLW} -20		t _{CLW} -12		ns
40	t _{WDZ}	PHI Rise to Write Data Float		10		10	ns
I/O Read							
41	t _{AVIR}	Address Valid to \overline{IORQ} Fall (/IOC = 1)	t _{CHW} -10		t _{CHW} -10		ns
42		Address Valid to \overline{IORQ} Fall (/IOC = 0)	t _{CYC} -10		t _{CYC} -10		ns
43	t _{CSVIR}	Chip Selects Valid to \overline{IORQ} Fall (/IOC = 1) ³	t _{CHW} -10		t _{CHW} -10		ns
44		Chip Selects Valid to \overline{IORQ} Fall (/IOC = 0) ³	t _{CYC} -10		t _{CYC} -10		ns
45	t _{ISTL}	PHI Fall to \overline{IORQ} , \overline{RD} Fall (/IOC = 1)		15		15	ns
46		PHI Rise to \overline{IORQ} , \overline{RD} Fall (/IOC = 0)		15		15	ns
47	t _{ISTW}	\overline{IORQ} , \overline{RD} Width Low (/IOC = 1)	3t _{CYC} -15		3t _{CYC} -10		ns
48		\overline{IORQ} , \overline{RD} Width Low (/IOC = 0)	2t _{CYC} + t _{CHW} -15		2t _{CYC} + t _{CHW} -10		ns
49	t _{ISTH}	PHI Fall to \overline{IORQ} , \overline{RD} Rise		15		15	ns
50	t _{IRAC}	\overline{IORQ} Rise to Address Change	t _{CLW} -20		t _{CLW} -12		ns
51	t _{IRCSC}	\overline{IORQ} Rise to Chip Selects Change ³	TBS		TBS		ns
I/O Write							
52	t _{OWPL}	\overline{WR} Width Low	2t _{CYC} + t _{CHW} -15		2t _{CYC} + t _{CHW} -10		ns
Bus Exchange Timing							
53	t _{BRS}	\overline{BUSREQ} Setup to PHI Fall ²	10		10		ns
54	t _{BRH}	\overline{BUSREQ} Hold After PHI Fall ²	10		10		ns
55	t _{RLAL}	\overline{BUSREQ} Low to \overline{BUSACK} Low	t _{CYC} + t _{CLW}		t _{CYC} + t _{CLW}		ns

TABLE 150. AC CHARACTERISTICS, $T_A = -40^{\circ}\text{C}$ TO $+85^{\circ}\text{C}$, $C_L = 100\text{ pF}$ (CONTINUED)

No	Symbol	Parameter	20 MHz		33 MHz		Units
			Min	Max	Min	Max	
56	t_{BAL}	PHI Rise to $\overline{\text{BUSACK}}$ Fall		15		15	ns
57	t_{BZ}	PHI Rise to Bus Float		10		10	ns
58	t_{RHAH}	$\overline{\text{BUSREQ}}$ High to $\overline{\text{BUSACK}}$ High	t_{CYC}	$2t_{CYC}$	t_{CYC}	$2t_{CYC}$	ns
59	t_{BAH}	PHI Fall to $\overline{\text{BUSACK}}$ Rise		15		15	ns
60	t_{BV}	PHI Rise to Bus Valid		30		20	ns
Interrupt Timing							
61	t_{INTS}	$\overline{\text{INT0}}$ Setup to PHI Fall ²	15		15		ns
62	t_{INTH}	$\overline{\text{INT0}}$ Hold after PHI Fall ²	5		5		ns
63	t_{IELD}	IEI Low to IEO Low		10		10	ns
64	t_{IEHD}	IEI High to IEO High		TBS		TBS	ns
65	t_{IEHD}	$\overline{\text{M1}}$ Low to IEO Low ⁵		TBS		TBS	ns
66	t_{IORL}	PHI Low to $\overline{\text{IORQ}}$ Low (Int Ack Cycle)		15		15	ns
67	t_{IEIS}	IEI setup to $\overline{\text{IORQ}}$ Low (Int Ack Cycle) ¹	TBS		TBS		ns
68	t_{INTS}	D7-0 Hold from $\overline{\text{M1}}$ High (Int Ack Cycle) ¹	0		0		ns
69	t_{INTS}	$\overline{\text{INT1-2}}$ Setup to PHI Fall (Level Sense) ²	15		15		ns
70	t_{INTH}	$\overline{\text{INT1-2}}$ Hold after PHI Fall (Level Sense) ²	5		5		ns
71	t_{INTL}	$\overline{\text{INT1-2}}$ Pulse Width (Edge Sense) ¹	15		15		ns
72	t_{NMIL}	$\overline{\text{NM1}}$ Width Low ¹	35		25		ns
Refresh Timing							
73	t_{REFL}	PHI rise to RFSH Low		TBS		TBS	ns
74	t_{REFH}	PHI rise to RFSH High		TBS		TBS	ns
75	t_{MRFWL}	$\overline{\text{MREQ}}$ width Low in 2-clock refresh	$t_{CYC}-15$		$t_{CYC}-10$		ns
76	t_{MRQWL}	$\overline{\text{MREQ}}$ width Low in any other cycle	$2t_{CYC}-15$		$2t_{CYC}-10$		ns
DMA Timing							
77	t_{DROS}	$\overline{\text{DREQ0-1}}$ Setup to PHI Rise (Level Sense) ²	15		15		ns
78	t_{DROH}	$\overline{\text{DREQ0-1}}$ Hold from PHI Rise (Level Sense) ²	5		5		ns
79	t_{DRQL}	$\overline{\text{DREQ0-1}}$ Low Width (Edge Sense) ¹	t_{CYC}		t_{CYC}		ns

TABLE 150. AC CHARACTERISTICS, $T_A = -40^{\circ}\text{C}$ TO $+85^{\circ}\text{C}$, $C_L = 100\text{ pF}$ (CONTINUED)

No	Symbol	Parameter	20 MHz		33 MHz		Units
			Min	Max	Min	Max	
80	t_{DROH}	$\overline{\text{DREQ0-1}}$ High Width (Edge Sense) ¹	t_{CYC}		t_{CYC}		ns
81	t_{TEV}	PHI Fall to $\overline{\text{TEND0-1}}$ Valid		TBS		TBS	ns
82	t_{TEW}	$\overline{\text{TEND0-1}}$ Pulse Width Low	$2t_{\text{CYC}}-15$		$2t_{\text{CYC}}-10$		ns
Halt, Sleep Timing							
83	t_{HAL}	PHI Rise to $\overline{\text{HALT}}$ Low		TBS		TBS	ns
84	t_{HAH}	PHI Rise to $\overline{\text{HALT}}$ High		TBS		TBS	ns
PRT Timing							
85	t_{CKATX}	PHI Fall to TOUT Valid		TBS		TBS	ns
ASCI Timing							
86	t_{CKATX}	CKAn fall to TXA Valid		TBS		TBS	ns
87	t_{RXSCKA}	RXAn Setup to CKAn Rise (1X Mode)	TBS		TBS		ns
88	t_{RXHCKA}	RXAn Hold From CKAn rise (1X Mode)	TBS		TBS		ns
89	f_{OLDBRG}	Max Data Rate (/16 Clock, Old BRG)		$f_{\text{PHI}}/160$		$f_{\text{PHI}}/160$	bits/s
90	f_{NEWBRG}	Max Data Rate (/16 Clock, New BRG)		$f_{\text{PHI}}/64$		$f_{\text{PHI}}/64$	bits/s
CSI/O Timing							
91	t_{STDI}	CKS Low to TXS Valid		10		10	ns
92	t_{SRSI}	RXS Valid to CKS High ¹	15		15		ns
93	t_{SRHI}	CKS High to RXS Invalid ¹	5		5		ns
WDT Timing							
94	t_{WDTL}	PHI rise to $\overline{\text{WDTOU}}$ Low		TBS		TBS	ns
95	t_{WDTH}	PHI rise to $\overline{\text{WDTOU}}$ High		TBS		TBS	ns
PIO Timing							
96	t_{MOPD}	$\overline{\text{IORQ}}$ High (mode 0 port write) to Px7-0 Valid		TBS		TBS	ns
97	t_{RDYH}	PHI Fall to xRDY High		TBS		TBS	ns
98	t_{STBL}	$\overline{\text{xSTB}}$ Pulse Width Low ¹	TBS		TBS		ns
99	t_{STBS}	$\overline{\text{xSTB}}$ Rise to PHI Fall ²	TBS		TBS		ns
100	t_{RDYL}	PHI Fall to xRDY Low		TBS		TBS	ns
101	t_{STBH}	$\overline{\text{xSTB}}$ Hold after PHI Fall ²	TBS		TBS		ns

TABLE 150. AC CHARACTERISTICS, $T_A = -40^{\circ}\text{C}$ TO $+85^{\circ}\text{C}$, $C_L = 100\text{ pF}$ (CONTINUED)

No	Symbol	Parameter	20 MHz		33 MHz		Units
			Min	Max	Min	Max	
102	t_{M1DS}	Px7-0 Setup to xSTB High (Mode 1, 2) ¹	TBS		TBS		ns
103	t_{M1DH}	Px7-0 Hold After xSTB High (Mode 1, 2) ¹	TBS		TBS		ns
104	t_{RDLSTL}	xRDY fall to $\overline{\text{xSTB}}$ Fall (Mode 1, 2)	0		0		ns
105	t_{SLDV}	$\overline{\text{xSTB}}$ Low to Px7-0 Valid (Mode 2 In)		TBS		TBS	ns
106	t_{SHDZ}	$\overline{\text{xSTB}}$ High to Px7-0 released (mode 2 In)		TBS		TBS	ns
107	t_{SHXDNZ}	$\overline{\text{xSTB}}$ High to Px7-0 Ext Driven (Mode 2)	t_{SHDZ}		t_{SHDZ}		ns
108	t_{PS}	Px7-0 Setup to $\overline{\text{RD}}$ Low (Mode 3)	TBS		TBS		ns
109	t_{PH}	Px7-0 Hold After $\overline{\text{RD}}$ Low (Mode 3)	TBS		TBS		ns
CTC Timing							
110	t_{CTCY}	CLK/TRGn Cycle Time	$2 t_{CYC}$		$2 t_{CYC}$		ns
111	t_{CTW}	CLK/TRGn High or Low Pulse Width	t_{CYC}		t_{CYC}		ns
112	t_{CTS}	CLK/TRGn Active Edge Setup to PHI Rise ²	TBS		TBS		ns
113	t_{CTH}	CLK/TRGn Active Edge Hold After PHI Rise ²	TBS		TBS		ns
114	t_{ZCH}	PHI rise to ZC/TOn rise		TBS		TBS	ns
115	t_{ZCL}	PHI Fall to ZC/TOn Fall		TBS		TBS	ns
SIO Timing							
116	t_{PWH}	$\overline{\text{CTS}}$, $\overline{\text{DCD}}$, $\overline{\text{SYNC}}$ Pulse Width High	TBS		TBS		ns
117	t_{PWL}	$\overline{\text{CTS}}$, $\overline{\text{DCD}}$, $\overline{\text{SYNC}}$ Pulse Width Low	TBS		TBS		ns
118	t_{TXCY}	$\overline{\text{TxC}}$ Cycle Time ¹	TBS		TBS		ns
119	t_{TXL}	$\overline{\text{TxC}}$ Width Low ¹	TBS		TBS		ns
120	t_{TXH}	$\overline{\text{TxC}}$ Width High ¹	TBS		TBS		ns
121	t_{TXD}	$\overline{\text{TxC}}$ Fall to TxD Valid		TBS		TBS	ns
122	t_{TXRDY}	$\overline{\text{TxC}}$ Fall to $\overline{\text{W/RDY}}$ (Ready) Low		TBS		TBS	ns
123	t_{RXCy}	$\overline{\text{RxC}}$ Cycle Time ¹	TBS		TBS		ns
124	t_{RXL}	$\overline{\text{RxC}}$ Width Low ¹	TBS		TBS		ns
125	t_{RXH}	$\overline{\text{RxC}}$ Width High ¹	TBS		TBS		ns

TABLE 150. AC CHARACTERISTICS, $T_A = -40^{\circ}\text{C}$ TO $+85^{\circ}\text{C}$, $C_L = 100\text{ pF}$ (CONTINUED)

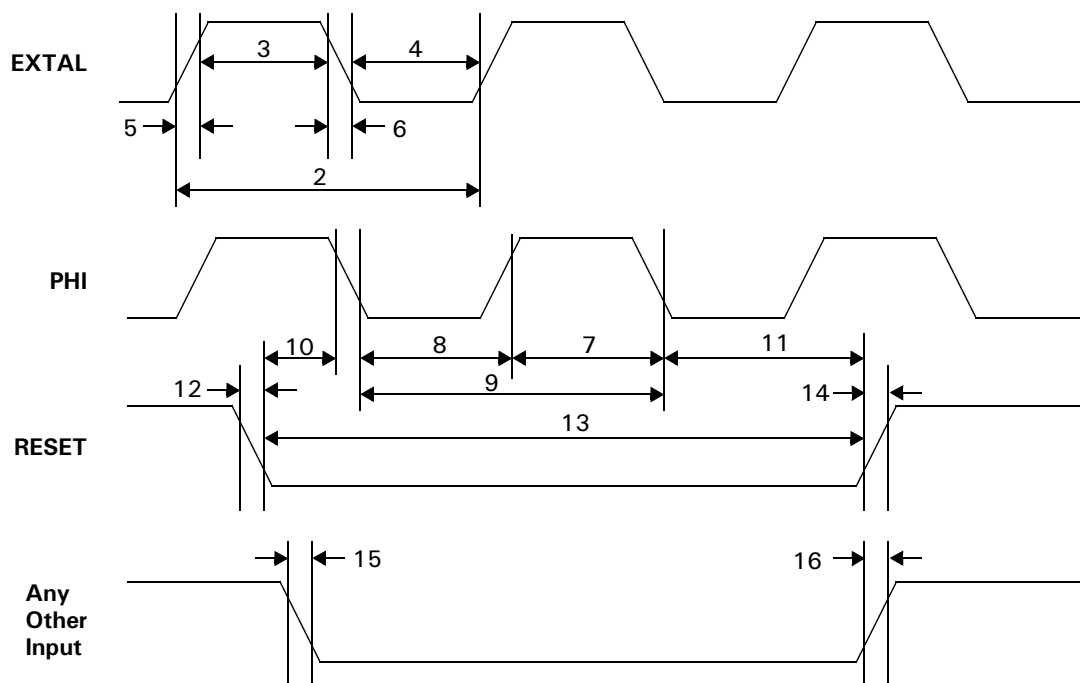
No	Symbol	Parameter	20 MHz		33 MHz		Units
			Min	Max	Min	Max	
126	t_{RXDS}	RxD Valid Setup to $\overline{\text{RxC}}$ Rise (X1 Mode)	TBS		TBS		ns
127	t_{RXDS}	RxD Valid Hold From $\overline{\text{RxC}}$ Rise (X1 Mode)	TBS		TBS		ns
128	t_{RXRDY}	$\overline{\text{RxC}}$ Rise to $\overline{\text{W/RDY}}$ (Ready) Low		TBS		TBS	ns
129	t_{RXSY}	$\overline{\text{RxC}}$ Rise to $\overline{\text{SYNC}}$ (Output) Low		TBS		TBS	ns

CAPACITANCE

The capacitance of each pin on the Z80S188 depends on whether the pin is an input, output, or both. The total capacitance associated with outputs affects their AC characteristics (switching time) as described in “Characteristic Curves”, on page 224.

Input	5 pF
Output	10 pF
I/O	12 pF

TIMING DIAGRAMS


FIGURE 25. BASIC TIMING

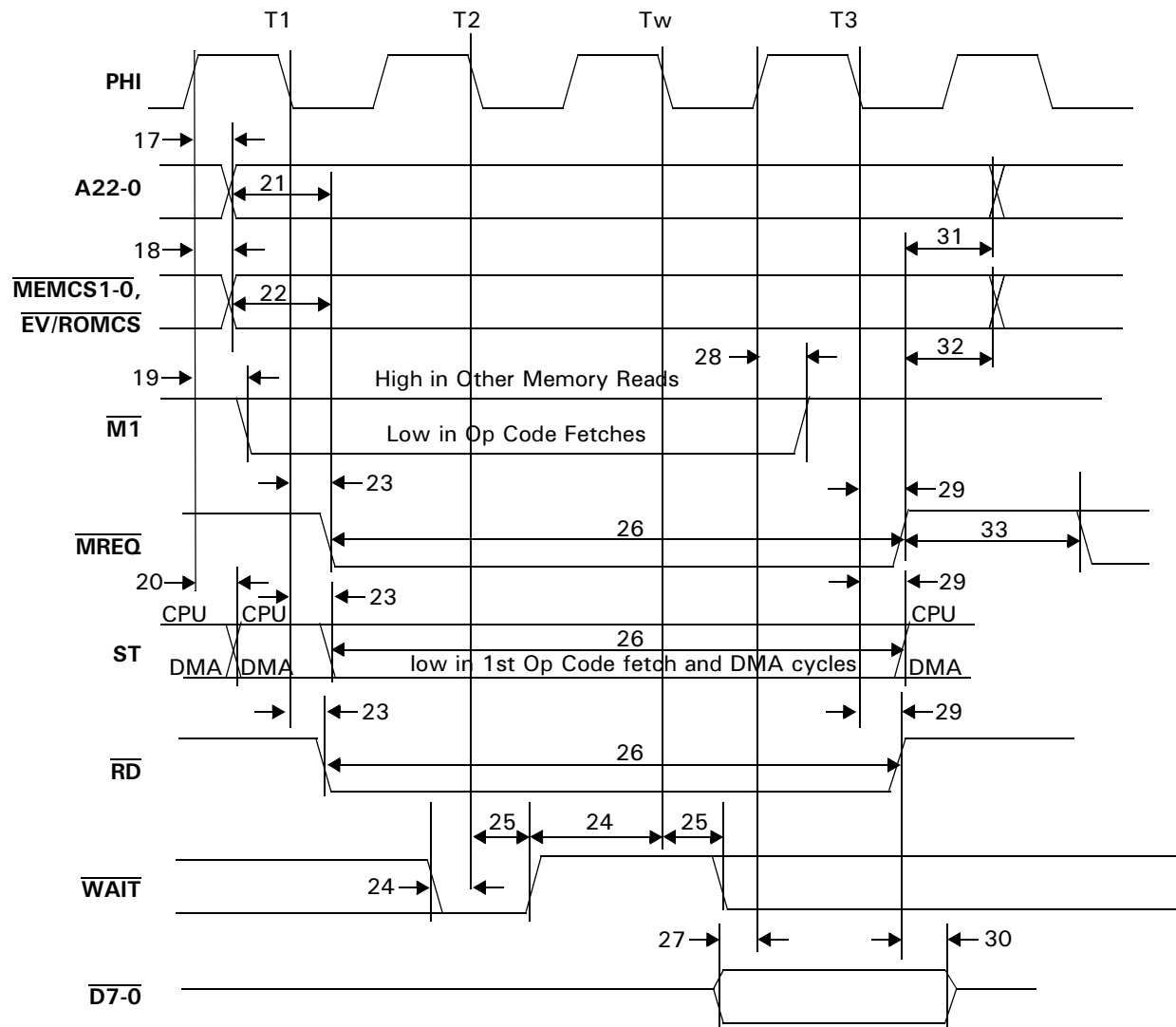


FIGURE 26. MEMORY READ TIMING (ONE WAIT STATE)

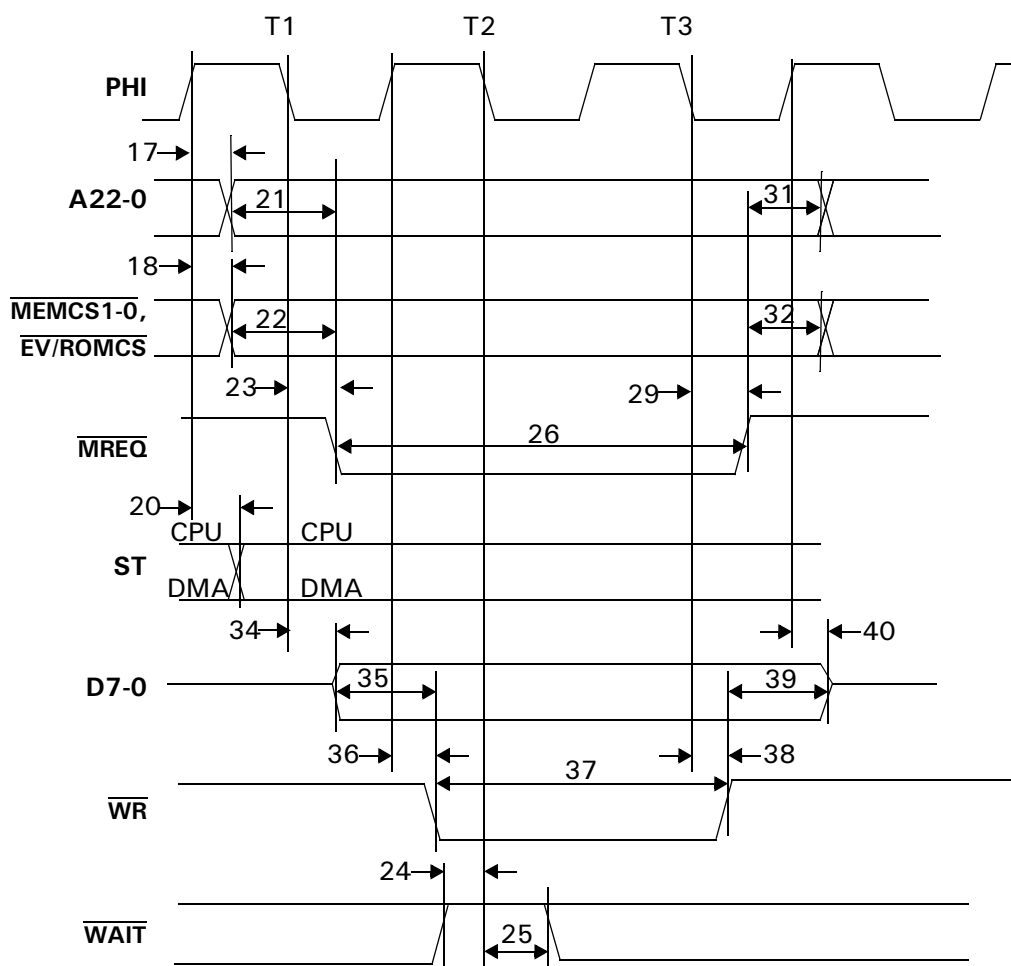


FIGURE 27. MEMORY WRITE TIMING (NO WAIT STATES)

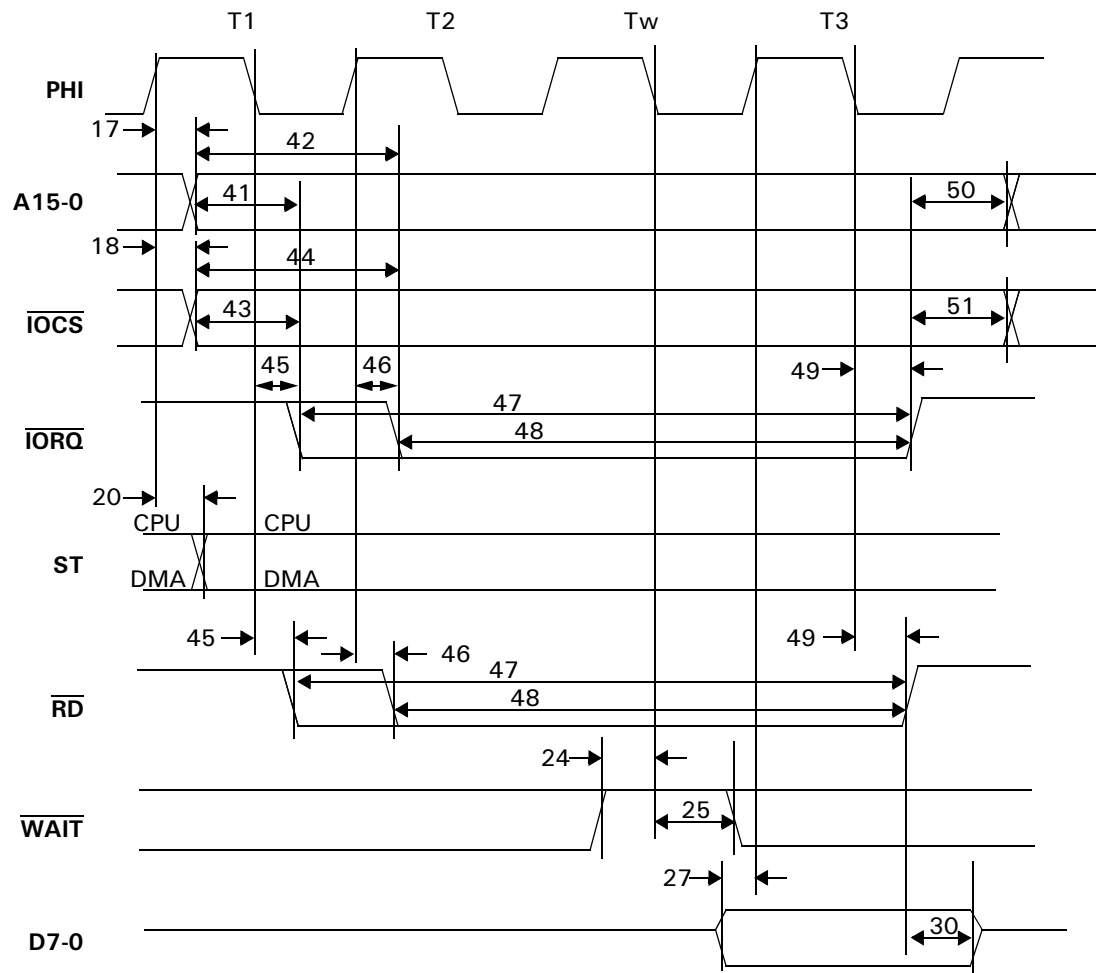


FIGURE 28. I/O READ TIMING (AUTO WAIT STATE)

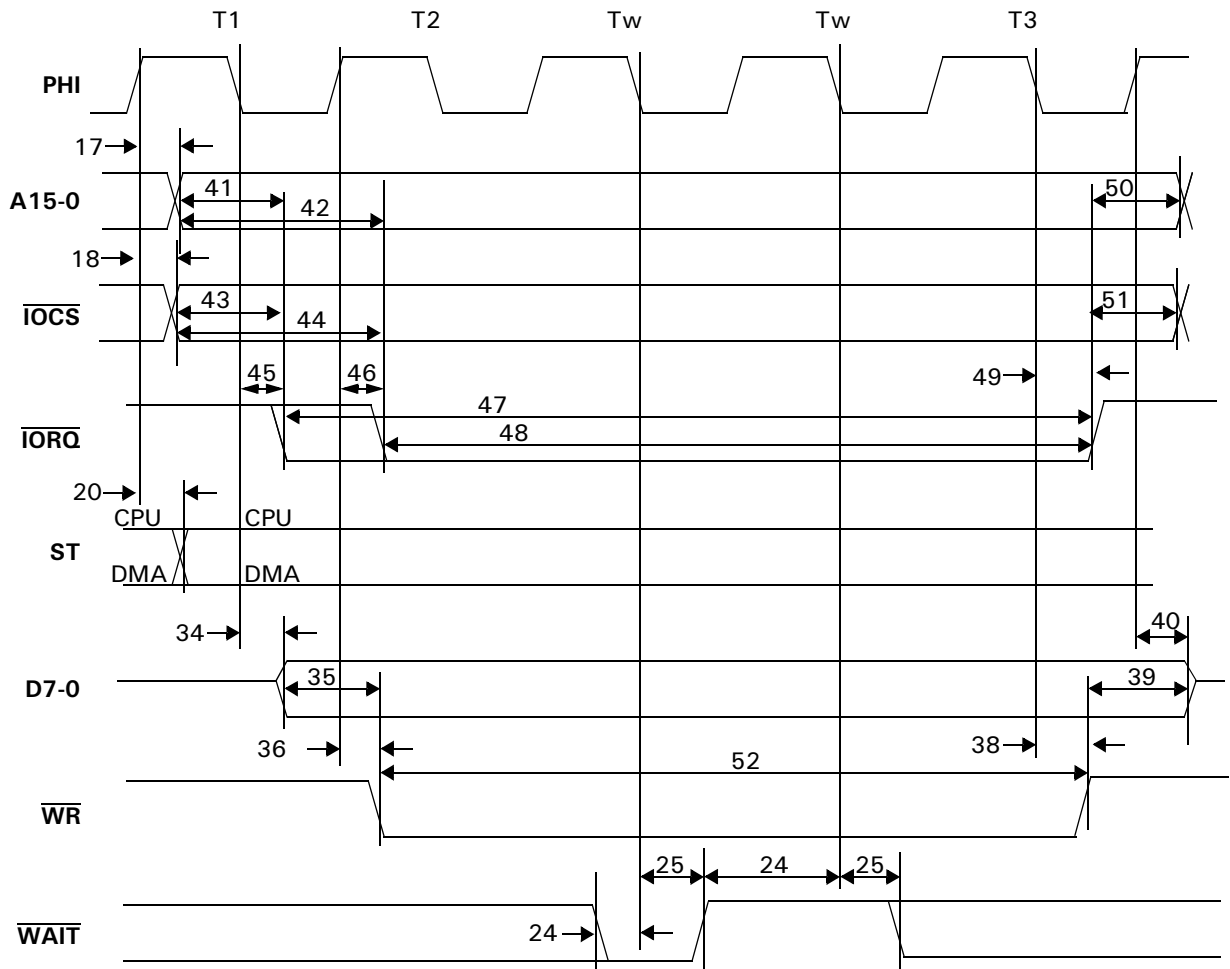


FIGURE 29. I/O WRITE TIMING (AUTO AND ONE WAIT STATE)

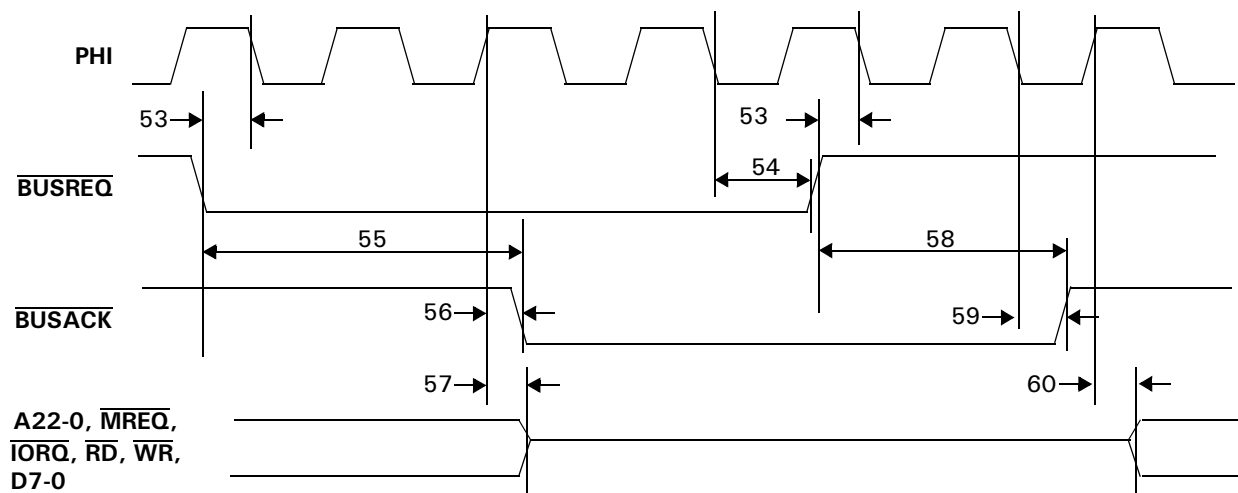


FIGURE 30. BUS EXCHANGE TIMING

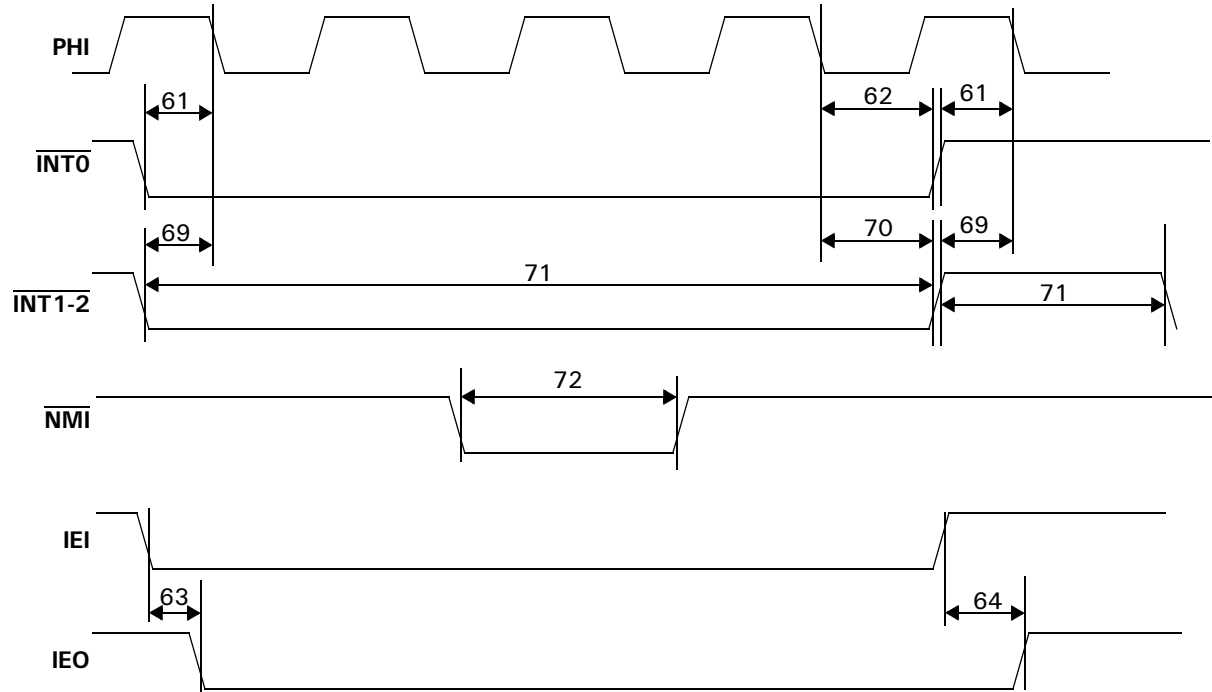


FIGURE 31. INTERRUPT TIMING

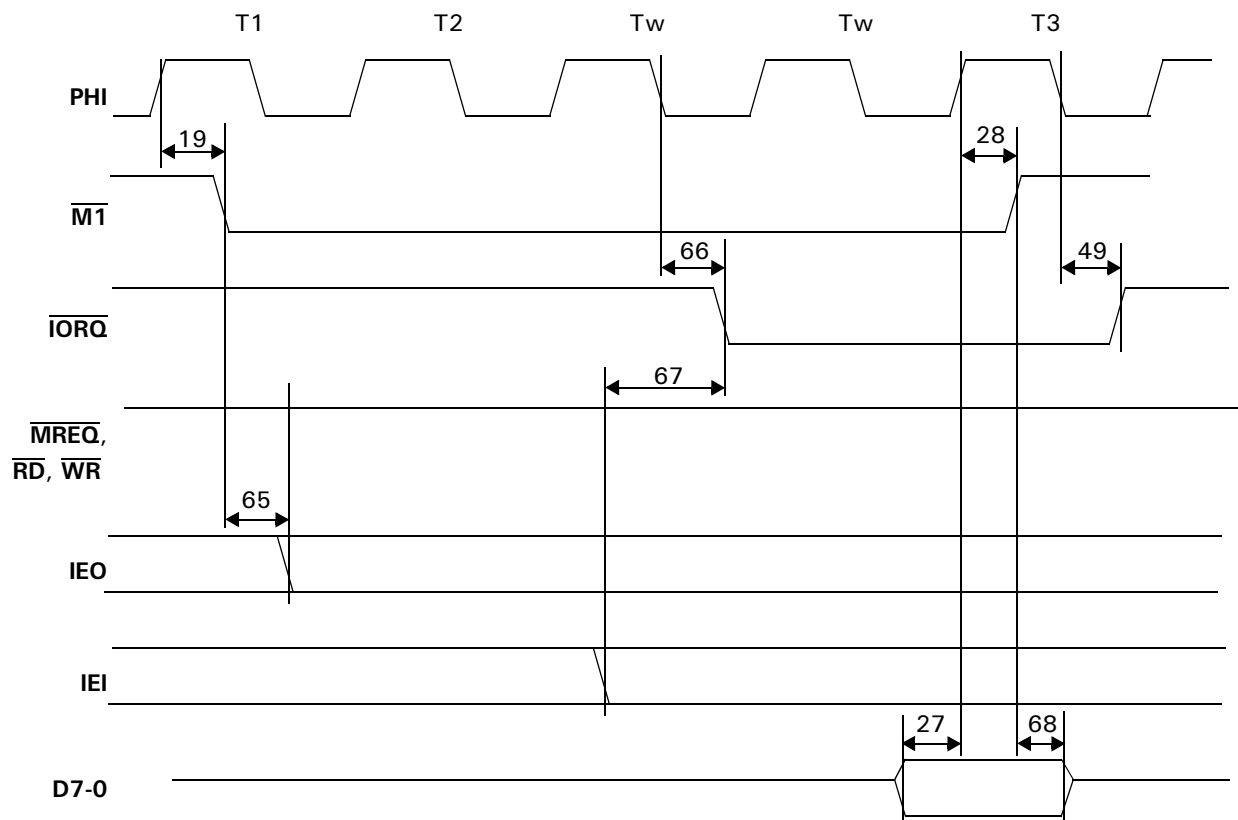
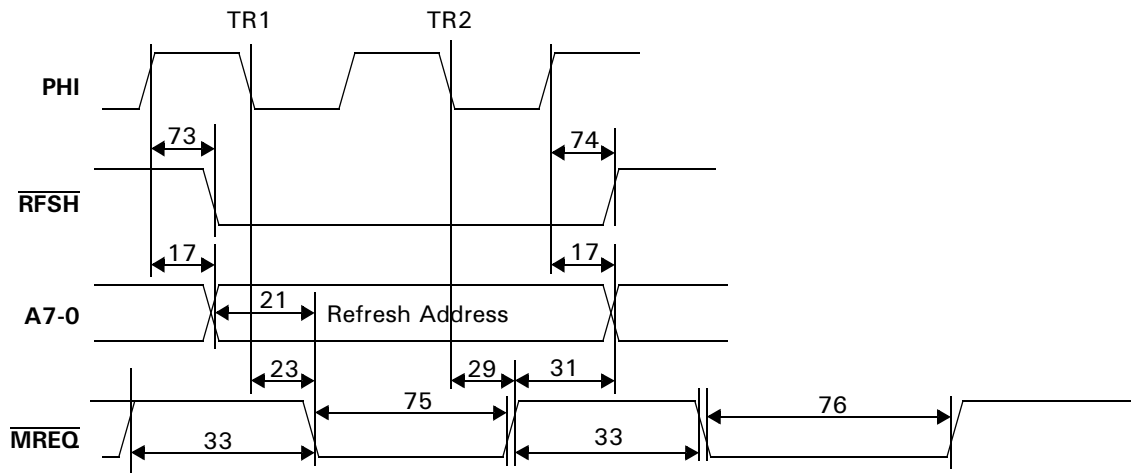


FIGURE 32. INTERRUPT ACKNOWLEDGE TIMING



Note: 3-Clock refresh simply adds another clock TRW between T_{R1} and T_{R2} , during which nothing changes.

FIGURE 33. REFRESH TIMING (2-CLOCK CYCLE)

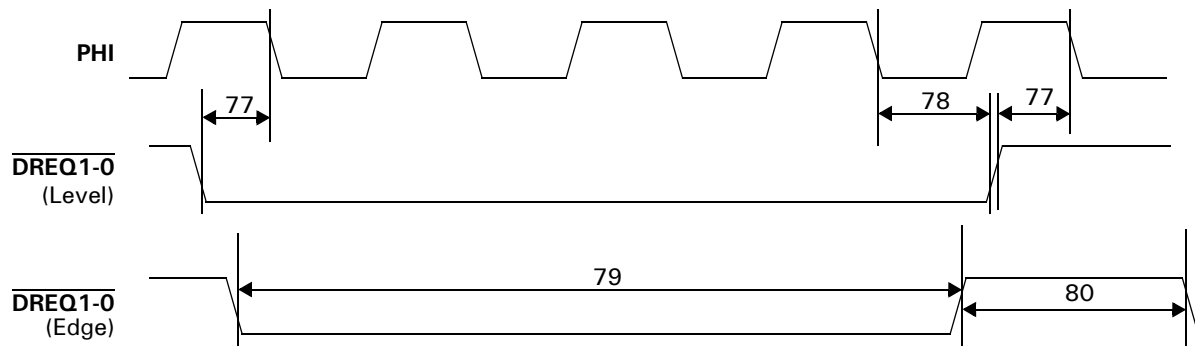


FIGURE 34. DMA REQUEST TIMING

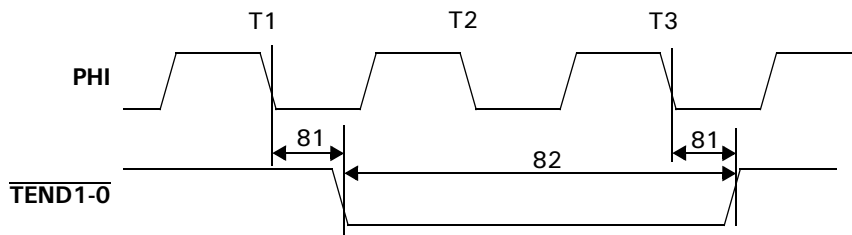
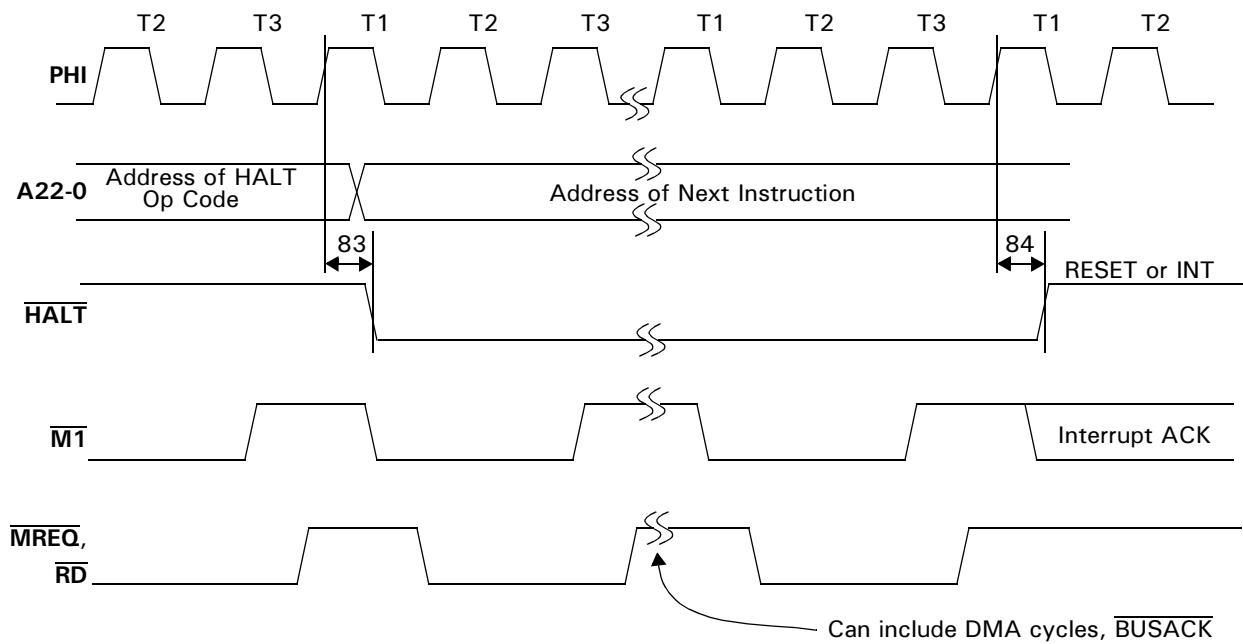
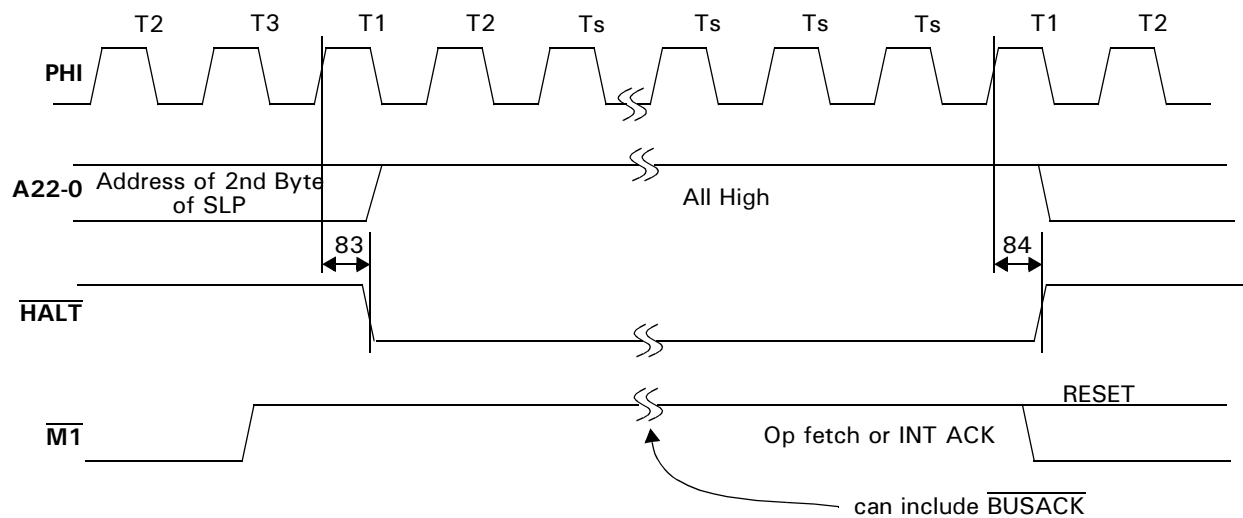


FIGURE 35. DMA TERMINATION TIMING



Note: All other timing relationships are as described in the Memory Read figure.

FIGURE 36. HALT TIMING



Note: Other timing relationships are as described in the Memory Read figure.

FIGURE 37. SLEEP TIMING

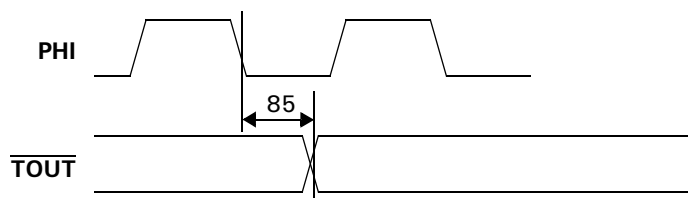


FIGURE 38. PRT TIMING

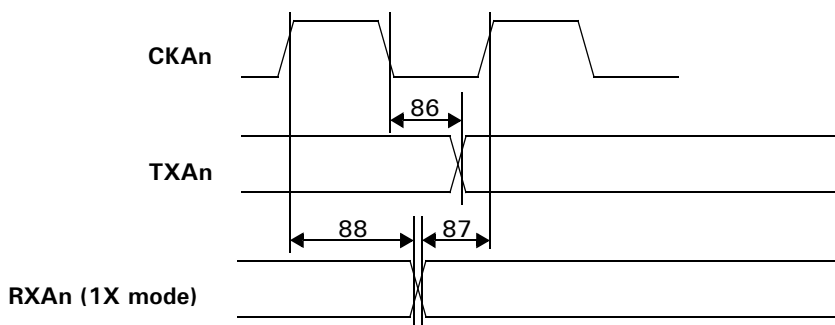


FIGURE 39. ASCII TIMING

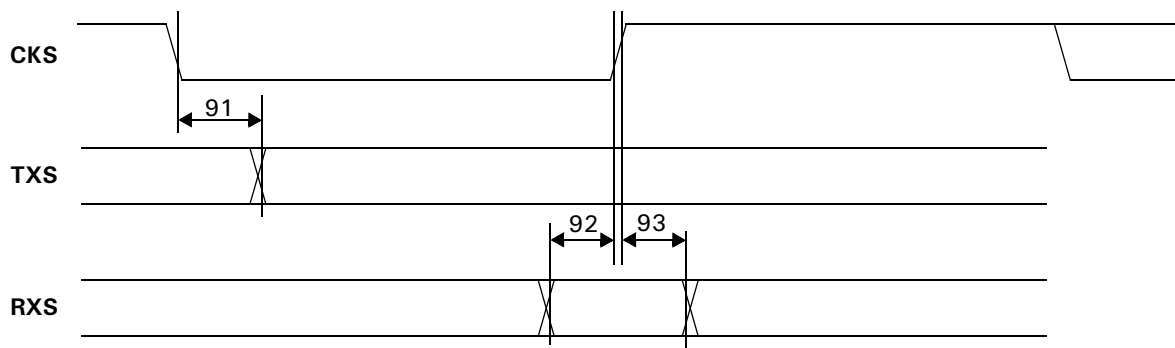


FIGURE 40. CSI/O TIMING

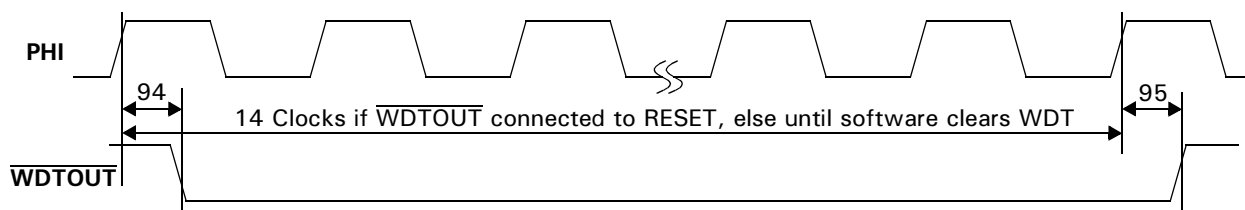


FIGURE 41. WATCH-DOG TIMER TIMING

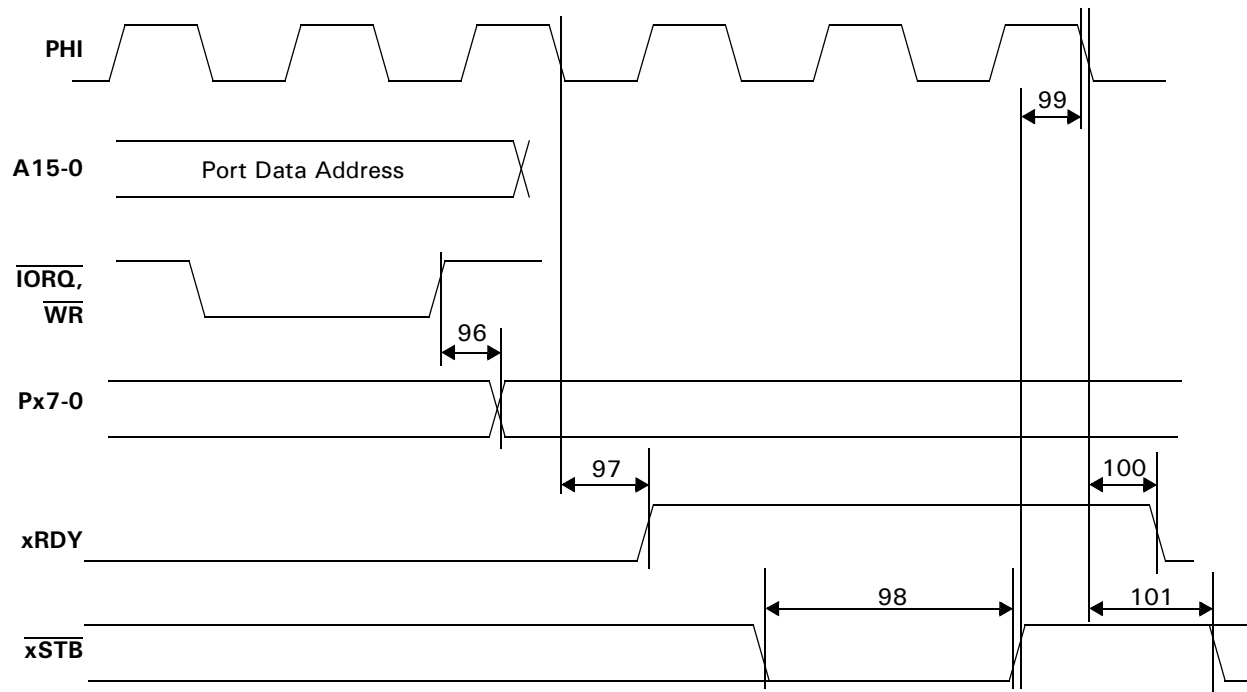


FIGURE 42. PIO MODE 0 (OUTPUT) TIMING

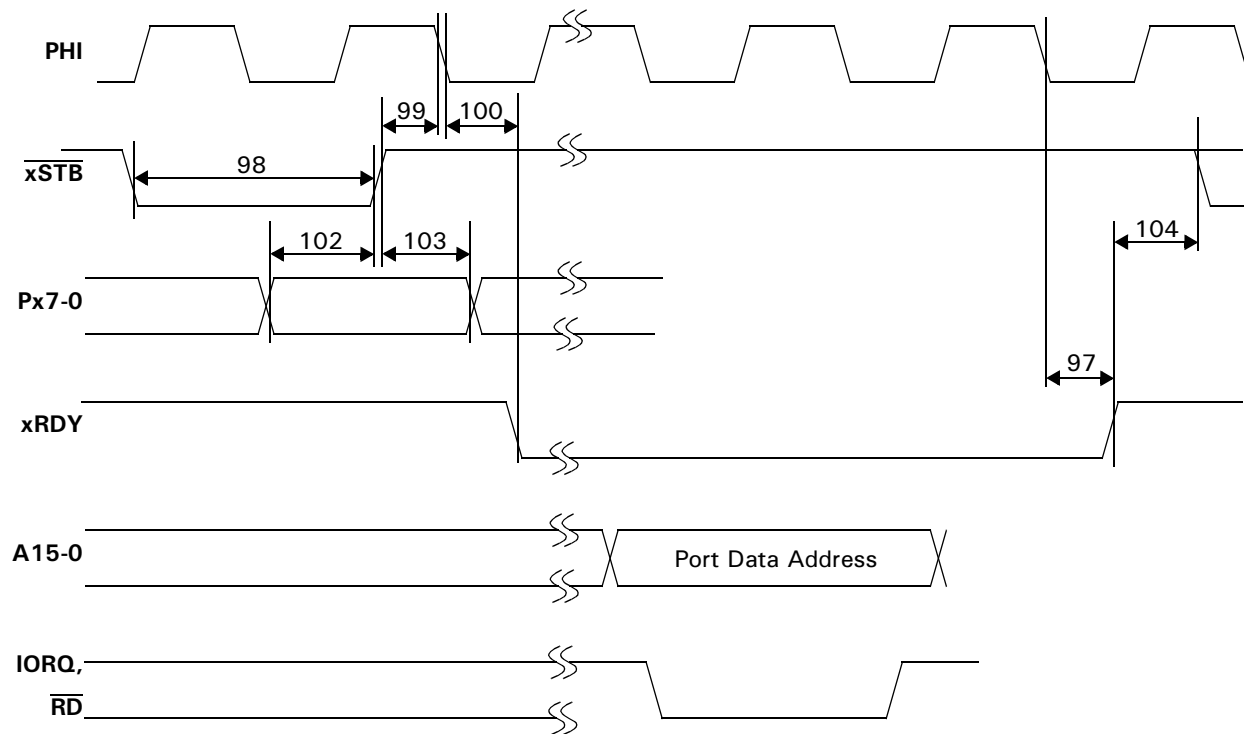
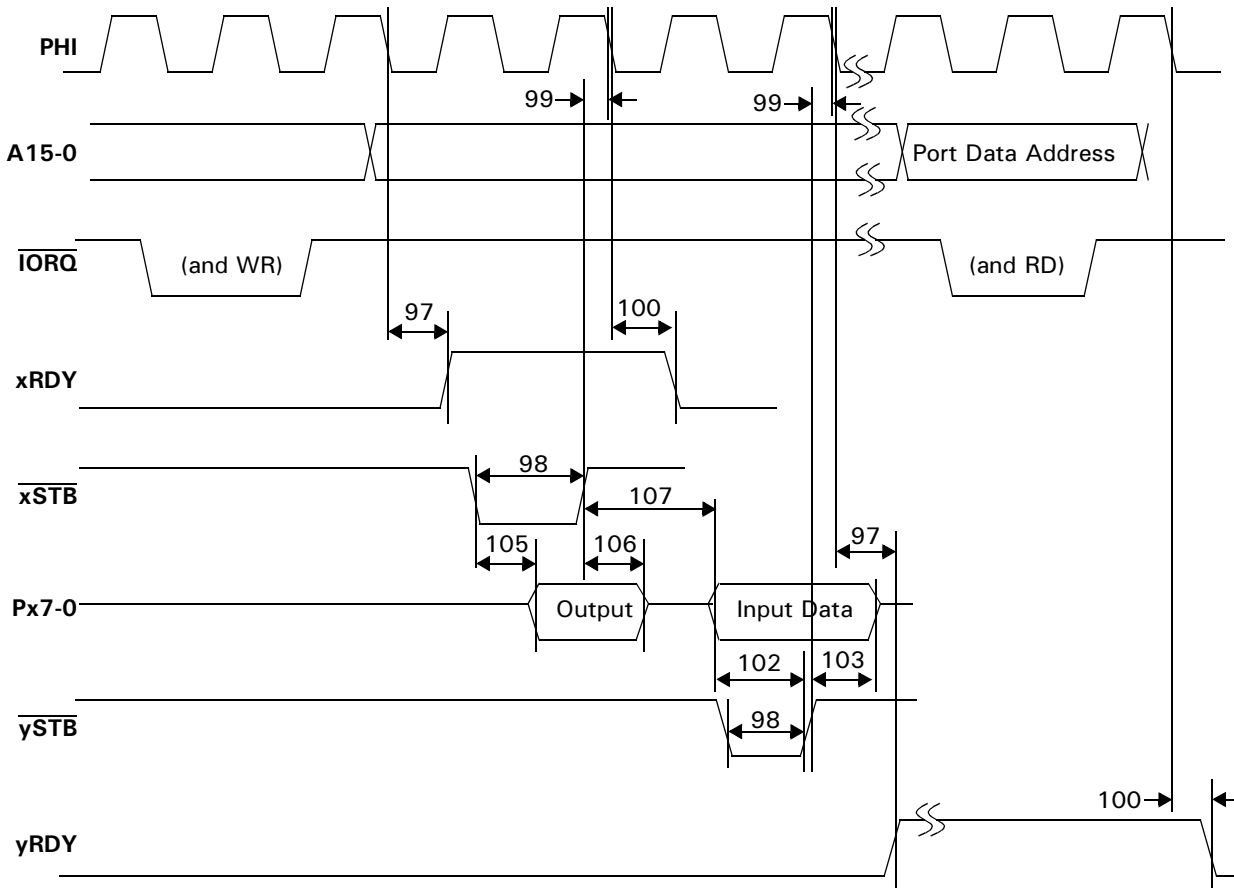


FIGURE 43. PIO MODE 1 (INPUT) TIMING



Note: "x" stands for port A or C, "y" for port B or D correspondingly.

FIGURE 44. PIO MODE 2 (BIDIRECTIONAL) TIMING

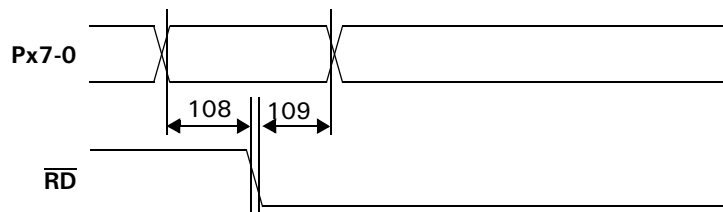


FIGURE 45. PIO MODE 3 (BIT CONTROL) TIMING

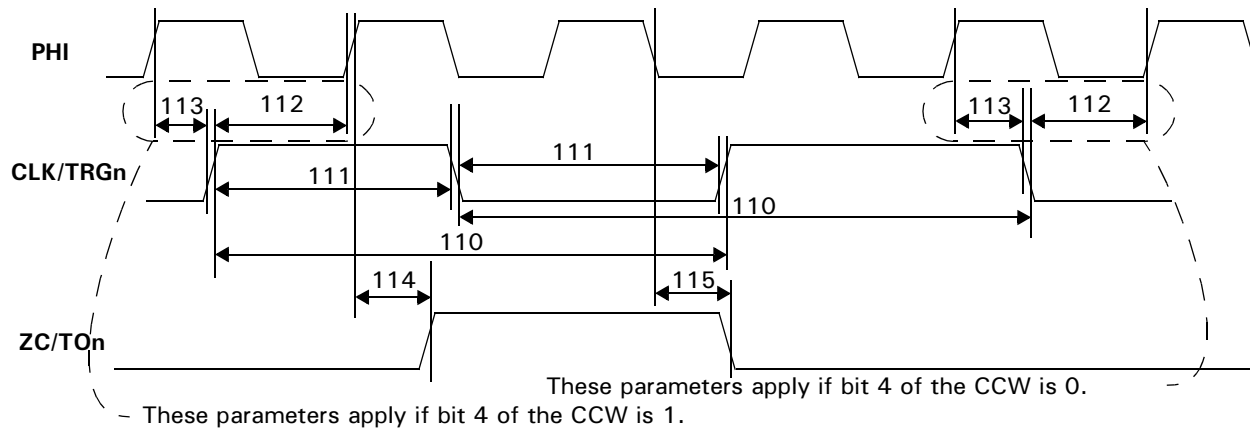


FIGURE 46. CTC TIMING

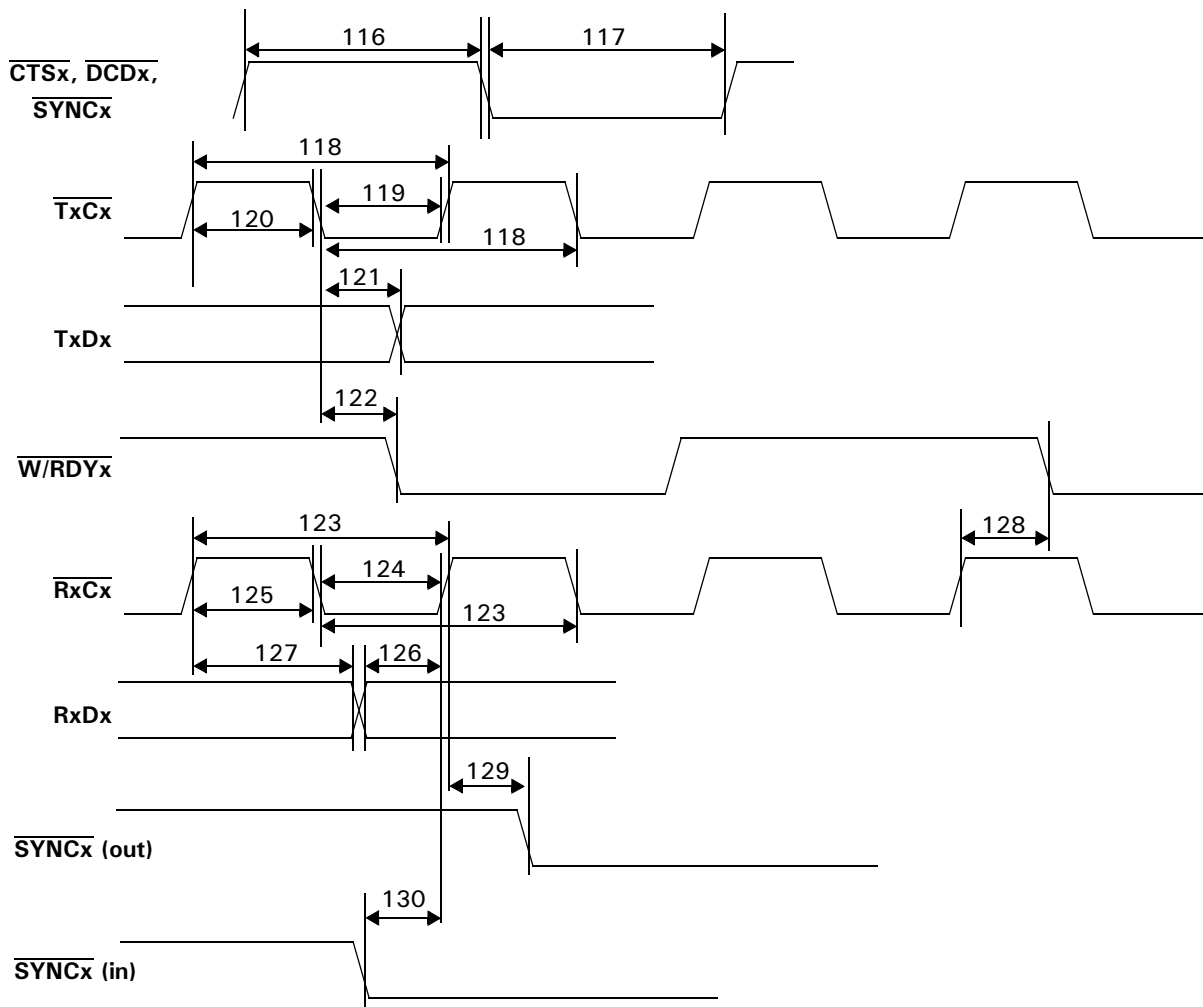


FIGURE 47. SIO TIMING

CHARACTERISTIC CURVES

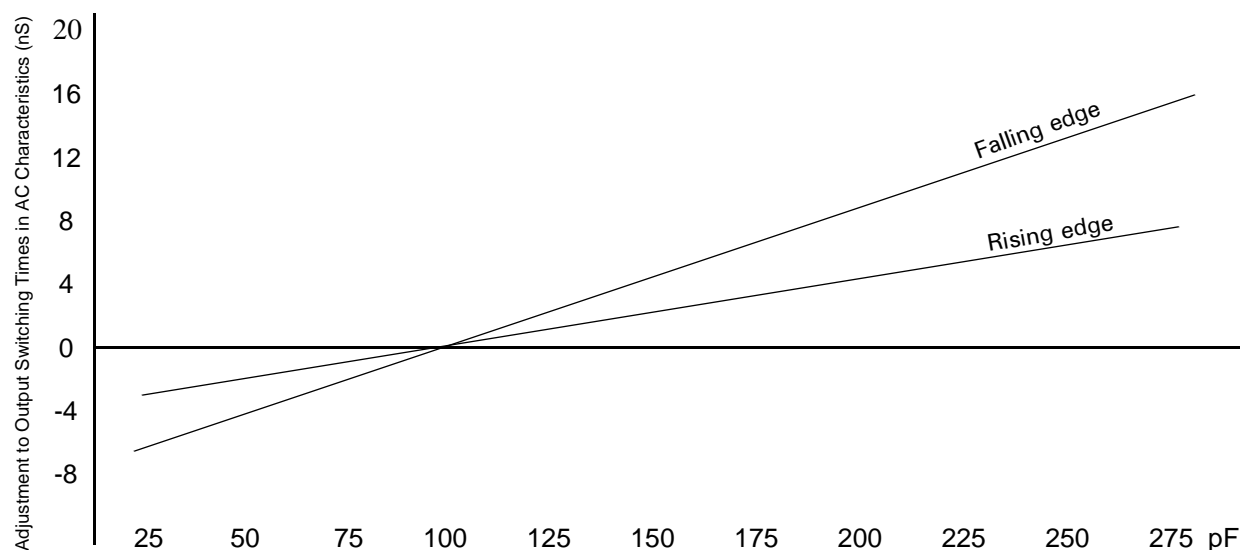


FIGURE 48. CAPACITIVE LOAD C_L VS. SWITCHING TIME

SYSTEM DESIGN CONSIDERATIONS

ERRATA

The following errata apply to revision AB of the Z80S188, which is identified by a 0 in the Device Revision register, I/O address 003DH.

Schmitt Triggers

The $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, ZCL and ZDA pins are specified to include Schmitt triggers, but these were not included on Rev AB. The 50 mS max rise and fall time specified for $\overline{\text{RESET}}$ thus becomes much shorter, 50 nS or less. Since $\overline{\text{RESET}}$ is an open-drain output, perhaps the best work around is to simply connect a pullup resistor, and not drive $\overline{\text{RESET}}$ externally.

No RTC Alarm

The Alarm function of the Real Time Clock is not operative.

PWRSWTCH Negative Logic

The PWRSWTCH pin is specified as positive logic, but in Rev AB is implemented as negative logic (1 is Low, 0 is High).

Output Control Register Reset

The OCR is specified to reset to 0xx0 0000, but instead resets to 0xx1 1000.

No Reset From OPMOD1

A rising edge on OPMOD1 does not trigger a Power On Reset sequence.



PIOS Port Initialization

The Programmable I/O Sequencer enables Port C outputs when it is enabled. Software needs to define the state of Port C outputs before enabling the PIOS, by writing to the Port C Data Register.

PRT Race Condition

If software writes to the PRT Reload registers while the PRT is operating, and a write coincides with a window near the end of the PRT's countdown cycle, the down counter may end up reloaded with the wrong value.

50/60 Hz RTC Clock Taken From Wrong Pin

The RTC specs read that if bit 4 of the RTC Control/Status Register is 1, the RTC takes a 50- or 60-Hz clock from the PC0 pin. Rev AB makes the frequency adjustment when this bit is 1, but takes the clock from LFEXTAL, as when the bit is 0.

Register Write Enable State Indeterminate After Reset

The Register Write Enable state that is readable as bit 0 of the Watch-Dog Timer Master register is specified to reset to 1. However, its value is indeterminate after power up and unchanged by other Resets. Reset-initialization software writes a 0BH to the Watch-Dog Timer Command Register to set this bit, before writing any of the System Configuration, Power Control, Port Data Direction, or Real Time Clock registers.

CSI/O Clock Pin Direction Reversed

The tri-state control signal for the CKS pin is inverted. Thus when the CSI/O module thinks the pin should be an input, the pad is driving out, and vice-versa. This renders the CSI/O unusable in any mode.

APPLICATION NOTES/DEVELOPMENT TOOLS

ZiLOG DEBUG INTERFACE

The Z80S188 includes this serial interface to allow ZiLOG and third-party development systems and emulators to control and monitor the processor and other on-chip resources, during application development and debugging. This two-wire interface is intended to be a standard feature of ZiLOG processors developed in the future, eliminating the need for expensive and cumbersome pods and clip-on emulation equipment.

In order to use the ZiLOG Developer Studio (ZDS) or equivalent equipment with an application or target board, include a standard right angle, 0.1 in spaced, 0.025 in square post, 6-pin header on the board (Berg P/N 75867-131 or equivalent). Connect the ZCL and ZDA pins of the Z80S188 to pins 4 and 6 of this header as illustrated in Figure 49, which is a top view of the board.

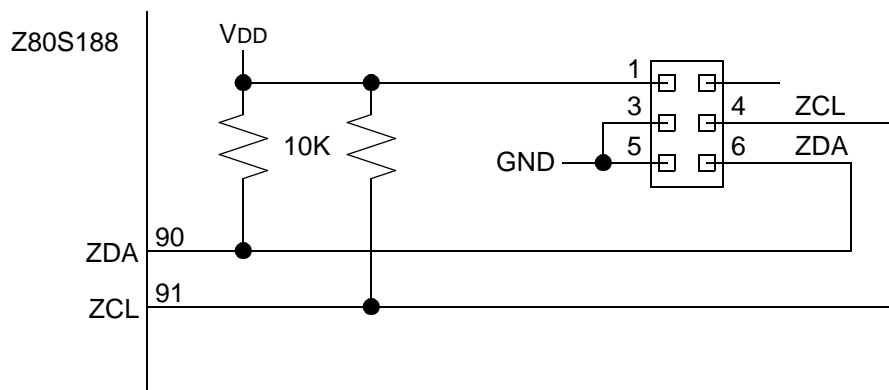


FIGURE 49. ZDI CONNECTOR ON TARGET BOARD

ORDERING INFORMATION

Z80S188 (33 MHz)

Standard Temperature

160 Pin QFP

Z80S18833ASCRxxxx

Extended Temperature

160 Pin QFP

Z80S18833AECDxxxx

Z80L188 (20 MHz)

Standard Temperature

160 Pin QFP

Z80L18820ASCRxxxx

Extended Temperature

160 Pin QFP

Z80L18820AECDxxxx

For fast results, contact your local ZiLOG sale offices for assistance in ordering the part(s) desired.

**PART NUMBER DESCRIPTION**

ZiLOG part numbers consist of a number of components.

EXAMPLE:

Part number Z80S188 33 A S C, a Z80S188, 33 MHz, Quad Flat Pack, 0° to 70° C, Plastic Standard Flow, is made up of the codes described in the following table.

Z	ZiLOG Prefix
Z80S188	Product Number
33	Speed
A	Package
S	Temperature
C	Environmental Flow

**ZiLOG Z80S188 CODE SUBMISSION FORM**

To submit a ROM Code:

1. Complete ROM code submission form.
2. E-mail this form and the hex file (in Intel Hex format) as separate attachments to: codes@ZiLOG.com

Company Name: _____ Disty/Subcon: _____ Date: _____

ZiLOG P/N: _____ Package Code Legend: _____

Checksum: _____ - 00 _____ REV: (Input by
ZiLOG

Company P/N: _____ File P Input: _____ (Input by ZiLOG

Expected Annual Volume in Units: _____

Application: _____

Special Instructions: (Optional) _____

ZiLOG Sales Office (or your city and country): _____

Send ROM verification to: _____

Phone: _____ E-mail address: _____ Fax: _____

To submit a topmark:

1. Check box for default or custom for the preferred package.
2. On default, 9999 indicates ROM number assigned to part by ZiLOG.
3. If custom topmark is selected, enter characters in the space provided. ZiLOG adds the date code (described below as XXYY BB) as bottom line and align as described on default topmark.
4. If all the lines on a custom topmark are not used, leave the top line blank.
5. For [™] (Trademark) symbol, place lowercase tm.
6. For © (Copyright) symbol, place lowercase c followed by a space.
7. For custom logo, attach a BMP, JPEG or GIF file to this form.
8. To use ZiLOG in a custom topmark, type pZiLOG. The p is a placemark for the ZiLOG name.

The Z logo is used when applicable.

QFP 160 Pin	Custom
Default	11 char max



DISCLAIMER

©1999 by ZiLOG, Inc. All rights reserved. Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZiLOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZiLOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. Except with the express written approval of ZiLOG, use of information, devices, or technology as critical components of life support systems is not authorized. No licenses are conveyed, implicitly or otherwise, by this document under any intellectual property rights.

PRECHARACTERIZATION PRODUCT

The product represented by this document is newly introduced and ZiLOG has not completed the full characterization of the product. The document states what ZiLOG knows about this product at this time, but additional features or non-conformance with some aspects of the document may be found, either by ZiLOG or its customers in the course of further application and characterization work. In addition, ZiLOG cautions that delivery may be uncertain at times, due to start-up yield issues.

ZiLOG, Inc.
910 East Hamilton Avenue, Suite 110
Campbell, CA 95008
Telephone (408) 558-8500
FAX 408 558-8300
Internet: [HTTP://WWW.ZILOG.COM](http://www.zilog.com)

DOCUMENT INFORMATION

CHANGE LOG

Rev	Date	Purpose	By
00	0999	Original issue	ggamble

A

AC characteristics	
Extended temperature range	209
Normal temperature range	203
ADC instruction	184
ADD instruction	184
Addressing modes	30
alarm, RTC	227
AND instruction	185
application notes/development tools	228
Arithmetic instructions.	178
ASCI	
ASCI interrupts	103
Basic clocking.	95
Baud rate generator (BRG).	95
Clock mode.	96
Clocking summary.	98
DMA operation	101
Multiprocessor mode	104
Operation	86
Options.	99
Polled reception.	103
Polled transmission	103
Programming techniques	102
Reception	97
Status	100
Timing diagram.	223
Transmission.	97

B

Basic	
Device registers.	111
Timing diagram.	215
Baud rate generator (BRG).	95
BIT	
Instruction.	185
Manipulation instructions.	179
Block transfer instructions	179
Bus exchange timing diagram.	219
Byte	
Reception	108
Transmission.	107

C

CALL instruction	185
Cancel CSI/O transmission or reception .	108

Capacitance	214
CCF instruction	185
Channels	
Counter/Timer (CTC).	63
DMA.	57
Characteristics	
Curves.	227
Electrical	200
Chip Select Decoding for IOCS pin	39
Circuits	
Fundamental mode crystal	42
Third-overtone crystal	42
Clock, CSI/O selection	106
Clocking	
Circuits	41
Crystal specifications	42
Divide by 2 option 41	
Multiply by 2 option.	41
Reduced oscillator drive option	42
Reset conditions.	43
Sample rate	98
Summary	98
Condition codes	181
Configuration, MMU	32
Control timing diagram.	37
Counter/Timer Channel	
Addresses and registers.	64
Channel Control register.	64
Down Counter	65
Interrupt priority daisy chaining	66
Interrupt Vector register	65
Interrupts	65
Software sequences	66
Time Constant register	65
CP instruction	185
CPD instruction	185
CPDR instruction	185
CPI instruction.	185
CPIR instruction.	185
CPL instruction	186
Crystal	
Circuits	41
Specifications.	42
CSI/O	
Byte reception	108
Byte transmission.	107
Cancelling transmission or reception	108

C (CONTINUED)

CSI/O (Continued)	
Clock selection	106
Interrupts	108
Operation	106
Timing diagram	223
CTC, Timing diagram	226

D

DAA instruction	186
Daisy chain signalling	19
DAR0 registers	60
DC characteristics	
Extended temperature range	202
Normal temperature range	201
DEC instruction	186
Description	
General	1
Part number	230
Pin	4
Destination	
And source registers (DAR, SAR)	57
Mode DMA0	61
DI instruction	186
Diagram, pin	4
Disclaimer	232
DJNZ instruction	186
DMA	
Async operation	101
Basics	57
Channel completion	62
Channels	57
Edge- vs. level-sensitive requests	58
Edge-sense request timing	59
Edge-sensitive request	101
Handling interrupts	63
Interrupts	59
Level-sense request timing	58
Memory-to-memory modes	59
Reception	93
Request timing diagram	221
Requests	58
Setting up a transfer	60
Termination timing diagram	221
Transmission	93
DSTAT register	57

E

EI instruction	186
Electrical characteristics	
Absolute maximum ratings	200
AC characteristics	203
Capacitance	214
Characteristic curves	227
DC characteristics	201
Standard test conditions	200
Test load circuit	201
Errata	227
Exchange instructions	178
External status interrupts	94
EXX instruction	186

F

Features of the Z80S188	1
Flag settings definitions	181
Flags	180

G

General description	1
-------------------------------	---

H

Halt 180	
And I/O Stop mode	44
Instruction	186
Mode	44
Timing diagram	222

I

I register	16
I/O	
Memory address registers (IAR, MAR) 57	
Chip select high 127	
Read timing diagram 218	
Stop mode 44	
Idle mode	45
IL register	16
Illegal instruction traps	13
IM instruction	186
IN A instruction	186
IN instruction	186
IN0 instruction	186
INC instruction	186
IND instruction	187

I (CONTINUED)

INDR instruction	187
INI instruction	187
INIR instruction	187
Input/Output	
Central I/O waits	40
Chip select	39
I/O cycle timing diagram	41
I/O instructions	38
IORQ and RD timing	40
Relocating the 80180 registers	38
Space	37
Input/output	
Instructions	180
Instruction	
Classes	178
Input/Output	38
Notation	183
RET	26
RET	26
RETI Bit 6 (M1TE)	27
RETI Bit 7 (M1E)	26
Set	177
Summary	184
INT0	18
Mode 0 timing diagram	22
Mode 1 timing diagram	23
Mode 2 timing diagram	26
modes	18
Interrupt	
Acknowledge timing diagram	221
ASCI	103
ASCI, PRT, DMA, CSI/O, and ZDI	28
CSI/O	108
CTC	65
DMA	59
External/Status	94
Maskable requests	16
Non-maskable	17, 62
Offsets and priorities	29
PIO	49
Priority	118
PRT	71
Relative priority of PIOs, SIO, and CTC	20
Resources in the Z80S188	16
SIO	85

Interrupt (Continued)

Timing diagram	220
Trap control register 17	
Interrupt-driven	
Reception	92
Transmission	91
IORQ and RD	40
ITC register	17

J

JP instruction	187
JR instruction	187

L

LD instruction	187
LDD instruction	188
LDI instruction	188
LDIR instruction	189
Load instructions	178
Logic, negative	227
Logical instructions	178
Low power modes	43

M

Mark state	97
Maskable interrupt requests	16
Maximum ratings	200
Memcs0-1, size/enable values	35
Memory (ROM and RAM)	
Addressing modes	30
Clocking	41
DRAM refresh	36
External memory	
with external decoding	35
External, using MEMSC0-1	34
Input/output space	37
Memory Management Unit (MMU)	31
On-chip RAM	34
On-chip ROM	34
Structure	29
Wait state generator	36
Memory and I/O address registers	
(MAR, IAR)	57
Memory Management Unit (MMU)	31
Memory read timing diagram	216
Memory write timing diagram	217
Memory-to-memory modes	59

M (CONTINUED)

MLT instruction	189
Mode	
0 output timing diagram	47
1 input timing diagram	48
2 bidirectional timing diagram	48
3 bit control timing	49
Async clock	96
Async multiprocessor	104
Destination DMA0	61
Halt	44
Halt and I/O	44
I/O Stop	44
Idle	45
Low power	43
Memory-to-memory	59
Nine-bit	104
Normal	44
Operating DMA1	62
Sleep	44
Source, DMA0	61
Standby	45
Standby with quick recovery	45
System Stop	44

N

NEG instruction	189
Nine-bit mode	104
Non-maskable interrupt	62
Non-maskable interrupt (NMI)	17
NOP instruction	189
Normal low power mode	44

O

Op code map	
1st op code	193
2nd op code after OCBH	194
2nd op code after ODDH	195
2nd op code after OEDH	196
2nd op code after OFDH	197
4th byte after ODDH, OCBH, and D	198
4th byte after OFDH, OCBH, and D	199
OR instruction	189
Ordering information	229
Oscillator drive	42
OTDM instruction	189
OTDMR instruction	189

OTDR instruction	189
OTIM instruction	189
OTIMR instruction	189
OTIR instruction	190
OUT instruction	190
OUT0 instruction	190
OUTD instruction	190
OUTI instruction	190
Overview	1

P

Part number description	230
Pin description	4
PIO	
Addresses	46
DMA channels	57
Interrupt control word	46
Interrupts	49
Mode 0 (output) timing diagram	224
Mode 1 (input) timing diagram	224
Mode 2 (bidirectional) timing diagram	225
Mode 3 (bit control) timing diagram	225
Mode control word	46
Operation modes	46
Output control words	46
Registers	45
Reset	56
Software sequences	50
PIOS port 1 initialization	228
Polled reception	90
POP instruction	190
Power management	43
Precharacterization	232
Processor control instructions	180
Processor description	
Control registers	12
Flags	13
I/O space	12
Interrupt high address register (I)	13
Memory Management Unit (MMU)	12
Program Counter (PC)	13
Program registers	12
Rcounter register (R)	13
Stack Pointer (SP)	13
Processor flags	180
Program control instructions	179
Programming techniques	102

P (CONTINUED)

PRT

Handling interrupts	71
Operation	71
Reset	72
Starting	70
Stopping	70
Timing diagram	223
PRT race condition	228
PUSH instruction	190

Q

Quick recovery (standby) mode	45
---	----

R

RAM Protect option	15
Receiver, Async 102	
Reception	
Async	97
DMA	93
Interrupt-driven	92
Refresh	37
Control register	36
DRAM	36
Timing diagram (2-clock cycle)	221
Register	111
Addresses	110
Bank Area (BAR)	32
Bank Base (BBR)	32
Bank Base High and Low	32
Channel Control	64
Clock Multiplier (CMR)	41
Common Area (CAR)	32
Common Base (CBR)	32
Common base area (CBAR)	32
Common Base High and Low	32
DSTAT	57
I	16
IL	16
Input/Output Chip Select High and Low	39
Interrupt High address (I)	13
Interrupt Vector	65
Interrupt/trap control	17
ITC	17
Memory and I/O address (MAR, IAR)	57
On-chip memory control	34
PIO	45

Register (Continued)

Processor control	12
R counter (R)	13
Refresh control	36
Relocating the 80180	38
SAR0, DAR0	60
SIO Read	81
SIO Write	73
Source and destination (SAR, DAR)	57
Summary	109
Time Constant	65

Register, ASCII

ASCII 0 Rx data	172
ASCII0 control	167
ASCII0 control B	168
ASCII0 extension control	173
ASCII0 status	170
ASCII0 time constant high	175
ASCII0 time constant low	174
ASCII0 time constatn high	175
ASCII0 Tx data	171
ASCII1 control A	168
ASCII1 control B	169
ASCII1 extension control	174
ASCII1 Rx data	172
ASCII1 status	171
ASCII1 time constant low	175
ASCII1 Tx data	172

Register, basic device

Clock multiplier	112
CPU control	113
Free-running counte	111
I/O control	115
On-chip memory control	115
Operating mode control	114
Refresh control	114

Register, chip select and wait

Chip select and wait	127
I/O chip select low	127
Memory chip select 0 high	125
Memory chip select 0 low	123
Memory chip select 1 high	126
Memory chip select 1 low	125

Register, CSI/O

CSI/O control	176
CSI/O data	177

R (Continued)

Register, CTC

CTC0	144
CTC2	146
CTC3	146

Register, DMA

DMA mode	141
DMA status	140
DMA/wait control	142
DMA0 byte count high	135
DMA0 byte count low	135
DMA0 destination address B	134
DMA0 destination address high	134
DMA0 destination address low	133
DMA0 source address B	133
DMA0 source address high	132
DMA0 source address low	132
DMA1 byte count high	139
DMA1 byte count low	139
DMA1 I/O address B	137
DMA1 I/O address high	137
DMA1 I/O address low	136
DMA1 memory address B	136
DMA1 memory address high	136
DMA1 memory address low	135

Register, interrupt

Interrupt vector low	116
Trap control	117

Register, MMU

Bank area	121
Bank base	119
Bank base high, extended mode	123
Bank base low, extended mode	122
Common area, extended mode	122
Common base	118
Common base high	121
Common base low	120
Common/bank area	120

Register, PIO

Port A control	128
Port A data	128
Port B control	130
Port B data	129
Port C control	131
Port C data	131
Port D control	131
Port D data	131

Register, PRT

PRT0 reload high	148
PRT0 reload low	147
PRT0 timer data low	147
PRT1 reload high	151
PRT1 reload low	151
PRT1 timer data high	150
PTRT0 timer data high	147
Timer control	148
Timer prescale	150

Register, SIO

RR0	160
RR1	163
SIO A control	152
SIO A data	152
SIO B control	165
SIO B data	165
SIO B rr2	166
SIO B wr2	165
WR0	154
WR1	155
WR3	156
WR4	157
WR5	158
WR6	159
WR7	160

Register, WDT

Master	143
Watch-Dog Timer command	143

Requests

DMA	58
Edge vs. level sensitive	58

RES instruction

RES instruction	190
-----------------------	-----

Reset

Conditions	43
Watch-Dog timer	43
None from OPMOD1	227
Output control register	227

RET instruction

INT0 mode 2 timing	26
Instruction summary	190

RETI instruction

Bit 6 (M1TE)	27
Bit 7 (M1E)	26
Timing with M1E=0	27

RETN instruction

RETN instruction	190
------------------------	-----

RL instruction

RL instruction	191
----------------------	-----

R (Continued)

RLA instruction	191
RLC instruction	191
RLCA instruction	191
RLD instruction	191
ROM Protect option	15
ROMCS device	34
ROMless option	15
Rotate and shift instructions	179
RR instruction	191
RRA instruction	191
RRC instruction	191
RRCA instruction	191
RRD instruction	191
RST instruction	191

S

SAR0 registers	60
SBC instruction	192
SCF instruction	192
Security features	15
SET instruction	192
Setting up a DMA transfer	60
Signalling, Daisy chain	19
SIO	
Addresses	72
Interrupts	85
Read registers	81
Registers per channel	73
Timing diagram	226
Write registers	73
SLA instruction	192
Sleep	
Mode	44
Timing diagram	222
Source	
And destination registers (SAR, DAR)	57
Mode, DMA0	61
SRA instruction	192
SRL instruction	192
Stack pointer (SP)	13
Standard test conditions	200
Standby mode	45
Status, Async	100
Stopping a PRT	70
SUB instruction	192
Sync operation	87
System Stop mode	44

T

Test load circuit	201
Timing diagrams	
ASCI	223
Basic	215
Bus exchange	219
Control timing diagram	37
CSI/O	223
CTC	226
DMA request	221
CSI/O	223
CTC	226
DMA Request	221
DMA termination	221
DMA Transmission	93
Edge-sense request	59
Halt	222
I/O cycle	41
I/O read timing	218
I/O write timing	219
INT0 mode 0	22
INT0 mode 1	23
INT0 mode 2	26
Interrupt	220
Interrupt acknowledge timing	221
Level-sense request	58
Memory read timing	216
Memory write timing	217
Mode 0 output	47
Mode 1 input	48
Mode 2 bidirectional	48
Mode 3 bit control	49
PIO mode 0 (output)	224
PIO mode 1 (input)	224
PIO mode 2 (bidirectional)	225
PIO mode 3 (bit control)	225
PRT	223
Refresh cycle	37
Refresh timing (2-clock cycle)	221
RETI instruction with M1E=0	27
SIO	226
Sleep	222
Watch-Dog timer	223
Transmission	
Async	97
DMA	93
Interrupt-driven	91

T (CONTINUED)

Transmitter, Async	102
Traps, illegal instruction.	13
Triggers, Schmitt.	227
TST instruction	192
TSTIO instruction	192

W

Wait state generator.	36
Waits, Central I/O	40
Watch-Dog	
Time.	63
Timing diagram.	223

X

XOR instruction	192
---------------------------	-----

Z

ZDI	
Connector on target board	229
Enable	16
ZiLOG	
Debug Interface (ZDI)	228
Developer Studio (ZDS)	228