Z80185/Z80195

Smart Peripheral Controllers

User Manual

UM001001-1000

ZiLOG Worldwide Headquarters • 910 E. Hamilton Avenue • Campbell, CA 95008
Telephone: 408.558.8500 • Fax: 408.558.8300 • www.ZiLOG.com

Z80185/Z80195
User Manual

This publication is subject to replacement by a later edition. To determine whether a later edition exists, or to request copies of publications, contact

ZiLOG Worldwide Headquarters
910 E. Hamilton Avenue
Campbell, CA 95008
Telephone: 408.558.8500
Fax: 408.558.8300
www.ZiLOG.com

Document Disclaimer

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concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded.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 ZiLOG, use of information, devices, or technology as critical components of life support systems is not authorized. No licenses or other rights are conveyed, implicitly or otherwise, by this document under any intellectual property rights.

Preface

DOCUMENT ASSUMPTIONS AND CONVENTIONS

The following assumptions and conventions have been adopted to provide clarity and ease of use:

- Use of the Words *Set* and *Clear*

The words *set* and *clear* imply that a register bit or a condition has the value of *logical 1* and *logical 0* respectively. When the terms set and clear are followed by a number, often in parentheses, the word *logical* may not be included, but it is implied.

- Notation for Bits and Similar Registers

A field of bits within a register are designated as: Register (*n*..*n*). For example: `PWM_CR` (31..20). A field of bits within a bus are designated as: $\text{Bus}_{n..n}$. For example: $\text{PCntl}_{7..4}$. A range of similar (whole) registers is designated as:
Register*n*..Register*n*. For example: OPBCS5..OPBCS0.

- Use of the Terms *LSB* and *MSB*

In this document, the terms *LSB* and *MSB* mean *least significant bit* and *most significant bit* respectively.

- Courier Font

Commands, code lines and fragments, register and other mnemonics, values, equations, and various executable items are distinguished from general text by the use of the Courier font. This convention is not used within tables. Where the use of the font is not possible, as in the Index, the name of the entity is capitalized. For example: The STP bit in the `CNTR` register must be `1`.

- Hexadecimal Values Designated by H

Hexadecimal values are designated by an upper-case letter H as well as the use of Courier font. For example: `STAT` is set to `F8H.`

- Use of All Upper-Case Letters

The use of all upper-case letters designates the names of states and commands. For example: The receiver can force the SCL line to Low for force the transmitter into a WAIT state. The bus is considered BUSY after the Start condition. A `START` command triggers the processing of the initialization sequence.

- Use of Initial Upper-Case Letters

Initial upper-case letters designate settings, modes, and conditions in general text. For example: The Slave receiver leaves the data line High. In Transmit mode, the byte is sent most significant bit first. The Master can generate a Stop condition to abort the transfer.

- Register Access Abbreviations

Register access is designation by the following abbreviations:

Designation	Description
R	Read Only
R/W	Read/Write
W	Write Only
–	Unspecified or indeterminate

- Use of Fewer Bits Than in a Register Field

When a register field is comprised of multiple bits, a value for the field may be stated as a single number. For example: The reset value for an 8-bit field may be described as `0` when the register contains 8 bits that each have the value `0`.

TRADEMARKS

Several trademarks appear in this product specification.

The following items are trademarks of ZiLOG, Inc.:

- Z80
- Z180
- ESCC
- SCC
- Z80185
- Z80195

Table of Contents

Enhanced Serial Communications Controller 233

List of Figures

Asynchronous Serial Communications Interface 131

Clocked Serial I/O Port 159

Programmable Reload Timers 169

Counter/Timer Channels 177

Bidirectional Centronics P1284 Controller 195

Enhanced Serial Communications Controller 233

List of Tables

The Processor 29

Direct Memory Access 103

Asynchronous Serial Communications Interface 131

Clocked Serial I/O Port 159

Bidirectional Centronics P1284 Controller 195

Enhanced Serial Communications Controller 233

Overview

INTRODUCTION

This manual describes the operation and programming of the Z80185/Z80195 Smart Peripheral Controllers. The Z80185/Z80195 Product Specification describes the quantitative performance characteristics of the device. This chapter provides an overview of the controllers and describes its pinout and pin functions.

Features

- Enhanced Z8S180 MPU
  - Code Compatible with ZiLOG's Z80®/Z180™ CPUs
  - Extended Instructions
  - Two Enhanced DMA Channels
  - On-Chip Interrupt Controllers
  - Three On-Chip WAIT-State Generators
  - On-Chip Clock Oscillator/Generator
  - Expanded MMU Addressing (to 1 MB)
  - Clocked Serial I/O Port
  - Two 16-Bit Counter/Timers
  - Two Enhanced UARTs (to 512 Kbps)
- Four Z80 CTC Channels
- One ESCC™ Channel
- Two 8-Bit Parallel Ports
- Bidirectional Centronics™ (IEEE P1284) Controller
- Clock Speeds: 20, 33 MHz

- Operating Range: 5 Volts (3.3V @ 20 MHz)
- Operating Temperature Range: 0°C to +70°C
- 100-Pin QFP Package Style

GENERAL DESCRIPTION

The Z80185/Z80195 is a smart Peripheral controller device specifically designed to manage input and output (I/O) requirements for both serial and parallel connectivity. The Z80185/Z80195 is the version of the device that does not contain ROM.

These devices are well-suited for use in general-purpose data communication equipment, external modems using a parallel interface, protocol translators, and low-cost WAN adapters. It is ideal for managing laser printer I/O, as well as serving as the main processor in low-cost printer applications.

The Z80185/Z80195 features an enhanced Z8S180 microprocessor linked with one enhanced channel of ZiLOG’s industry-standard ESCC serial communications controller, and 25 bits of parallel I/O. Of these 25 I/O lines, 17 can be configured as a Bidirectional Centronics™ (IEEE P1284) port. When configured as a P1284 port, the device can operate in either the Host or Peripheral role in compatible, nibble, byte, or ECP mode.

Note: The Z80185/Z80195 device contains a single-channel Enhanced SCC™. In this user’s manual, it may be called enhanced SCC or ESCC.

For additional information on ZiLOG microprocessors referenced in this user’s manual, please refer to the following technical documentation:

- Z80180, Z8S180, Z80181 Microprocessors
  - *Z80180/Z180 MPU User’s Manual*, DC 8276-04
  - *Z180 Databook*, DB971800100

- SCC, ESCC
  - *Serial Communication Controllers User's Manual*, DC 8293-02
- Z84C15, Z8430/C30
  - *Z80 Microprocessor Family Databook*, DC 8321-00

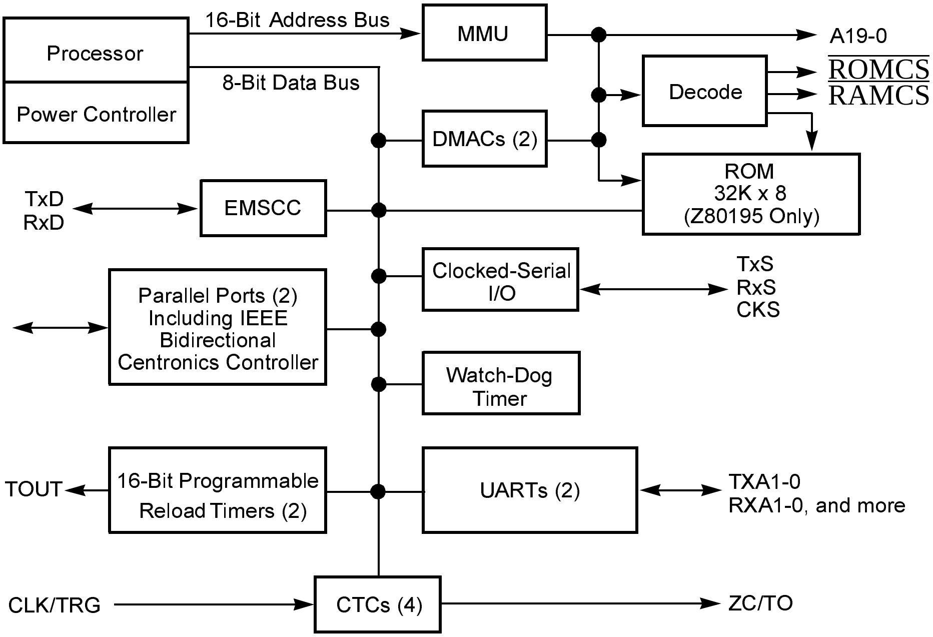


Figure 1. Functional Block Diagram

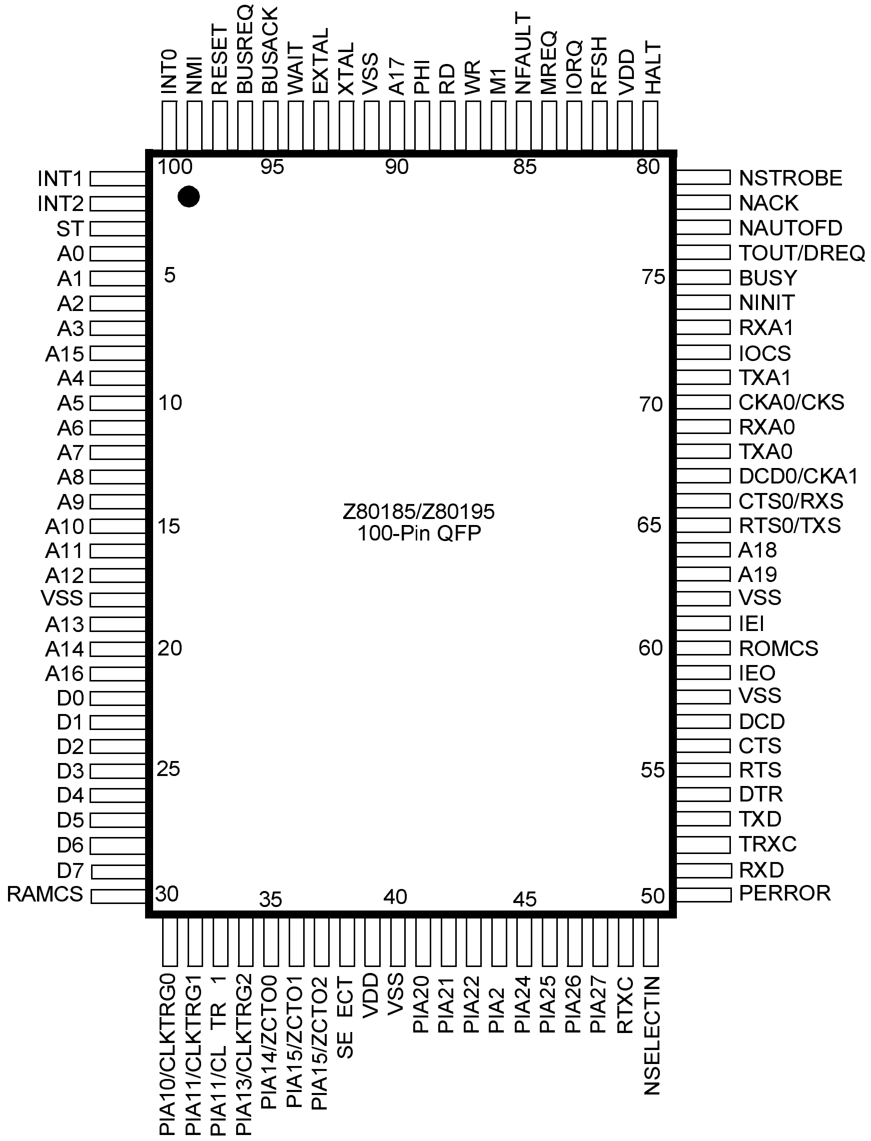

INT0
NMI
RESET
BUSREQ
BUSACK
WAIT
EXTAL
XTAL
VSS
A17
PHI
RD
WR
M1
NFAULT
MREQ
IORQ
RFSH
VDD
HALT
100
95
90
85
80
INT1
INT2
ST
A0
A1
A2
A3
A15
A4
A5
A6
A7
A8
A9
A10
A11
A12
VSS
A13
A14
A16
D0
D1
D2
D3
D4
D5
D6
D7
RAMCS
5
10
15
20
25
30
Z80185/Z80195
100-Pin QFP
NSTROBE
NACK
NAUTOFD
TOUT/DREQ
BUSY
NINIT
RXA1
IOCS
TXA1
CKA0/CKS
RXA0
TXA0
DCD0/CKA1
CTS0/RXS
RTS0/TXS
A18
A19
VSS
IEI
ROMCS
IEO
VSS
DCD
CTS
RTS
DTR
TXD
TRXC
RXD
PERROR
75
70
65
60
55
50
35
40
45
PIA10/CLKTRG0
PIA11/CLKTRG1
PIA11/CL TR 1
PIA13/CLKTRG2
PIA14/ZCTO0
PIA15/ZCTO1
PIA15/ZCTO2
SE ECT
VDD
VSS
PIA20
PIA21
PIA22
PIA2
PIA24
PIA25
PIA26
PIA27
RTXC
NSELECTIN

Figure 2. 100-Pin QFP Pin Assignments

PIN DESCRIPTIONS

CPU Signals

A19-A0. *Address Bus* (Input/Output, Active High, 3-State). A0-A19 are a 20-bit address bus that provides the address for up to 1 MB of memory, and for up to 64 KB of space. The address bus enters a High impedance state during reset and external bus acknowledge cycles. This bus is an input when $\overline{\text{BUSACK}}$ is Low. No address lines are multiplexed with any other signals.

D7-D0. *Data Bus* (Bidirectional, Active High, 3-State). D7-D0 constitute an 8-bit bidirectional data bus, used to transfer information to and from I/O and memory devices. The data bus enters the High impedance state during reset and external bus acknowledge cycles, as well as during SLEEP and HALT states.

$\overline{\text{RD}}$. *Read* (Input/Output, Active Low, 3-State). $\overline{\text{RD}}$ indicates that the CPU is reading data from memory or an I/O device. The addressed I/O or memory device should use this signal to gate data onto the CPU data bus. This pin is an input during bus acknowledge cycles.

$\overline{\text{WR}}$. *Write* (Input/Output, Active Low, 3-State). $\overline{\text{WR}}$ indicates that the CPU data bus holds valid data to be stored at the addressed I/O or memory location. This pin is an input during bus acknowledge cycles.

$\overline{\text{IORQ}}$. *I/O Request* (Input/Output, Active Low, 3-State). $\overline{\text{IORQ}}$ indicates that the address bus contains a valid I/O address for a read or write operation. $\overline{\text{IORQ}}$ is also generated, along with $\overline{\text{M1}}$, during the acknowledgment of the $\overline{\text{INT0}}$ input signal to indicate that an interrupt response vector can be placed onto the data bus. This pin is an input during bus acknowledge cycles.

$\overline{\textbf{M1}}$. *Machine Cycle 1* (Input/Output, Active Low). Together with $\overline{\text{MREQ}}$, $\overline{\text{M1}}$ indicates that the current cycle is the opcode fetch cycle of an instruction execution. Together with $\overline{\text{IORQ}}$, $\overline{\text{M1}}$ indicates that the current cycle is for an interrupt acknowledge. It is also used with the $\overline{\text{HALT}}$ and ST signals to indicate the status of the CPU machine cycle. The processor can be configured so that this signal is compatible with the $\overline{\text{M1}}$ signal of the Z80, or with the $\overline{\text{LIR}}$ signal of the 64180. This pin is 3-stated during bus acknowledge cycles.

$\overline{\textbf{MREQ}}$. *Memory Request* (Input/Output, Active Low, 3-State). $\overline{\text{MREQ}}$ indicates that the address bus holds a valid address for a memory read or memory write operation. It is included in the $\overline{\text{RAMCS}}$ and $\overline{\text{ROMCS}}$ signals, and because of this may not be needed in some applications. This pin is an input during bus acknowledge cycles.

$\overline{\textbf{WAIT}}$. (Input/Open-Drain Output, Active Low.) $\overline{\text{WAIT}}$ indicates to the MPU that the addressed memory or I/O devices are not ready for a data transfer. This input is used to induce additional clock cycles into the current machine cycle. External devices should also drive this pin in an open-drain fashion. This results in a wired OR of the Wait indications produced by external devices and those produced by the two separate Wait State generators in the Z80185/Z80195. If the wire-ORed input is sampled Low, then additional wait states are inserted until the $\overline{\text{WAIT}}$ input is sampled High, at which time the cycle is completed.

$\overline{\textbf{HALT}}$. *Halt/Sleep Status* (Output, Active Low). This output is asserted after the CPU has executed either the HALT or SLP instruction, and is waiting for non-maskable or maskable interrupt before operation can resume. It is also used with the $\overline{\text{M1}}$ and $\overline{\text{ST}}$ signals to indicate the status of the CPU machine cycle. On exit from the Halt/Sleep state, the first instruction fetch is delayed for 16 clock cycles after the $\overline{\text{HALT}}$ pin goes High.

$\overline{\textbf{BUSACK}}$. *Bus Acknowledge* (Output, Active Low). $\overline{\text{BUSACK}}$ indicates to the requesting device that the MPU address and data bus, as well as some control signals, have entered their High impedance state.

$\overline{\text{BUSREQ}}$. *Bus Request* (Input, Active Low). This input is used by external devices (such as DMA controllers) to request access to the system bus. This request has a higher priority than $\overline{\text{NMI}}$ and is always recognized at the end of the current machine cycle. This signal stops the CPU from executing further instructions and places the address and data buses, and other control signals, into the High impedance state.

$\overline{\text{NMI}}$. *Non-Maskable Interrupt* (Input, Negative Edge Triggered). $\overline{\text{NMI}}$ has a higher priority than the $\overline{\text{INT}}$ pins and is always recognized at the end of an instruction, regardless of the state of the interrupt enable flip-flops. This signal forces CPU execution to location `0066H`.

$\overline{\text{INT0}}$. *Maskable Interrupt Request 0* (Input/Open-Drain Output, Active Low). This signal is generated by internal and external I/O devices. External devices should also drive this signal in an open-drain fashion. The CPU honors 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. The CPU acknowledges this interrupt request with an interrupt acknowledge cycle. During this cycle, both the $\overline{\text{M1}}$ and $\overline{\text{IORQ}}$ signals become active.

$\overline{\text{INT1}}$, $\overline{\text{INT2}}$. *Maskable Interrupt Requests 1 and 2* Inputs, Active Low). These signals are generated by external I/O devices. The CPU honors these requests at the end of the current instruction cycle as long as the $\overline{\text{NMI}}$, $\overline{\text{BUSREQ}}$, and $\overline{\text{INT0}}$ signals are inactive. The CPU responds to these interrupt requests with an idle time period of the same duration as an interrupt acknowledge cycle. Neither the $\overline{\text{M1}}$ nor the $\overline{\text{IORQ}}$ signals become active during this period. These pins may be programmed to provide active Low level, rising or falling edge interrupts. The level of the external $\overline{\text{INT1}}$ and $\overline{\text{INT2}}$ pins may be read in the Interrupt Edge Register.

$\overline{\text{RFSH}}$. *Refresh* (output, Active Low, 3-state). $\overline{\text{RFSH}}$ and $\overline{\text{MREQ}}$ Active indicate that the current CPU machine cycle and the contents of the address bus should be used for refresh of dynamic memories. The low order eight bits of the address bus (A7-A0) contain the refresh address.

UART and CSIO Signals

CKA0/CKS. *Asynchronous Clock 0 or Serial Clock* (Input/Output). An optional clock input or output for ASCI channel 0 or the Clocked Serial I/O Port.

$\overline{\text{DCD0}}$/CKA1. *Data Carrier Detect 0 or Asynchronous Clock 1* (Input/Output). A Low-Active modem status input for ASCI channel 0, or a clock input or output for ASCI channel 1.

$\overline{\text{RTS0}}$/TxS. *Request to Send 0 or Clocked Serial Transmit Data* (output). A programmable modem control output for ASCI channel 0, or the serial output from the CSIO channel.

$\overline{\text{CTS0}}$/RxS. *Clear to Send 0 or Clocked Serial Receive Data* (Input). A Low-Active modem control input for ASCI channel 0, or the serial data input to the CSIO channel.

TXA0. *Transmit Data 0* (Output). This output transmits data from ASCI channel 0.

RXA0. *Receive Data 0* (Input). This input receives data for ASCI channel 0.

RXA1. *Receive Data 1* (Input). This input receives data for ASCI channel 1.

TXA1. *Transmit Data 1* (Output). This output transmits data from ASCI Channel 1.

Multiplexed Signal

T_{OUT}/$\overline{\text{DREQ}}$. *Timer Out or External DMA Request* (Input or Output). This pin can be programmed to be either T_{OUT}, the High-Active pulse output from PRT channel 1, or a Low- Active DMA Request input from an external Peripheral.

ESCC Signals

TXD. *Transmit Data* (Output). This output transmits serial data at standard TTL levels.

RXD. *Receive Data* (Input). This input receives serial data at standard TTL levels.

$\overline{\text{TRXC}}$. *Transmit/Receive Clock* (Input or Output). This pin functions under program control. $\overline{\text{TRXC}}$ may supply the receive clock or the transmit clock in the input mode or supply the output of the digital phase-locked loop, the crystal oscillator, the baud rate generator, or the transmit clock in the output mode.

$\overline{\text{RTXC}}$. *Receive/Transmit Clock* (Input). This pin functions under program control. $\overline{\text{RTXC}}$ may supply the receive clock, the transmit clock, the clock for the baud rate generator, or the clock for the digital phase-locked loop. The receive clock may be 1, 16, 32, or 64 times the data rate in asynchronous mode.

$\overline{\text{CTS}}$. *Clear To Send* (Input, Active Low). If this pin is programmed as an auto enable, a Low on it enables the ESCC transmitter. If not programmed as an auto enable, it can be used as a general-purpose input. This pin is Schmitt-trigger buffered to accommodate slow rise-times. The ESCC detects transitions on this input and can interrupt the processor on either logic level transition.

$\overline{\text{DCD}}$. *Data Carrier Detect* (Input, Active Low). This pin functions as an ESCC receiver enable when programmed as an auto enable; otherwise it can be used as a general-purpose input pin. The pin is Schmitt-trigger buffered to accommodate slow rise-times. The ESCC detects transitions on this pin and can interrupt the processor on either logic level transition.

$\overline{\text{RTS}}$. *Request to Send* (Output, Active Low). When the Request to Send (RTS) bit in Write Register 5 is set, the RTS signal goes Low. When the RTS bit is reset in the Asynchronous mode and auto enables is on, the signal goes High after the transmitter is empty. In Synchronous mode, or in Asynchronous mode with auto enables off, the RTS pin strictly follows the state of the RTS bit. Thus the pin can be used as a general-purpose output. In a special AppleTalk mode on the Z80185/Z80195, the pin is under hardware control.

$\overline{\textbf{DTR}}$. *Data Terminal Ready* (Output, Active Low). The $\overline{\text{DTR}}/\overline{\text{REQ}}$ functionality found in other SCC family members has been reconfigured in the Z80185/Z80195 ESCC. The $\overline{\text{DTR}}$ output is routed to this pin, while the $\overline{\text{REQ}}$ signal is routed to the DMA request multiplexing logic as described in a later section on the ESCC. This pin follows the state of the DTR bit in WR5 of the ESCC.

Note: The $\overline{\text{W}}$/REQ pin present on other SCC family members has its two possible functions reconfigured in the Z80185/Z80195. The Wait output of the ESCC drives the $\overline{\text{WAIT}}$ signal in a wire-ORed fashion with other internal and external Peripherals. The $\overline{\text{REQ}}$ component is routed to the DMA request multiplexing logic as described in a later section on the ESCC.

Parallel Ports

PIA16-14. *Port 1, Bits 6-4 or CTC ZC/TO2-0* (Input/Output). These lines can be configured as inputs or outputs, or as the 0-count/timeout outputs of three of the four CTC channels, on a bit-by-bit basis.

PIA13-10. *Port 1, Bits 3-0 or CTC CLK/TRG3-0* (Input/Output). These lines can be configured as inputs or outputs, or as the clock/trigger inputs of the four CTC channels, on a bit-by-bit basis.

PIA27-20. *Port 2, Data, or Bidirectional* (Input/Output). These lines can be configured as inputs or outputs on a bit-by-bit basis when not used for Bidirectional Centronics™ operation. However, when used for Bidirectional Centronics operation, software and hardware controls the direction of all eight as a unit.

Bidirectional Centronics Pins

nStrobe, nAutoFd, nSelectIn, nInit (Input/Output). These are inputs when using P27-20 for a Peripheral or outputs when using P27-20 for a Host. In certain P1284 modes, these pins assume other names as described in the section on the P1284 controller. When not using P27-20 for a parallel port, these pins can be used as general-purpose inputs or outputs.

Busy, nAck, PError, nFault, Select (Input/Output). These are outputs when using P27-20 for a Peripheral or inputs when using P27-20 for a Host. In certain P1284 modes, these pins have other names as described in the section on the P1284 controller. When not using P27-20 for a parallel port, these pins can be used as general-purpose outputs or inputs. These pins function in the opposite direction from the preceding group.

System Control Signals

ST. *Status* (Output, Active High). This signal is used with the $\overline{\text{M1}}$ and $\overline{\text{HALT}}$ output to indicate the nature of each CPU machine cycle.

$\overline{\textbf{RESET}}$. *Reset Signal* (Input, Active Low). $\overline{\text{RESET}}$ signal is used for initializing the Z80185/Z80195 and other devices in the system. It must be kept Low for at least three system clock cycles.

IEI. *Interrupt Enable Signal* (Input, Active High). IEI is used with IEO to form a priority daisy-chain when there are external interrupt-driven Z80-compatible Peripherals.

IEO. *Interrupt Enable Output Signal* (Output, Active High). In an interrupt daisy-chain, IEO enables interrupts from external Peripherals that have lower priority than on-chip Peripherals. IEO is Active when IEI is 1, the CPU is not servicing an interrupt from an on-chip Peripheral, and no interrupt is being requested by the ESCC channel, the P1284 controller, or any of the CTC channels.

IOCS. IOCS decodes IORQ, M1, and the address lines to ensure it is activated for an I/O space access to any register in any block of eight registers that does not contain any on-chip registers. Also included in the decode is any programmed relocation of the 180 register set in the ICR, and the Decode High I/O bit in the System Configuration Register. If the 180 registers are not relocated, and Decode High I/O is `0`, IOCS is Active from address `XX40` though `XXD7`, `XXF8` through `XXFF`, and `NN00` through `NN3F`, where `NN` are non-zero. If the 180 registers are not relocated and Decode High I/O is `1`, IOCS is Active from `0040` through `00D7`, and `00F8` through `FFFF`. IOCS is Active when an external master is in control of the bus, as well as when the Z80185/Z80195 processor has control.

RAMCS. *RAM Chip Select* (Output, Active Low). This signal is driven Low for memory accesses at addresses that fall between the values programmed into the RAMLBR and RAMUBR registers. It is Active when an external master has control of the bus, as well as when the Z80185/Z80195 processor is in control.

ROMCS. *ROM Chip Select* (Output, Active Low). This output is driven Low for memory accesses between the top of on-chip ROM (if on-chip ROM is enabled) and the value programmed into the ROMBR register. It is Active when an external master has control of the bus, as well as when the Z80185/Z80195 processor is in control.

XTAL. *Crystal* (Input, Active High). This pin functions as the Crystal oscillator connection and should be left open if an external clock is used instead of a crystal. The oscillator input is not a TTL level (see the “DC Characteristics” section).

EXTAL. *External Clock/Crystal* (Input, Active High). This pin functions as a Crystal oscillator connection. An external clock can be input to the Z80185/Z80195 on this pin when a crystal is not used. This input is Schmitt-triggered.

PHI. *System Clock* (Output, Active High). This output is the processor’s reference clock, and is provided for the use of external logic. The frequency of this output may be equal to, or one-half that of the crystal or input clock frequency, depending on an internal register bit. After Reset, the output frequency is one-half the crystal or input clock frequency.

MPU FUNCTIONAL DESCRIPTION

The Z80185/Z80195 includes a ZiLOG Z8S180 MPU (Static Z80180 MPU). This feature allows software code compatibility with existing Z180 software code. The following is an overview of the major functional units of the Z80185/Z80195.

The MPU portion of the Z80185/Z80195 is the Z8S180 core with added features and modifications. The single-channel EMSCC of the Z80185/Z80195 is compatible with the Z85233 EMSCC with additional enhancements for LocalTalk and the demultiplexing of the $\overline{\text{DTR}}$/$\overline{\text{REQ}}$ and $\overline{\text{WT}}$/$\overline{\text{REQ}}$ lines.

ARCHITECTURE

The Z80185/Z80195 combines a high-performance CPU core with a variety of system and I/O resources useful in a broad range of applications. The CPU core consists of four functional blocks:

- Central Processing Unit (CPU)
- Memory Management Unit (MMU)
- Clock Generator
- Bus State Controller (Dynamic Memory Refresh)

The integrated I/O resources make up the remaining functional blocks:

- Direct Memory Access (DMA Control, Two Channels)
- Asynchronous Serial Communications Controller (ASCI, Two Channels)
- Programmable Reload Timers (PRT, Two Channels)

- Clocked Serial I/O Channel (CSIO)
- Enhanced Z85C33 (ESCC)
- Counter/Timer Channels (CTC)
- Parallel I/O
- Bidirectional P1284 Controller

Memory Management Unit. The MMU allows the user to map the memory used by the CPU (logically only 64 KB) into the 1 MB addressing range supported by the Z80185/Z80195. The organization of the MMU object code maintains compatibility with the Z80 CPU while offering access to an extended memory space. This is accomplished by using an effective “common area-banked area” scheme.

Central Processing Unit. The CPU is microcoded to provide a core that is object-code compatible with the Z80 CPU. It also provides a superset of the Z80 instruction set, including 8-bit multiply. This core has been modified to allow many of the instructions to execute in fewer clock cycles than on the Z80.

Clock Generator. This logic generates the system clock from either an external crystal or clock input. The external clock is divided by one or two, and is provided to both internal and external devices.

Bus State Controller. This logic performs the status and bus control activity associated with both the CPU and some on-chip Peripherals. This includes wait state timing, reset cycles, DRAM refresh, and DMA bus exchanges.

Interrupt Controller. This logic monitors and prioritizes internal and external interrupts and traps to provide the correct responses from the CPU. To maintain compatibility with the Z80 CPU, three different interrupt modes are supported.

DMA Controller. The DMA controller provides high speed transfers between memory and I/O devices. Transfer operations supported are memory to memory, and memory to/from I/O. Transfer modes supported are request, burst and cycle steal. DMA transfers can access the full 1 MB addressing range with a block length up to 64 KB, and can cross over 64K boundaries. On the Z80185/Z80195, the two DMA channels can handle I/O from the ESCC, the bidirectional Centronics interface, both UARTs (ASCIs), or an external device. Furthermore, the Z80185/Z80195's two channels can be used for the same device with hardware alternation, allowing continuous operation at very high data rates.

ESCC. This multiprotocol serial channel can handle asynchronous formats, character-oriented synchronous protocols, or bit-oriented synchronous protocols such as HDLC, LocalTalk, SDLC, X.25 and frame relay. It is enhanced from the industry-standard SCC by the inclusion of an 8-character receive FIFO and 4-character transmit FIFO and other improvements. For the Z80185/Z80195, the ESCC includes a special LocalTalk (AppleTalk) mode in which it automatically handles start- and end-of-frame transmission that required dedicated processor attention with previous (E)SCCs.

Bidirectional P1284 Interface. This controller, when combined with one of the DMA channels, can transfer large blocks of data into or out of one of the Z80185/Z80195's parallel ports without processor attention. It can operate as the Host or Peripheral side of such a parallel link in the standard Centronics compatible mode or any of the IEEE-defined Nibble, Byte or ECP modes. Data rates up to 700 KB/second in compatible mode, and up to about 2.5 MB/second in ECP mode can be achieved. Throughput and software overhead are further improved by hardware RLE-expansion provided by the interface during ECP reception.

Asynchronous Serial Communications Interface (ASCI). The ASCI logic provides two full-duplex UARTs. Each channel includes a choice of two programmable baud rate generators and modem control signals. On the Z80185/Z80195, the baud rate generators can be used with any crystal frequency and can handle data rates up to the oscillator rate divided by 64 (520 Kbits/second at 33.33 MHz). A new 4-byte receive FIFO helps make such data rates achievable in most applications.

Programmable Reload Time (PRT). This logic consists of two separate channels, each containing a 16-bit counter (timer) and count reload register. The time base for the counters is derived from the system clock (divided by 20) before reaching the counter. PRT channel 1 provides an optional output to allow for waveform generation.

Counter/Timers (CTCs). Four of these flexible channels add to the counting and timing facilities provided by the two PRTs. Each channel includes a prescaler and 8-bit downcounter with programmable reload value, and can be programmed for counter/timer, or triggered-timer operation. On the Z80185/Z80195, a Long Counter mode provides for extended time intervals despite today's fast oscillator rates.

Clocked Serial I/O (CSIO). The CSIO channel provides a half-duplex serial transmitter and receiver. This channel can be used for simple high-speed data connection to another microprocessor or microcomputer, or to serial memories.

Memory And Input/Output Cycle Timing

TIMING

The basic CPU operation consists of one or more Machine Cycles (MC). A machine cycle consists of an access to internal or external memory or I/O and includes at least three system clocks called T_1, T_2, and T_3. Optional Wait states may be inserted between T_2 and T_3, either by one of the on-chip Wait-state generators or by external logic driving the $\overline{\text{WAIT}}$ pin. In this manual, external system clock-cycles that are idle during instruction execution, are not considered machine cycles. Thus, the execution of an instruction requires one or more machine cycles and for some instructions, one or more externally-idle system clocks.

A system clock cycle may be one or two cycles on the XTAL pin(s) depending on the setting for the clock divide option in the CPU Control Register (`CCR`), as described in the “CPU Options” section.

To avoid undue complexity, the following descriptions and waveforms assume that the MIE and IOC bits in the Operating Mode Control Register (`OMCR`) are both `1`. The “CPU Options” section, describes the effects of the setting of these bits.

Op Code Fetch Timing

Figure 3 describes the Op Code instruction fetch timing with no Wait states. An Op Code fetch cycle is
externally indicated when the $\overline{\text{M1}}$ output pin is Low.

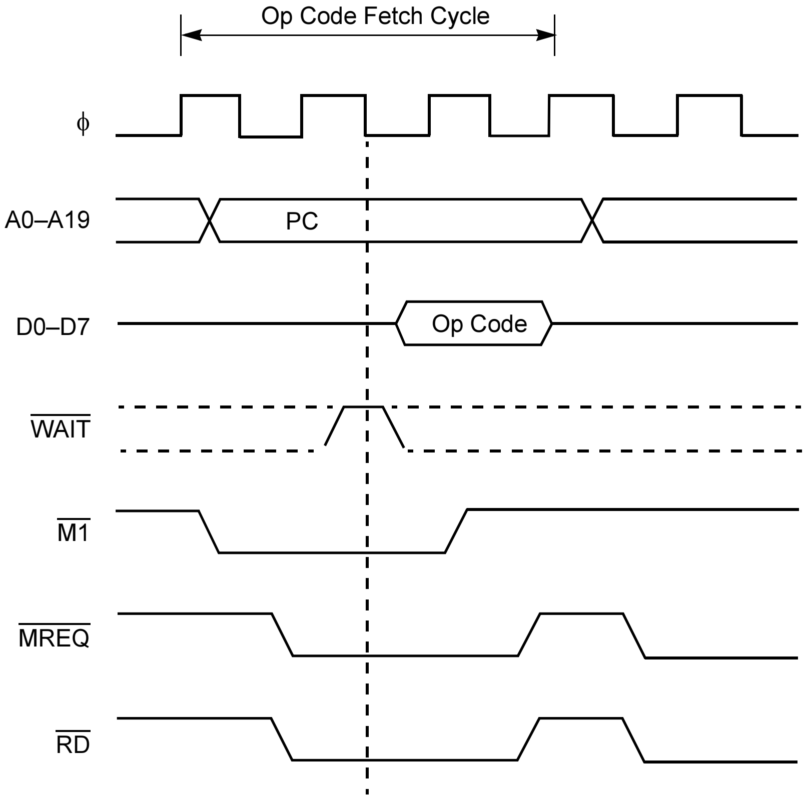


Figure 3. Op Code Fetch Timing Without Wait States

In the first half of T_1, the address bus (A19-A0) is driven by the contents of the Program Counter (PC). The address bus contains the translated address Output of the on-chip MMU.

In the second half of T_1, the $\overline{\text{MREQ}}$ and $\overline{\text{RD}}$ signals are asserted Low, enabling memory.

The Op Code on the data bus is latched at the Rising edge of $T_{3,}$ and the bus cycle terminates at the end of T_3.

Figure 4 illustrates the insertion of Wait states (T_W) into the Op Code fetch cycle. Wait states (T_W) are controlled by the external $\overline{WAIT}$ input combined with on-chip programmable Wait-state generators. At the Falling edge of $T_{2,}$ the combined $\overline{WAIT}$ input is sampled. If $\overline{WAIT}$ input is asserted Low, a Wait state (T_W) is inserted. The address bus, $\overline{MREQ}$, $\overline{RD}$ and $\overline{M1}$ are held stable during Wait states. When $\overline{WAIT}$ is sampled Inactive High at the Falling edge of T_W, the bus cycles enter T_3 and completes at the end of T_3.

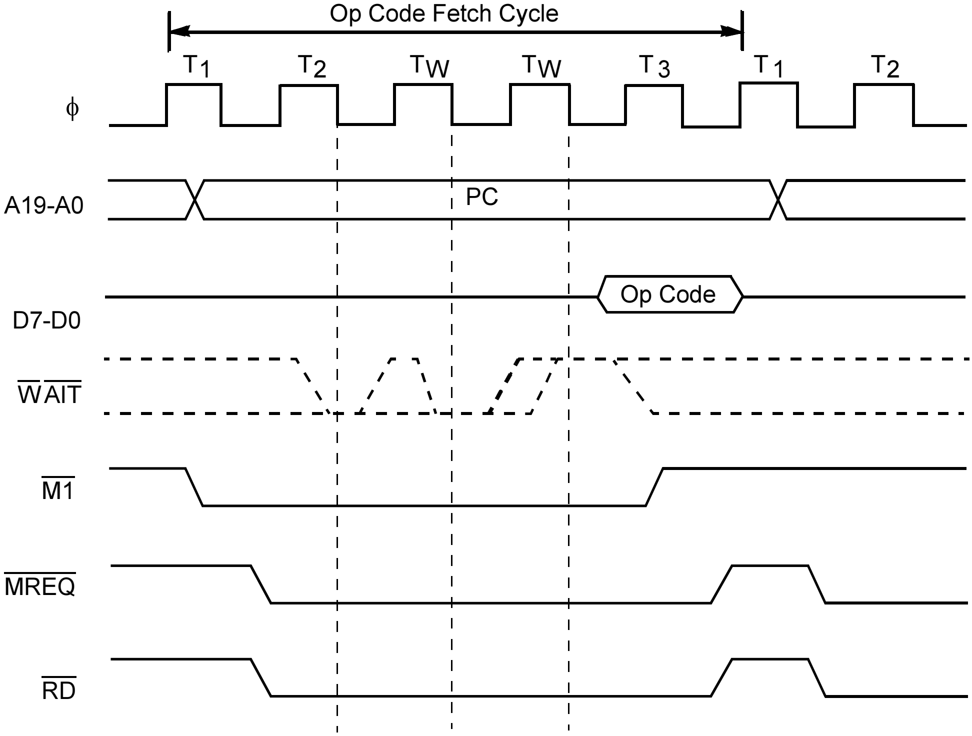


Figure 4. Op Code Fetch Timing With Wait States

Operand and Data Read/Write Timing

Operand and data read/write timing differs from Op Code fetch timing in two ways. First, the $\overline{\text{M1}}$ output is held Inactive. Second, read-cycle timing is relaxed by one-half clock cycle because data is latched at the Falling edge of T_3.

Instruction operands include immediate data, displacements, and extended addresses and have the same timing as memory-data reads.

During memory-write cycles, the $\overline{\text{MREQ}}$ signal goes Active in the second half of T_1, and the data bus is driven with the write data.

At the start of T_2, the $\overline{\text{WR}}$ signal is asserted Low enabling memory. $\overline{\text{MREQ}}$ and $\overline{\text{WR}}$ go Inactive in the second half of T_3 followed by disabling of the write data on the data bus.

Wait states (T_W) may be inserted as previously described for Op Code fetch cycles. Figure 5 illustrates read/write timing without Wait states, while Figure 6 illustrates read/write timing with Wait states.

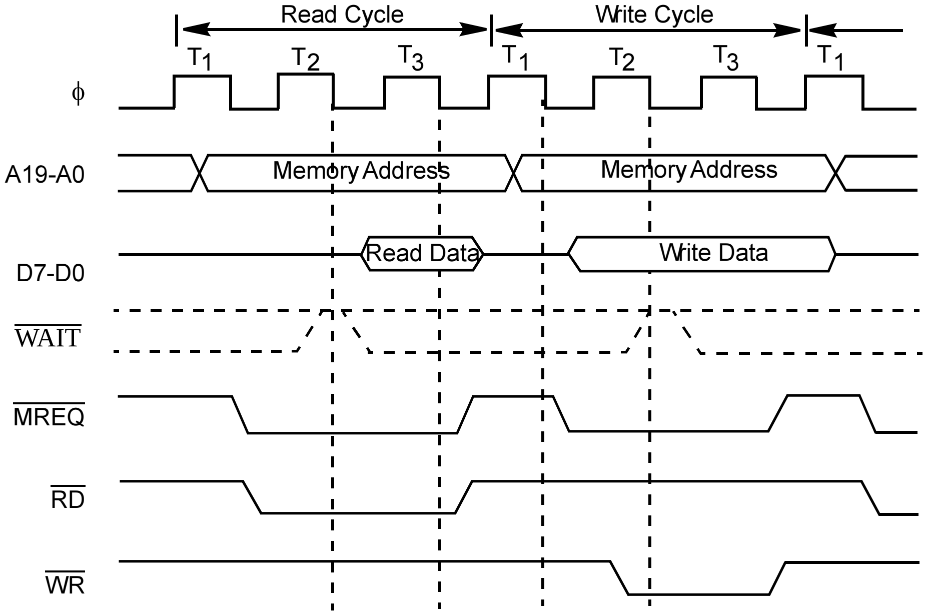


Figure 5. Memory Read/Write Timing Without Wait States

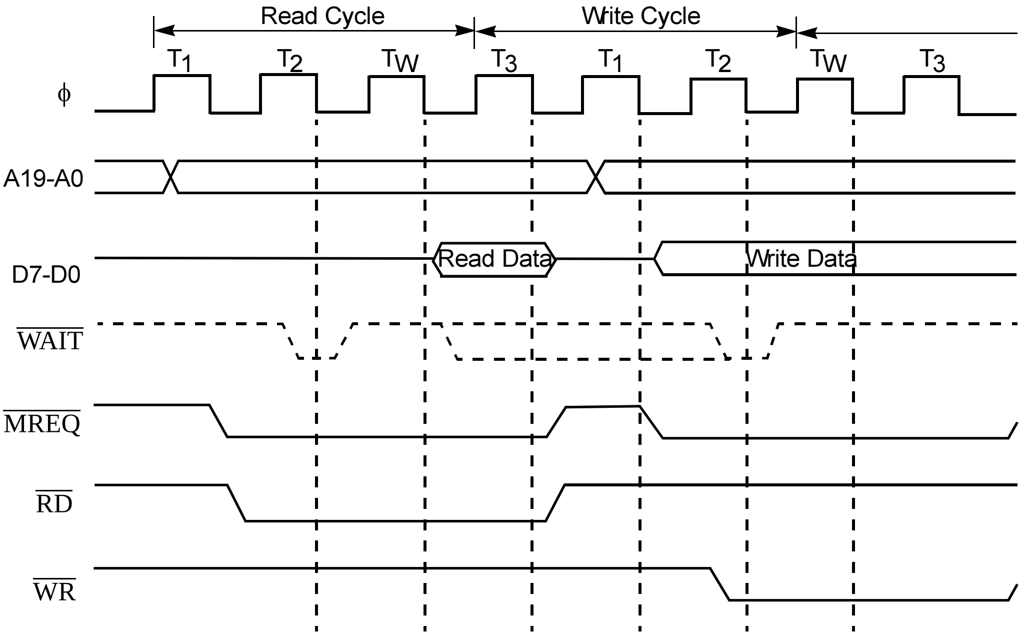
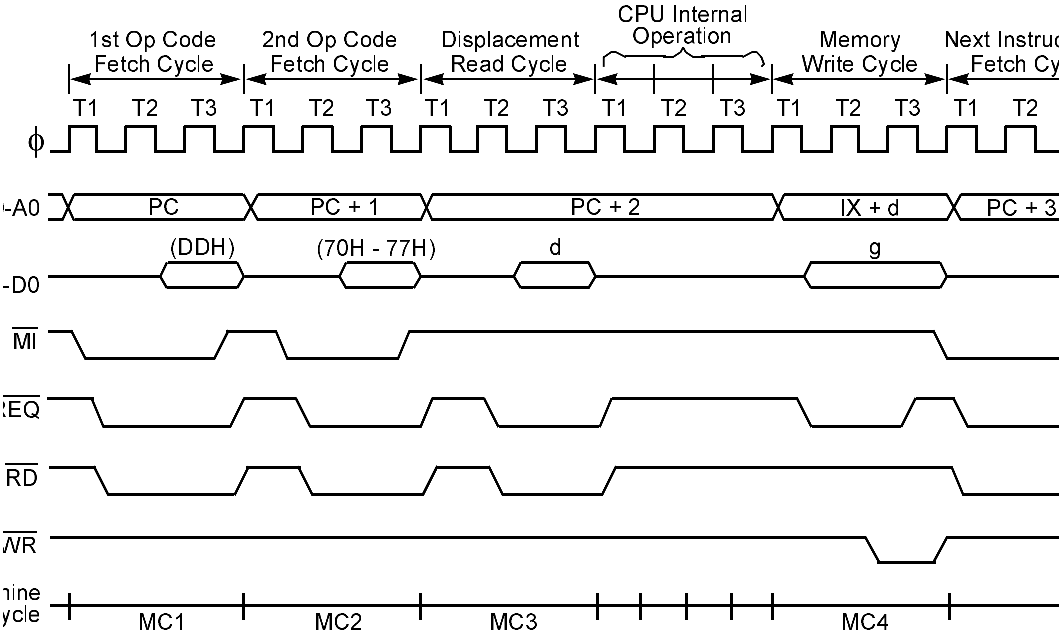


Figure 6. Memory Read/Write Timing With Wait States

Basic Instruction Timing

An instruction may consist of a number of machine cycles including Op Code fetch, operand fetch, and data read/write cycles. An instruction may also include clock-cycles for internal processes, during which the bus is Idle.

Figure 7 illustrates the bus timing for the data-transfer instruction (`LD (IX+d),g`. This instruction moves the contents of a CPU register (`g`) to the memory location with address computed by adding a signed 8-bit displacement (`d`) to the contents of an index register (`IX`).



e:
Displacement
Regular contents

Figure 7. Instruction Timing

The instruction cycle starts with the two machine cycles required to read the two byte instruction Op Code as indicated by $\overline{\text{M1}}$ Low. The instruction operand (d) is fetched.

The external bus is IDLE while the CPU computes the effective address. Finally, the computed memory location is written with the contents of the CPU register (g).

Reset Timing

Figure 8 illustrates hardware Reset timing. If the $\overline{\text{RESET}}$ pin is Low for six or more clock cycles, processing is terminated and the execution restarts from address (logical and physical) `00000H`.

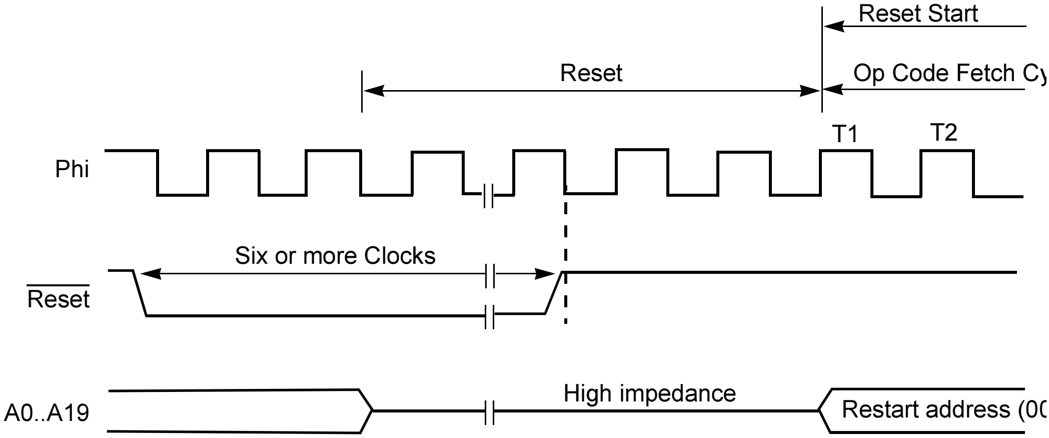


Figure 8. Reset Timing

Bus-Exchange Timing

The processor coordinates the exchange of control, address and data bus ownership with another bus Master. The alternate bus Master may request the bus release by asserting the $\overline{\text{BUSREQ}}$ Input Low. After the Z80185/ Z80195 releases the bus, control passes to the alternate bus Master by asserting the $\overline{\text{BUSACK}}$ Output Low.

The bus may be released by the Z80185/Z80195 at the end of a machine cycle or an external Idle clock cycle.

When the bus is released, the address bus (A1..A19), data bus (D0..D7), and control signals ($\overline{\text{MREQ}}$, $\overline{\text{IORG}}$, $\overline{\text{RD}}$, and $\overline{\text{WR}}$) are placed in the high-impedance state to on-chip memory or I/O. These signals are monitored for a cycle by the alternate bus Master. Dynamic RAM refresh is not performed when the Z80185/Z80195 has released the bus. The alternate bus Master must provide dynamic-memory refreshing if the bus is released for long periods of time.

Figure 9 illustrates $\overline{\text{BUSREQ}}$/$\overline{\text{BUSACK}}$ bus-exchange timing during a memory-read cycle. Figure 10 illustrates bus exchange during a CPU internal operation. $\overline{\text{BUSREQ}}$ is sampled at the Falling edge of the system clock prior to T_3, Ti and Tx (BUS RELEASE state). If $\overline{\text{BUSREQ}}$ is asserted Low at the Falling edge of the clock state prior to Tx, another Tx is executed.

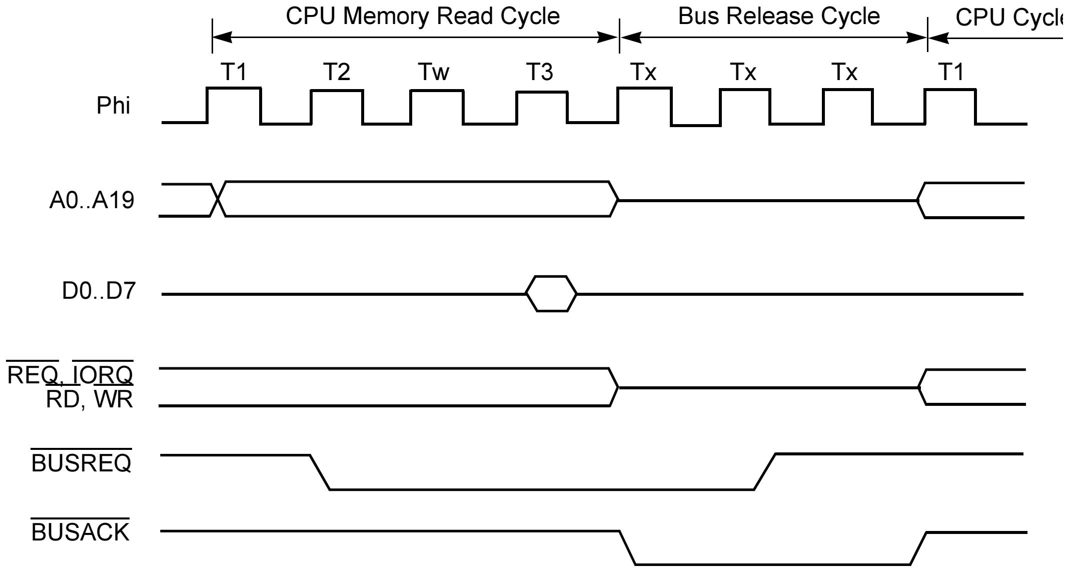


Figure 9. Bus Exchange Timing During A Memory Read Cycle

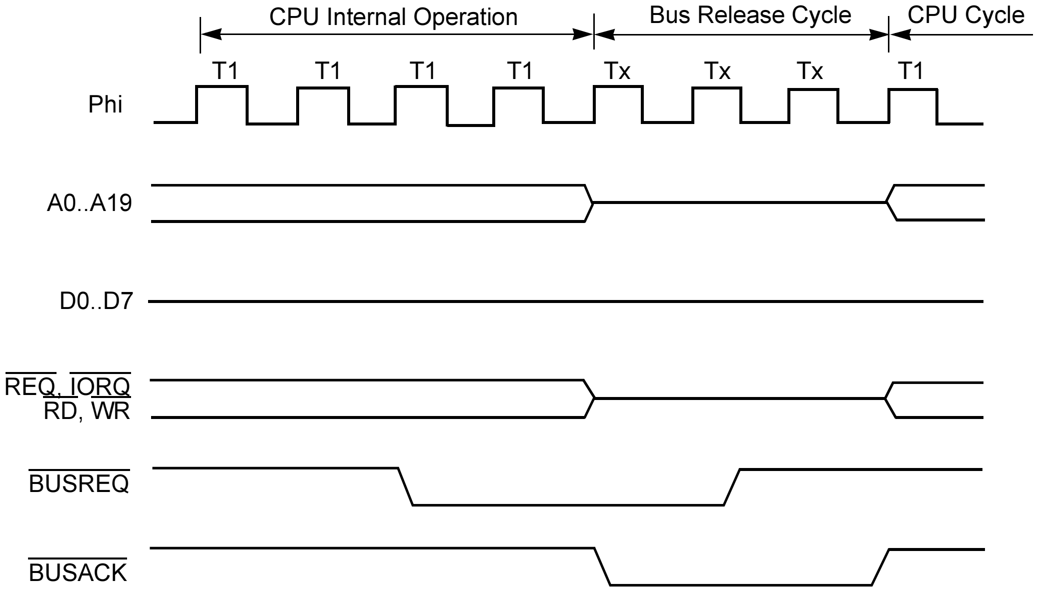


Figure 10. Bus Exchange Timing During CPU Internal Operation

INPUT/OUTPUT

Input/Output devices and registers reside in a separate address space from memory. In the original Z80, I/O addresses were 8 bits wide and were carried on the A7..A0 lines, with A15..A8 were driven from various CPU registers and were mostly ignored by I/O devices. With the Z8018X family, I/O addresses are extended to 16 bits, and special instructions like `IN0` and `OUT0` are included which ensure that the A15..8 lines are all `0`.

Internal I/O Registers

The Z80185/Z80195 contain two groups of I/O registers. Group 1 includes the registers for the MMU, DMA, ASCIs, PRT and CSI/O, and has the following characteristics:

- Decodes 16-bit I/O addresses.
- Resides at addresses `0000-003F` after Reset, but may be relocated to `00040-007F` or `0080-00BF` by programming the I/O Control Register (`ICR`).
- Is accessed in three clock-machine cycles.

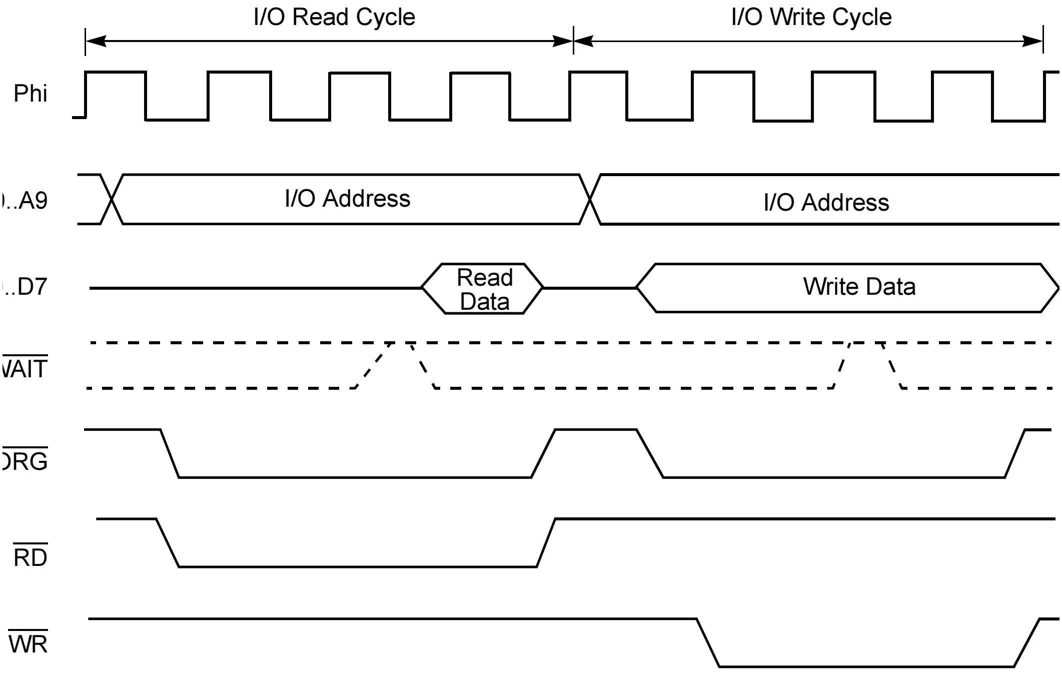
Group 2 includes the registers for the ESCC, parallel ports, Bidirectional Centronics controller, CTCs, chip select, and Watch-Dog Timer, and has the following characteristics:

- May decode 8- or 16-bit addresses depending on the Decode High bit in the System Configuration Register.
- Resides at addresses `xxD8-xxF1`.
- Is accessed using four machine clock-cycles, like off-chip I/O devices.

I/O Read/Write Timing

I/O instructions cause data read/write transfers which differ from memory-data transfers in the following ways:

1. The $\overline{\text{IORQ}}$ signal is asserted Low instead of the $\overline{\text{MREQ}}$ signal.
2. The 16-bit I/O address is not translated by the MMU.
3. A19..A16 are held Low.
4. A15..A8 may or may not be decoded by the I/O device. At least one Wait state (T_W) is always inserted for I/O ready and write cycles (except Group 1 internal I/O cycles). See Figure 11.



Note: A19..A16 = 0 for I/O Cycles

Figure 11. I/O Read/Write Timing

The Processor

INTRODUCTION

This chapter describes the processor core of the Z80185/Z80195, with particular emphasis on the operational options provided by its various I/O Registers.

TIMING

Figure 12 illustrates the RETI instruction sequence. Figure 13 depicts the MI Temporary Enable timing diagram. Figure 14 illustrates the I/O Read and Write Cycles with IOC set to 1.

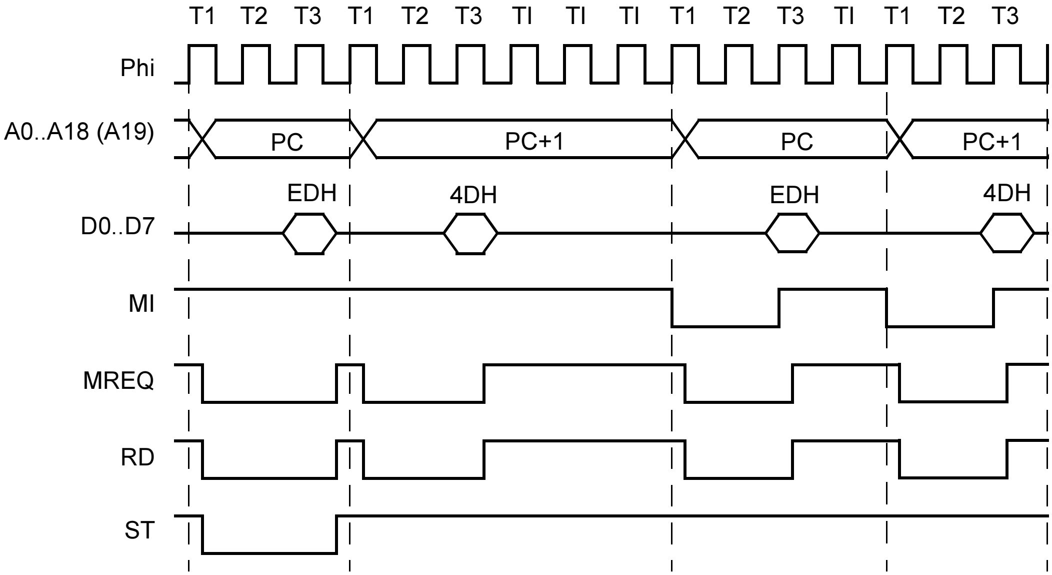


Figure 12. RETI Instruction Sequence with MIE Set to 0

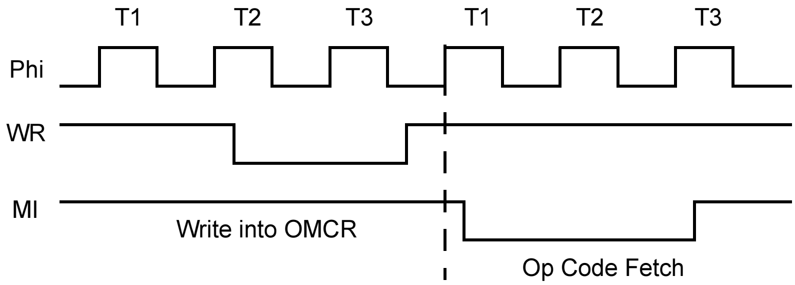


Figure 13. MI Temporary Enable Timing

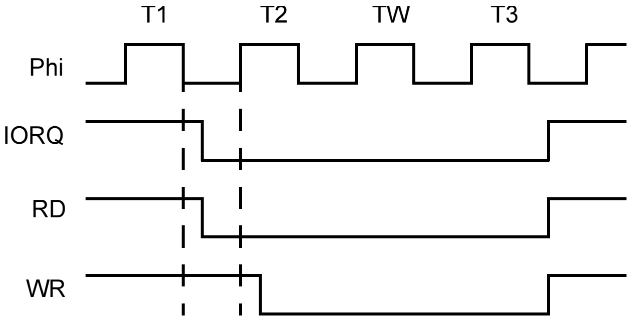


Figure 14. I/O Read and Write Cycles with $\overline{\text{IOC}}$ Set to 1

Figure 15 depicts the I/O Read and Write Cycles with IOC set to 0. Figure 16 illustrates the I/O address relocation.

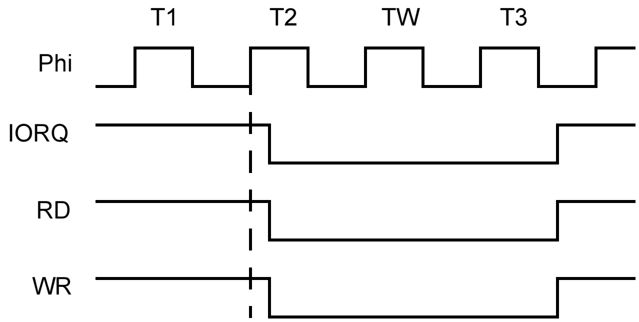


Figure 15. I/O Read and Write Cycles with $\overline{\text{IOC}}$ Set to 0

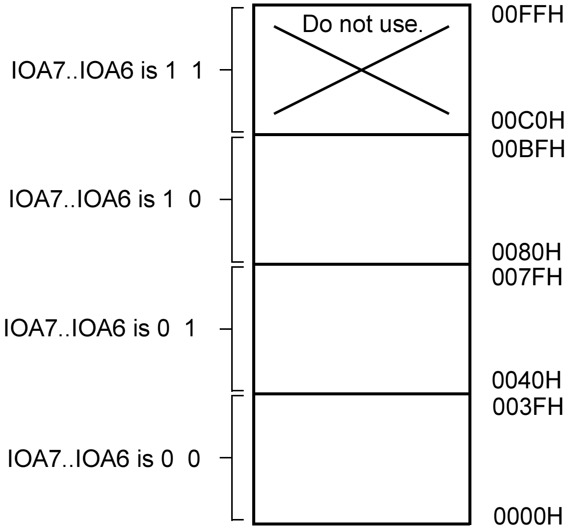


Figure 16. I/O Address Relocation

Figure 17 depicts Halt timing, while Figure 18 represents Sleep timing. Figure 19 illustrates the Z80185/Z80195 Idle Mode Exit because of External Interrupt.

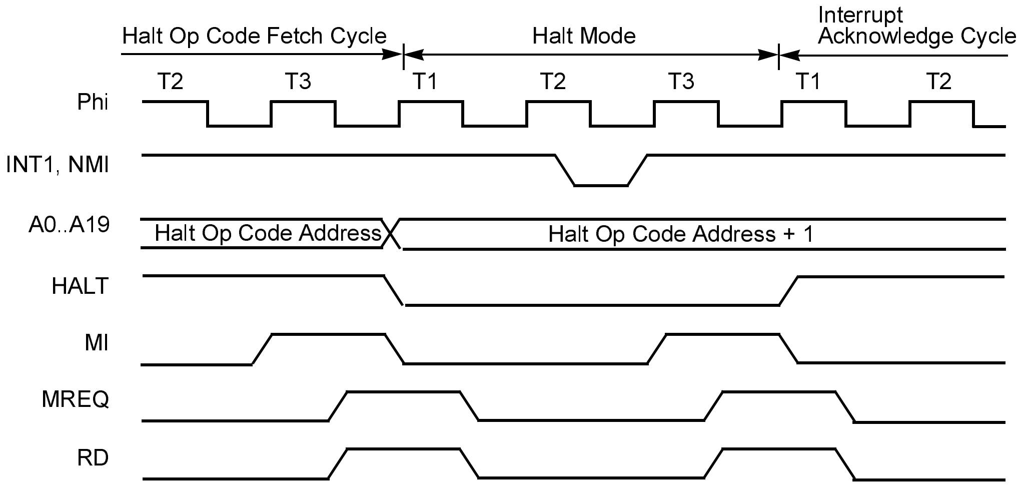


Figure 17. Halt Timing

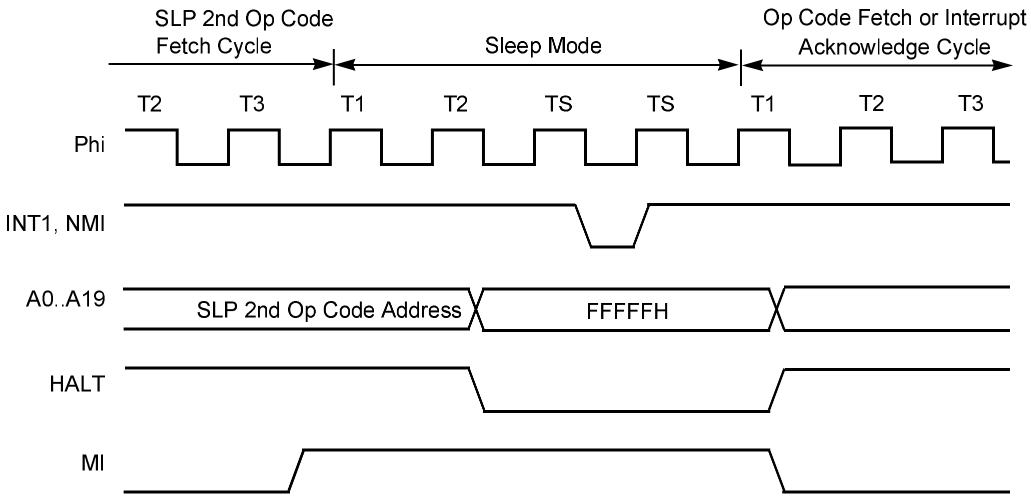


Figure 18. Sleep Timing

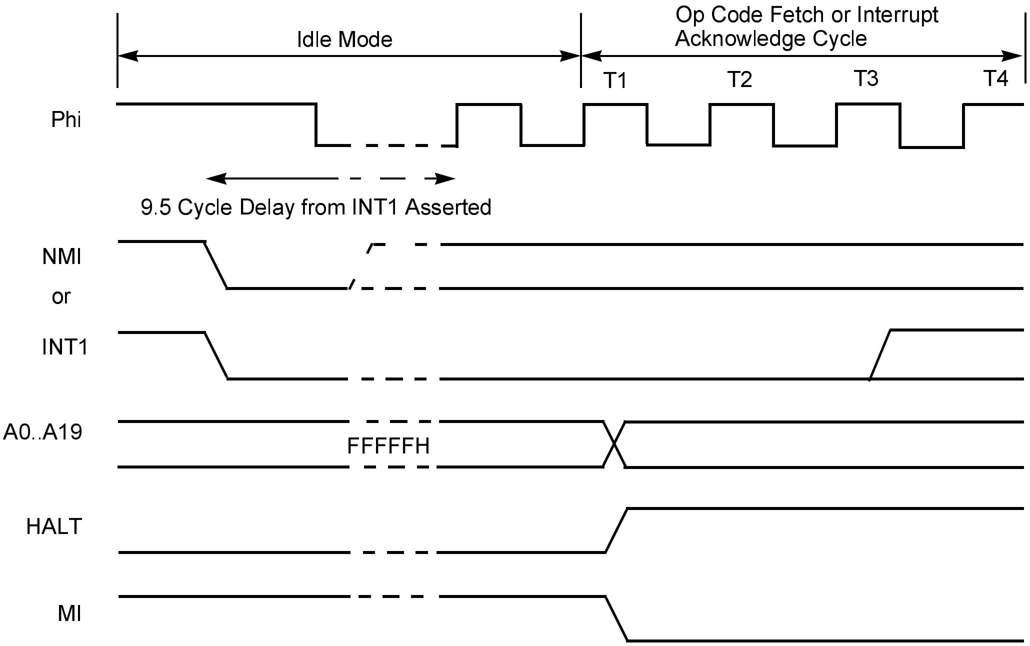


Figure 19. Z80185/Z80195 Idle Mode Exit Because of External Interrupt

Figure 20 illustrates Bus Granting to External Master in Idle Mode. Figure 21 depicts Z8S180 and Z80Z8L180 Standby Mode Exit because of External Interrupt. Figure 22 represents Bus Granting to External Master During Standby Mode.

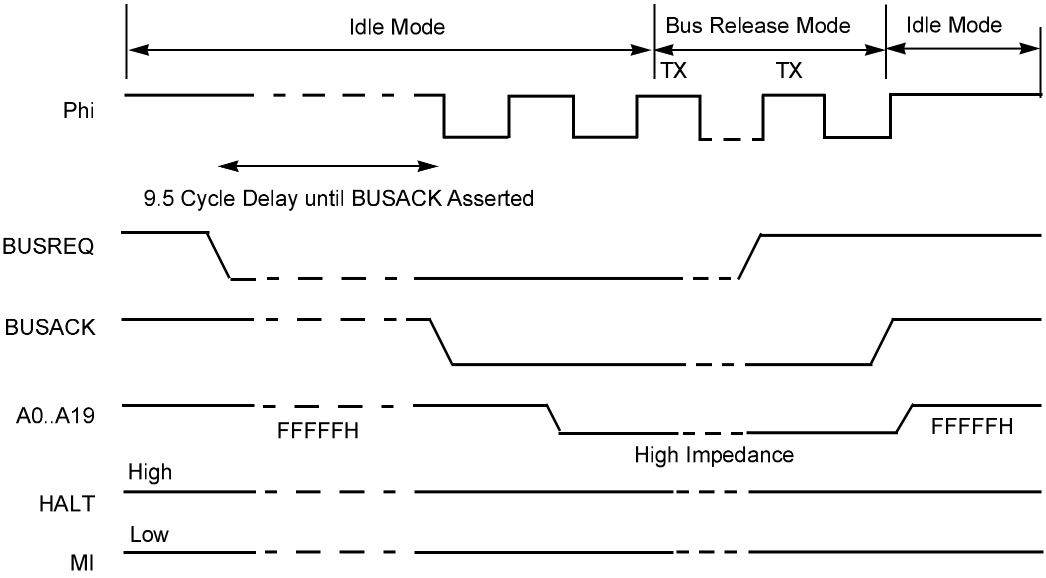


Figure 20. Bus Granting to External Master in Idle Mode

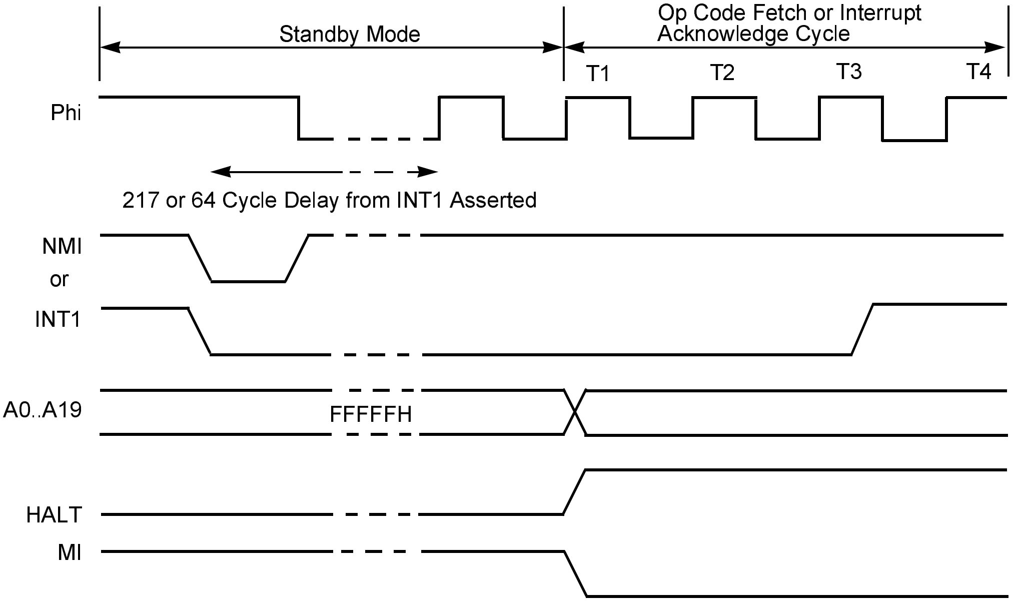


Figure 21. Z8S180 and Z80Z8L180 Standby Mode Exit Because of External Interrupt

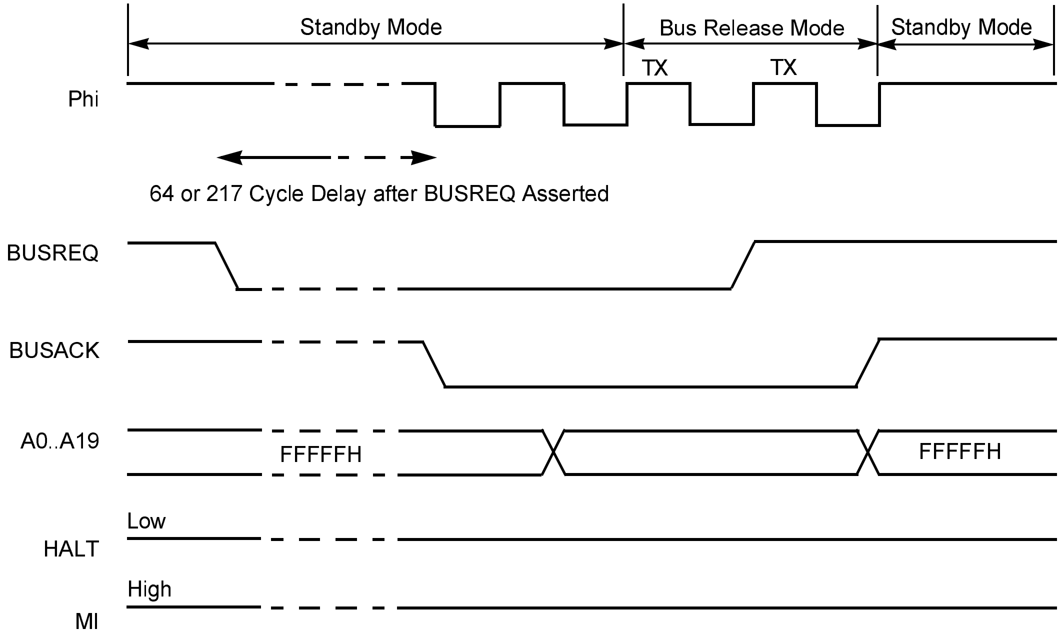


Figure 22. Bus Granting to External Master During Standby Mode

Figure 23 represents TRAP Timing - 2nd Op Code Undefined. Figure 24 depicts TRAP Timing - 3rd Op Code Undefined.

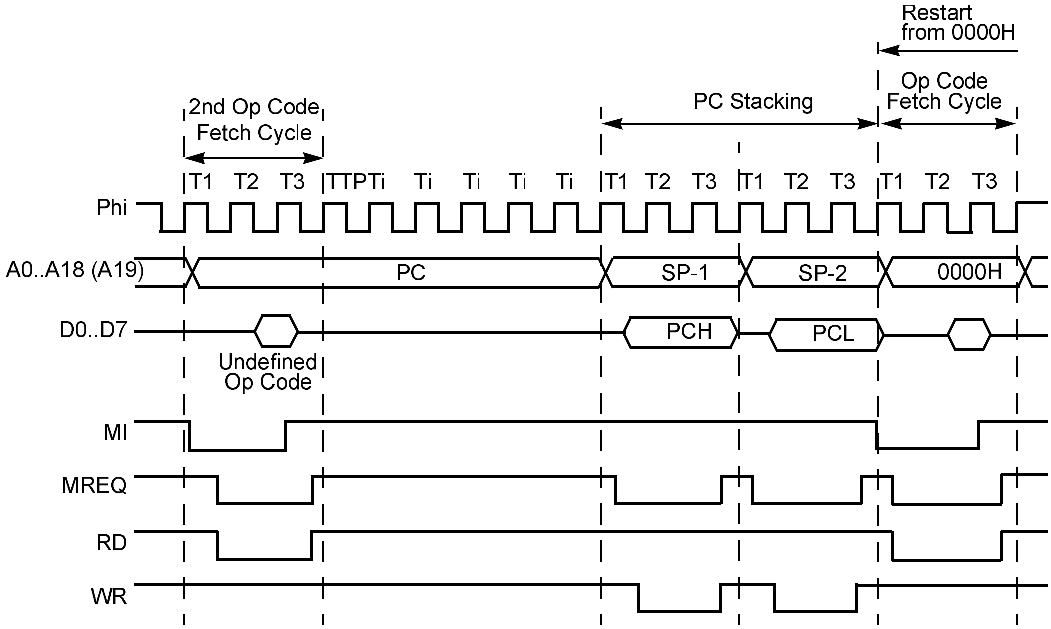


Figure 23. TRAP Timing - 2nd Op Code Undefined

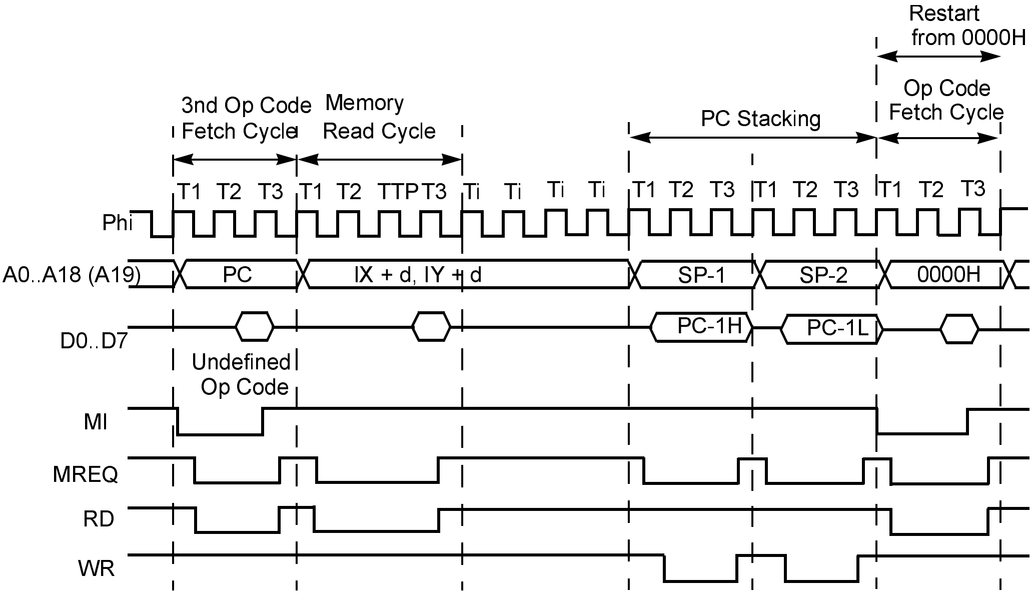


Figure 24. TRAP Timing - 3rd Op Code Undefined

Figure 25 depicts $\overline{\text{NMI}}$ timing. Figure 26 illustrates Level 0 Mode 0 Timing. Figure 27 represents Level 0 Mode 1 Timing.

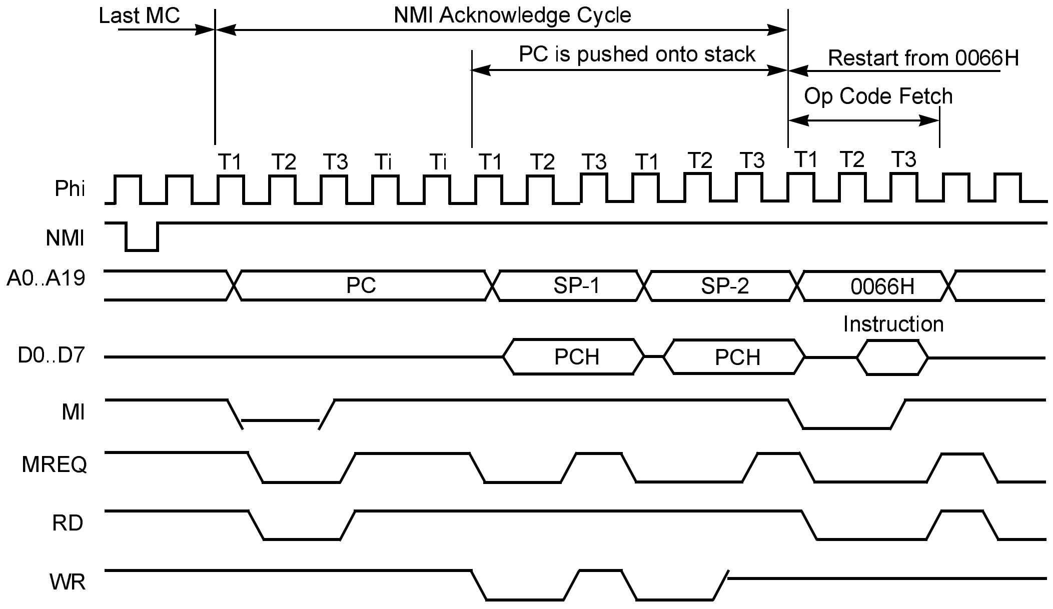


Figure 25. $\overline{\text{NMI}}$ Timing

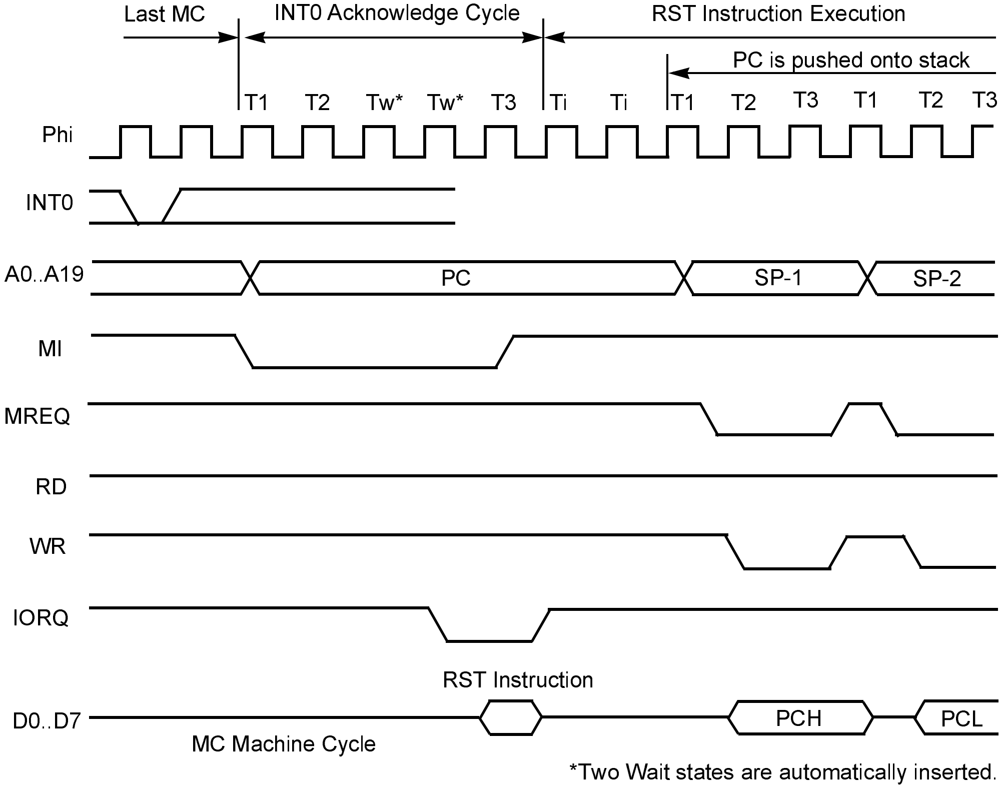


Figure 26. Level 0 Mode 0 Timing

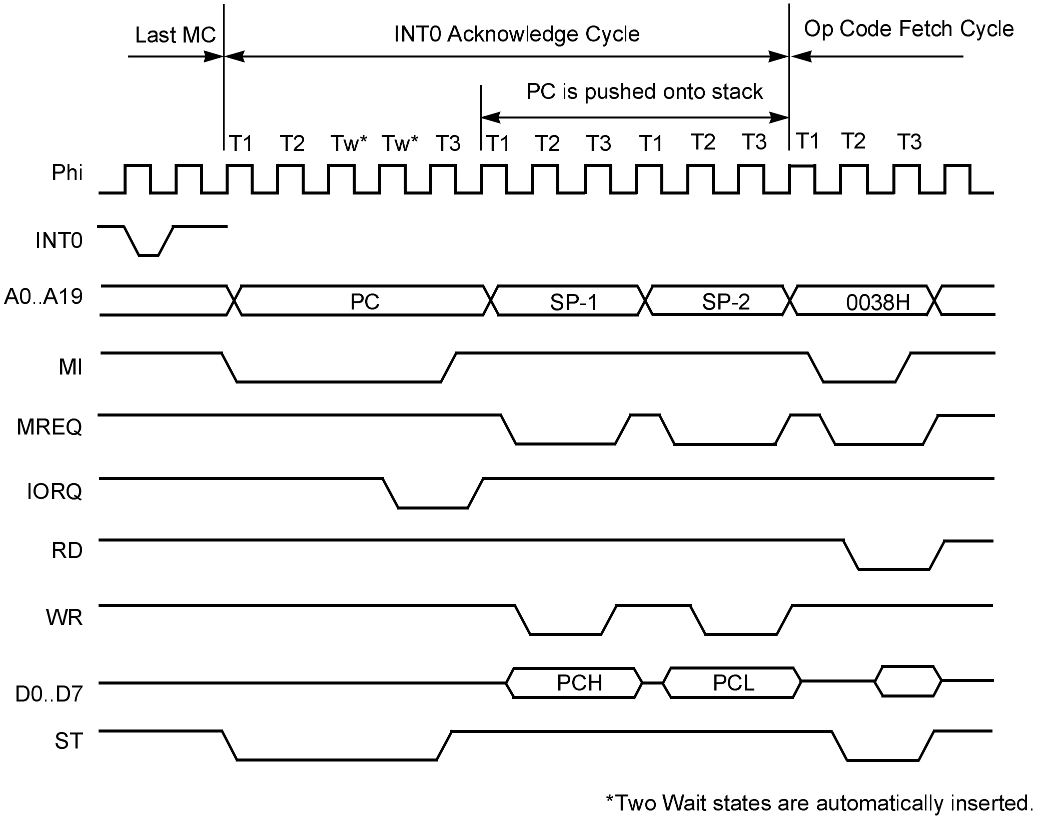


Figure 27. Level 0 Mode 1 Timing

Figure 28 represents Level 0 Interrupt Mode 2 timing. Figure 29 depicts INT1, INT2, DMA, ASCI, PRT, CSI/O Interrupt timing.

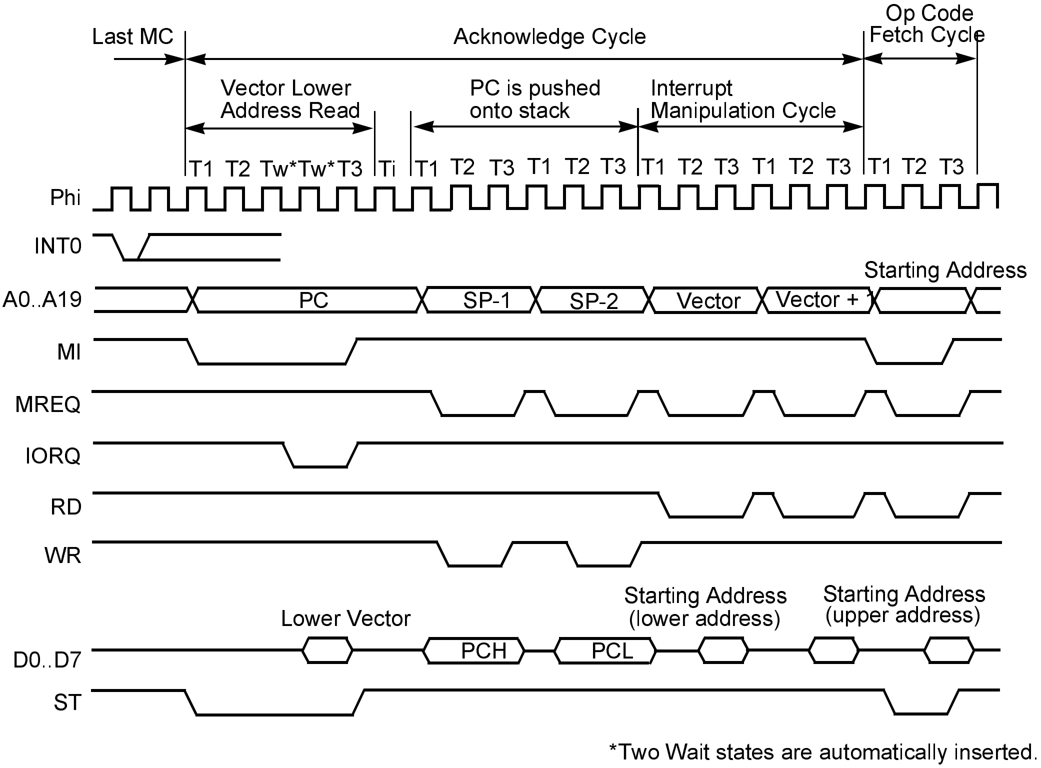


Figure 28. Level 0 Interrupt Mode 2 Timing

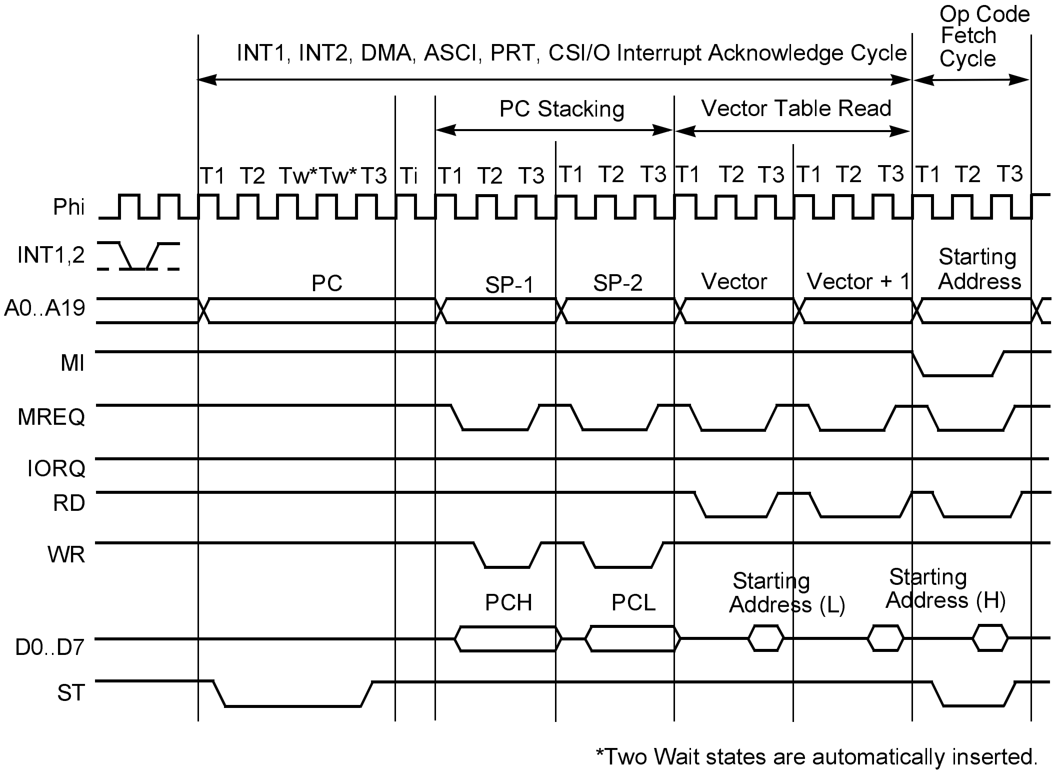
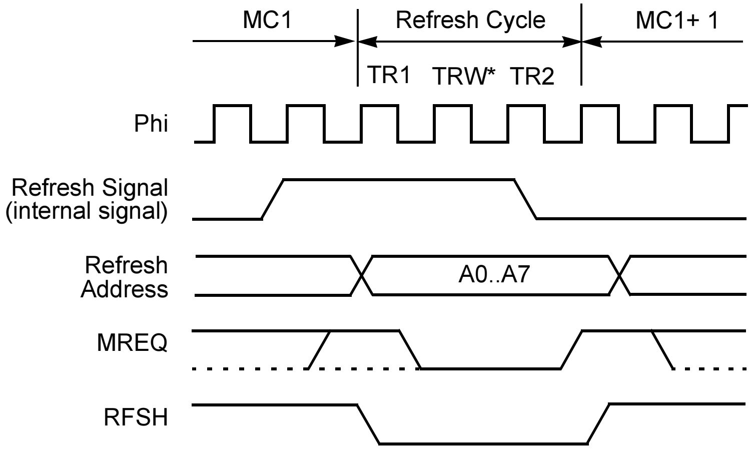


Figure 29. $\overline{\text{INT1}}$, $\overline{\text{INT2}}$, **DMA, ASCI, PRT, CSI/O Interrupt Timing**

Figure 30 depicts Refresh Cycle timing. Figure 31 illustrates the Refresh Control Register timing.



Note: * If three refresh cycles are specified, TRW, is inserted. Otherwise, TRW is not inserted.
MC: Machine Cycle

Figure 30. Refresh Cycle Timing

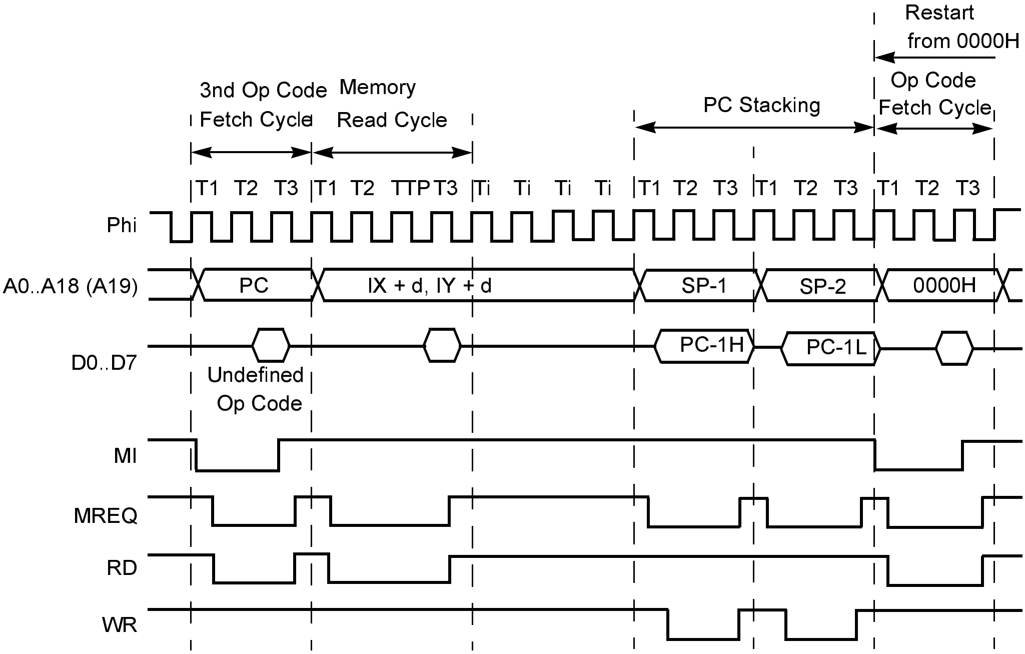


Figure 31. Refresh Control Register Timing

CPU OPTIONS

Z80 versus 64180 Compatibility

The Z80185/Z80195 is descended from two different *ancestor* processors, ZiLOG's original Z80 and the Hitachi 64180. The Operating Mode Control Register (OMCR) may be programmed to select between certain differences between the Z80 and the 64180.

Operating Mode Control Register (`OMCR: 3EH`)

Table 1:

7	6	5	4	0
MIE	$\overline{\text{MITE}}$	$\overline{\text{IOC}}$	Reserved	
1	1	1		
R/W	W	R/W		

Bit Number	Field	R/W	Reset Value	Description
7	MIE	R/W	1	**Enable** `MIE` controls the MI Output. 1: The $\overline{\text{MI}}$ Output is asserted Low during opcode fetch cycles, $\overline{\text{INT0}}$ acknowledge cycles, and the first machine cycle of the $\overline{\text{NMI}}$ acknowledge. On the Z80185/Z80195, this choice makes the processor fetch an `RETI` instruction once, and when fetching an `RETI` from Zero-Wait-state memory uses three clock-machine cycles. This mode is not fully Z80-timing compatible but is compatible with the on-chip CTCs. 0: The processor does not drive $\overline{\text{MI}}$ Low during instruction fetch cycles, and after fetching an `RETI` instruction once with normal timing, it goes back and re-fetches the instruction using fully Z80-compatible cycles that include driving $\overline{\text{MI}}$ Low. This may be needed by some external Z80 peripherals to properly decode the `RETI` instruction. Refer to Figure 12 in the front of this Chapter. Table 3-1 describes the `RETI` sequence when `MIE` is `0`.

Bit Number	Field	R/W	Reset Value	Description
6	$\overline{\text{MITE}}$	W	1	**Temporary Enable** This bit controls the temporary assertion of the $\overline{\text{MI}}$ signal. When Bit 7 (`MIE`) is set to `0` to accommodate certain external Z80 peripheral(s), those same device(s) may require a pulse on $\overline{\text{MI}}$ after programming certain of their registers to complete the function being programmed. For example, when a control word is written to the Z80 PIO to enable interrupts, no enable actually takes place until the PIO sees an Active $\overline{\text{MI}}$ signal. 1: There is no change in the operation of the $\overline{\text{MI}}$ signal and Bit 7 (`MIE`) controls its function. 0: The $\overline{\text{MI}}$ Output is asserted during the next Op Code Fetch cycle regardless of the state programmed into the Bit 7 (`MIE`). This is only momentary (one time) and the user need not reprogram a `1` to disable the function. Refer to Figure 13 in the front of this Chapter.
5	$\overline{\text{IOC}}$	R/W	1	**Timing Control** This bit controls the timing of the $\overline{\text{IORQ}}$ and $\overline{\text{RD}}$ signals. 1: The $\overline{\text{IORQ}}$ and $\overline{\text{RD}}$ signals function the same as the Z64180. 0: The timing of the $\overline{\text{IORQ}}$ and $\overline{\text{RD}}$ signals match the timing of the Z80. The $\overline{\text{IORQ}}$ and $\overline{\text{RD}}$ signals go Active as a result of the Rising edge of T2. Refer to Figure 14 and Figure 15 in the front of this Chapter.
4..0	Reserved		0	**Reserved** Must be `0`.

				MI							
Machine Cycle	States	Address	Data	R D	W R	MRE Q	IOR Q	$\overline{\text{IOC}}$ is 1	IOC is 0	Halt	ST
1	T1..T3	1st Op Code	EDH	0	1	0	1	0	1	1	0
2	T1..T3	2nd Op Code	4DH	0	1	0	1	0	1	1	0
	Ti	NA	3-State	1	1	1	1	1	1	1	1
	Ti	NA	3-State	1	1	1	1	1	1	1	1
	Ti	NA	3-State	1	1	1	1	1	1	1	1
3	T1..T3	1st Op Code	EDH	0	1	0	1	0	0	1	1
	Ti	NA	3-State	1	1	1	1	1	1	1	1
4	T1..T3	2nd Op Code	4DH	0	1	0	1	0	1	1	1
5	T1..T3	SP	Data	0	1	0	1	1	1	1	1
6	T1..T3	SP+1	Data	0	1	0	1	1	1	1	1

I/O Control Register

ICR allows relocating of the internal I/O addresses, and also controls enabling/disabling of the IOStop Mode.

I/O Control Register (ICR: 3FH)

7	6	5	4	0
IOA7	IOA6	IOSTP	Reserved	
0	0	0		
R/W	R/W	R/W		

Bit Number	Field	R/W	Reset Value	Description
7..6	IOA7.. IOA6	R/W	0	**I/O Address Relocation** Bit 7 (IOA7) and Bit 6 (IOA6) relocate internal I/O. The High-order 8 bits of 16-bit internal I/O address are always 0. Refer to Figure 16 in the front of this Chapter.
5	IOSTP	R/W	0	**I/O Operation** IOStop Mode is enabled when Bit 5 (IOSTP) is set to 1. Normal I/O operation resumes when IOStop is reprogrammed or reset to 0.
4..0	Reserved		0	**Reserved** Must be 0.

CPU Control Register

This register controls the basic clock rate, certain aspects of Power-Down Modes, and Output Drive/Low noise options.

CPU Control Register (`CCR: 1FH`)

7	6	5	4	3	2	1	0
Clock Divide	Standby/ Idle Enable	BREXT	LNPHI	Standby/ Idle Enable	LNIO	LNCPUCTL	LNAD/ DATA
0	0	0	0	0	0	0	0

Bit Number	Field	R/W	Reset Value	Description
7	Clock Divide		0	**Clock Divide Select**. 1: XTAL divided by 1. 0: XTAL divided by 2. If an external oscillator is used in divide-by-one Mode, the minimum pulse width requirement given in the AC Characteristics must be satisfied.

Bit Number	Field	R/W	Reset Value	Description
6 and 3	Standby/ Idle Enable		0	**Standby/Idle Control** 1: Setting IOStop Bit 5 (ICR5) and executing a SLP instruction puts the part into Quick Recovery Standby Mode, in which the on-chip oscillator is stopped, and the part allows only 64 clock cycles for the oscillator to stabilize when it's restarted. 0: A SLP instruction makes the Z80185/Z80195 enter Sleep or System Stop Mode, depending on the IOStop Bit 5 (ICR5). When Bit 6 is 0 and Bit 3 is 1, setting the IOStop Bit 5 (ICR5) and executing a SLP instruction puts the Z80185/ Z80195 into Idle Mode, in which the on-chip oscillator runs but its output is blocked from the rest of the part including ϕ out. When Bit 6 is 1 and Bit 3 is 0, setting IOStop Bit 5 (ICR5) and executing a SLP instruction puts the part into Standby Mode, in which the on-chip oscillator is stopped and the part allows 2^{17} (128K) clock cycles for the oscillator to stabilize when it is restarted.
5	BREXT		0	**Bus Request** This bit controls the ability of the Z80185/Z80195 to honor a bus request during Standby Mode. 1: If the part is in Standby Mode, a BUSREQ is honored after the clock stabilization timer has timed out.
4	LNPHI		0	**Clock Output** This bit controls the drive capability on the ϕ Clock Output. 1: The ϕ Clock Output is reduced to 33% of its full drive capability.

Bit			Reset	
Number	Field	R/W	Value	Description
2	LNIO		0	**I/O Pin Drive Capability Control** This bit controls the drive capability of certain external I/O pins of the Z80185/Z80195. 1: The output drive capability of the following pins is reduced to 33% of the full drive capability: - $\overline{\text{RTSO}}$/TxS - CKA1 - CKAO - TXAO - TXAI - TOUT
1	LNCPUCTL		0	**CPU Control Pins Drive Capability Control** This bit controls the drive capability of the CPU Control pins. 1: The output drive capability of the following pins is reduced to 33% of the full drive capability: - $\overline{\text{BUSACK}}$ - $\overline{\text{RD}}$ - $\overline{\text{WR}}$ - $\overline{\text{MI}}$ - $\overline{\text{MREQ}}$ - $\overline{\text{IORQ}}$ - $\overline{\text{RFSH}}$ - $\overline{\text{HALT}}$

Bit Number	Field	R/W	Reset Value	Description
0	LNAD/ DATA		0	**Address/Data bus Drive Capability Control** This bit controls the drive capability of the Address/Data bus Output drivers. 1: The output drive capability of the Address and Data bus Output is reduced to 33% of its original drive capability. In addition to the bits in the CCR, if Bit 1 of the Interrupt Edge Register (IER, address DF) is set to 1, the output drive capability of the following pins is reduced to 33% of their full drive capability: - PIA10..13 Busy - PIA14..16 nAcK - PIA27..20 nAutoFd - $\overline{\text{ROMCS}}$ nFault - $\overline{\text{RAMCS}}$ nInit - $\overline{\text{IOCS}}$ nSelectIn - IEO nStrobe - $\overline{\text{RTS}}$ PError - $\overline{\text{DTR}}$ Select - TXD - $\overline{\text{TRXC}}$

System Configuration Register

The System Configuration Register controls a number of Device-Level Features on the Z80185/Z80195 and includes the following control bit.

System Configuration Register (EDH)

7	6	5	4	3	2	1	0
Decode High I/O	Daisy-Chain	Disable $\overline{\text{ROMCS}}$	EntHunt	ESCC CLK	ROM Emulator Mode (RRME)	Daisy-Chain	
0	0	0	0	0	0	0	

Bit Number	Field	R/W	Reset Value	Description
7	Decode High I/O		0	**Decide High I/O** If this bit is `1`, A15..A8 must all be `0` to access these registers, as well as the other registers in the Z80185/Z80195. If this bit is `0`, A15..A8 are not decoded for those registers for which A7..A6 are `11`; that is, the register is for the ESCC, CTC, I/O Ports, and Bicentronics Controller. 1: This bit allows more extensive offchip I/O. 0: Internal I/O address decoding is compatible with the Z80181, and Z80182, and allows shorter, and more basic I/O instructions to be used to access these registers.
6	Daisy-Chain		0	**Daisy Chain Configuration** This bit is described with Bits 1..0, at end of this section.
5	Disable $\overline{\text{ROMCS}}$		0	**Disable** $\overline{\text{ROMCS}}$ 1: $\overline{\text{ROMCS}}$ is disabled. 0: $\overline{\text{ROMCS}}$ is enabled.

Bit Number	Field	R/W	Reset Value	Description
4	EntHunt		0	**CTS Auto-Enable Function** When this bit is 1 and ASCI0 is used, the CTS auto-enable function must not be enabled. The multiplexing of CKA0 is important only with respect to output; the same external clock may be used for both ASCI0 and CSI0. 1: The pins have the TXS, RXS and CKS functions, and the CSIO facility may be used. 0: The $\overline{\text{RTS0}}$/TXS, $\overline{\text{CTS0}}$/RXS and CKA0/CKS pins have the $\overline{\text{RTS0}}$, $\overline{\text{CTS0}}$, and CKA0 functions, respectively.
3	ESCC CLK		0	**ESCC CLK** 1: ESCC CLK is ϕ/2 0: ESCC CLK is ϕ.
2	RRME		0	**ROM Emulator Mode Enable** 1: Data Bus in RRME. 0: Data Bus Normal Mode.
1..0	Daisy-Chain		0	**Routing Determination** These bits, plus Bit 6, determine the routing of the on-chip interrupt daisy-chain, and thus the relative interrupt priority of the on-chip interrupt sources to the daisy chain as described in the following table.

Bits			
6	1	0	Daisy-Chain Routing
0	0	0	IEI pin => ESCC =>CTC =>Bidirectional Centronics Controller =>IE0 pin
0	0	1	IEI pin => ESCC=> Bidirectional Centronics Controller=>CTC=>IE0 pin
0	1	X	IEI pin =>Bidirectional Centronics Controller=>ESCC=>CTC=>IE0 pin
1	0	0	IEI pin => CTC=>ESCC=>Bidirectional Centronics Controller=>IE0 pin
1	0	1	IEI pin=>CTC=>Bidirectional Centronics Controller=>ESCC=>IE0 pin
1	1	X	IEI pin => Bidirectional Centronics Controller=>CTC=>ESCC=>IE0 pin

On-Chip ROM

The Z80185/Z80195 includes 32,768 bytes of Read-Only Memory, while the Z80195 is a ROMless version. Even with a Z80185/Z80195, if External Logic drives $\overline{\text{WAIT}}$ Low when $\overline{\text{RESET}}$ is Low, on-chip ROM is disabled.

Also, if the contents of the ROM Boundary Register (`ROMBR`), described in the next section, are less than `03`, the latter portion of the on-chip ROM is disabled as follows:

Contents of ROMBR	Portion of On-Chip ROM Enabled
00	First 8K: 00000..01FFF
01	First 16K: 00000..03FFF
10	First 24K: 00000..05FFF

On-chip ROM requires no Wait states, even at the maximum rated clock frequency.

Chip Select Outputs

Z80185/Z80195 includes flexible address decoding logic that controls the $\overline{\text{ROMCS}}$, $\overline{\text{RAMCS}}$ and $\overline{\text{IOCS}}$ Outputs, as well as affecting the amount of on-chip ROM (if any) that is enabled. $\overline{\text{ROMCS}}$, $\overline{\text{RAMCS}}$ and $\overline{\text{IOCS}}$ are Active during cycles by the on-chip processor, on-chip DMA and by external Masters. $\overline{\text{ROMCS}}$ and $\overline{\text{RAMCS}}$ are Active only in cycles in which $\overline{\text{MREQ}}$ is also Active.

ROM Boundary Register (`ROMBR: ECH`)

7		0
	A19..A12 $\overline{\text{ROMBR}}$	
	1	

Bit Number	Field	R/W	Reset Value	Description
7..0	A19..A12 $\overline{\text{ROMBR}}$		1	**Memory Access** The Z80185/Z80195 drives the $\overline{\text{ROMBR}}$ pin Low during a memory access if: - Bit 3 of the System Configuration Register is `0`, and - Bits A19..A12 are less than or equal to the value in `ROMBR`, and - The part is a ROMless Z80195, or - External hardware drove $\overline{\text{WAIT}}$ Low during Reset, or - The address is greater than or equal to the size of on-chip ROM (32K, `08000H`) If the contents of `ROMBR` are less than `03H`, and on-chip ROM is enabled, then the value in `ROMBR` also limits the effective size of on-chip ROM, as described in the previous section.

RAM Upper Boundary Register (RAMUBR: EAH)

7		0
	A19..A12 $\overline{\text{RAMUBR}}$	
	1	

RAM Lower Boundary Register (RAMLBR: EBH)

7		0
	A19..A12 RAMLBR	
	1	

Bit Number	Field	R/W	Reset Value	Description
7..0	RAMUBR and RAMLBR		1	**Memory Access** The Z801x5 drives the $\overline{\text{RAMCS}}$ pin Low during a memory access if: - Bits A19..A12 are greater than the contents of ROMBR, and - Bits A19..A12 are greater than or equal to the contents of RAMLBR and - Bits A19..A12 are less than or equal to the contents of RAMUBR.

$\overline{\text{IOCS}}$

The processor drives this output Low during an I/O cycle if the address does not select an on-chip register. The decoding extends only down through A3. The decoding includes the relocation of Group 1 registers per the I/O Control Register (ICR) and the Decode-High bit in the System Configuration Register.

Decode High	IOA7..A6	Address for which $\overline{\text{IOCS}}$ is Driven Low
0	00	0040..00D7, 00F8..01D7, 01F8..02D7, ..., FEF8..FFD7, FFF8..FFFF
0	01	0000..003F, 0080..00D7, 00F8.. 01D7, 01F8..02D7, ... , FEF8..FF07, FFF8..FFFF
0	10	0000..007F, 00C0..00D7, 00F8..01D7, 01F8..02D7, ..., FEF8..FFD7, FFF8..FFFF.
1	00	0040..00D7, 00F8..FFFF
1	01	0000..003F, 0080..00D7, 00F8..FFFF
1	10	0000..007F, 00C0..00D7, 00F8..FFFF

WAIT STATE GENERATORS

The Z80185/Z80195 includes several on-chip Wait-state generation facilities. The outputs of these facilities and the external $\overline{\text{WAIT}}$ Input are logically ORed (Positive-logic ANDed) to produce the effective $\overline{\text{WAIT}}$ Input of the processor. Thus, the number of Wait states in a cycle is the maximum number requested by any of these sources.

Wait States in I/O Cycles

All accesses to Group 1 registers, those normally in address range 0000..003F including the MMU, DMA, ASCIs, PRT and CSI/O, have no internally-generated Wait states. Unless the $\overline{\text{WAIT}}$ pin is for some reason pulled Low during such cycles, they execute in 3 ϕ cycles.

All access to Group 2 registers, those in address range D8..F1 including the ESCC, parallel ports, Bidirectional Centronics controller, CTCs, chip select and Watch-Dog Timer, and to off-chip I/O devices, have one to four

Wait states inserted depending on the value of Bits 5..4 (IWI1..IWI0) in the DMA Control Register (`DCTRL`):

IWI1	IWI0	Wait States
0	0	1
0	1	2
1	0	3
1	1	4

Wait States in Interrupt Acknowledge Cycles

During the first cycle of an interrupt-acknowledge sequence (the one in which $\overline{\text{MI}}$ is driven Low), for Group 2 devices (the ESCC, Bidirectional Centronics and CTCs) as well as off-chip requests via the $\overline{\text{INT0}}$ pin, two to six Wait states are inserted by the on-chip Wait-state generators, depending on Bits 5..4 (`IWI1..IWI0`) of the DMA Control Register (`DCTRL`):

IWI1	IWI0	Wait States
0	0	2
0	1	4
1	0	5
1	1	6

During the first cycle of a interrupt-acknowledge sequence (the one in which $\overline{\text{MI}}$ is driven Low) for Group 1 devices (PRTs, DMAs, CSI/O and ASCIs) as well as interrupts and off-chip requests via the $\overline{\text{INT1}}$ and $\overline{\text{INT2}}$ pins, two Wait states are inserted by the on-chip Wait-state generators.

During the first cycle of an NMI sequence (in which $\overline{\text{MI}}$ and $\overline{\text{RD}}$ are driven Low), the on-chip Wait-state generators do not insert any Wait states.

Subsequent cycles of interrupt sequences are categorized as memory accesses and the internal Wait-state generators operate as described in the next section.

Wait States in Memory-Space Cycles

A global Wait-state generator for all memory-space cycles (including memory-mapped I/O) is controlled by Bits 7..6 (`MWI1..MWI0`) in the DMA Control Register (`DCTRL`):

MWI1	MWI0	Wait States
0	0	0
0	1	1
1	0	2
1	1	3

These bits reset to `11` to insert 3 Wait states into the initial reset code.

If Bits 7..6 (MWI1..MWI0) are subsequently programmed to `00`, the Wait-State Generator Chip-Select Register (`WSGCS`) allows individual Wait-state generation for four subdivisions of memory space.

WSG Chip Select Register (`WSGCS: D8`)

7	6	5	4	3	2	1	0
$\overline{\text{RAMCS}}$ Wait Insertion		$\overline{\text{ROMCS}}$ Wait Insertion		On-Chip ROM Wait Insertion		Other Memory Wait Insertion	
1		1		1		1	

Bit Number	Field	R/W	Reset Value	Description
7–6	$\overline{\text{RAMCS}}$ Wait Insertion		1	**Wait Insertion** This field controls how many Wait states are inserted for accesses to external memory in which $\overline{\text{RAMCS}}$ is asserted: 00: None 01: 1 10: 2 11: 4 Wait states
5..4	$\overline{\text{ROMCS}}$ Wait Insertion		1	**Wait Insertion** This field controls how many Wait states are inserted for accesses to external memory in which $\overline{\text{ROMCS}}$ is asserted. 00: None 01: 1 10: 2 11: 4 Wait states

Bit Number	Field	R/W	Reset Value	Description
3..2	On-Chip ROM Wait Insertion		1	**Wait Insertion** This field controls how many Wait states are inserted for access to external memory in which neither $\overline{\text{RAMCS}}$ nor $\overline{\text{ROMCS}}$ is asserted. The 4-Wait state feature is included to allow the use of commodity DRAMs with a clock rate at, or near, the maximum. 00: None 01: 1 10: 2 11: 4 Wait states
1..0	Other Memory Wait Insertion		1	**Wait Insertion** This bit is used to initiate CRC calculation at the beginning of the last byte transferred from the Receiver Shift Register to the Receive FIFO. This operation occurs independently of the number of bytes in the Receive FIFO. When a particular byte is to be excluded from the CRC calculation, this bit must be reset before the next byte is transferred to the Receive FIFO. If this feature is used, care must be taken to ensure that eight bits per character is selected in the receiver because of an inherent delay from the Receive Shift Register to the CRC checker. This bit is internally 1 in SDLC Mode and the receiver calculates the CRC on all bits except 0s inserted between the opening and closing flags. This bit is ignored in asynchronous modes. 00: None 01: 1 10: 2 11: 4 Wait states

HALT AND LOW-POWER OPERATING MODES

The Z801x5 may operate in 7 modes with respect to activity and power consumption:

- Normal Operation
- Halt Mode
- IOStop Mode
- Sleep Mode
- System Stop Mode
- Idle Mode
- Standby Mode (with or without Quick Recovery)

Normal Operation

The Z80185/Z80195 processor is fetching and running a program. All enabled functions and portions of the device are Active, and the HALT pin is High.

Halt Mode

This mode is entered by the `Halt` instruction. Thereafter, the Z80185/Z80195 processor continually fetches the following opcode but does not execute it, and drives the HALT, ST and MI pins all Low. The oscillator and PHI pin remain Active, as well as interrupts and bus granting to external Masters, and DRAM refresh may occur and all on-chip I/O devices continue to operate including the DMA Channels.

The Z80185/Z80195 leaves Halt Mode in response to a Low on RESET, or to an interrupt from an enabled on-chip source, an external request on NMI, or an enabled external request on INT0, INT1, or INT2. In case of an interrupt, the return address is the instruction following the `Halt` instruction; at that point the program may either branch back to the `Halt` instruction to Wait for another interrupt, or may examine the new state of the system/application and respond appropriately. Refer to Figure 17 in the front of this Chapter.

Sleep Mode

This mode is entered by keeping the IOStop Bit 5 (`ICR5`) and CPU Control Register Bit 3 (`CCR3`) and Bit 6 (`CCR6`) all `0` and executing the `SLP` instruction. The oscillator and ϕ Output continue operating, but are blocked from the CPU core and DMA Channels to reduce power consumption. DRAM refresh stops but interrupts and granting to external Masters may occur. When the bus is granted to an external Master, A19..A0 and all control signals except $\overline{\text{HALT}}$ are maintained High. $\overline{\text{HALT}}$ is Low. I/O operations continue as before the `SLP` instruction, except for the DMA Channels.

The Z80185/Z80195 leaves Sleep Mode in response to a Low on $\overline{\text{RESET}}$, an interrupt request from an
on-chip source, an external request on $\overline{\text{NMI}}$, or an external request on $\overline{\text{INT0}}$, $\overline{\text{INT1}}$, or $\overline{\text{INT2}}$.

If an interrupt source is individually disabled, it cannot bring the Z80185/Z80195 out of Sleep Mode. If an interrupt source is individually enabled, and the `IEF` bit is `1` so that interrupts are globally enabled (by an `EI` instruction), the highest priority Active interrupt occurs, with the return address being the instruction after the `SLP` instruction. If an interrupt source is individually enabled, but the `IEF` bit is `0` so that interrupts are globally disabled (by a `DI` instruction), the Z80185/Z80195 leaves Sleep Mode by simply executing the following instruction(s).

This provides a technique for synchronization with High-speed external events without incurring the latency imposed by an interrupt-response sequence.

Refer to Figure 18 in the front of this Chapter. This Figure describes the timing for exiting Sleep Mode due to an interrupt request. The Z80185/ Z80195 takes about 1.5 clocks to restart.

IOStop Mode

IOStop Mode is entered by setting the IOStop bit of the I/O Control Register (`ICR`) to `1`. In this case, on-chip I/O (ASCI, CSI/O, PRT) stops operating but the CPU continues to operate. Recovery from IOStop Mode is by resetting the IOStop bit in `ICR` to `0`.

System Stop Mode

System Stop Mode is the combination of Sleep and IOStop Modes. System Stop Mode is entered by setting the IOStop bit in `ICR` to `1` followed by execution of the `SLP` instruction. In this mode, on-chip I/O and CPU stop operating, reducing power consumption, but the ϕ Output continues to operate. Recovery from System Stop Mode is the same as recovery from Sleep Mode except that internal I/O sources (not clocked due to IOStop) may only generate recovery interrupts that are combinatorial (not dependent on clocking).

Idle Mode

Software may put the Z80185/Z80195 into this mode by setting the IOStop Bit 5 (`ICR5`) to `1`, Bit 6 (`CCR6)` to `0`, Bit 3 (`CCR3)`to `1` and executing the `SLP` instruction. The oscillator keeps operating but its output is blocked to all circuitry including the ϕ pin. DRAM refresh and all internal devices stop, but external interrupts may occur. Bus granting to external Masters may occur if the BREXT bit in the CPU Control Register (`CCR5`) is set to `1` before Idle Mode is entered.

The Z80185/Z80195 leaves Idle Mode in response to a Low on $\overline{\text{RESET}}$, an external-interrupt request on $\overline{\text{NMI}}$, or an external-interrupt request on $\overline{\text{INT0}}$, $\overline{\text{INT1}}$ or $\overline{\text{INT2}}$ that is enabled in the INT/TRAP Control Register.

As described above for Sleep Mode, when the Z80185/Z80195 leaves Idle Mode due to an $\overline{\text{NMI}}$, or due to an enabled external-interrupt request when the IEF Flag is 1 due to an EI instruction, it starts by performing the interrupt with the return address being that of the instruction after the SLP instruction.

If the Z80185/Z80195 leaves Idle Mode due to an external-interrupt request that is enabled in the INT/TRAP Control Register but the IEF1 bit is 0 due to a DI instruction, the processor restarts by executing the instruction(s) following the SLP instruction.

Refer to Figure 19 in the front of this Chapter. The Z80185/Z80195 takes about 9.5 clocks to restart.

While the Z80185/Z80195 is in Idle Mode, it grants the bus to an external Master if Bit 5 (BREXT) in CCR is 1. Refer to Figure 20 in the front of this Chapter. The processor takes 8 clock cycles longer to respond to the Bus Request than in normal operation.

After the external Master negates the bus request, the Z80185/Z80195 disables the ϕ clock and remains in IDLE Mode.

Standby Mode With or Without Quick Recovery

Software may put the Z80185/Z80195 into this mode by setting the Bit 5 (IOStop) of the ICR register to 1, Bit 6 (CCR) to 1, and executing the SLP instruction. This mode stops the on-chip oscillator and thus draws the least power of any mode, less than 10 µA.

As with Idle Mode, the Z80185/Z80195 leaves Standby Mode in response to a Low on $\overline{\text{RESET}}$ or on $\overline{\text{NMI}}$, or a Low on $\overline{\text{INT0}}$-2 that is enabled by a 1 in the corresponding bit in the INT/TRAP Control Register, and grants the bus to an external Master if Bit 5 (BREXT) in the CPU Control Register (CCR) is 1. But the time required to restart is greatly increased by the need to restart the on-chip oscillator and ensure that it has stabilized to square-wave operation.

When an external clock is connected to the EXTAL pin rather than a crystal to the XTAL and EXTAL pins, and the external clock runs continuously, there is little need to use Standby Mode because there is no time required to restart the oscillator, and other modes restart faster. However, if external logic stops the clock during Standby Mode (that is, by decoding $\overline{\text{HALT}}$ Low and $\overline{\text{MI}}$ High for several clock cycles), then Standby Mode may be useful to allow the external clock source to stabilize after it is re-enabled.

When external logic drives $\overline{\text{RESET}}$ Low to being a Z80185/Z80195 out of Standby Mode, and a crystal is used or an external clock source has been stopped, the external logic must hold $\overline{\text{RESET}}$ Low until the on-chip oscillator or external clock source has restarted and stabilized.

The clock stability requirements of the Z80185/Z80195 are much less in the *Divide-By-Two* Mode that is selected by a Reset sequence and thereafter controlled by Bit7 (Clock Divide) in the CPU Control Register (CCR). Because of this, software must:

1. Program Bit 7 to 0 to select *Divide-By-Two* Mode, before the SLP instruction that enters Standby Mode.
2. After a Reset, interrupt or in-line restart after the SLP 01 instruction, delay programming CCR
   Bit 7 back to 1 to set Divide-By-One Mode, as long as possible to allow additional
   clock-stabilization time.

When software programs Bit 6 to 1 before a SLP instruction that enters Standby Mode, the value that it writes to the Bit 3 determines how long the processor waits for oscillator restart and stabilization, when it leaves Standby Mode because of an external-interrupt request. If Bit 3 is 0, the processor waits 2^{17} (131,072) clock cycles, while Bit 3 is 1, it waits only 64 clock cycles. The latter is called Quick Recovery Mode. The same delay applies to granting the bus to an external Master during Standby Mode, when Bit 5 (BREXT) is 1.

As described previously for Sleep and Idle Modes, when a Z801x5 leaves Standby Mode due to $\overline{\text{NMI}}$ Low, or due to an enabled $\overline{\text{INT0}}$..$\overline{\text{INT2}}$ Low when the IEF1 Flag is `1` due to an `IE` instruction, it starts by performing the interrupt with the return address being that of the instruction following the `SLP` instruction. If the processor leaves Standby Mode due to an external-interrupt request that is enabled in the INT/TRAP Control Register, but the `IEF1`, bit is `0` due to a `DI` instruction, the processor restarts by executing the instruction(s) following the `SLP` instruction. If $\overline{\text{INT0}}$, $\overline{\text{INT1}}$, or $\overline{\text{INT2}}$ goes Inactive before the end of the clock stabilization delay, the processor stays in Standby Mode.

Refer to Figure 21 in the front of this Chapter. The Z80185/Z80195 takes either 64 or 2^{17} (131,072) clocks to restart, depending on Bit 3 of the `CCR` register.

While the Z80185/Z80195 is in Standby Mode, it grants the bus to an external Master if Bit 5 (BREXT) of `CCR` is `1`. Refer to Figure 22 in the front of this Chapter. The part takes 64 or 2^{17} (131,072) clock cycles to grant the bus depending on the `CCR` Bit 3.

The latter (non-Quick-Recovery) case may be prohibitive for many *demand driven* external Masters. If so, Quick Recovery or Idle Mode may be used.

TRAPS AND INTERRUPTS

The Z80185/Z80195 has 12 priority levels for interrupts. As depicted in Figure 32, eight of them are assigned to specific on-chip sources. Three levels are assigned to the $\overline{\text{NMI}}$, $\overline{\text{INT1}}$ and $\overline{\text{INT2}}$ pins, each of which may be connected to a specific external-interrupt source or shared among multiple sources. In the latter case, identification of the source of an interrupt is left to software polling.

The twelfth interrupt level is shared among several on-chip interrupt sources (the ESCC, CTCs, and the Bidirectional Centronics controller) and the $\overline{\text{INT0}}$ pin. $\overline{\text{INT0}}$ may be connected to a specific external interrupt source or shared among multiple sources. The on-chip logic associated with this interrupt level allows for the automatic identification of the highest-priority interrupting device within the level, by means of an interrupt-acknowledge daisy-chain.

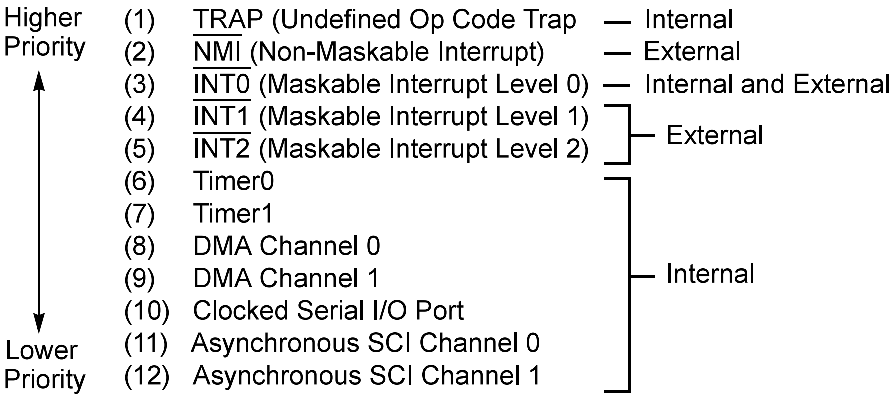

Higher Priority
(1) $\overline{\text{TRAP}}$ (Undefined Op Code Trap — Internal
(2) $\overline{\text{NMI}}$ (Non-Maskable Interrupt) — External
(3) $\overline{\text{INT0}}$ (Maskable Interrupt Level 0) — Internal and External
(4) $\overline{\text{INT1}}$ (Maskable Interrupt Level 1)
(5) INT2 (Maskable Interrupt Level 2)
External
(6) Timer0
(7) Timer1
(8) DMA Channel 0
(9) DMA Channel 1
(10) Clocked Serial I/O Port
(11) Asynchronous SCI Channel 0
(12) Asynchronous SCI Channel 1
Internal
Lower Priority

Figure 32. Interrupt Levels

INT/TRAP Control Register

This register is used in handling TRAP interrupts and to enable or disable Maskable Interrupt Level 0 and the $\overline{\text{INT1}}$ and $\overline{\text{INT2}}$ pins.

Int/Trap Control Register (`ITC: 34H`)

7	6	5	3	2	1	0
TRAP	UFO	Reserved		ITE2	ITE1	ITE0
0	0			0	0	1
R/W	R			R/W	R/W	R/W

Bit Number	Field	R/W	Reset Value	Description
7	TRAP	R/W	0	**Undefined Op Code** 1: Bit 7 (`TRAP`) is set when an undefined Op Code is fetched. TRAP may be reset under program control by writing it with a `0`, however, it cannot be written with `1` under program control.
6	UFO	R	0	**Undefined Fetch Object** When a TRAP interrupt occurs the contents of UFO allow determination of the starting address of the undefined instruction. This is necessary because the TRAP may occur on either the second or third byte of the Op Code. UFO allows the stacked `PC` value to be correctly adjusted. UFO is Read-Only. 1: The first Op Code address is the stacked `PC-2`. 0: The first Op Code is at the stacked `PC-1`.
5..3	Reserved		0	**Reserved** Must be `0`.

Bit Number	Field	R/W	Reset Value	Description
2..0	ITE2.. ITE0	R/W	0 1	**Interrupt Enable** ITE2 and ITE1 enable and disable the External-Interrupt Inputs $\overline{\text{INT2}}$ and $\overline{\text{INT1}}$, respectively. ITE0 enables and disables interrupts from the on-chip ESCC, CTCs and Bidirectional Centronics controller as well as the External-Interrupt Input $\overline{\text{INT0}}$. 1: Enables the corresponding interrupt level. 0: Disables it. Reset sets ITE0 to 1 and clears ITE1 and ITE2 to 0.

TRAP Interrupt

The Z80185/Z80195 generates a non-maskable (not affected by the state of IEF1) TRAP interrupt when an undefined Op Code fetch occurs. This feature may be used to increase software reliability, implement an extended instruction set, or both. TRAP may occur during Op Code fetch cycles and also if an undefined Op Code is fetched during the interrupt-acknowledge cycle for $\overline{\text{INT0}}$ when Mode 0 is used.

When a TRAP interrupt occurs, the Z80185/Z80195 operates as follows:

1. The TRAP bit in the Interrupt TRAP/Control (ITC) Register is set to 1.
2. The current Program Counter (PC) value, reflecting the location of the undefined Op Code, is saved on the stack.
3. The Z80180 restarts execution at logical address 0. If logical address 0000H is mapped to physical address 00000H, the vector is the same as for Reset and for the RST 0 instruction. In this case, testing the TRAP bit in ITC reveals whether the restart at physical address 00000H was caused by TRAP.

TRAP interrupts occur after fetching an undefined second Op Code byte following one of the prefix Op Codes CBH, DDH, EDH, or FDH, or after fetching an undefined third-opcode byte following one of the double prefix Op Codes DDCBH or FDCBH. Refer to Figure 23 and Figure 24 in the front of this Chapter.

The state of Bit 6 (UFO) in ITC allows TRAP software to correctly adjust the stacked PC, depending on whether the second or third byte of the opcode generated the TRAP. If UFO is 0, the starting address of the invalid instruction is equal to the stacked PC-1. If UFO is 1, the starting address of the invalid instruction is equal to the stacked PC-2.

Interrupt Enabling and Disabling

All of the interrupt levels except TRAP and NMI are subject to global enabling and disabling by means of the EI and DI instructions which control two internal bits called IEF1 and IEF2.

IEF1 controls the overall enabling and disabling of all internal and external maskable interrupts (in other words, all interrupts except $\overline{\text{NMI}}$ and TRAP).

If IEF1 is 0, all maskable interrupts are disabled. IEF1 may be reset to 0 by the DI instruction and set to 1 by the EI instruction.

The purpose of IEF2 is to correctly manage the occurrence of NMI. During NMI, the prior state is saved by copying the state of IEF1 into IEF2 and maskable interrupts are then disabled by clearing IEF1 to 0.

At the end of the NMI interrupt-service routine, execution of the RETN instruction automatically restores the interrupt-receiving state (by copying IEF2 to IEF1) prior to the occurrence of NMI.

The state of IEF2 may be copied to the P/V bit of the CPU Status Register by executing an LD A, I or
LD A, R instruction.

The following table summarizes the relationship between various operations and the IEF1 and IEF2 Flags.

CPU Operation	IEF1	IEF2	Remarks
Reset	0	0	Inhibits all interrupts except $\overline{\text{NMI}}$ and TRAP
NMI	0	IEF1	Copies the contents of IEF1 to IEF2.
RETN	IEF2	Not Affected	
Interrupt Except $\overline{\text{NMI}}$ and TRAP	0	0	Inhibits all interrupts except $\overline{\text{NMI}}$ and TRAP.
RETI	Not Affected	Not Affected	
TRAP	Not Affected	Not Affected	
EI	1	1	
DI	0	0	
LD A, I	Not Affected	Not Affected	Copies the contents of IEF2 to the P/V Flag.
LD A, R	Not Affected	Not Affected	Copies the contents of IEF2 to the P/V Flag.

In addition to the global interrupt enabling and disabling afforded by the `EI` and `DI` instructions and the IEF1 and IEF2 bits, each interrupt source in the Z80185/Z80195 (other than TRAP and $\overline{\text{NMI}}$) has its own individual enabling/disabling mechanism which is described in later sections.

$\overline{\text{NMI}}$ Non-Maskable Interrupt

The $\overline{\text{NMI}}$ Interrupt Input is Edge sensitive and cannot be masked by software. When $\overline{\text{NMI}}$ is detected, the Z80185/Z80195 operates as follows.

1. DMA Channel operation is suspended by clearing Bit 0 (DME) in the DSTAT register.
2. The `PC` is pushed onto the stack.

3. The contents of IEF1 are copied to IEF2. This saves the interrupt reception state that existed prior to $\overline{\text{NMI}}$.
4. IEF1 is cleared to `0`. This disables all external and internal maskable interrupts (in other words, all interrupts except $\overline{\text{NMI}}$ and TRAP).
5. Execution commences at logical address `0066H`.

The `RETN` instruction provides a convenient way to return from a non-maskable interrupt if the software has not done any `EI` or `DI` instructions since the NMI occurred. It copies the saved state of IEF2 to IEF1 and restores the return address from the stack.

Alternatively, the NMI software may sense the state of IEF2 by performing an `LD A, I` or `LD A, R` instruction, which copies IEF2 to the P/V Flag, branching on P/V, and setting a status bit in memory accordingly. The software may then perform `EI` and `DI` instructions as needed. When it is time to return from the NMI, the software may test the status bit in memory and use either an `IE-RET` or `DI-RET` sequence to return to the interrupted process.

An NMI service routine may protect against multiple edges on the $\overline{\text{NMI}}$ line (such as, contact bounce on a push-button) by maintaining an in NMI status bit in memory. One of the first things the service routine must do is to test this status bit, and perform an immediate `RETN` if the bit is set. If not, it sets the bit. When it is time to return from the NMI, the software clears the bit.

$\overline{\text{NMI}}$ is Edge sensitive. A Falling edge sets an internal latch that remains set until the interrupt occurs. This latch is sampled by the Falling edge of ϕ in the second-last clock cycle of each instruction. If the latch has been set by the time of that Falling edge, the non-maskable interrupt occurs at the end of the instruction. Refer to Figure 25 in the front of this Chapter.

Maskable Interrupt Level 0

This is the next highest priority interrupt level after $\overline{\text{NMI}}$ and is shared by the on-chip ESCC, CTCs,
Bidirectional Centronics interface, and the $\overline{\text{INT0}}$ pin. The logical OR (positive logic AND) of these four requests is sampled at the Falling edge of ϕ in the second-last clock cycle of the execution of most instructions. (Certain instructions such as `EI` do not allow an interrupt after them.) If the composite request is Low at a Falling edge, and Bit 0 (ITE0) in the INT/TRAP Control Register (`ITC`) is `1`, and the `IEF1`, bit is `1` (due to an `EI` instruction) then the Z80185/Z80195 performs an interrupt after the instruction.

The type of interrupt processing the Z80185/Z80195 performs for an interrupt at this level is controlled by the Interrupt Mode (IM) instruction. There are three cases called Modes 0, 1, and 2.

Level 0 Mode 0 Interrupts

After a Reset and/or an IM 0 instruction, the Z80185/Z80195 begins an interrupt sequence by clearing the IEF1 and IEF2 bits to disable further interrupts, and fetching an Op Code byte from the D7..D0 lines using a special cycle in which it drives $\overline{\text{MI}}$ and $\overline{\text{IORQ}}$ both Low. Refer to Figure 26 in the front of this Chapter.

Often this instruction is one of the eight single-byte `RST` (Restart) instructions which stack the `PC` and restart execution at one of the Fixed-Logical Addresses `0`, `8`, ..., `38H`. However, multibyte instructions may be processed if the device providing the instruction may provide such a multibyte sequence. Unlike all other interrupts, the Z80185/Z80195 does not automatically stack the `PC` in this mode.

A TRAP interrupt occurs if an invalid multibyte instruction is provided in this mode.

This mode may be supported by the ESCC, CTCs, Bidirectional Centronics interface and other devices that are designed to provide an interrupt vector in Mode 2, by programming the Interrupt Vector Register with the op- code for one of the `RST` instructions. Devices that may only supply an even vector value is limited to
`RST 0`, `10H`, `20H` and `30H`.

Level 0 Mode 1 Interrupts

After an `IM 1` instruction, the Z80185/Z80195 performs a Level 0 interrupt by clearing IEF1 and IEF2, stacking the `PC`, and beginning execution at logical address `0038H`. In effect this is like a Level 0 Mode 0 interrupt, in which the responding device provides an `RST 38H` instruction (`FFH`). Refer to Figure 27 in the front of this Chapter.

Before describing Level 0 Mode 2 it is helpful to describe the `I` Register.

The I Register and LD A, I Instructions

The Z80185/Z80195 includes an internal I Register that is used during Level 0 Mode 2 interrupts, $\overline{\text{INT1}}$ and $\overline{\text{INT2}}$ interrupts and interrupts from the on-chip DMAs, ASCIs, PRTs and CSI/0. Rather than having an address in I/O space like most registers in the Z80185/Z80195, the `I` Register is loaded by the special
instruction `LD I, A`, and may be read back by the `LD A, I` instruction.

During the interrupts noted above, the Z80185/Z80195 places the contents of `I` on the A15..A8 lines while it fetches a two-byte starting address of the interrupt service routine from memory.

Level 0 Mode 2 Interrupts

After an IM 2 instruction, the Z801x5 starts a Level 0 interrupt by clearing the IEF1 and IEF2 bits to disable further maskable interrupts, and performing a special interrupt-acknowledge cycle in which it drives $\overline{\text{MI}}$ and $\overline{\text{IORQ}}$ both Low. Refer to Figure 28 in the front of this Chapter.

Through this point, operation is the same as Mode 0. But rather than treating the byte supplied by the interrupting device as an instruction Op Code as in Mode 0, in Mode 2 the Z80185/Z80195 treats the value from the device as an *interrupt vector.* (The value from the device must have its units Bit (D0) in this mode.)

The Z80185/Z80195 automatically stacks the return address from PC, and thereafter fetches two bytes from memory. It fetches the first byte from the logical address formed by using the contents of the I Register as the most-significant byte (A15..A8) and the vector value obtained from the device as the least-significant byte (A7..A0). It treats this first byte as the least-significant byte of the starting address of the interrupt-service routine (ISR). It then fetches a second byte from the next higher address, which it treats as the most-significant byte of the starting address of the ISR. Finally, it begins executing the instruction(s) at the address it has fetched from memory.

Figure 33 depicts the process of fetching the vector for Level 0 Mode 2 Interrupts.

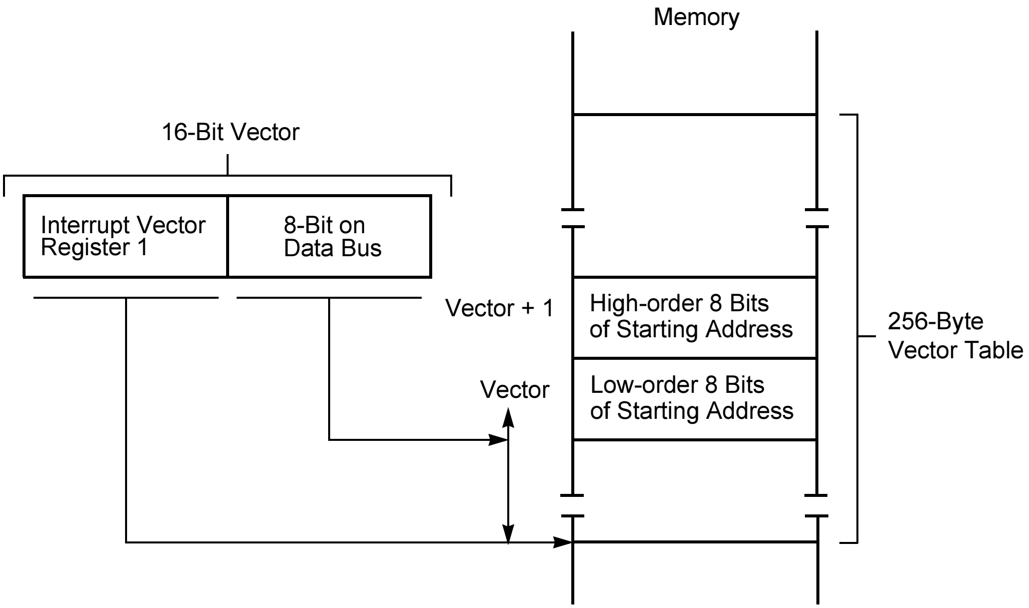

Memory
16-Bit Vector
Interrupt Vector Register 1
8-Bit on Data Bus
Vector + 1
High-order 8 Bits of Starting Address
Vector
Low-order 8 Bits of Starting Address
256-Byte Vector Table

Figure 33. Level 0 Mode 2 Vector Acquisition

Interrupt Vector Low Registers

Bits 7..5 of IL are used as Bits 7..5 of the synthesized interrupt vector during interrupts for the $\overline{\text{INT1}}$ and $\overline{\text{INT2}}$ pins and for the DMAs, ASCIs, PRTs, and CSI/O. These three bits are cleared to 0 during Reset.

Interrupt Vector Low Register (IL: 33H)

7	6	5	4 0
IL7	IL6	IL5	Reserved
0	0	0	
R/W	R/W	R/W	

Bit Number	Field	R/W	Reset Value	Description
7..5	IL7.. IL5	R/W	0	**Programmable**.
4..0	Reserved		0	**Reserved** Must be 0.

Interrupt Edge Register

Interrupt Edge Register (DF)

7	6	5	4	3	2	1	0
$\overline{\text{INT2 Sense/Unlatch}}$		$\overline{\text{INT1 Mode Select}}$		$\overline{\text{INT2}}$ $\overline{\text{Mode}}$ $\overline{\text{Select}}$	$\overline{\text{INT1}}$ $\overline{\text{Sense/}}$ $\overline{\text{Unlatch}}$	Drive Select	$\overline{\text{DCD0}}$/ CKA0

Bit Number	Field	R/W	Reset Value	Description
7..6	$\overline{\text{INT2}}$ $\overline{\text{Sense/}}$ $\overline{\text{Unlatch}}$			**Sense/Unlatch** These bits control the interrupt capture logic for the $\overline{\text{INT2}}$ pin. 0X: The $\overline{\text{INT2}}$ pin is Level sensitive and Low Active. 10: Negative edge detection is enabled. Any Falling edge latches an Active Low on the internal $\overline{\text{INT2}}$ to the processor. This interrupt must be cleared by writing a `1` to Bit 3 ($\overline{\text{INT2}}$) of this register. 11: Enables Rising edge interrupts to be latched. The latch must be cleared in the same fashion as for a Falling edge.

Bit Number	Field	R/W	Reset Value	Description
5..4	$\overline{\text{INT1}}$ Mode Select			**Mode Select** These bits control the interrupt-capture logic for the external $\overline{\text{INT1}}$ pin. 0X: The $\overline{\text{INT1}}$ pin is Level sensitive and Low Active. 10: Negative edge detection is enabled. Any Falling edge latches an Active Low on the internal $\overline{\text{INT1}}$ to the processor. This interrupt must be cleared by writing a `1` to Bit 2 ($\overline{\text{INT1}}$)of this register. Programming these bits to `11` enables Rising-edge interrupt to be latched. The latch must be cleared in the same fashion as for a Falling edge.
3	$\overline{\text{INT2}}$ Mode Select			**Sense/Unlatch** Software may read this register to sense the state of the $\overline{\text{INT2}}$ pin. 1 in: Clears the edge detection logic for $\overline{\text{INT2}}$. 0 in: INT2 is Low 1 out: Unlatch Edge Detect 0 out: No operation
2	$\overline{\text{INT1}}$ Sense/ Unlatch			**Sense/Unlatch** Software may read this register to sense the state of the $\overline{\text{INT1}}$ pin. 1 in: Clears the Edge detection logic for INT1. 0 in: $\overline{\text{INT1}}$ is Low 1 out: Unlatch Edge Detect 0 out: No operation

Bit Number	Field	R/W	Reset Value	Description
1	Drive Select			**Low-Noise Option** This bit selects Low noise or normal drive for the following parallel port, bidirectional-Centronics controller, Chip Select, and ESCC pins: PIA 10..13 $\overline{\text{RTS}}$ nFault PIA 14..16/ZCT0 0..2 $\overline{\text{DTR}}$ nlnit PIA 27..20 TXD nSelectln $\overline{\text{ROMCS}}$ $\overline{\text{TRXC}}$ nStrobe $\overline{\text{RAMCS}}$ BUSY PError $\overline{\text{IOCS}}$ nAck Select IEO nAutoFd 1: Selects the Low-noise option, which is a 33% reduction in drive capability. 0: Selects normal drive, and is the default after power-up. Refer to the CPU Register (CCR) for a list of the pins that are programmable for Low drive, by means of the CCR Register.
0	$\overline{\text{DCD0}}$/ CKA0			**CKA1 Function** The $\overline{\text{DCD0}}$/CKA1 pin has the CKA1 function. The pin is always connected to the DCD Input of ASCI0, so if this pin is 1, and ASCI0 is used, it must not be programmed to use DCD as a receive auto-enable. 1: $\overline{\text{DCD0}}$/CKA0 is $\overline{\text{DCD0}}$ 0: $\overline{\text{DCD0}}$/CKA0 is CKA0

$\overline{\text{INT1}}$ and $\overline{\text{INT2}}$ Interrupts

These two pins may be programmed for Level-sensitive operation like $\overline{\text{INT0}}$, or for Edge-sensitive operation, as described in the preceding section on the Interrupt Edge Register. Interrupt on Rising or Falling edges may be selected and may be switched dynamically by software. When an Active edge has been detected, software must clear the Edge-detection hardware by writing to the Interrupt Edge Register.

The level of $\overline{\text{INT1}}$ and $\overline{\text{INT2}}$, or the Edge detection latches for these pins, are sampled at the Falling of ϕ in the second-last clock cycle of most instructions. (Certain instructions such as `EI` do not allow an interrupt after them.) If Bit 1 or 2 in the INT/TRAP Control Register (`ITC`) is `1`, and IEF1 `is 1` (due to an `EI` instruction), then the Z80185/Z80195 performs an interrupt after the instruction.

During such an interrupt, the Z80185/Z80195 stacks the return address from `PC` and then fetches two bytes from memory at addresses having the contents of the I Register as A15..A8, bits 7..5 of the `IL` Register as A7..A5, and `00000` followed by `00001` on A4..A0 for $\overline{\text{INT1}}$, or `00010` followed by `00011` on A4..A0 for $\overline{\text{INT2}}$. It treats the first byte as the least-significant byte of the logical address of an interrupt service routine, and the second byte as the most-significant byte and begins normal instruction execution at that address.

The same sequence is followed for interrupts for the DMAs, ASCIs, PRTs and CSI/0, differing only in the value on A4..A1. Refer to Figure 29 in the front of this Chapter.

Figure 34 depicts the fetching of the interrupt-service routine address from memory.

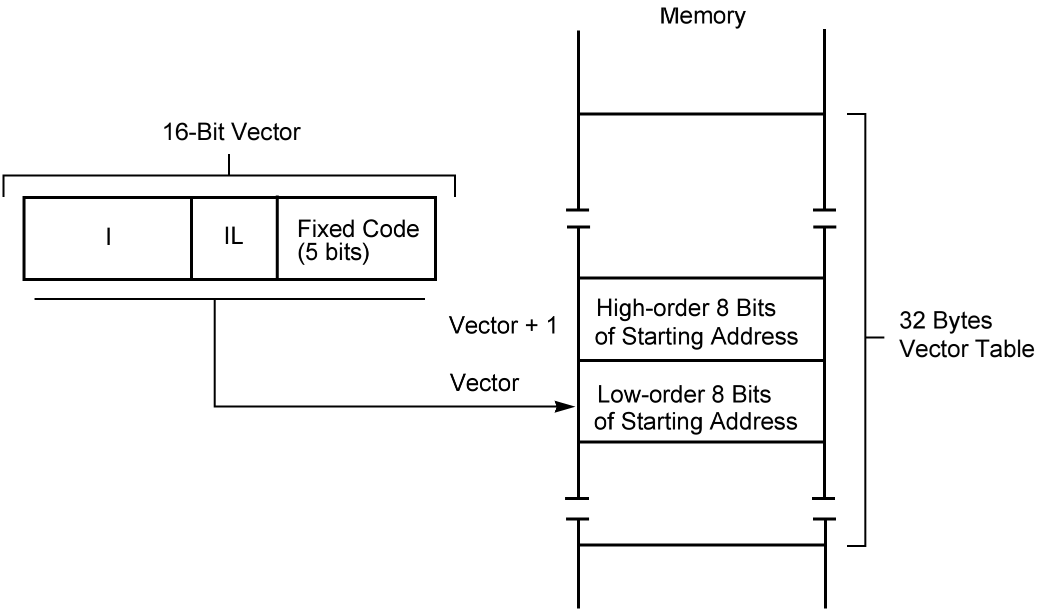


Figure 34. **$\overline{\text{INT1}}$, $\overline{\text{INT2}}$, DMA, ASCI, PRT and CSI/0 Interrupts**

DMA, ASCI, PRT and CSI/0 Interrupts

Each of these devices includes one or more interrupt conditions or sources which may be individually enabled or armed to interrupt the Z80185/Z80195 as described in the later sections about these devices. The processor samples enabled-request signals from each device, at the Falling edge of ϕ in the second-last clock cycle of most instructions. (Certain instructions such as `EI` do not permit interrupts after them.) If an *enable request* line is sampled as *asserted*, and IEF1, is `1` (due to an `EI` instruction), then the Z80185/Z80195 performs an interrupt after the instruction.

When the Z80185/Z80195 performs an interrupt, it does so as appropriate for the highest-requesting priority level among the 12 levels described at the start of this Chapter. If the highest-requesting level is a DMA, ASCI, PRT or CSI/O, the Z80185/Z80195 performs the interrupt exactly as described above for the $\overline{\text{INT1}}$ and $\overline{\text{INT2}}$ interrupts, except that the code on A4..A0 differs according to the highest-priority requesting device, as described in the following table.

		IL			**Fixed Code**				
Interrupt Source	**Priority**	**A7**	**A6**	**A5**	**A4**	**A3**	**A2**	**A1**	**A0**
INT1	Highest	IL7	IL6	IL5	0	0	0	0	0then 1
INT2		IL7	IL6	IL5	0	0	0	1	0 then 1
PRT Channel 0		IL7	IL6	IL5	0	0	1	0	0 then 1
PRT Channel 1		IL7	IL6	IL5	0	0	1	1	0 then 1
DMA Channel 0		IL7	IL6	IL5	0	1	0	0	0 then 1
DMA Channel 1		IL7	IL6	IL5	0	1	0	1	0 then 1
CSI/O		IL7	IL6	IL5	0	1	1	0	0 then 1
ASCI Channel 0		IL7	IL6	IL5	0	1	1	1	0 then 1
ASCI Channel 1	Lowest	IL7	IL6	IL5	1	0	0	0	0 then 1

The interrupt-service routine for one of these devices takes appropriate action for the current conditions, including register accesses to the device that typically causes it to negate its enabled-request signal, before re-enabling interrupts and returning to the interrupted process. These actions and register accesses are described later, in the sections about the devices.

The RETI Instruction

The original ZiLOG Z80 peripheral chips (PIO, SIO, CTC and DMA) include a special circuit that monitors the instruction stream being fetched by a Z80 processor and recognizes the special instruction `RETI`. To the processor, `RETI` operates exactly like `RET`, but to Z80 peripherals, it has the additional meaning that a peripheral that has its Interrupt Under Service (`IUS`) bit set and its IEI

Input asserted, must clear its IUS bit because its interrupt-service routine has concluded.

The CTCs in the Z80185/Z80195 are compatible with the original Z80-CTC and interrupt-service routines for the CTCs must end with RETI instructions, as must ISRs for any external Z80-PIO, SIO, CTC or DMA. In ISRs for other on-chip or external devices, RETI is no different from RET and the latter is preferred because it is shorter and faster. RETI does not re-enable interrupts and must be preceded by an EI instruction just like RET.

The IUS mechanism is intended to allow nested interrupts, wherein a higher-priority device may interrupt the interrupt-service routine for a lower-priority device, the priority being determined by the IEO-IEI daisy-chain. More recent daisy-chainable devices like the ESCC and Bidirectional Centronics controller include explicit means to clear their IUS bits and do not recognize RETI. Non-daisy-chainable devices like the DMAs, ASCIs, PRTs and CSI/O do not support nested interrupts at all, and interrupt-service routines for such devices typically run to completion without being interrupted. They end with an EI directly followed by an RET instruction.

The Z80185/Z80195 processor performs an RETI instruction in two different ways depending on Bit 7 (MIE) in the Operating Mode Control Register (OMCR) as described in the “Z80 versus 64180 Compatibility” section.

MEMORY MANAGEMENT UNIT

The processor includes an on-chip MMU which performs the translation of the CPU 64 KB (16-bit addresses 0000H to FFFFH) logical-memory address space into a 1024 KB (20-bit addresses 00000H to FFFFFH) physical memory address space. Address translation occurs internally in parallel with other CPU operation.

Logical Address Spaces

The 64 KB CPU logical-address space is interpreted by the MMU as consisting of up to three separate logical address areas, Common Area 0, Bank Area, and Common Area 1.

As depicted in Figure 35, a variety of logical-memory configurations are possible. The boundaries between the Common and Bank Areas may be programmed with 4 KB resolution.

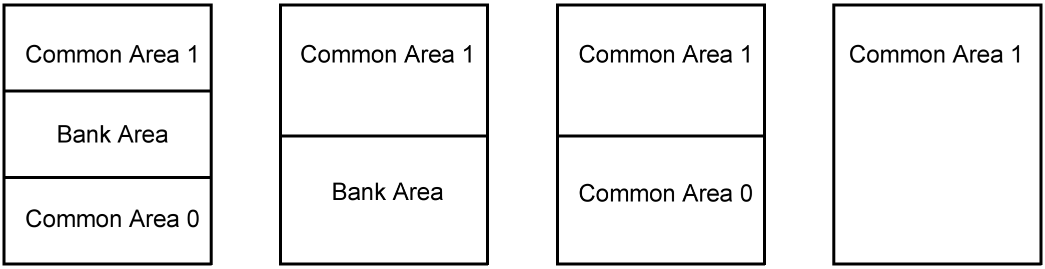


Figure 35. Logical Address Mapping Examples

Logical to Physical Address Translation

Figure 36 describes an example in which the three logical-address space portions are mapped into a 1024 KB physical-address space. The important points to note are that Common and Bank Areas may overlap and that Common Area 1 and Bank Area may be freely relocated (on 4 KB physical-address boundaries). Common Area 0 (if it exists) is always based at physical address `00000H`.

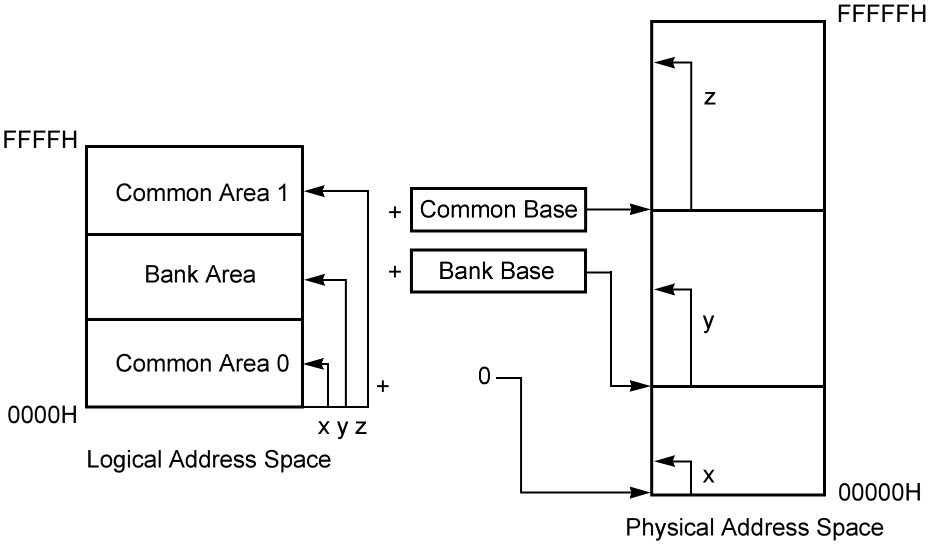


Figure 36. Physical Address Translation

MMU Block Diagram

The MMU block diagram is illustrated in Figure 37. The MMU translates internal 16-bit logical addresses to external 20-bit physical addresses.

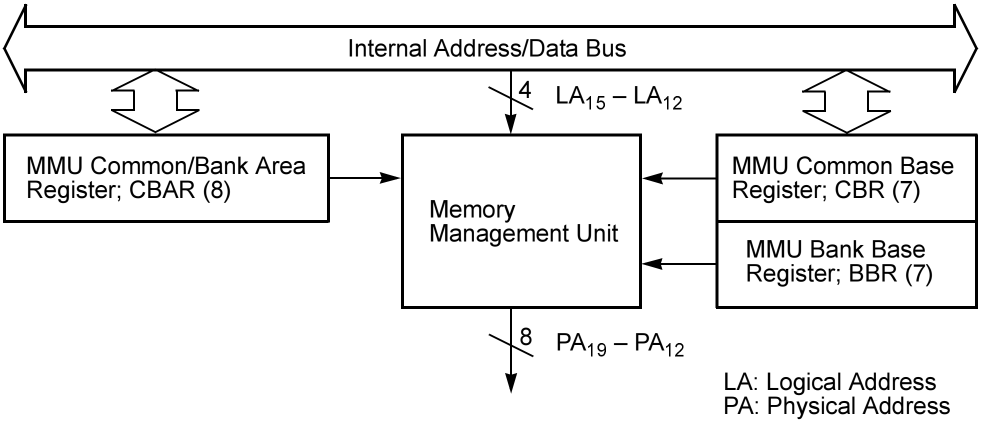

Internal Address/Data Bus
4
$LA_{15} – LA_{12}$
MMU Common/Bank Area Register; CBAR (8)
Memory Management Unit
MMU Common Base Register; CBR (7)
MMU Bank Base Register; BBR (7)
8
$PA_{19} – PA_{12}$
LA: Logical Address
PA: Physical Address

Figure 37. MMU Block Diagram

Whether address translation takes place depends on the type of CPU cycle, listed as follows:

Memory Cycles

Address Translation occurs for all memory-access cycles including instruction and operand fetches,
memory-data reads and writes, hardware-interrupt-vector fetch, and software-interrupt restarts.

I/O Cycles

The MMU is logically bypassed for I/O cycles. The 16-bit logical I/O address space corresponds directly with the 16-bit physical I/O address space. The four High-order Bits (A16..A19) of the physical address are always `0` during I/O cycles (Figure 38).

DMA Cycles

When the Z80180 on-chip DMAC is using the external bus, the MMU is physically bypassed. The 20-bit source and destination registers in the DMAC are directly output on the physical-address bus (A19..A0).

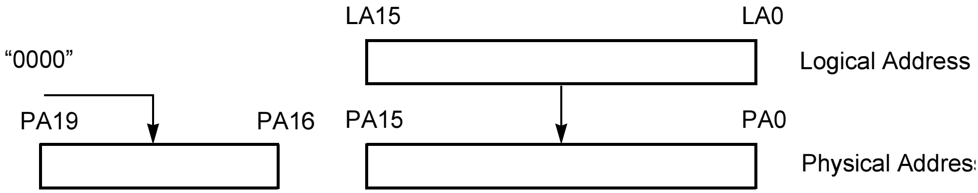


Figure 38. I/O Address Translation

MMU Registers

Three MMU Registers are used to program a specific configuration of logical and physical memory.

- MMU Common/Bank Area Register (CBAR)
- MMU Common Base Register (CBR)
- MMU Bank Base Register (BBR)

CBAR defines the logical memory organization, while CBR and BBR are used to relocate logical areas within the 1024 KB physical-address space. The resolution for both setting boundaries within the logical space and relocation within the physical space is 4 KB.

The CA field of CBAR determines the start address of Common Area 1 (Upper Common) and by default, the end address of the Bank Area. The BAR field determines the start address of the Bank Area and by default, the end address of Common Area 0 (Lower Common).

The CA and BA fields of CBAR may be freely programmed subject only to the restriction that CA may never be less than BA. Figure 39 and

Figure 40 illustrate examples of logical-memory organizations associated with different values of CA and BA.

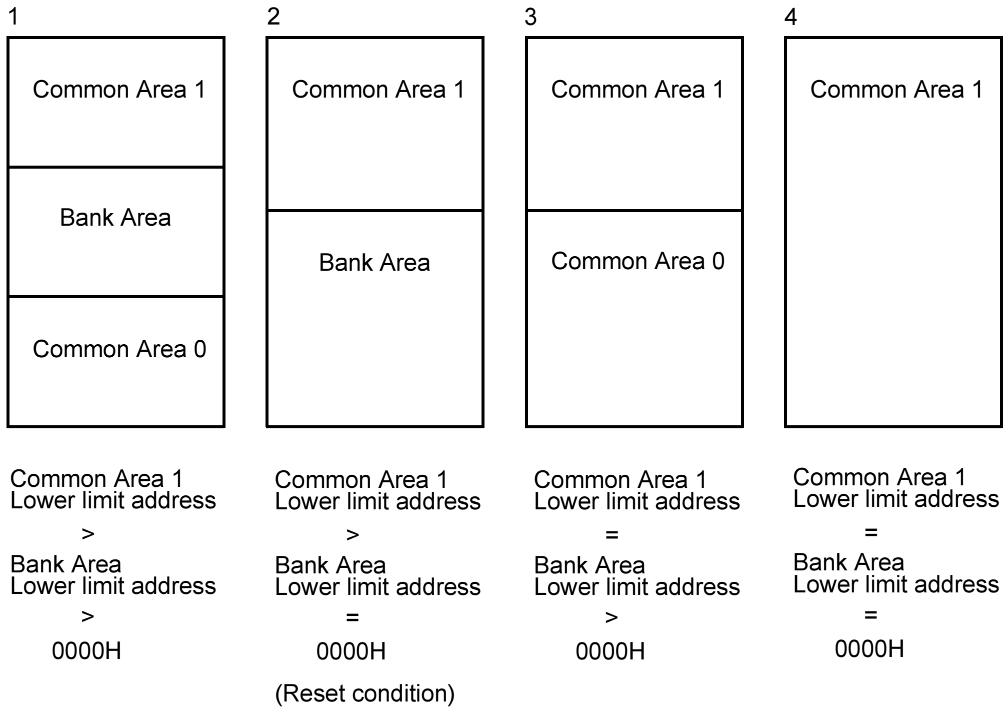


Figure 39. Logical Memory Organization

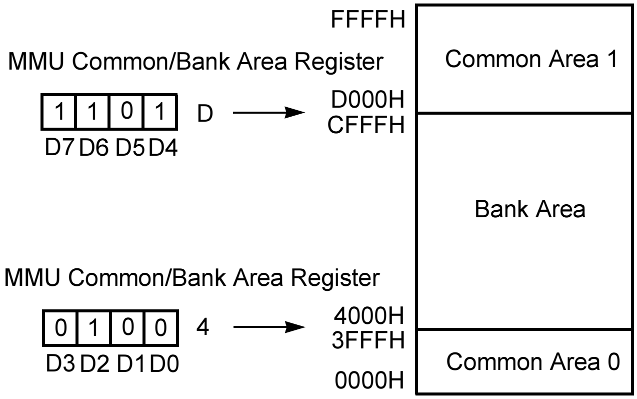


Figure 40. Logical Space Configuration Example

MMU REGISTER DESCRIPTION

MMU Common/Bank Area Register

CBAR specifies boundaries within the Z80180 64 KB logical-address space for up to three areas; Common Area, Bank Area and Common Area 1.

MMU Common/Bank Area Register (CBAR: 3AH)

7	4	3	0
CA		BA	
1		1	
R/W		R/W	

Bit Number	Field	R/W	Reset Value	Description
7..4	CA	R/W	1	**Logical-Address Space Boundaries** CA specifies the start (Low) address (on 4 KB boundaries) for the Common Area 1. This also determines the last address of the Bank Area.
3..0	BA	R/W	1	**Logical-Address Space Bounders** BA specifies the start (Low) address (on 4 KB boundaries) for the Bank Area. This also determines the last address of the Common Area 0.

MMU Common Base Register

CBR specifies the base address (on 4 KB boundaries) used to generate a 20-bit physical address for Common Area 1 accesses. All bits of CBR are reset to 0 during Reset.

MMU Bank Base Register (CBR: 39H)

7	6	5	4	3	2	1	0
CB7	CB6	CB5	CB4	CB3	CB2	CB1	CB0
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

MMU Bank Base Register

BBR specifies the base address (on 4 KB boundaries) used to generate a 19-bit physical address for Bank Area accesses. All bits of BBR are reset to 0 during Reset.

MMU Bank Base Register (BBR:39H)

7	6	5	4	3	2	1	0
BB7	BB6	BB5	BB4	BB3	BB2	BB1	BB0
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Register Description Missing

Physical Address Translation

Figure 41 describes the way in which physical addresses are generated based on the contents of CBAR, CBR and BBR. MMU comparators classify an access by logical area as defined by CBAR. Depending on which of the three potential logical areas (Common Area 1, Bank Area, or Common Area 0) is being accessed, the appropriate 8-bit base address is added to the High-order 4 bits of the logical address, yielding a 20-bit physical address. CBR is associated with Common Area 1 accesses. Common Area 0, if defined, is always based at physical address 00000H.

MMU and Reset

During Reset, all bits of the CA field of CBAR are set to 1 while all bits of the BA field of CBAR, CBR and BBR are reset to 0. The logical 64 KB address space corresponds directly with the first 64 KB (0000H to FFFFH) of the 1024 KB (00000H to FFFFFH) physical-address space. After Reset, the Z80180 begins execution at logical and physical address 0.

MMU Register Access Timing

When data is written into CBAR, CBR or BBR, the value is effective from the cycle immediately following the I/O write cycle which updates these registers.

Take care during MMU programming to ensure that CPU-program execution is not disturbed. Observe that the next cycle following MMU-Register programming normally is an Op Code Fetch from the newly translated address. One simple technique is to localize all MMU programming routines in a Common Area that is always enabled.

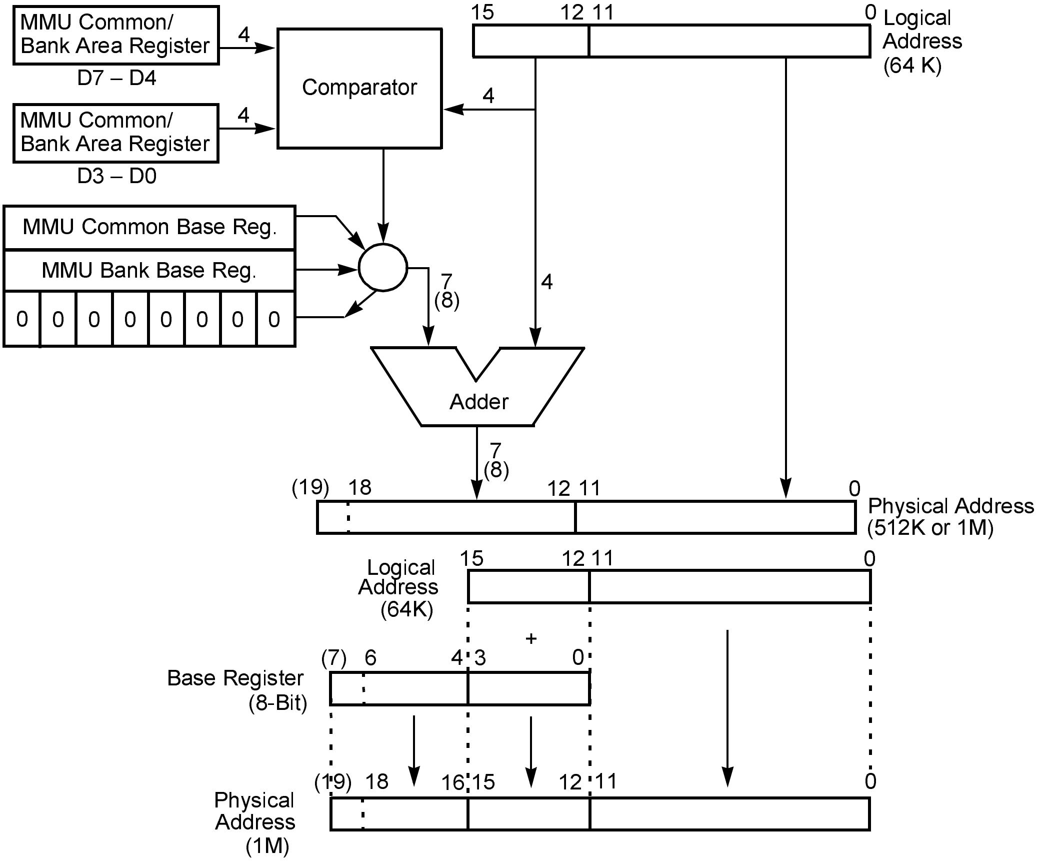


Figure 41. Physical Address Generation

DYNAMIC RAM REFRESH CONTROL

The Z80185/Z80195 incorporates a dynamic RAM refresh-control circuit including 8-bit refresh-address generation and programmable-refresh timing. This circuit generates asynchronous-refresh cycles inserted at the programmable interval independent of CPU-program execution. For systems which do not use dynamic RAM, the refresh function may be disabled.

When the internal-refresh controller determines that a refresh cycle must occur, the current instruction is interrupted at the first breakpoint between machine cycles. The refresh cycle is inserted by placing the refresh address on A0..A7 and the $\overline{\text{RFSH}}$ Output is driven Low.

Refresh cycles may be programmed to be either 2 or 3 clock cycles in duration by programming the REFW (Refresh Wait) bit in the Refresh Control Register (RCR). The external $\overline{\text{WAIT}}$ Input and the internal Wait-state generator are not effective during refresh. Refer to Figure 30 in the front of this Chapter.

Refresh Controller

Refresh Control Register (RCR: 36H)

The RCR specifies the interval and length of refresh cycles, while enabling or disabling the refresh function. Refer to Figure 3-20 in the front of this Chapter.

7	6	5	2	1	0
REFE	REFW	Reserved		CYC	
1	1			0	

Bit Number	Field	R/W	Reset Value	Description
7	REFE		1	**Refresh Enable** When REFE is 0 it disables the refresh controller and when REFE is 1 it enables the refresh-cycle insertion.
6	REFW		1	**Refresh Wait** Bit 6 (REFW) set to 0 causes the refresh cycle to be two clocks in duration. REFW set to 1 causes the refresh cycle to be three clocks in duration by adding a refresh-Wait cycle (TR$_W$).
5..2			0	**Reserved** Must be 0.
1..0	CYC		0	**Cycle Interval** CYC specify the interval (in clock cycles) between refresh cycles. In the case of dynamic RAMs requiring 128 refresh cycles every 2 ms (0r 256 cycles in every 4 ms), the required refresh interval is less than or equal to 15.625 µs. The underlined values indicate the best refresh interval depending on CPU clock frequency (see the following table).

			Processor Interval (in microsec)		
CYC1	CYC0	Insertion Interval	20 MHz	10 MHz	8 MHz
0	0	10 states	0.5	1.0 µ	1.25
0	1	20 states	1.0	2.0	2.5
1	0	40 states	2.0	4.0	5.0
1	1	80 states	4.0	8.0 µ	10.0

Refresh Control and Reset

After Reset, based on the initialized value of RCR, refresh cycles occur with an interval of 10 clock cycles and are 3 clock cycles in duration.

Dynamic RAM Refresh Operation Notes

1. Refresh-Cycle insertion is stopped when the CPU is in the following states:
   a. During Reset
   b. When the bus is released in response to $\overline{\text{BUSREQ}}$
   c. During Sleep Mode
   d. During $\overline{\text{WAIT}}$ states
2. Refresh cycles are suppressed when the bus is released in response to $\overline{\text{BUSREQ}}$. However, the refresh timer continues to operate. The time at which the first refresh cycle occurs after the Z80180 re-acquires the bus depends on the refresh timer and has no timing relationship with the bus exchange.
3. Refresh cycles are suppressed during Sleep Mode. If a refresh cycle is requested during Sleep Mode, the refresh-cycle request is internally

latched (until replaced with the next refresh request). The latched-refresh cycle is inserted at the end of the first machine cycle after Sleep Mode is exited. After this initial cycle, the time at which the next refresh cycle occurs depends on the refresh time and has no relationship with the exit from Sleep Mode.

4. The refresh address is incremented by one for each successful refresh cycle, not for each refresh. Thus, independent of the number of missed refresh requests, each refresh-bus cycle uses a refresh address incremented by one from that of the previous refresh-bus cycles.

Direct Memory Access

INTRODUCTION

This chapter describes the Direct Memory Access (DMA) channels of the Z80185/Z80195 their characteristics, operation, and programming.

DMA OVERVIEW

The Z80185/Z80195 includes a two-channel DMA controller which supports high speed data transfer.

Common Capabilities

Both channels (Channel 0 and Channel 1) have the following capabilities:

Memory Address Space

Memory source and destination addresses may be directly specified anywhere within the 1024 KB physical address space using 20-bit source and destination memory addresses. In addition, memory transfers can arbitrarily cross 64 KB physical address boundaries without CPU intervention.

I/O Address Space

I/O source and destination addresses may be directly specified anywhere within the 64 KB I/O address space (16-bit source and destination I/O addresses).

Transfer Length

Up to 64 KB may be transferred based on a 16-bit byte count register.

Request Handshaking

Transfers involving an I/O device are controlled by a Request line from the device, indicating data available from an Input/Source device or data request from an Output/Destination device. Requests may be internally routed to both DMA channels from the ESCC, ASCIs, Bidirectional Centronics controller or from an external device on the T_{OUT}/DREQ pin.

Requests from Input/Source devices are typically programmed as Level sensitive because these devices have a relatively long time to update their request signal after the DMA channel reads data from them.

Requests from Output/Destination devices are typically programmed as Edge sensitive, because such devices have much less time to update their request signal from the start of the cycle in which the DMA controller writes data to them.

Transfer Rate

A byte transfer may occur every six clock cycles. Wait states may be inserted in DMA cycles for slow memory or I/O devices.

Channel Complete Interrupt

Each DMA channel may request an interrupt when the transfer of the programmed byte count is complete.

Alternating-Channel Capability

When both channels are programmed to service the same high-speed device, special request-routing logic may be invoked. The logic enables the processor software to load a new buffer address and byte count into one channel while the other channel continues its activity. This methodology provides the efficiency required for handling fast, demanding I/O devices.

Channel 0 Unique Capabilities

Channel 0 may perform several types of data transfers:

- Memory to memory
- Memory to I/O
- I/O to memory
- Memory to memory mapped I/O
  - These transfers may include a memory address increment, decrement, or no-change
  - Burst or cycle steal memory to memory transfers
  - Channel 0 transfers have higher priority than DMAC Channel 1 transfers

Channel 1 Unique Capabilities

Channel 1 may perform memory-to-I/O and I/O-to-memory transfers. These transfers may include a memory address increment or decrement.

DMAC Registers

Each DMA channel includes twoaddress registers and one byte count register.

The Channel 0 registers are the Source Address Register (SAR0), the Destination Address Register (DAR0), and the Byte Count Register (BCR0).

The Channel 1 registers are the Memory Address Register (MAR1), I/O Address Register (IAR1), and the Byte Count Register (BCR1).

SAR0, DAR0, MAR0, and IAR1 are each composed of three registers which are called xxRnB, xxRnH, and xxRnL, xxRnB contains the most-significant bits and xxRnL the least-significant bits.

The two channels share the DMA Status Register (DSTAT), the DMA Mode Register (DMODE), and the DMA Control Register (DCNTL).

DMAC BLOCK DIAGRAM

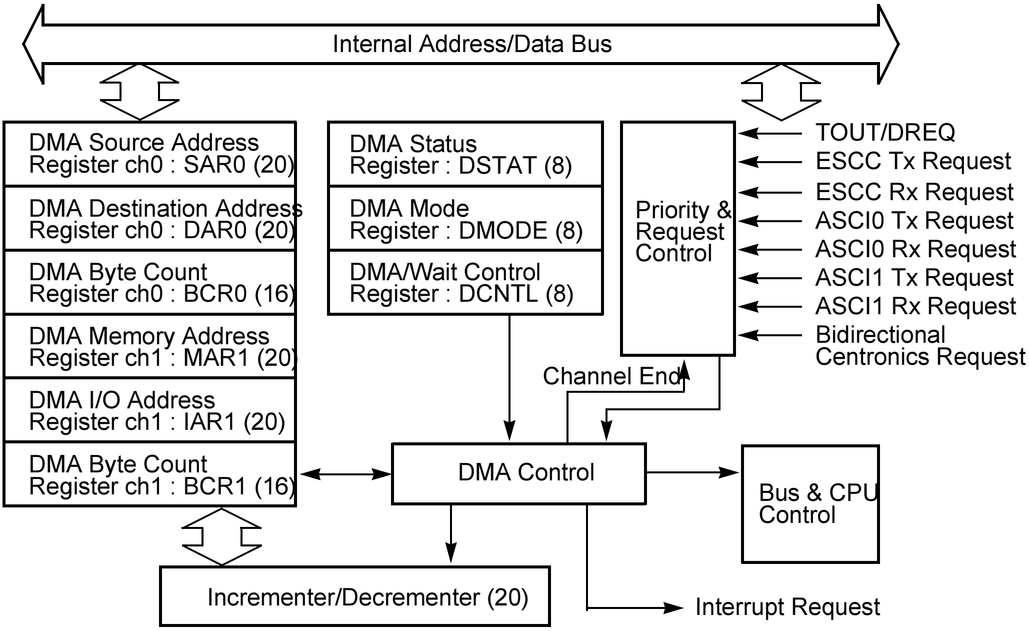

Internal Address/Data Bus
DMA Source Address Register ch0 : SAR0 (20)
DMA Destination Address Register ch0 : DAR0 (20)
DMA Byte Count Register ch0 : BCR0 (16)
DMA Memory Address Register ch1 : MAR1 (20)
DMA I/O Address Register ch1 : IAR1 (20)
DMA Byte Count Register ch1 : BCR1 (16)
DMA Status Register : DSTAT (8)
DMA Mode Register : DMODE (8)
DMA/Wait Control Register : DCNTL (8)
Priority & Request Control
TOUT/DREQ
ESCC Tx Request
ESCC Rx Request
ASCI0 Tx Request
ASCI0 Rx Request
ASCI1 Tx Request
ASCI1 Rx Request
Bidirectional Centronics Request
Channel End
DMA Control
Bus & CPU Control
Incrementer/Decrementer (20)
Interrupt Request

Figure 42. DMA Block Diagram

DMAC REGISTER DESCRIPTION

DMA Source Address Register

The DMA Source Address Register (`SAR0`: `20H` to `22H`) specifies the physical source address for Channel 0 transfers. The register contains 20 bits and can specify up to 1024 KB memory addresses or up to 64 KB I/O addresses. Channel 0 source can be memory, I/O, or memory mapped I/O. For I/O, the most-significant bits of this register identify the Request Handshake Signal for Channel 0.

DMA Destination Address Register

The DMA Destination Address Register (`DAR0`: `23H` to `25H`) specifies the physical destination address for Channel 0 transfers. The register contains 20 bits and can specify memory addresses up to 1024 KB or I/O addresses up to 64 KB. Channel 0 destination can be memory, I/O, or memory mapped I/O. For I/O, the most-significant bits of this register identify the Request Handshake Signal for Channel 0.

DMA Byte Count Register

The DMA Byte Count Register (`BCR0`: `26H` to `27H`) specifies the number of bytes to be transferred. This register contains 16 bits and may specify transfers up to 64 KB. When one byte is transferred, the register is decremented by `1`. If *n* bytes must be transferred, the value *n* must be loaded into this register before the DMA operation begins.

DMA Memory Address Register

The DMA Memory Address Register (`MAR1`: `28H` to `2AH`) specifies the physical memory address for Channel 1 transfers. This register may be destination or source memory address. The register contains 20 bits and may specify up to 1024 KB memory address.

DMA I/O Address Register

The DMA I/O Address Register (IAR1: 2BH to 2DH) specifies the I/O address for Channel 1 transfers. This register may be destination or source I/O address. The register contains 16 bits of I/O address; its most-significant byte (IAR1B) identifies the Request Handshake Signal and controls the Alternating-Channel feature, and is described in Figure 43 in the "DMA Operation" section. All bits in IAR1B reset to 0.

DMA I/O Address Register Most-Significant Byte Register (`IAR1B: 2DH`)

7	6	5	4	3	2	0
AltE	AltF	Reserved		T_{OUT}/ DREQ	Req1Sel	
0	0			0	0	
R/W	R/W			R/W	R/W	

Bit Number	Field	R/W	Reset Value	Description
7	AltE	R/W	0	**Alternating Channel Enable** Set this bit to `1` only when both DMA channels are programmed for the same I/O device. 1: A *channel-end* condition (byte count is `0`) on Channel 0 sets Bit 6 (AltF), which then routes the device's Request signal to Channel 1 rather than Channel 0. Similarly, a channel-end condition on Channel 1 clears Bit 6 (AltC), which then routes the device's Request to Channel 0 rather than Channel 1.
6	AltF	R/W	0	**Alternating Channel Select** 1: When Bit 7(AltE) is 1, the Request selected by SAR18-16 or DAR18-16 is not presented to Channel 0, but Channel 1's request operates normally. This bit may written by the user's software, but a write is performed only when both channels are stopped (both `DE1` and `DE0` are `0`), to select which channel operates first. 0: When Bit 7 (AltE) is `1`, the Request signal selected by Bits 2..0 is not presented to Channel 1, and Channel 0's Request operates normally.

Bit Number	Field	R/W	Reset Value	Description
5..4	Reserved		0	**Reserved** Must be 0.
3	TOUT/ DREQ	R/W	0	**T_{OUT}/DREQ Function Select** 1: The pin carries the T_{OUT} output from PRT Channel 1. 0: As Reset, the T_{OUT}/DREQ pin acts as a DREQ Input.
2..0	Req1Sel	R/W	0	**Channel 1 Request Signal Select** These 3 bits select which Request signal is presented to DMA Channel 1: 000: External T_{OUT}/DREQ 001: ASCI 0 Request (TDRE or RDRF) 010: ASCI 1 Request (TDRE or RDRF) 011: ESCC Request 111: Bidirectional Centronics Request

DMA Byte Count Register

The DMA Byte Count Register (BCR1: 2EH to 2FH) specifies the number of bytes to be transferred. This register contains 16 bits and may specify up to 64 KB transfers. When one byte is transferred, the register is decremented by 1.

DMA Status Register

The DMA Status Register (DSTAT) is used to enable and disable DMA transfer an DMA termination interrupts. DSTAT also indicates DMA transfer status; in other words, whether DMA transfers has been completed or are in progress.

DMA Status Register (DSTAT: 30H)

7	6	5	4	3	2	1	0
DE1	DE0	$\overline{\text{DWE1}}$	$\overline{\text{DWE0}}$	DIE1	DIE0	Reserved	DME
0	0	1	1	0	0		0
R/W	R/W	W	W	R/W	R/W		R

Bit Number	Field	R/W	Reset Value	Description
7	DE1	R/W	0	**DMA Enable Channel 1** To perform a software write to DE1, Bit 5 (DWE1) must be written as 0 during the same register Write access. 1: Enables Channel 1 DMA and automatically sets (DME) to 1. When a DMA transfer terminates (BCR1 is 0), DE1 is reset to 0 by the DMAC. 0: Disables Channel 1 DMA, but DMA is restartable.
6	DE0	R/W	0	**DMA Enable Channel 0** To perform a software write to DE0, Bit 4 (DWE0) must be written as 0 during the same register Write access. 1: Enables Channel 0 DMA and automatically sets (DME) to 1. DE0 is reset to 0 by the DMAC. 0: Disables Channel 0 DMA. A DMA interrupt request is made to the CPU.
5	$\overline{\text{DWE1}}$	W	1	**DE1 Bit Write Enable** To write to Bit 7 (DE1), DWE1 must be written as 0 during the same access. DWE1 always reads as 1.
4	$\overline{\text{DWE0}}$	W	1	**DE0 Bit Write Enable** To write to Bit 7 (DE0), DWE0 must be written as 0 during the same access. DWE0 always reads as 1.

Bit Number	Field	R/W	Reset Value	Description
3	DIE1	R/W	0	**DMA Interrupt Enable Channel 1** 1: The termination of Channel 1 DMA transfer (indicated when Bit 7, DE1, is 0) causes a CPU interrupt request to be generated. 0: The Channel 0 DMA termination interrupt is disabled.
2	DIE0	R/W	0	**DMA Interrupt Enable Channel 0** 1: The termination of Channel 0 DMA transfer (indicated when Bit 6, DE0, is 0) causes a CPU interrupt request to be generated. 0: The Channel 0 DMA termination interrupt is disabled.
1	Reserved		0	**Reserved** Must be 0.
0	DME	R	0	**DMA Main Enable** A DMA operation is only enabled when the channel's DE bit, Bit 6 (DE0) for Channel 0 for Bit 7 (DE1) for Channel 1, and the DME bit are both set. When $\overline{\text{NMI}}$ interrupt occurs, DME is reset to 0, thus disabling DMA activity during the $\overline{\text{NMI}}$ interrupt-service routine. To restart, DMA, DE0 and/or DE1 are 1 (even if the contents are already 1). This sequence automatically sets DME to 1, allowing DMA operations to continue. DME cannot be written directly. It is cleared to 0 by $\overline{\text{NMI}}$ or indirectly set to 1 by setting DE0 and/or DE1 to 1.

DMA Mode Register

DMODE is used to set the addressing and transfer mode for Channel 0 (Tables 2 and 2).

DMA Mode Register (`DMODE: 31H`)

7	6	5	4	3	2	1	0
Reserved		DM1	DM0	SM1	SM0	MMOD	Reserved
		0	0	0	0	0	
		R/W	R/W	R/W	R/W	R/W	

Bit Number	Field	R/W	Reset Value	Description
7..6	Reserved		0	**Reserved** Must be 0.
5..4	DM1.. DM0	R/W	0	**Destination Mode Channel 0** Specifies whether the destination for Channel 0 transfers is memory or I/O, and whether the address is incremented or decremented for each byte transferred. Values for this bit are described in Table 2.
3..2	SM1.. SM0	R/W	0	**Source Mode Channel 0** Specifies whether the source for Channel 0 transfers is memory or I/O, and whether the address is incremented or decremented for each byte transferred. Values for this bit are described in Table 3

Bit Number	Field	R/W	Reset Value	Description
1	MMOD	W	0	**Memory Mode Channel 0** When Channel 0 is configured for memory to memory transfers there is no Request Handshake Signal to control the transfer timing. Instead, two automatic transfer timing modes are selectable: Burst (MMOD is `1`) and Cycle Steal (MMOD is `0`). For burst memory to memory transfers, the DMAC controls the bus continuously until the DMA transfer completes (when the byte count register reaches `0`). In Cycle Steal mode, the CPU is granted one cycle for each DMA byte-transfer cycle until the transfer is completed. For Channel 0 DMA with I/O source or destination, the selected Request signal controls the transfer, and MMOD is ignored.
0	Reserved		0	**Reserved** Must be 0.

Table 2. Addressing Mode for Channel 0

			Address
DM1	**DM0**	**Memory/ I/O**	**Increment/Decrement**
0	0	Memory	+1
0	1	Memory	–1
1	0	Memory	Fixed
1	1	I/O	Fixed

Table 3. Transfer Mode for Channel 0

			Memory
SM1	SM0	Memory I/O	Increment/Decrement
0	0	Memory	+1
0	1	Memory	–1
1	0	Memory	Fixed
1	1	I/O	Fixed

Table 4 describes all DMA transfer mode combinations of DM0, DM1, SM0, SM1. Because I/O-to-I/O transfers are not implemented, 12 combinations are available.

Table 4. DMA Transfer Mode Combinations

					Address
DM1	DM0	SM1	SM0	Transfer Mode	Increment/Decrement
0	0	0	0	Memory to Memory	SAR0+1, DAR0+1
0	0	0	1	Memory to Memory	SAR0–1, DAR0+1
0	0	1	0	Memory* to Memory	SAR0 fixed, DAR0+1
0	0	1	1	I/O to Memory	SAR0 fixed, DAR0+1
0	1	0	0	Memory to Memory	SAR0+1, DAR0–1
0	1	0	1	Memory to Memory	SAR0–1, DAR0–1
0	1	1	0	Memory* to Memory	SAR0 fixed, DAR0–1
0	1	1	1	I/O to Memory	SAR0 fixed, DAR0–1
1	0	0	0	Memory to Memory*	SAR0+1, DAR0 fixed
1	0	0	1	Memory to Memory*	SAR0–1, DAR0 fixed
1	0	1	0	Reserved	
1	0	1	1	Reserved	
1	1	0	0	Memory to I/O	SAR0+1, DAR0 fixed

The * indicates that memory-mapped I/O is included.

Table 4. DMA Transfer Mode Combinations

					Address
DM1	**DM0**	**SM1**	**SM0**	**Transfer Mode**	**Increment/Decrement**
1	1	0	1	Memory to I/O	SAR0–1, DAR0 fixed
1	1	1	0	Reserved	
1	1	1	1	Reserved	

The * indicates that memory-mapped I/O is included.

DMA $\overline{\text{Wait}}$ Control Register

DCNTL controls the insertion of Wait states into DMAC and CPU accesses of memory or I/O. Also, it defines the Request signal for each channel as Level or Edge sense. DCNTL also sets the DMA transfer mode for Channel 1, described in Table 5, which is limited to memory I/O transfers.

DMA $\overline{\text{Wait}}$ Control Register (`DCNTL: 32H`)

7	6	5	4	3	2	1	0
MWI1	MWI0	IWI1	IWI0	DMS1	DMS0	DIM1	DIM0
1	1	1	1	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit Number	Field	R/W	Reset Value	Description
7..6	MWI1.. MWI0	R/W	1	Memory Wait Insertion Specifies the number of Wait states introduced into CPU and DMAC memory access cycles. See the “Wait State Generation” section for details.
5..4	IWI1.. IWI0	R/W	1	**I/O Wait Insertion** Specifies the number of Wait states introduced into CPU and DMAC I/O access cycles. See the “Wait State Generation” section for details.

Bit Numbe r	Field	R/W	Reset Value	Description
3..2	DMS1.. DMS0	R/W	0	**DMA Request Sense** Specifies the DMA request sense for Channel 0 and Channel 1, respectively. Typically, for an Input/Source device, the associated DMS bit is programmed as 0 for Level sense. The device has a relatively long time to update its Request signal after the DMA channel reads data from it in the first of the two machine cycles involved in transferring a byte. An Output/Destination device has much less time to update its Request signal, after the DMA channel starts a Write operation to it. The update occurs in the second of two machine cycles involved in transferring a byte. With Zero-Wait state I/O cycles, which apply only to the ASCIs, it is impossible for a device to update its Request signal in time, and Edge sensing must be used.
				With one-Wait-state I/O cycles (the fastest possible except for the ASCIs), it is unlikely that an output device is able to update its Request in time. Edge sense is required for output to the ESCC and Bidirectional Centronics controller, and is recommended for External Output devices connected to T_{OUT}/DREQ. With two or more Wait states in I/O cycles, External Output devices on T_{OUT}/DREQ may use Edge or Level sense depending on their characteristics; Edge sense is still recommended for output on the ESCC and Bidirectional Centronics controller. 1: The Input is Edge sense. 0: The Input is Level sense.
1..0	DIM1.. DIM0	R/W	0	**DMA Channel 1 I/O and Memory Mode** Specifies the source, destination, and address for Channel 1 memory to/from I/O transfer modes.

Table 5. DMA Transfer Mode for Channel 1

			Address
DIM1	DMI0	Transfer Mode	Increment/Decrement
0	0	Memory to I/O	MAR1 +1, IAR1 fixed
0	1	Memory to I/O	MAR1–1, IAR1 fixed
1	0	I/O to Memory	IAR1 fixed, MAR1 + 1
1	1	I/O to Memory	IAR1 fixed, MAR1 –1

DMA OPERATION

This section discusses the various operating modes of DMA Channels 0 and 1.

Memory to Memory

For memory-to-memory transfers, no Request signal governs DMA transfer timing. Rather, the DMA operation is timed in one of two programmable modes - Burst or Cycle Steal. In both modes, the DMA operation automatically proceeds until the byte count in BCR0 has been decremented to 0.

In Burst mode, the DMA operation proceeds until termination. In this case, the CPU cannot perform any program execution until the DMA operation is completed.

In Cycle Steal mode, DMA and CPU operation alternated until the DMA is completed. The following sequence is repeated until DMA is completed.

1. 1 CPU Machine Cycle
2. 1 DMA Byte Read
3. 1 DMA Byte Write

Figure 43 describes Cycle Steal mode DMA timing.

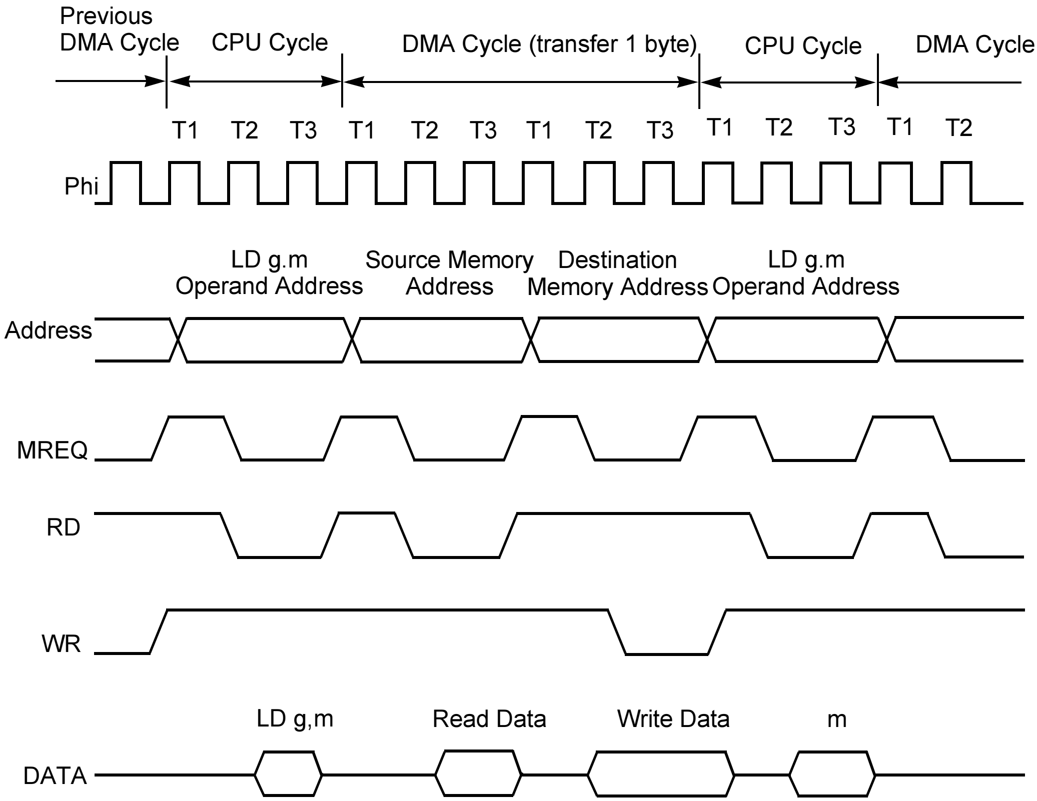


Figure 43. DMA Timing-Cycle Steal Mode

To initiate memory-to-memory DMA transfer for Channel 0, perform the following operations.

1. Load the memory source and destination addresses into `SAR0` and `DAR0`.

2. Specify memory-to-Memory mode and Address Increment/ Decrement in the SM0, SM1, DM0 and DM1 bits of DMODE.
3. Load the number of bytes to transfer into BCR0.
4. Specify Burst or Cycle Steal mode in the MMOD bit of DCNTL.
5. Program Bit 6 (DE0) as 1 (with Bit 4 $\overline{\text{DWE0}}$ written as 0 in the same access) in DSTAT. If an interrupt is required after the programmed byte count has been transferred, set Bit 2 (DIE0) to 1 in the same write to DSTAT. The DMA operation starts one machine cycle later.

Memory-Mapped I/O

For memory to/from memory-mapped I/O, a Request signal controls DMA operation. The DMA channels may be programmed to use this Request in a Level- or Edge-Sensitive Mode.

When Level sense is programmed, DMA operation is triggered when the Request line (such as, T_{OUT}/DREQ) is sampled Low. As depicted in Figure 44, the Request line is sampled again at the Rising edge of 0 that begins the second-last clock cycle of the machine cycle that writes the data byte to the destination. If the Request is low at that time, as in Figure 44, DMA operation continues for another byte. If the Request is High at that time, the DMA channel relinquishes use of the bus after the current Write operation.

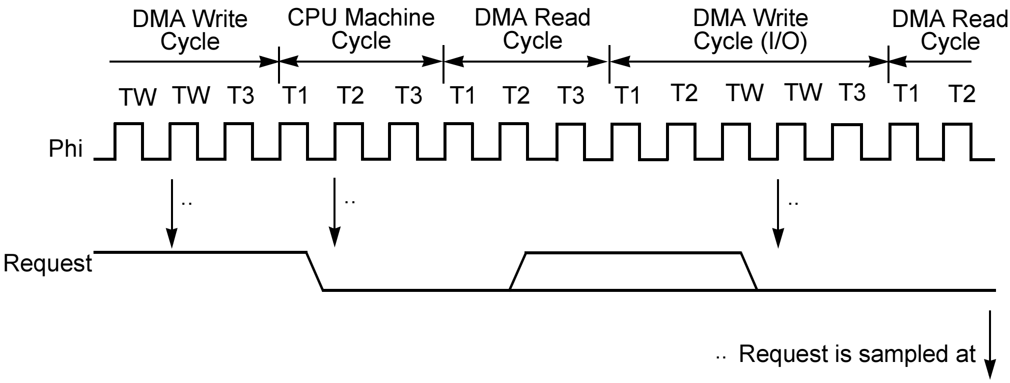


Figure 44. CPU Operation and DMA Operation With Level Sense Request

When Edge sense is programmed, DMA operation is triggered by a Falling edge on the Request line (T_{OUT}/DREQ or Request from an ASCI, ESCC or the Bidirectional Centronics controller). If another Falling edge is detected before the Rising edge of ϕ at the start of the second-last clock cycle of the machine cycle that writes the data to the destination, DMA operation continues for another byte. If not (as in Figure 45) the DMA channel relinquishes use of the bus after the Write operation. DMA operation by the channel is triggered again when a Falling edge on the Request line precedes the Rising edge of ϕ at the start of the second-last clock cycle in a machine cycle.

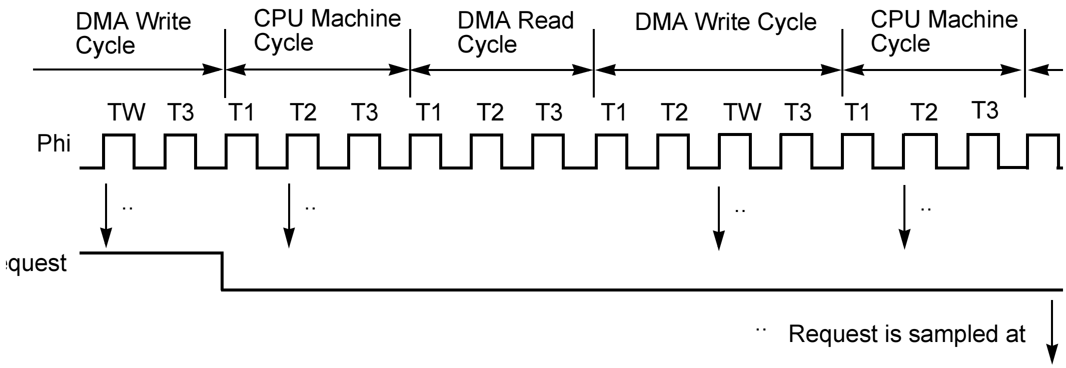


Figure 45. CPU Operation and DMA Operation With Edge Sense Request

To initiate a memory to/from mapped I/O, DMA transfer for Channel 0, including DMA transfers with an ASCI, ESCC or Bidirectional Centronics Controller, perform the following operations:

1. Load the memory and I/O or memory mapped I/O source and destination addresses into SAR0 and DAR0. I/O addresses (not memory mapped I/O) are limited to 16 bits (A15..A0). For an I/O source or destination, program A18..16 to select the Request signal that controls the operation, as described in Table 6. (For memory-mapped I/O, T_{OUT}/DREQ is automatically selected as the Request signal.)
2. Specify memory to/from I/O or memory to/from memory-mapped I/O Mode and Address Increment/
Decrement in the SM0, SM1, DM0 and DM1 bits of DMODE.
3. Load the number of bytes to transfer into BCR0.
4. Specify whether the Request line is Edge or Level-sense by programming the DMS0 bit of DCNTL.
5. Enable or disable DMA termination interrupt with the DIE0 bit in DSTAT.

6. Program DE0 to `1` and DWE0 to `0` in the same access (in `DSTAT`). DMA operation then begins under the control of the Request signal.

7. For an Edge-sensed Request line from a destination device, if the Request line was asserted before step 6, write one byte of data to the device under program control, so that the Request line is negated. Refer to Table 6.

Table 6. DMA Source Transfer Request

SAR18	SAR17	SAR16	DMA Transfer Request	Edge/Level Sense
0	0	0	T_{OUT}/DREQ	Level recommended
0	0	1	ASCI0 RDRF	Level
0	1	0	ASCI1 RDRF	Level
0	1	1	ESCC Rx	Level
1	0	x	Reserved	
1	x	0	Reserved	
1	1	1	Bidirectional Centronics Input	Either

Table 7. DMA Destination Transfer Request

DAR18	DAR17	DAR16	DMA Transfer Request	Edge/Level Sense
0	0	0	T_{OUT}/DREQ	Edge recommended
0	0	1	ASCI0 TDRF	Edge
0	1	0	ASCI1 TDRF	Edge
0	1	1	ESCC Tx	Edge
1	0	x	Reserved, do not program	
1	x	0	Reserved, do not program	
1	1	1	Bidirectional Centronics output	Edge

Channel 1 DMA

DMAC Channel 1 performs memory to/from I/O transfers. Except for different registers and status/control bits, operation is similar to that described for Channel 0 memory to/from I/O DMA.

To initiate DMA Channel 1 memory to/from I/O transfer including DMA transfers with an ASCI, ESCC or Bidirectional Centronics controller, perform the following operations:

1. Load the memory address (20 bits) into MAR1.
2. Load the I/O address (16 bits) into the least- significant 16 bits of IAR1.
3. Program the 3 least-significant bits of IAR1B (which corresponds to A18..A16 if IAR1 holds a memory address) to select the Request line that controls the transfer. These bits are encoded as depicted in Table 6 for Channel 0.
4. Program the Source/Destination and Address Increment/Decrement mode using Bit 3 (DIM1) and Bit 0 (DIM0) in DCNTL.
5. Specify whether the Request signal is Level or Edge sense in Bit 3 (DMS1) in DCNTL. The Request line is sampled as described for Channel 0, and the same considerations described previously apply to this choice.
6. Enable or disable DMA termination interrupt with Bit 3 (DIE1) in DSTAT.
7. Program Bit 3 (DE1) to 1 and Bit 4 ($\overline{\text{DWE1}}$) to 0 in the same access in DSTAT. The DMA operation then begins under the control of the Request signal.
8. For an Edge-sensed Request line from a destination device, if the Request line was asserted before step 7, write one byte of data to the device under program control, so that the Request line is negated.

DMA Bus Timing

When memory (or memory mapped I/O) is specified as a source or destination, $\overline{\text{MREQ}}$ goes Low during the memory access. When I/O is specified as a source or destination, $\overline{\text{IORQ}}$ goes Low during the I/O access.

When I/O (and memory mapped I/O) is specified as a source or destination, DMA operation is controlled by the Request signal from the I/O device, which may be an internal signal or the T_{OUT}/DREQ pin from an external device.

For accesses to I/O devices other than the ASCIs, one Wait state is automatically inserted. Additional Wait states may be inserted by programming the on-chip Wait state generator or, for external devices, by using the external $\overline{\text{WAIT*}}$ Input. For memory mapped I/O accesses, the only automatic Wait-state generation available is via Bits 1..0 of the `WSGCS`.

For memory-to-memory transfers (Channel 0 only) no Request signal is used. Automatic DMA timing is programmed as either burst or cycle steal.

When a DMA memory address carry/borrow between Bits A15 and A16 or the address bus occurs (when crossing a 64 KB boundary), the minimum bus cycle is extended to 4 clocks by automatic insertion of one internal clock cycle.

DMAC Channel Priority

In case of simultaneous requests, Channel 0 has priority over Channel 1. When Channel 0 is performing a memory to/from memory transfer, Channel 1 cannot operate until the Channel 0 operation has terminated. If Channel 1 is operating, Channel 0 cannot operate until Channel 1 releases control of the bus.

DMAC and BUSREQ, BUSACK

The $\overline{\text{BUSREQ}}$ and $\overline{\text{BUSACK}}$ Inputs allow another bus Master to take control of the bus. $\overline{\text{BUSREQ}}$ and $\overline{\text{BUSACK}}$ have priority over the on-chip DMAC, and suspend DMAC operation. The DMAC releases the bus to the external-bus Master between DMAC memory or I/O accesses. Because a single byte DMAC transfer requires a read and a Write cycle, it is possible for the DMAC to be suspended after the DMAC read, but before the DMAC write. When the external Master releases the bus ($\overline{\text{BUSREQ}}$ High), the on-chip DMAC continues the suspended DMA operation.

DMAC Internal Interrupts

Figure 46 illustrates the internal DMA interrupt request generation circuit.

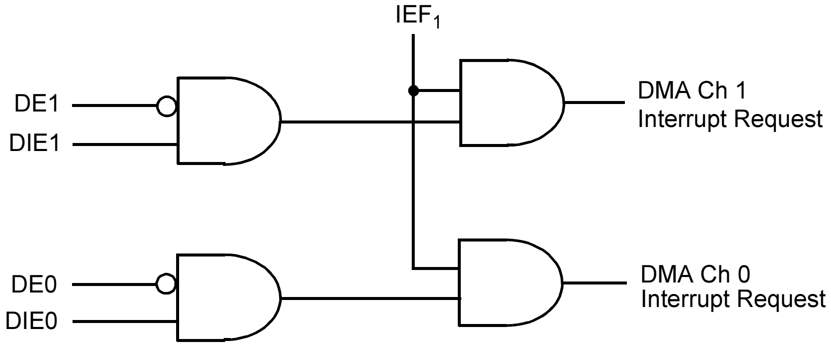


Figure 46. DMA Interrupt Request Generation

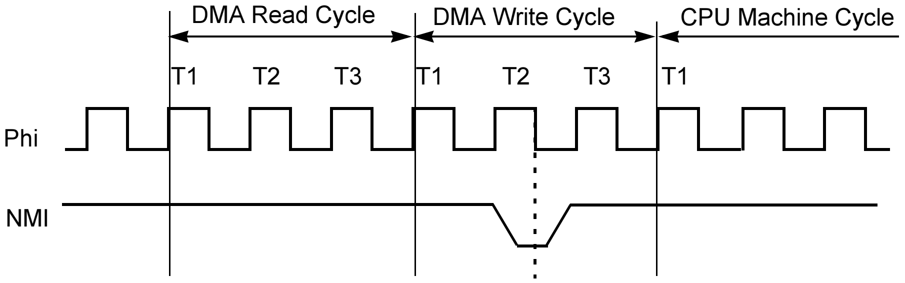
DSTAT Bit 6 (DE0) and Bit 7 (DE1) are automatically cleared to 0 at the completion (byte count is 0) of a DMA operation for Channel 0 and Channel 1, respectively. They remain 0 until a 1 is written. A Level-sensitive interrupt occurs any time a DE bit (Bit 6, DE0, or Bit 7, DE1) is

`0`, the corresponding DIE bit (Bit 2, DIE0, and Bit 3, DIE1) is `1`, and the CPU IEF1 flag is `1`. Therefore, the DMA termination interrupt service routine disables further DMA interrupts by programming the channel DIE bit to `0`, before enabling CPU interrupts by executing an `IE` instruction to set IEF1 to `1`. Alternatively, after reloading the DMAC address and count registers, software sets the DIE bit to `1` to reenable the channel interrupt in the same instruction that reenables DMA operation by programming the channel DE bit to `1`.

DMAC and $\overline{\text{NMI}}$

$\overline{\text{NMI}}$, unlike other interrupts, automatically disables DMAC operation by clearing the Bit 0 (DME) in `DSTAT`. Thus, the $\overline{\text{NMI}}$ interrupt service routine may respond to time-critical events without delay due to DMAC bus usage. Also, $\overline{\text{NMI}}$ may be used as an external DMA abort input, in that both channels are suspended by the clearing of DME. Figure 47 illustrates $\overline{\text{NMI}}$ and DMA operation

If the Falling edge of $\overline{\text{NMI}}$ occurs before the Falling clock of the state prior to T3 (T2 or TW) of a DMA Write cycle, the DMAC is suspended and the CPU starts the NMI response at the end of the current cycle.



Note: DME is 0 (DMA Stop)

Figure 47. $\overline{\text{NMI}}$ and DMA Operation

When software sets a Bit 7 (DE1) or Bit 6 (DE0) in DMODE to 1, the channel correctly resumes operation from the point at which it was suspended by $\overline{\text{NMI}}$.

DMAC and Reset

During Reset the bits in DSTAT, DMODE, and DCNTL are initialized as stated in their individual register descriptions. Any DMA operation in progress is stopped, allowing the CPU to use the bus to perform the Reset sequence. However, the contents of the address registers (SAR, DAR0, MAR1, and IAR1) and byte count registers (BCR0, BCR1) contents are not changed during Reset.

Asynchronous Serial Communications Interface

OVERVIEW

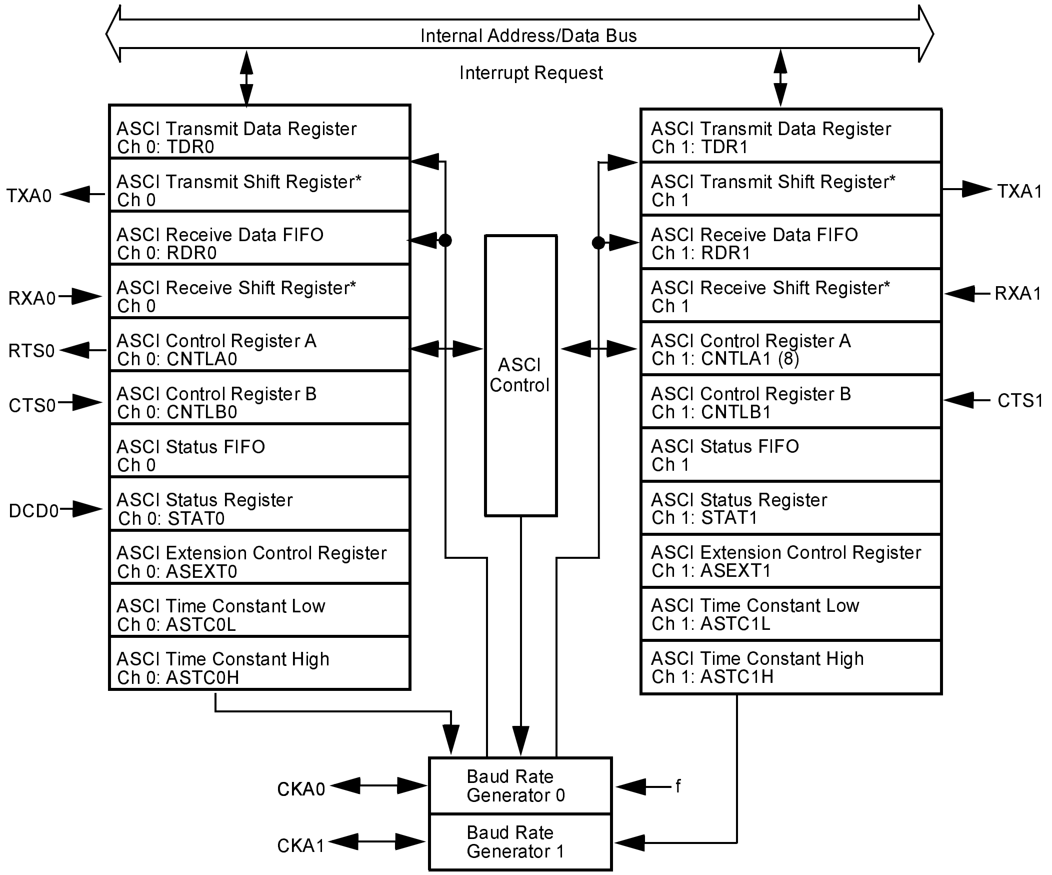
The Z80185/Z80195 on-chip Asynchronous Serial Communications Interface (ASCI) consists of two independent, full-duplex serial channels. The ASCI channels can communicate with a variety of standard Universal Asynchronous Receiver/Transmitters (UARTs) including the Z8440 SIO and the Z8530 SCC.

Features

- Each channel is independently programmable
- Full-Duplex communication
- 7- or 8-Bit data length
- Program-Controlled 9th data bit for multiprocessor communication
- 1 or 2 Stop bits
- Odd, Even, No parity
- Parity, Overrun, Framing Error, and Break detection
- Programmable baud rate generators for each channel
- Speed to 520 Kb per second (for CPU fc = 33.33 MHz /16 mode).
- Channel-0 Modem control signals: $\overline{\text{DCD0}}$, $\overline{\text{CTS0}}$ and $\overline{\text{RTS0}}$
- Programmable interrupt condition Enable and Disable
- Operation with on-chip DMAC
- 4-Character Receive FIFOs

- External clock input or output
- Divide by 1, 16, or 64 Clocking
- Break generation and detection

ASCI BLOCK DIAGRAM



Note: The * denotes registers that are not program accessible.

Figure 48. ASCI Block Diagram

ASCI REGISTERS

The following registers are not program accessible. The information contained in this section is for
reference only

ASCI Transmit Shift Registers

The ASCI Transmit Register 0 (TSRO) and ASCI Transmit Shift Register 1 (TSRI) receives data from the ASCI Transmit Data Register (TDR). Data is shifted out to the TXA pin. When transmission is completed, the next byte (if available) is automatically loaded from TDR into TSR and the next transmission starts. When no data is available for transmission, TSR idles by outputting a continuous High level. This register is not program accessible.

ASCI Transmit Data Registers

ASCI Transmit Data Register 0 (TDR0: 06H) and ASCI Transmit Data Register 1 (TDR1: 07H) contain data to be transferred to the TSR as soon as it is empty. Data can be written while the TSR is shifting out the previous byte of data. Therefore, the ASCI transmitter is double buffered.

Data can be written into and read from the ASCI Transmit Data Register. When data is read from the ASCI Transmit Data Register, operation is not affected.

ASCI Receive Shift Registers

This register receives data shifted in from the RXA pin. When a character is received, it is automatically transferred to the ASCI FIFO when the FIFO is not full. When the FIFO is full when the next incoming data byte is completed, an overrun error occurs. This register is not program accessible.

ASCI Receive Data FIFO Registers

ASCI Receive Data Register 0 (RDR0: 08H) and ASCI Receive Data Register 1 (RDR1: 09H) are read-only registers. When a complete incoming data byte is assembled in the RSR, it is automatically transferred to the 4-character Receive Data First-In First-Out (FIFO) memory. The oldest character in the FIFO (if any) can be read from the Receive Data Register (RDR). The next incoming data byte is shifted into the RSR while the FIFO is full. Therefore, the ASCI receiver is well buffered.

ASCI Status FIFO Registers

This four-entry, First-In First-Out (FIFO) Register contains Parity Error, Framing Error, Rx Overrun, and Break status bits associated with each character in the receive-data FIFO. The status of the oldest character (if any) can be read from the status registers.

ASCI Status Registers

ASCI Status Register 0 (STAT0) and ASCI Status Register 1 (STAT1) contain status information about ASCI communication, error and modem control signals. These registers support the enabling or disabling of ASCI interrupts.

ASCI Status Register 0 (STAT0: 04H)

7	6	5	4	3	2	1	0
RDRF	OVRN	PE	FE	RIE	$\overline{\text{DCD0}}$	TDRE	TIE
0	0	0	0	0	0	1	0
R	R	R	R	R/W	R	R	R/W

Bit Number	Field	R/W	Reset Value	Description
7	RDRF	R	0	Receive Data Register Full 1: An incoming data byte is remains loaded into an empty Rx FIFO. When a framing or parity error occurs, RDRF is set, and the receive data (which generated the error) remains loaded into the FIFO. 0: RDRF is cleared to 0 by reading the last character in the FIFO from the RDR, in IOSTOP Mode, during Reset, and, for ASCI0, if the $\overline{\text{DCD0}}$ input is Auto-enabled and is High.
6	OVRN	R	0	**Overrun Error** An overrun condition occurs when the receiver is finished assembling a character and the Rx FIFO is too full to receive another character. When an overrun occurs, the receiver does not place the character in the shift register into the FIFO, or any subsequent characters, until the last good character reaches the top of the FIFO so that OVRN is set, and software then writes a 1 to EFR to clear it. 1: This status bit is set when the last character received before the overrun becomes the oldest byte in the FIFO. This bit is cleared to 0 when software writes a 0 to the EFR bit in the CNTLA register, and also by Reset, in IOSTOP Mode, and for ASCI0 if the $\overline{\text{DCD0}}$ pin is Auto-enabled and is High.

Bit Number	Field	R/W	Reset Value	Description
5	PE	R	0	**Parity Error** A parity error is detected when parity checking is enabled by the MOD1 bit in the CNT1LA register (set to1), and a character is assembled in which the parity does not match the PEO bit in the CNTLB register. 1: PE is set when the error character becomes the oldest character in the RxFIFO. 0: PE is cleared to 0 when software writes a 0 to the EFR bit in the CNTRLA register, and also by Reset, in IOSTOP Mode, and for ASCI0 if the $\overline{\text{DCD0}}$ pin is Auto-enabled and is High.
4	FE	R	0	**Framing Error** A framing error is detected when the Stop bit of a character is sampled as 0/Space. 1: FE is set when the error character becomes the oldest character in the RxFIFO. 0: FE is cleared to 0 when software writes a 0 to the EFR bit in the CNTLA register, and also by Reset, in IOSTOP Mode, and for ASCI0 if the $\overline{\text{DCD0}}$ pin is Auto-enabled and is High.
3	RIE	R/W	0	**Receive Interrupt Enable** 1: Enable ASCI receive interrupt requests. When RIE is 1, the Receiver requests an interrupt when a character is received and RDRF is set, but only if Bit 7 of the ASCI Extension Control Register is 0. When Bit 7 is 1, the ASCI does not request an interrupt for RDRF. When RIE is 1, an ASCI requests an interrupt when OVRN, PE or FE is set, and ASCI0 requests an interrupt when $\overline{\text{DCD0}}$ goes High. 0: Disable ASCI receive interrupt requests.

Bit Number	Field	R/W	Reset Value	Description
2	$\overline{\text{DCD}}_0$	R	0	**Data Carrier Detect** 1: When Bit 0 of the Interrupt Edge Register (IER0) is 0, the $\overline{\text{DCD0}}$ $\overline{\text{CKA1}}$ pin performs the $\overline{\text{DCD0}}$ function, and this bit is set to 1 when the pin is High. 0: The first read of STAT0 following the pin's transition from High to Low and during Reset. When IER0 is 0, Bit 6 of the ASEXT0 register is 0 to select Auto-enabling, and the pin is High, the receiver is reset and receiver operation is inhibited. Bit 2 of STAT1 is reserved.
1	TDRE	R	1	**Transmit Data Register Empty** The TDR is empty and the next transmit data byte can be written to the TDR. After the byte is written to the TDR, TDRE is cleared to 0 until the ASCI transfers the byte from the TDR to the TSR and then TDRE is again set to 1. TDRE is set to 1 in IOSTOP Mode and during Reset. On ASCI0, the TDRE bit is inhibited (forced to 0) if the $\overline{\text{CTS0}}$ pin is Auto-enabled in the ASEXT0 register and the pin is High. 1: TDR is empty; or, IOSTOP Mode is in effect. 0: TDR is not empty; or, $\overline{\text{CTS0}}$ is High and Bit 5 of ASEXT0 is 0.
0	TIE	R/W	0	**Transmit Interrupt Enable** 1: Enable ASCI transmit interrupt requests. When TIE is 1, an interrupt is requested when TDRE is set to1. 0: Disable ASCI receive interrupt requests.

ASCI Status Register 1 (STAT1: 05)

7	6	5	4	3	2	1	0
RDRF	OVRN	PE	FE	RIE	Reserved	TDRE	TIE
0	0	0	0	0		1	0
R	R	R	R	R/W		R	R/W

The fields of the ACSI Status Register 1 (STAT1) are identical to those for ACSI Status Register 0 (STAT0).

ASCI CONTROL REGISTERS

ASCI-Channel Control Register A (CNTLA0) and ASCI-Channel Control Register B (CNTLA1) configure the major operating modes such as receiver/transmitter enable and disable, data format, and multiprocessor communication modes.

ASCI Control Register A0 (CNTLA0: 00H)

7	6	5	4	3	2	1	0
MPE	RE	TE	$\overline{\text{RTS0}}$	MPBR/ EFR	MOD2	MOD1	MOD0
0	0	0	1	-	-	-	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit Number	Field	R/W	Reset Value	Description
7	MPE	R/W	0	**Multi-Processor Mode Enable** The ASCI performs a multiprocessor communication mode that utilizes an extra data bit for selective communication when a number of processors share a common serial bus. Multiprocessor data format is selected when the MP bit in CNTLB is set to 1. When multiprocessor mode is not selected (MP bit in CNTLB is 0), MPE has no effect. When multiprocessor mode is selected, MPE enables or disables the wake-up feature as follows. When MPE is set to 1, only received bytes in which the multiprocessor bit (MPB) is 1 are received and affect the RDRF and error flags. Other bytes (when MPB is 0) are ignored by the ASCI. When MPE is reset to 0, all bytes, regardless of the state of the MPB data bit, are received and affect the RDRF and error flags.
6	RE	R/W	0	**Receiver Enable** 1: ASCI receiver is enabled. 0: Receiver is disabled, and any receive operation in progress is interrupted. However, the RDRF and error flags are not reset and the previous contents of RDRF and error flags are held. RE is cleared to 0 in IOSTOP Mode and during Reset.

Bit Number	Field	R/W	Reset Value	Description
5	TE	R/W	0	**Transmitter Enable** 1: ASCI transmitter is enabled. 0: Transmitter is disabled, and any transmit operation in progress is interrupted. However, the TDRE flag is not reset and the previous contents of TDRE are held. TE is cleared to `0` in IOSTOP Mode and during Reset.
4	RTS0	R/W	1	**Request to Send Channel 0** When Bit 5 of the System Configuration Register is `0`, the $\overline{\text{RTS0}}$ $\overline{\text{TxS}}$ pin performs the $\overline{\text{RTS0}}$ function and tracks this bit in a real-time, positive-logic fashion (`1` makes the pin High and a `0` makes it Low). Bit 4 in `CNTLA1` is not used. 1: $\overline{\text{RTS0}}$ $\overline{\text{TxS}}$ pin is High 0: $\overline{\text{RTS0}}$ $\overline{\text{TxS}}$ pin is Low

Bit Number	Field	R/W	Reset Value	Description
3	MPBR/ EFR	R/W	-	**Multiprocessor Bit Receive/Error Flag Reset** When multiprocessor mode is enabled (MP in CNTLB is 1), MPBR, when read, contains the value of the MPB bit for the last receive operation. 1: No effect. 0: The EFR function is selected to reset all error flags (OVRN, FE, PE and BRK in the ASEXT Register) to 0. MPBR EFR is undefined during Reset.
2-0	MOD2 MOD1 MOD0	R/W	-	**ASCI Data Format Mode** These bits program the ASCI data format as follows: MOD2 1: 8-Bit data 0: 7-Bit data MOD1 1: Parity enabled 0: No parity MOD0 1: 2 Stop bit 0: 1 Stop bits The data formats available based on all combinations of MOD2, MOD1, and MOD0 are shown below.

Table 8. ASCI Data Formats

MOD2	MOD1	MOD0	Data Format
0	0	0	Start + 7 bit data + 1 stop
0	0	1	Start + 7 bit data + 2 stop
0	1	0	Start + 7 bit data + parity + 1 stop
0	1	1	Start + 7 bit data + parity + 2 stop
1	0	0	Start + 8 bit data + 1 stop
1	0	1	Start + 8 bit data + 2 stop
1	1	0	Start + 8 bit data + parity + 1 stop
1	1	1	Start + 8 bit data + parity + 2 stop

ASCI Control Register A1 (CNTLA1: 01H)

7	6	5	4	3	2	1	0
MPE	RE	TE	Reserved	MPBR/ EFR	MOD2	MOD1	MOD0
1	1	0		0	x	x	x
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The fields of the ACSI Control Register 1 (CNTLA1) are identical to those for ACSI Control Register 0 (CNTLA0).

ASCI CONTROL REGISTER B0, 1

Each ASCI-Channel Control Register B 0 (CNTLB0) and ASCT Control Register B1 (CNTLB1) configure multiprocessor mode, parity, and baud-rate selection.

ASCI Control Register B0 (CNTLB0: 02H)

7	6	5	4	3	2	1	0
MPBT	MP	$\overline{\text{CTS}}$/PS	PEO	DR	SS2	SS1	SS0
-	0	0	0	0	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit Number	Field	R/W	Reset Value	Description
7	MPBT	R/W	-	**Multiprocessor Bit Transmit** When multiprocessor communication format is selected (Bit 6 is 1), MPBT is used to specify the MPB data bit for transmission. 1: The value 1 is transmitted. 0: The value 0 is transmitted. MPBT state is undefined during and after Reset.
6	MP	R/W	0	**Multiprocessor Mode** 1: The data format is configured for multiprocessor mode based on the MOD2 (number of data bits) and MOD0 (number of stop bits) bits in CNTLA. The format is as follows: Start bit + 7 or 8 data bits + MPB bit + 1 or 2 stop bits The multiprocessor format (MP is 1) has no provision for parity. 0: The data format is based on MOD0, MOD1, MOD2, and may include parity.

Bit Number	Field	R/W	Reset Value	Description
5	$\overline{\text{CTS}}$/PS	R/W	0	**Clear to Send/Prescale** When Bit 5 of the System Configuration Register is 0, the $\overline{\text{CTS0 RxS}}$ pin performs the $\overline{\text{CTS0}}$ function, and the state of the pin can be read in Bit 5 of CNTLB0 in a real-time, positive-logic fashion (High is 1, Low is 0). When Bit 5 in the System Configuration Register is 0, Bit 5 of ASEXT0 is 0 to Auto-enable $\overline{\text{CTS0}}$, and the pin is High, the TDRE bit is Inhibited (forced to 0). Bit 5 of CNTLB1 reads back as 0. When the Bits 2..0 in this register are not 111 and Bit 3 (BRG0 Mode bit) in the ASEXT register is 0, then writing Bit 5 CNTLB0 sets the prescale (PS) control as described in the “Clock Modes” section. 1: Divide by 30 0: Divide by 10
4	PEO	R/W	0	**Parity Even Odd** PEO selects Even or Odd parity. PEO does not affect the enabling/disabling of parity, which is controlled by Bit 1 (the MOD1 bit) of CNTLA. 1: Odd parity 0: Even parity

Bit Number	Field	R/W	Reset Value	Description
3	DR	R/W	0	**Divide Ratio** This bit specifies the divider used to obtain the baud rate from the data sampling clock, when Bit 4 (the X1 bit) in the ASEXT register is 0 1: Divide by 64 0: Divide by 16
2..0	SS2, SS1, SS0	R/W	1	Source/Speed Select When these bits are 111, as they are after a Reset, the CKA pin is specified as a clock input, and is divided by 1, 16, or 64 depending on the DR bit and Bit 4 (the X1 bit) in the ASEXT register. When these bits are not 111 and Bit 3 (the BRG Mode bit) in the ASEXT is 0, then these bits specify a power-of-two devisor for the PHI clock as shown in the table below. Setting or leaving these bits at 111 is appropriate for a channel only when its CKA pin is selected for the CKA function. CKA0 $\overline{\text{CKS}}$ performs the CKA0 function when Bit 4 of the System Configuration Register is 0. $\overline{\text{DCD0}}$ $\overline{\text{CKA1}}$ performs the CKA1 function when Bit 0 of the Interrupt Edge Register is 1. SS2SS1SS0Divide Ratio 000³1 001÷2 010÷4 011÷8 100÷16 101÷32 110÷64 11 1External Clock

ASCI Control Register B1 (CNTLB1: 03H)

7	6	5	4	3	2	1	0
MPBT	MP	$\overline{\text{CTS}}$/PS	PEO	DR	SS2	SS1	SS0
-	0	0	0	0	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The fields of the ACSI Control Register B1 (CNTLB1) are identical to those for ACSI Control Register B0 (CNTLB0).

ASCI Extension Control Registers

ASCI Extension Control Register 0 (ACEXT0) and ASCI Extension Control Register 1 (ASEXT1) control functions that have been added to the ASCIs in the Z8018x family.

ASCI Extension Control Register 0 (ASEXT0: 12H)

7	6	5	4	3	2	1	0
RDRF0 DI	Disable DCD0	CTS0 Disable	X1	BRG0 Mode	Break Enab	Break	Send Break
–	–	–	–	–	–	–	–
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit Number	Field	R/W	Reset Value	Description
7	RDRF0 D	R/W	–	**RDRF Interrupt Disable** When this bit is `1`, an ASCI does not request receive data interrupts. This bit sets to `1` when a DMA channel is used to handle an ASCI's receive data.
6	Disable DCD0	R/W	–	**DCD0 Disable ASCI0 Only** When Bit 0 of the Interrupt Edge Register is `0` (to select the DCD0 function for the DCD0 CKA1 pin), and this bit is `0`, then the DCD0 pin Auto-enables the ASCI0 receiver, so that when the pin is negated/High, the Receiver is held in a reset state. When Bit 0 of the `IER` is `0` and this bit is `1`, the state of the DCD0 pin has no effect on receiver operation. In either state of this bit, software can read the state of the DCD0 pin in the `STAT0` register, and the receiver interrupts on a rising edge of DCD0 when its RIE bit is `1`.
5	CTS0 Disable	R/W	–	**CTS0 Disable ASCI0 Only** When Bit 5 of the System Configuration Register is `0` (to select the CTS0 function of the CTS0 RxS pin), and this bit is `0`, then the CTS0 pin Auto-enables the ASCI0 transmitter, so that when the pin is negated/High, the `TDRE` bit in the `STAT0` register is forced to `0`. When Bit 5 of the System Configuration Register is `0` and this bit is `1`, the state of the CTS0 pin has no effect on the transmitter. Regardless of the state of this bit, software can read the state of the CTS0 pin in the `CNTLB0` register.

Bit Number	Field	R/W	Reset Value	Description
4	X1	R/W	–	**X1** 1: The clock from the Baud Rate Generator or CKA pin is taken as a 1X bit clock (this is sometimes called isochronous mode). In this mode, receive data on the RXA pin must be synchronized to the clock on the CKA pin, regardless of whether CKA is an input or an output. 0: The clock from the Baud Rate Generator or CKA pin is divided by 16 or 64 as specified by the DR bit in the CNTLB register, to obtain the actual bit rate. In this mode, receive data on the RXA pin need not be synchronized to a clock.
3	BRG0 Mode	R/W	–	**BRG Mode** When the SS2-0 bits in the CNTLB register are not 111, and this bit is 0, this ASCI's Baud Rate Generator divides PHI by 10 or 30 depending on the DR bit in CNTLB. Then this number is ------------ by a power of two, which is selected by the SS2-0 bits, to obtain the clock that is presented to the transmitter and receiver, which is then output on the CKA pin. When SS2-0 are not 111, and this bit is 1, the Baud Rate Generator divides PHI by the quantity (twice the 16-bit value programmed into the Time Constant Registers, plus two). This mode is identical to the operation of the baud rate generator in the ESCC.
2	Break Enab	R/W	–	**Break Enable** 1: The receiver detects Break conditions and reports them in Bit 1, and the transmitter sends Breaks to the control of Bit 0.

Bit Number	Field	R/W	Reset Value	Description
1	Break	R/W	–	**Break Detect** 1: The receiver sets this read-only bit to `1` when an all-zero character with a Framing Error becomes the oldest character in the Rx FIFO. 0: The bit is cleared to `0` when software writes a `0` to the EFR bit in `CNTLA` register, also by Reset, by `IOSTOP` Mode, and for ASCI0 if the $\overline{\text{DCD0}}$ pin is Auto-enabled and is High.
0	Send Break	R/W	–	**Send Break** 1: When this bit and Bit 2 are both `1`, the transmitter holds the TXA pin Low to send a Break condition. The duration of the Break is under software control (one of the PRTs or CTCs can be used to time it). 0: The TXA carries the serial output of the transmitter

ASCI Extension Control Register 1 (`ASEXT1: 13H`)

7	6	5	4	3	2	1	0
RDRF1 DI			X1	BRG1 Mode	Break Enab	Break	Send Break
–	0	0	0	0	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The fields of the ACSI Extension Control Register 1 (`ASEXT1`) are identical to those for ACSI Extension Control Register 0 (`ASEXT0`).

ASCI TIME CONSTANT REGISTERS

When the SS2-0 bits of the `CNTLA` register are not `111`, and the BRG Mode bit in the `ASEXT` register is `1`, the ASCI divides the PHI clock by (twice the 16-bit value in these registers, plus two), to obtain the clock that is presented to the transmitter and receiver for division by 1, 16, or 64, and that value is output on the CKA1 pin.

ASCI Time Constant Register 0 Low (ASTC0L: 1AH) and ASCI Time Constant Register 1 Low (ASTC1L: 1CH)

Note: ASTC0L0,1 are 16 bits. Cannot be the 8 LSBs. The same is true for ASTC0H0,1.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Least Significant 8 Bits of Time Constant</td></tr>
</table>

ASCI Time Constant Register 0 High (ASTC0H: 1BH) and ASCI Time Constant Register 1 High (ASTC1H: 1DH)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Most Significant 8 Bits of Time Constant</td></tr>
</table>

CLOCKING SUMMARY

The sampling rate is 1, 16, or 64 depending on Bit 4 (the X1 bit) in the ASEXT register and the DR bit *What bit is that? CNTLA does not have a DR Bit.* in the CNTLA register:

Table 9. Sample Rate

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

When the SS2..SS0 bits in the `CNTLA` register are `111`, and the multiplexed CKA pin is selected for the CKA function, then the pin is used as a clock input, and is divided by the sampling rate by the receiver and transmitter:

```
bits/second = fCKAin / Sampling Rate
```

Note: In the following formulas:
PS selects between a prescaler of 10 and 30, and 2^SS2-0 is a power of two between 1 and 64.

When the SS2..SS0 bits are not `111`, and Bit 3 (the BRG0 Mode bit) in the `ASEXT` register is `0`, then the baud rate generator divides PHI for serial clocking.

```
bits/second = fPHI/ ((10+20*PS) *2^SS2-0 *Sampling
Rate)
```

When the multiplexed CKA pin is selected for the CKA function, it outputs the clock before the final division by the Sampling Rate:

```
fCKAout = fPHI/((10+20*PS)*2^SS2..SS0)
```

Note: In the following formulas:
TC is the 16-bit value programmed into the TC High and Low registers.

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

```
TC = (fPHI/(2*bits/second*Sampling Rate)) - 2
```

When the SS2..SS0 bits are not `111`, and Bit 3 (the BRG Mode) bit is `1`, the baud rate generator divides PHI for serial clocking as on the ESCC:

```
bits/second = fPHI/(2*(TC+2)*Sampling Rate)
```

When the multiplexed CKA pin is selected for the CKA function, it outputs the clock before the final division by the Sampling Rate:

```
fCKAout = fPHI/(2*(TC+2))
```

MODEM CONTROL

Signals

ASCI Channel 0 has $\overline{\text{CTS0}}$, $\overline{\text{DCD0}}$, $\overline{\text{RTS0}}$ and a clock (CKA0), which are multiplexed with the CSI/O signals and with the ASCI Channel 1 clock (CKA1). It is possible to run ASCI0, ASCI1, and the CSI/O simultaneously, as long as the only modem control or ASCI clock is either $\overline{\text{DCD0}}$ or CKA1.

Table 10. Modem Control Signals

Mnemonic	Name	Description
CTS0	Clear to Send 0 (Input)	When Bit 4 of the System Configuration Register is 0, the CTS0/RxS pin performs the CTS0 function. The state of the pin can be read by software as Bit 5 of the CNTLB0 register. When Bit 5 of the ASEXT0 register is 0, the pin provides external control (start/stop) of ASCI channel 0 transmit operations. When CTS0 is High, the channel 0 TDRE bit is held at 0 whether or not TDR0 (the Transmit Data Register) is full or empty. When CTS0 is Low, TDRE indicates the state of TDR0. The actual transmit operation is not disabled by CTS0 High, only TDRE is inhibited.
DCD0	Data Carrier Detect 0 (Input)	When Bit 0 of the Interrupt Edge Register is 0, the DCD0/CKA1 pin performs the DCD0 function. The state of the pin can be read by software as Bit 2 of the STAT0 register. When Bit 6 of the ASEXT0 register is 0, the pin provides external control of ASCI channel 0 receive operations. When DCD0 is High, the channel 0 RDRF bit is held at 0 whether or not the RDR0 (the Receive Data Register) is full or empty. The error flags (PE, FE, and OVRN bits) are also held at 0. Even after the DCD0 input goes Low, these bits do not resume normal operation until the status register (STAT0) is read. This first read of STAT0, while enabling normal operation, continues to indicate the DCD0 input is High (DCD0 bit is 1) even though it has gone Low. Therefore, read the STAT0 register twice to ensure that the DCD0 bit is reset to 0.
RTS0	Request to Send 0 (Output)	When Bit 4 of the System Configuration Register is 0, the RTS0/TxS pin performs the RTS0 function. RTS0 is essentially a 1 bit output port, having no other effects on other ASCI registers or flags.

Timing

Modem control signal timing is depicted in Figure 49 and Figure 50.

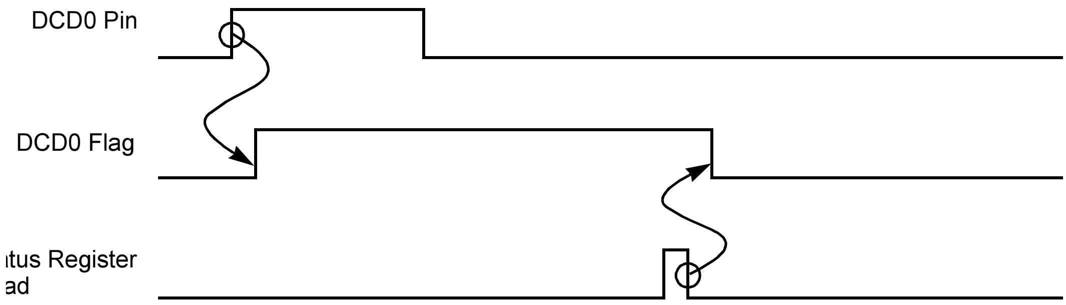


Figure 49. Read Timing

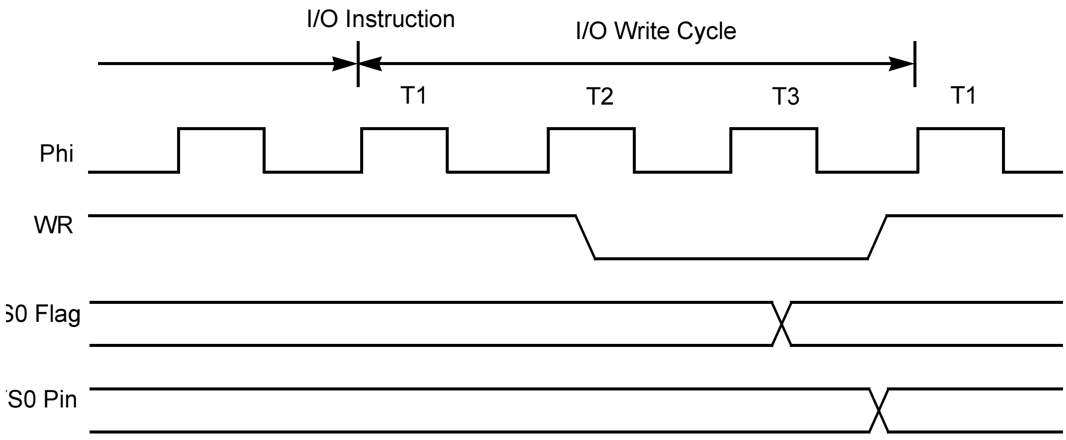


Figure 50. Write Timing

ASCI INTERRUPTS

Interrupts for the Transmitter and Receiver are enabled by the TIE and RIE bits in STAT register, respectively. When Bit 7 of the ASEXT register is 1, then the ASCI does not request Receive Data Interrupts. That is, if ASEXT Bit 7 is 1, an ASCI does not include RDRF in its Receive Interrupt Request Figure 51 illustrates the ASCI Interrupt Request Circuit diagram

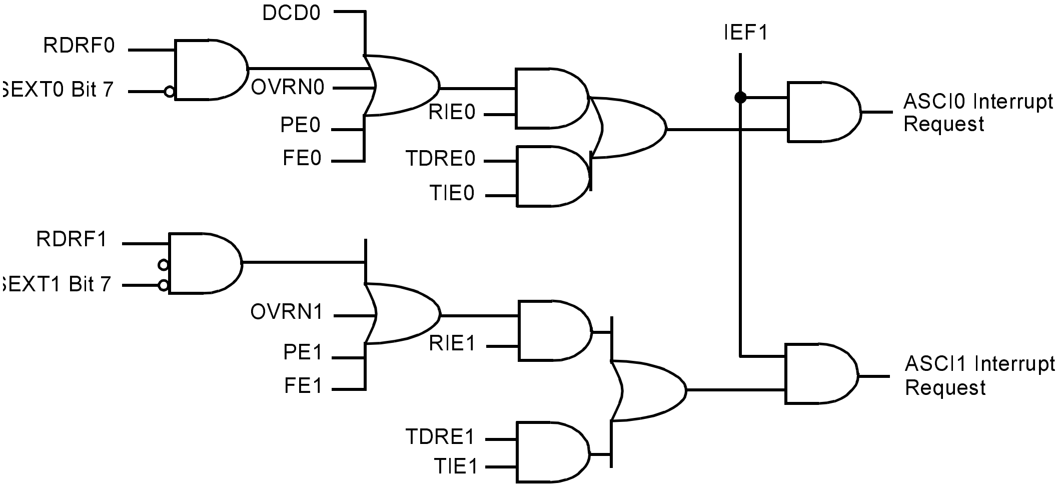


Figure 51. ASCI Interrupt Request Circuit Diagram

ASCI/DMAC OPERATION

Operation of the ASCIs with the on-chip DMA channels requires the DMAC be correctly configured to use the ASCI flags as DMA request signals. An ASCI disables its receive DMA request when any of the error flags PE, FE, OVERN, or BRK are set, so that software can analyze which character has a problem. Bit 7 of the ASEXT register can be set to 1 and the RIE bit can be set to 1, to enable receive interrupts only on error corrections.

ASCI AND RESET

During RESET, the ASCI status and control registers are initialized as defined in the individual register descriptions. Receive and Transmit operations are stopped during RESET, and the RxFIFO clears to 0.

Clocked Serial I/O Port

FEATURES

- A simple, high-speed clock
- Synchronous serial I/O port
- Transmit/receive (half-duplex), fixed 8-bit data
- Internal or external data clock selection
- High-speed operation (up to PHI/20 bits/second)

The CSI/O is ideal for implementing a multiprocessor communication link between multiple Z80185/Z80195 family members.

The three pins associated with the CSI/O are multiplexed with auxiliary pins for ASCI0. Bit 4 of the System Configuration Register must be `1` to use the CSI/O as described in this section.

CSI/O BLOCK DIAGRAM

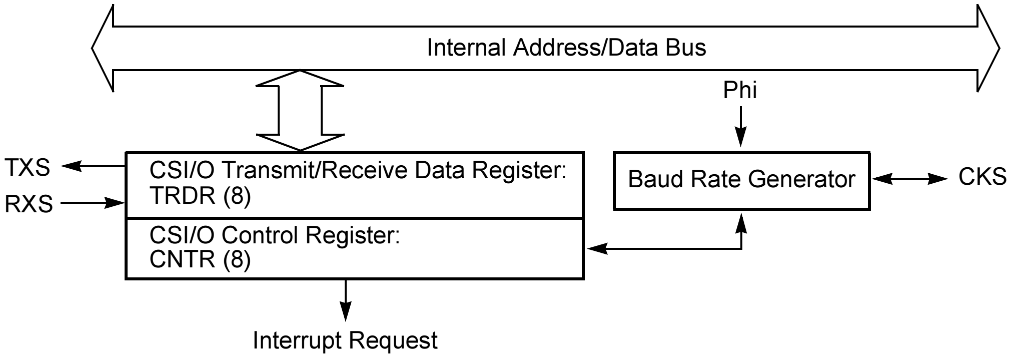


Figure 52. CSI/O Block Diagram

CSI/O REGISTERS

The CSI/O includes two registers: the Transmit/Receive Data Register (TRDR) and Control Register (CNTR).

CSI/O Transmit/Receive Data Register (TRDR:0BH)

The TRDR is used for both CSI/O transmission and reception. The system design must ensure that the constraints of half-duplex operation are met (Transmit and Receive operation cannot occur simultaneously). CSI/O transmission cannot be performed while the CSI/O is receiving data. The TRDR is not buffered. Therefore, attempting to perform a CSI/O transmit while the previous transmit data is still being shifted out causes the shift data to be immediately updated, thereby corrupting the transmit operation in progress. The TRDR must not be read while a transmit or receive is in progress

CSI/O Control/Status Register (CNTR:0AH)

CNTR is used to monitor CSI/O status, enable and disable the CSI/O, enable and disable interrupt generation, and select the data clock speed and source.

7	6	5	4	3	2	1	0
EF	EIE	RE	TE	–	SS2	SS1	SS0
R	R/W	R/W	R/W		R/W	R/W	R/W

Bit Number	Field	R/W	Reset Value	Description
7	End Flag E	R	0	EF is set to 1 by the CSI/O to indicate completion of an 8-bit data transmit or receive operation. If EIE (End Interrupt Enable) bit is 1 when EF is 1, a CPU interrupt request is generated. Program access of TRDR only occurs if EF is 1. The CSI/O clears EF to 0 when TRDR is read or written. EF is 0 during RESET and IOSTOP mode.
6	End Interrupt Enable	R/W	0	EIE is set to 1 to generate a CPU interrupt request. The interrupt request is inhibited if EIE is reset to 0. EIE is cleared to 0 during RESET.
5	Receive Enable	R/W	0	A CSI/O receive operation is started by setting RE to 1. When RE is set to 1, the data clock is enabled. In internal clock mode, the data clock is output from the CKS pin. In external clock mode, the clock is input on the CKS pin. In either case, data is shifted in on the RXS pin in synchronization with the (internal or external) data clock. After receiving 8 bits of data, the CSI/O automatically clears RE to 0, sets EF to 1, and generates an interrupt (if enabled by EIE is 1). RE and TE are never both set to 1 at the same time. RE is cleared to 0 during RESET and ISTOP mode.

Bit Number	Field	R/W	Reset Value	Description
4	Transmit Enable	R/W	0	A CSI/O transmit operation is started by setting TE to 1. When TE is set to 1, the data clock is enabled. When in internal clock mode, the data clock is output from the CKS pin. In external clock mode, the clock is input on the CKS pin. In either case, data is shifted out on the TXS pin synchronous with the (internal or external) data clock. After transmitting 8 bits of data, the CSI/O automatically clears TE to 0, sets EF to 1 and generates an interrupt if EIE is 1. TE and RE are never both set to 1 at the same time. TE is cleared to 0 during RESET and IOSTOP mode.
3				
2	SS2	R/W	0	SS2, SS1 and SS0 select the CSI/O transmit/receive clock source and speed. SS2, SS1 and SS0 are all set to 1 during RESET.
1	SS1	R/W	0	
0	SS0	R/W	0	

Table 11 describes the CSI/O Baud Rate Selection.

Table 11. CSI/O Baud Rate Selection

SS2	SS1	SS0	Divide Ratio
0	0	0	^{3}20
0	0	1	^{3}40
0	1	0	^{3}80
0	1	1	^{3}160
1	0	0	^{3}320
1	0	1	^{3}640
1	1	0	^{3}1280
1	1	1	External Clock Input (frequency must be less than PHI ^{3}20.)

After RESET, the CKS pin is configured as an external clock input (SS2, SS1, SS0 are all 1). Changing these values causes CKS to become an output pin; the selected clock is output only when transmit or receive operations are in operation.

CSI/O Interrupts

The CSI/O interrupt request circuit is depicted in Figure 53.

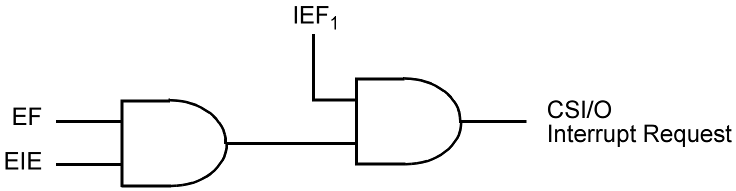


Figure 53. CSI/O Interrupt Request Generation

CSI/O OPERATION

The CSI/O is operated using status polling or interrupt-driven algorithms.

Transmit - Polling

1. Poll the TE bit in the CNTR until TE is 0.
2. Write the transmit data into the TRDR.
3. Set the TE bit in the CNTR to 1.
4. Repeat steps 1, 2 and 3 for each transmit data byte.

Transmit - Interrupts

1. Poll the TE bit in the CNTR until TE is 0.
2. Write the first transmit data into the TRDR.

3. Set the TE and EIE bits in the CNTR to 1.
4. When the transmit interrupt occurs, write the next transmit data byte into the TRDR.
5. Repeat steps 3 and 4 for each transmit data byte.

Receive - Polling

1. Poll the RE bit in the CNTR until RE = 0
2. Set the RE bit in the CNTR to 1
3. Poll the RE bit in the CNTR until RE = 0
4. Read the receive data from the TRDR
5. Read 2 to 4 for each receive data byte

Receive - Interrupts

1. Poll the RE bit in the CNTR until RE = 0.
2. Set the RE and EIE bits in the CNTR to 1.
3. When the receive interrupt occurs, read the receive data from the TRDR.
4. Repeat steps 2 and 3 for each receive data byte.

CSI/O OPERATION TIMING NOTES

Transmitter clocking and receiving sampling timing are different in internal and external timing modes. Figures 54 through 57 illustrate CSI/O Transmit/Receive Timing.

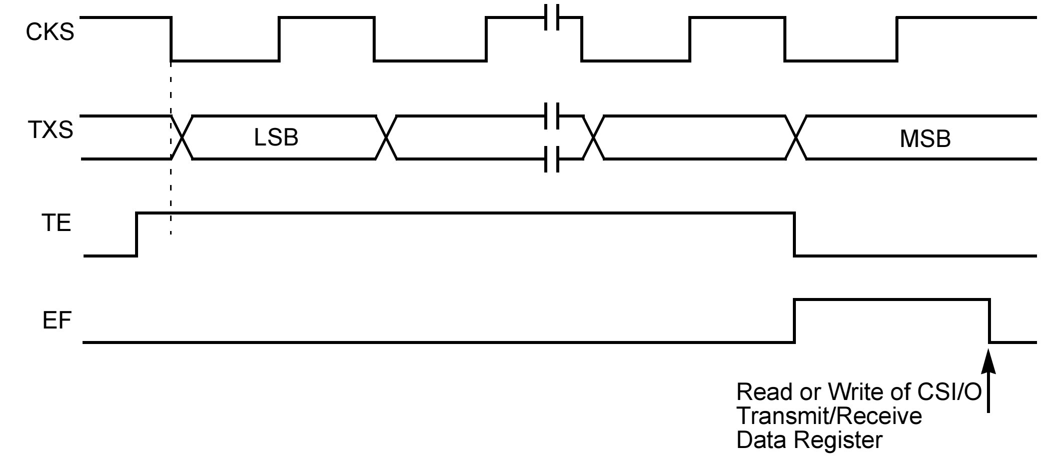


Figure 54. Transmit Timing Internal Clock

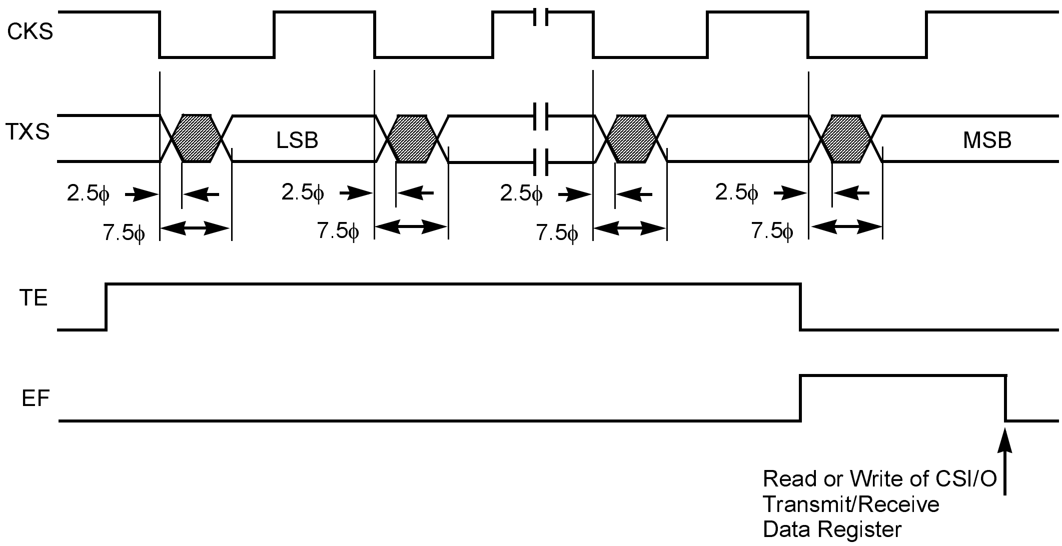


Figure 55. Transmit Timing External Clock

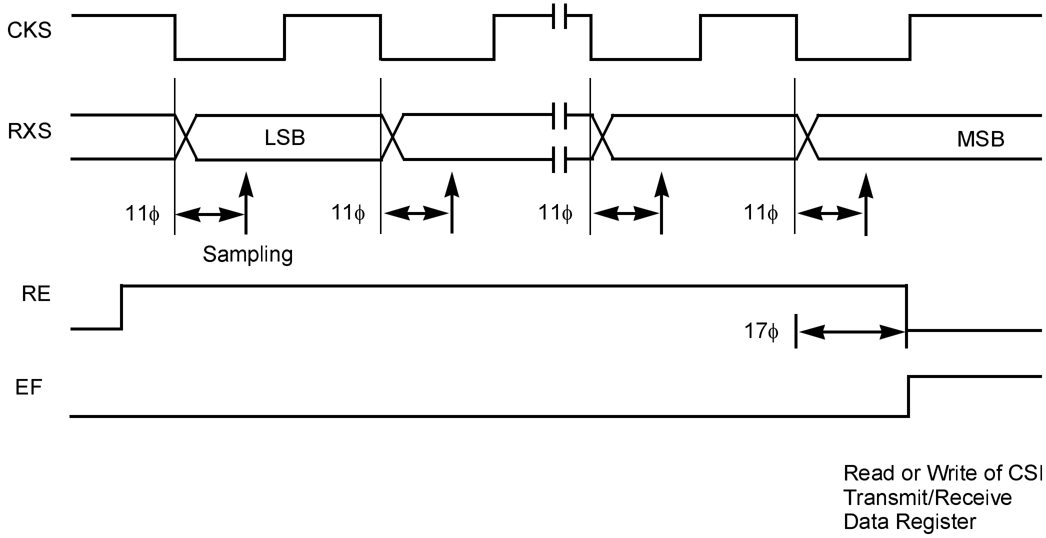


Figure 56. Receive Timing Internal Clock

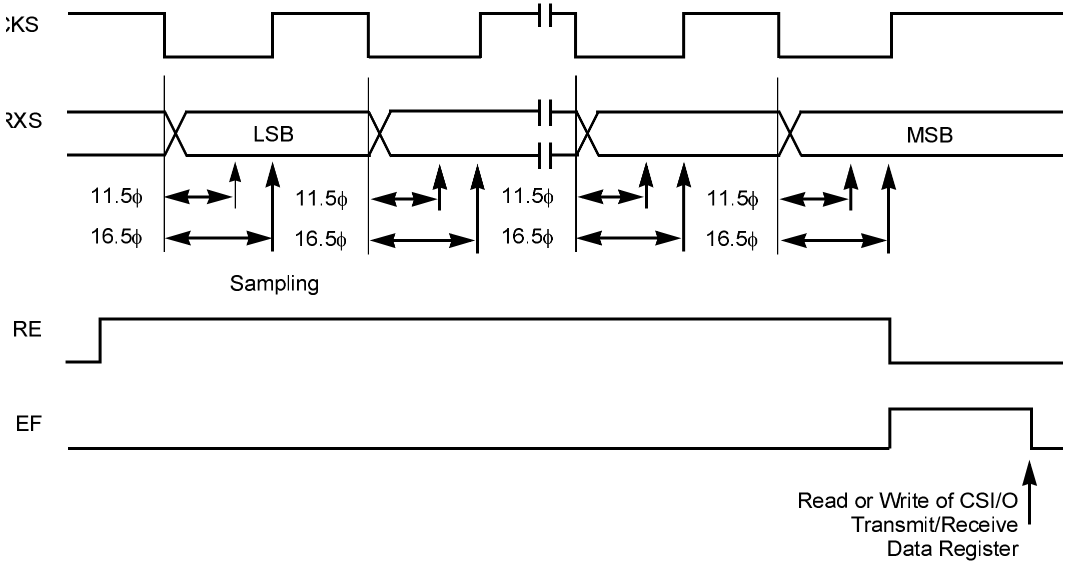


Figure 57. Receive Timing External Clock

CSI/O OPERATION NOTES

1. Disable the transmitter and receiver (TE and RE is 0) before initializing or changing the baud rate. When changing the baud rate after completion of transmission or reception, a delay of at least one bit time is required before baud rate modification.
2. When RE or TE is cleared to 0 by software, a corresponding receive or transmit operation is immediately terminated. Normally, TE or RE is only cleared to 0 when EF is 1.
3. Simultaneous transmission and reception is not possible. Thus, TE and RE must not both be 1 at the same time.

CSI/O AND RESET

During RESET each bit in the CNTR is initialized as defined in the CNTR register description.

CSI/O transmit and receive operations in progress are aborted during RESET. However, the contents of the TRDR are not changed.

Programmable Reload Timers

INTRODUCTION

The Z80185/Z80195 contains a two-channel 16-bit Programmable Reload Timer (PRT). Each PRT channel contains a 16-bit Down Counter and a 16-bit reload register. The Down Counter can be directly read and written, and a Down Counter Overflow interrupt can be programmably enabled or disabled. Also, if Bit 3 of the IAR1B register is 1, the $T_{OUT}\overline{DREQ}$ pin has the T_{OUT} function, and can be set High, Low, or toggled when PRT Channel 1 (PRT1) counts down to 0. Therefore, PRT1 can perform programmable-output waveform generation.

PRT BLOCK DIAGRAM

The PRT block diagram is illustrated in Figure 58. The two channels have separate timer data and reload registers and a common status/control register. The PRT input clock for both channels is equal to the system clock divided by 20.

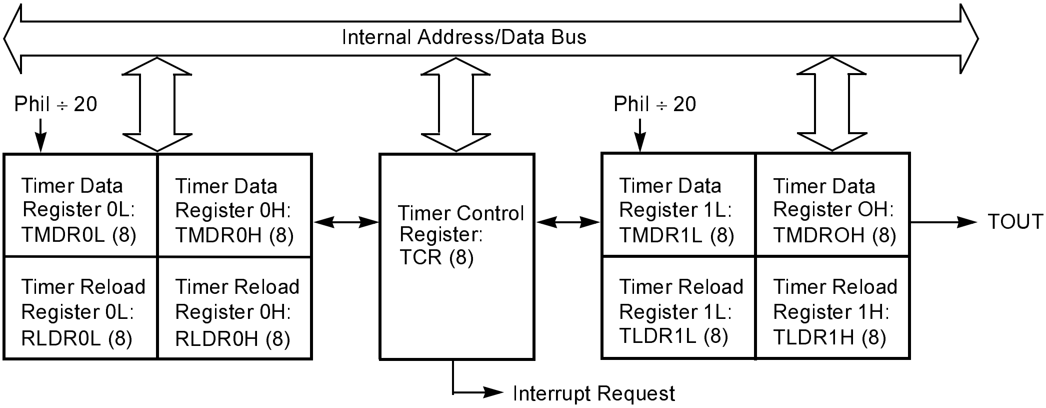

Internal Address/Data Bus
Phil ÷ 20
Phil ÷ 20
Timer Data Register 0L: TMDR0L (8)
Timer Data Register 0H: TMDR0H (8)
Timer Reload Register 0L: RLDR0L (8)
Timer Reload Register 0H: RLDR0H (8)
Timer Control Register: TCR (8)
Timer Data Register 1L: TMDR1L (8)
Timer Data Register OH: TMDROH (8)
Timer Reload Register 1L: TLDR1L (8)
Timer Reload Register 1H: TLDR1H (8)
TOUT
Interrupt Request

Figure 58. Programmable Reload Timer (PRT) Block Diagram

PRT REGISTERS

Timer Data Register

(TMDR: I/O Address = CH0, 0DH, 0CH, CH1: 15H, 14H). PRT0 and PRT1 each have 16-bit Timer Data Registers (TMDR). TMDR0 and TMDR1 are each accessed as Low and High byte registers (TMDR0H, TMDR0L and TMDR1H, TMDR1L). During reset, TMDR0 and TMDR1 are set to FFFFH.

The TMDR is decremented once every twenty clocks. When the TMDR counts down to 0, it is automatically reloaded with the value contained in the reload register (RLDR).

The TMDR is read and written by software using the following procedures. The read procedure uses a PRT internal temporary storage register to return accurate data without requiring the timer to be stopped. The write procedure requires the PRT to be stopped.

For reading (without stopping the timer), the TMDR is read in the order of lower byte/higher byte (TMDRnL, TMDRnH). The lower byte read (TMDRnL) stores the higher byte value in an internal register. The following higher byte read (TMDRnH) accesses this internal register. This procedure ensures timer data validity by eliminating the problem of potential 16-bit timer updating between the two 8-bit read. Specifically, reading the TMDR in higher byte/ower byte order may result in invalid data. Note the implications of TMDR higher byte internal storage for applications that may read only the lower or higher bytes. In normal operation, all TMDR read routines access both the lower and higher bytes, in that order. For writing, TMDR down counting must be inhibited using the Timer Down Count Enable (TDE) bits in the Timer Control Register (TCR). Then, the higher and lower bytes of TMDR can be written and read in any order.

Timer Reload Register (RLDR: I/O Address = CH0, 0EH, 0FH, CH1: 16H, 17H)

PRT0 and PRT1 each have 16-bit Timer Reload Registers (RLDR). RLDR0 and RLDR1 are each accessed as Low and High byte registers (RLDR0H, RLDR0L and RLDR1H, RLDR1L). During reset, RLDR0 and RLDR1 are set to FFFFH.

When the TMDR counts down to 0, it is automatically reloaded with the contents of RLDR.

Timer Control Register

Timer Control Register (TCR: 10H)

The TCR monitors both channels (PRT0, PRT1) status. It also controls enabling and disabling of down counting and interrupts.

7	6	5	4	3	2	1	0
TIF1	TIF0	TIE1	TIE0	TOC1	TOC0	TDE1	TDE0
0	0	0	0	0	0	1	0
R	R	R/W	R/W	R/W	R/W	R/W	R/W

Bit Number	Field	R/W	Reset Value	Description
7	TIF1	R	0	Timer Interrupt Flag 1 When TMDR1 decrements to 0, TIF1 is 1. This process generates an interrupt request if enabled by TIE1 as 1. TIF1 is reset to 0 when the TCR is read as well as when either byte of TMDR1 is read. During RESET, TIF1 is cleared to 0.
6	TIF0	R	0	Timer Interrupt Flag 0 When TMDR0 decrements to 0, TIF0 is set to 1. This process generates an interrupt request if enabled by TIE0 as 1. TIF0 is reset to 0 when TCR is read as well as when either byte of TMDR0 is read. During reset TIF0 is cleared to 0.
5	TIE1	R/W	0	Timer Interrupt Enable 1 When TIE1 is 1, when TIF1 is 1 TIE1 generates a CPU interrupt request. When TIE1 is reset to 0, the interrupt request is inhibited. During reset, TIE1 is cleared to 0.
4	TIE0	R/W	0	Timer Interrupt Enable 0 When TIE0 is 1, when TIF1 is 1 TIE0 generates a CPU interrupt request. When TIE0 is reset to 0, the interrupt request is inhibited. During reset, TIE0 is cleared to 0.

<table>
<tr><th>Bit Number</th><th>Field</th><th>R/W</th><th>Reset Value</th><th>Description</th></tr>
<tr><td>3..2</td><td>TOC1, 0</td><td>R/W</td><td>0</td><td>Timer Output Control
TOC1 and TOC0 control the output of PRT1 using the multiplexed $T_{OUT}\overline{DREQ}$ pin as shown in Figure 58. During RESET, TOC1 and TOC0 are cleared to 0. If Bit 3 of the IAR1B register is 1, the T_{OUT} function is selected. By programming TOC1 and TOC0, the $T_{OUT}\overline{DREQ}$ pin can be forced High, Low, or toggled when TMDR1 decrements to 0.
<table>
<tr><th>TOC1</th><th>TOC0</th><th></th><th>Output</th></tr>
<tr><td>0</td><td>0</td><td>Inhibited</td><td>The $T_{OUT}\overline{DREQ}$ pin is not affect the PRT.</td></tr>
<tr><td>0</td><td>1</td><td>Toggled</td><td rowspan="3">If Bit 3 of IAR1B is 1, the $T_{OUT}\overline{D}$ pin is toggled or set Low or High indicated.</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table>
</td></tr>
<tr><td>1..0</td><td>TDE1, 0</td><td>R/W</td><td>1</td><td>Timer Down Count Enable
TDE1 and TDE0 enable and disable down counting for TMDR1 and TMDR0, respectively. When TDEn (n is 0, 1) is set to 1, down counting is stopped and TMDRn can be read or written. TDE1 and TDE0 are cleared to 0 during RESET and TMDRn does not decrement until TDEn is 1.</td></tr>
</table>

PRT TIMING

Figure 59 illustrates timer initialization, count down, and reload timing. Figure 60 illustrates timer output ($T_{OUT}\overline{DREQ}$) timing.

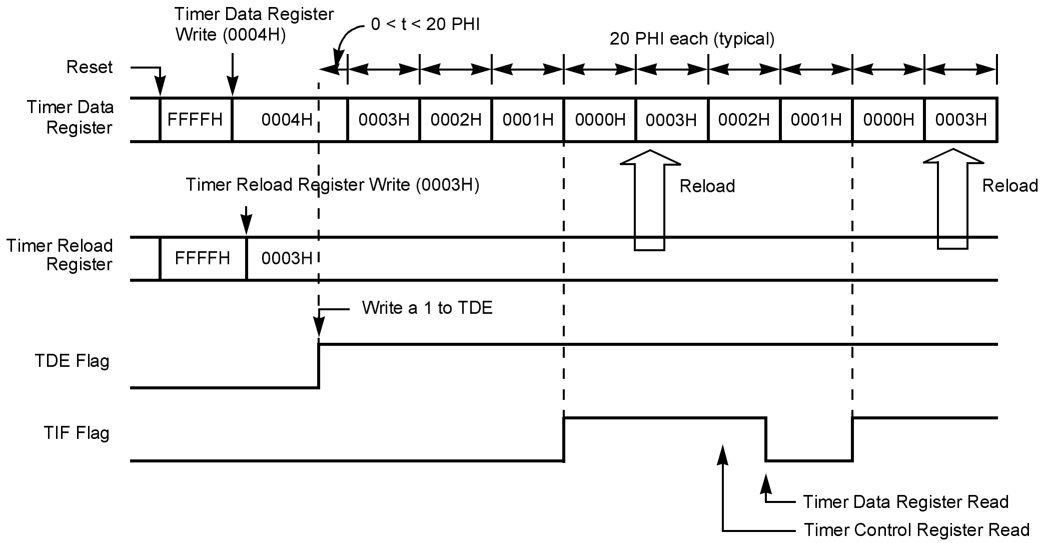


Figure 59. Timer Initialization, Count Down, and Reload Timing

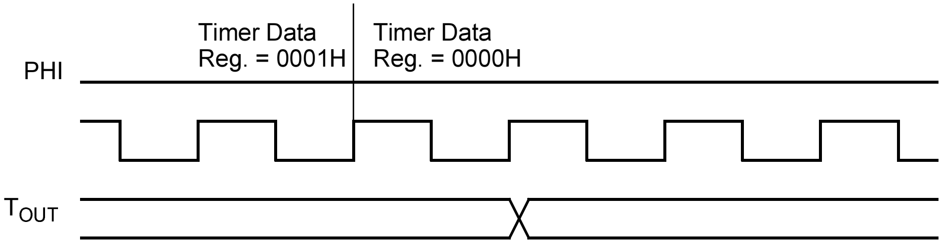


Figure 60. Timer Output Timing

PRT INTERRUPTS

The PRT interrupt request circuit is shown in Figure 61.

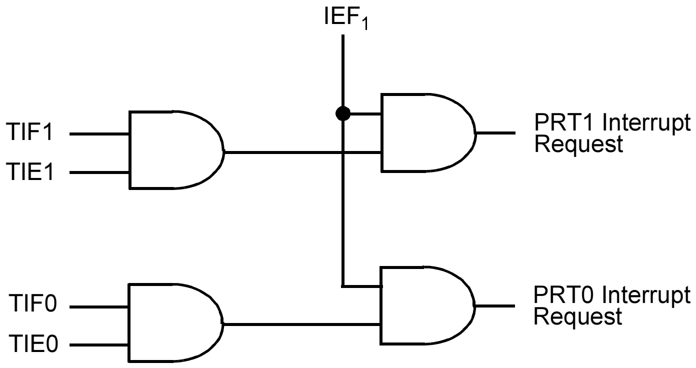


Figure 61. RPT Interrupt Generation

PRT AND RESET

During reset, the bits in the TCR are initialized as defined in the TCR register description. Down counting is stopped and the TMDR and RLDR registers are initialized to FFFFH.

PRT OPERATION NOTES

1. TMDR data can be accurately read without stopping down counting, by reading the lower (TMDRnL*) and higher (TMDRnH*) bytes in that order. Also, the TMDR can be read or written by stopping the down counting.
2. Take care that a timer reload does not occur during or between writing of the lower (RLDRnL*) and higher (RLDRnH*) byte writes. This

condition is guaranteed by system design/timing or by stopping down counting (with the TMDR containing a non-zero value) during the RLDR updating. Similarly, in applications where the TMDR is written at each TMDR overflow, the system/software design guarantees that RLDR can be updated before the next overflow occurs. Otherwise, time base inaccuracy occurs.

Note: **Note**: *n is 0, 1

Counter/Timer Channels

INTRODUCTION

The Z80185/Z80195 includes four independently programmable counter/timer channels called CTC0 through CTC3. Each Counter/Timer Channel (CTC) includes a prescaler that can divide by 16 or 256, and an 8-bit Down Counter that counts down from a programmable starting Time Constant value.

Each channel can operate in a Counter mode. Each channel counts transitions on a CLK/TRG input, or a Timer mode in which it is driven by the master PHI clock. Optionally, timing can be triggered by a transition on CLK/TRG. Each channel has a Zero Count/TimeOut (ZC/TO) output, three of which can be driven onto output pins. The CLK/TRG inputs and ZC/TO outputs are multiplexed with Parallel port 1 on pins of the Z80185/Z80195, and the ZC/TO outputs can be internally routed to CLK/TRG inputs of other CTCs, to produce longer timing intervals.

I/O ADDRESSES

Each CTC has one address in I/O space:

Channel	Address
CTC0	E4
CTC1	E5
CTC2	E6
CTC3	E7

As illustrated in Figure 62, at each of these addresses software can write to a Control Register or a Time Constant Register for that channel, or can write to an Interrupt Vector register that is shared in common among all four channels, and can read the value in the channel's Down Counter.

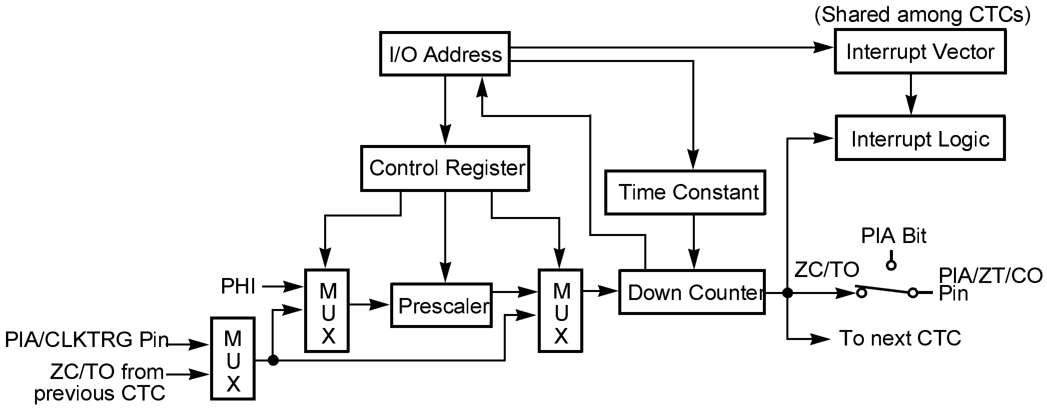


Figure 62. CTC Block Diagram

WRITING REGISTER

Normally, writing an even value (a value with Bit 0 at `0`) to any of the four CTCs writes to the Interrupt Vector register that is shared in common by all four CTCs. Writing an odd value (a value with Bit 0 is `1`) writes to the Control Register for that channel. After software writes a value with Bits 2 and 0 both `1` to the Control Register, that CTC enters a special mode in which the next value written to that CTC's I/O address are loaded into the CTC's Time Constant register, whether the value is odd or even. After software has written the Time Constant value, the CTC returns to its normal mode, in which the Control Register or Interrupt Vector register can be written.

CTC REGISTERS

Control Register (waiting for mnemonic)

7	6	54	4	3	2	1	0
Interrupt	Mode	Prescaler	CLK/TRG	Mode	Time Constant	Reset	Vector/ Control
R/W	R/W	R/W	R/W	R/W	W	R/W	R/W

Bit Number	Field	R/W	Reset Value	Description
7	Interrupt			1: This CTC requests an interrupt when its Down Counter reaches 0. 0: The CTC does not request an interrupt when its Down counter reaches 0.
6	Mode			1: Counter mode, in which the CTC counts transitions on its CLK/TRG input. 0: Timer mode, in which the PHI clock decrements the Prescaler, and the Prescaler decrements the Down Counter.
5	Prescaler Operation			1: Divide by 256. 0: Divide by 16.
4	CLK/ TRG Edge Selection			1: A rising edge on the CLK/TRG input enables timing and CTC counting. 0: A falling edge on the CLK/TRG input enables timing and CTC counting
3	Mode			Timer mode 1: CLK/TRG Pulse starts timing 0: Automatic trigger when Time Constant loads Counter mode 1: CLK/TRG decrements Prescaler. 0: CLK/TRG decrements Down Counter.

Bit Number	Field	R/W	Reset Value	Description
2	Time Constant			When this bit is 1, the CTC loads the next byte written to its I/O address into the Time Constant register. When this bit is 0, the CTC loads the next byte written into the Control Register or Interrupt Vector register, depending on Bit 0 of that byte. 1: Time Constant follows. 0: No Time Constant follows.
1	Reset			Software Reset. Writing a 1 to this bit and a 0 to Bit 2 stops the CTC. Writing a 1 to both this bit and Bit 2 resets the CTC, which then starts when the Time Constant is written. 1: Software reset. 0: Continued operation.
0	Vector or Control			This bit must be 1 for a write to the Control Register. When this bit is 0 (and the CTC is not expecting a Time Constant value as described for Bit 2) the byte is placed into the Interrupt Vector register. 1: Control 0: Vector.

Time Constant Register

Writing a value after writing the Control Register with Bits 1 and 2 both 1, writes the value into both the Time Constant register and the Down Counter, and starts the CTC into operation. Writing a value after writing the Control Register with Bits 2..1 written as 10, stores the value in the Time Constant register but does not affect the Down Counter until the next time it counts down to 0.

Because the CTC only checks the Down Counter for 0 after it counts down, a Zero Time Constant makes the CTC decrement the Down Counter 256 times before reaching the ZC/TO condition.

Interrupt Vector Register

There is a register for each CTC. When the processor acknowledges an interrupt from any of the CTCs, Bits 7..3 of the interrupt vector come from this register, while Bits 2..1 identify the highest-priority CTC currently requesting an interrupt, with CTC0 having the highest priority and CTC3 the lowest

:

	Vector	
Bit 2	**Bit 1**	**CTC**
0	0	CTC0
0	1	CTC1
1	0	CTC2
1	1	CTC3

Bit 0 of an interrupt vector must be 0, as required by the processor, by the mechanism used to write the Interrupt Vector register.

Down Counter Register (%DE)

Software reads the value in this counter from the CTC's I/O address at any time. (The Control Register, Time Constant register, and Interrupt Vector register are all write-only registers.)

7	6	5	4	3	2	1	0
Reserved	PIA16/ ZT/CO2	PIA15/ ZT/CO1	PIA14/ ZT/CO0	PIA13/ TRG3	PIA12/ TRG2	PIA11/ TRG1	Reserved
–	R/W	R/W	R/W	R/W	W	R/W	R/W

Bit Number	Field	R/W	Reset Value	Description
7	Reserved	–	–	Reserved. Must be 0.
6	PIA16/ ZT/CO2	R/W	0	When software has programmed one of the PIA16-14/ZT/ CO2-0 pins as an output in the PIA1 Data Direction register, and the corresponding one of these bits is 0, the Z80185/Z80195 drives the corresponding bit of the PIA1 Data Register onto the pin. When software has programmed such a pin as an output, and the corresponding bit here is 1, the Z80185/ Z80195 drives the pin with the ZC/TO output of the indicated CTC. 1: ZT/CO2 0: PIA16
5	PIA15/ ZT/CO1	R/W	0	See the description for Bit 6. 1: ZT/CO1 0: PIA15
4	PIA14/ ZT/CO0	R/W	0	See the description for Bit 6. 1: ZT/CO2 0: PIA14

Bit Number	Field	R/W	Reset Value	Description
3	PIA13/ TRG3	R/W	0	This bit control whether the CLK/TRG inputs of CTC3-1 are taken from the PIA13-11 pins respectively, or from the ZC/TO outputs respectively. The CLK/TRG input of CTC0 is always taken from the PIA10 pin. 1: TRG3 0: PIA13
2	PIA12/ TRG2	R/W	0	See the description for Bit 3. 1: TRG2 0: PIA12
1	PIA11/ TRG1	R/W	0	See the description for Bit 3. 1: TRG1 0: PIA11
0	Reserved	–	–	Reserved. Must be 0.

CTC OPERATION

The four major modes of operation of a CTC are controlled by Bits 6 and 3 of its Control Register:

Bit 6	Bit 3	Mode
0	0	Auto-Start Timer
0	1	Triggered Timer
1	0	8-Bit Counter
1	1	Long Counter

The difference between these modes is described in the section on the Control Register. In any of these modes, software starts a CTC using a Software Reset followed by writing a Time Constant value. When a CTC counts its downcounter down to 0, the following process occurs:

1. The CTC produces a pulse on its ZT/CO output.
2. It requests an interrupt When Bit 7 of its Control Register is 1.
3. It reloads its Down Counter from its Time Constant register.
4. The CTC continues operating.

There is no option to make a CTC stop automatically when it reaches 0. Software can stop a CTC at any time by writing a 1 to the Software Reset bit in its Control Register.

CTC INTERRUPTS

The CTCs are part of an interrupt acknowledge daisy-chain internal to the Z80185/Z80195, as are the ESCC channel and the Bidirectional Centronics controller. This daisy chain controls which of these three devices has the highest interrupt priority, and enables a higher-priority device to interrupt the interrupt service routine for a lower-priority device. (The DMAs, ASCIs, PRTs, and CSI/O use a fixed interrupt priority and do not allow such *nested* interrupts.)

The relative priority of the CTCs, ESCC, and Bidirectional Centronics controller is programmable as described in the section, “CPU Options” on page 3-1. The daisy chain can be extended to include external devices, by connecting the IEO output of a high-priority external device to the Z80185/Z80195’s IEI input, or by connecting the Z80185/Z80195’s IEO output to the IEI input of a low-priority external device.

The internal daisy chain includes the four CTCs in the fixed-priority order CTC0, which is the highest in priority, to CTC1, 2, and 3, which are the lowest in priority.

Each device on the daisy chain features its own Interrupt Under Service (IUS) latch, which it sets when the processor acknowledges an interrupt from that device. While a device’s IUS latch is set, it negates its IEO output to lower-priority devices so that they cannot interrupt this device’s interrupt service routine (ISR).

The ISR for a daisy-chained device must clear the device’s IUS latch to allow lower-priority devices to request interrupts. When the ISR re-enables processor interrupts to allow nested interrupts from higher-priority devices during its execution, it must clear IUS at the end of its execution; if not, the ISR can clear the IUS latch at any point in its execution.

The ESCC and Bidirectional Centronics controller include explicit operations in I/O space for clearing their IUS bits. The CTCs do not: the IUS bit of a CTC must be cleared by ending its interrupt service routine with the special instruction `RETI`. This instruction fetches a return address from the stack and resumes execution, just like the `RET` instruction, but the highest-priority CTC having its IUS bit set notes the execution of the RETI, and clears its IUS bit.

The `RETI` instruction is necessary only for an interrupt service routine for an internal CTC or an external Z80 PIO, SIO, CTC, or DMA. ISRs for other devices can and must end with an `RET` instruction because it is shorter and faster.

The CTCs inside the Z80185/Z80195 recognize an `RETI` instruction, and clears IUS as a result, whether or not the M1E bit is set as described in the section, “CPU Options” on page 46. When external Z80 devices are included in an application, software may need to program the M1E bit to `0` to ensure that these devices correctly recognize and respond to an `RETI`.

Watch-Dog Timer

INTRODUCTION

The Watch-Dog Timer (WDT) is enabled at Reset and must be disabled by software. When software does not disable the WDT, it must periodically clear the WDT in order to avoid a hardware Reset of the entire Z80185/Z80195.

WDT Registers

Watch-Dog Timer Master Register (waiting for address)

7	6	5	4	3	0
WDT Enable	WDT Periodic Field		Drive Reset	WDT Compatibility	
1	1		1	-	
R/W	R/W		R/W	R/W	

Bit Number	Field	R/W	Reset Value	Description
7	TIF1	R	1	WDT Enable When this bit is 1, as it is after a reset, the WDT is enabled. In order to disable the WDT, software must write this bit as 0 and then write the value B1H to the WDT Command Register. This two-step disabling procedure ensures than runaway software/ firmware does not disable the WDT. 1: WDT enabled. 0: WDT disabled.
6..5	TIF0	R/W	1	WDT Periodic Field These bits select how long software can go without writing a Clear command to the WDT Command Register, before the WDT Resets the entire Z80185/Z80195. They reset to 11, the longest interval 00: Period is (TcC x 2*16) 01: Period is (TcC x 2*18) 10: Period is (TcC x 2*20) 11: Period is (TcC x 2*22)

Bit Number	Field	R/W	Reset Value	Description
4	TIE1	R/W	1	Drive Reset When this bit is 1, as it is after a Reset, when the WDT times out, it drives the $\overline{\text{RESET}}$ pin Low to reset external logic as well as the Z80185/Z80195. When this bit is `0`, a WDT timeout only internally resets the Z80185/Z80195. 1: Output of WDT is driven onto $\overline{\text{RESET}}$ pin 0: WDT output only resets Z80185/Z80195
3..0	TOC1, 0	R/W	-	WDT Compatibility For compatibility with other ZiLOG WDTs, these bits are written as `0011`. They read back as `0011`

WDT Command Register `(WDTCR: F1H)`

The WDT decodes two values written to the WDTCR address.

7							0
D7	D6	D5	D4	D3	D2	D1	D0
0							
W							

Bit Number	Field	R/W	Reset Value	Description
7..0		W	0	**B1H.** Writing this value, when Bit 7 of the Master Register is 0, disables the WDT. **4EH.** Writing this value clears the WDT to `0`, therefore delaying the time when it would time out and Reset the Z80185/Z80195.

Parallel Ports

FEATURES

The Z80185/Z80195 features two 8-bit bidirectional ports. Each bit is individually programmable for input or output. Each port includes two registers: the Port Direction Control Register and the Port Data Register. The second port also includes an Alternate Address for its data register that is used with the Bidirectional Centronics feature.

PORT REGISTER

PIA1 Data Direction Register `(%E0)`

7	6	0
Reserved	PIA1 Data Direction	
–	R/W	

Bit Number	Field	R/W	Reset Value	Description
7	Reserved	–		**Reserved**. Do not program.
6..0	PIA1 Data Direction		0	The data direction register determines which of the PIA16-10 pins are inputs and outputs. When a bit is `1`, the corresponding bit in the PIA1 Data Register is an input. If the bit is `0`, then the corresponding pin is an output. These bits must be set appropriately even if these pins are used for CTC inputs and outputs.

PIA1 Data Register (%E1)

7	6 … 0
Reserved	PIA1 Data
	R/W

Bit Number	Field	R/W	Reset Value	Description
7	Reserved	–		**Reserved**. Do not program.
6..0	PIA1 Data	R/W	0	When the processor writes to the PIA1 Data Register, the data is stored in the internal buffer. Any bits that are output are then driven onto the pins. When the processor reads the PIA1 Data Register, the data on the external pins is returned.

PIA2 Data Direction Register (%E2)

7 … 0
PIA2 Data Direction
R/W

Bit Number	Field	R/W	Reset Value	Description
7..0	PIA2 Data Direction	R/W	0	.When the Bidirectional Centronics interface is not controlling the direction of the PIA20-27 pins, this register does so. When one of these bits is `1`, the corresponding pin among PIA20-27 is an input. If the bit is `0`, then the corresponding pin is an output.

PIA2 Data Register (%E3)

7 … 0
PIA2 Data Direction
R/W

Bit Number	Field	R/W	Reset Value	Description
7..0	PIA2 Data	R/W	0	When the processor writes to the PIA2 Data Register, the data is stored in the internal buffer. Any bits that are output are then driven on to the pins. In certain modes of the Bidirectional Centronics Controller, an intermediate register called the Output Holding Register is activated. The transfer of data from the OHR to the pins is under the control of the controller. When the processor reads the PIA2 Data Register, the data on the external pins is returned. In certain modes of the Bidirectional Centronics Controller, reading from this address reads the data that has been stored in the port register from PIA27-20 under the control of the controller.

PIA2 Data Alternate Address (%EE)

7	0
PIA2 Data Direction	
R/W	

Bit Number	Field	R/W	Reset Value	Description
7..0	PIA2 Data	R/W	0	Reading and writing this register is the same as reading and writing address E3 as described above. In certain modes of the Bidirectional Centronics Controller, writing to this address sets a *ninth bit* in the opposite sense from writing address E3. This process drives one of the control outputs with the opposite polarity.

Bidirectional Centronics P1284 Controller

INTRODUCTION

The Centronics P1284 Controller can operate in either the Host or Peripheral role in Compatibility mode (host to printer), Nibble or Byte mode (printer to host), and ECP mode (bidirectional). It provides no hardware support for the EPP mode, although it may be possible to implement this mode by software.

Nine control signals have dedicated hardware pins, and have ±12 mA drive (P1284 Level 2) capability as does the 8-bit data port PIA27-20.

Note: Signal names listed below are those for the original Compatible mode. The names shown in parentheses represent the same signal, but in a more recent mode. The Z80185 does not include hardware support for the P1284 EPP mode.

The following signals are outputs in a Peripheral mode, inputs in a Host mode:

- Busy (PtrBusy, PeriphAck)
- nAck (PtrClk, PeriphClk)
- PError (AckDataReq, nAckReverse)
- nFault (nDataAvail, nPeriphRequest)
- Select (Xflag)

The following signals are inputs in a Peripheral mode, outputs in a Host mode:

- nStrobe (HostClk)

- nAutoFd (HostBusy, HostAck)
- nSelectIn (P1284Active)
- nInit (nReverseRequest)

Because the Host/Peripheral mode is fully controlled by software, a Z80185-based product can operate as a Host in one system, or as a Peripheral in another, without any change to the hardware. A Z80185-based product could even act as a Host at one time and a Peripheral at another time within the same system, if a controller exists for the alternate use.

In general, the interface architecture automates operations that are seen as performance-critical, while leaving less frequent operations to software control. To achieve top performance, software assigns a DMA channel to the current direction of data flow.

Note: The IEEE 1284 Interface is used with the Bit 5 ($\overline{\text{IOC}}$) in the `OMCR` set to `0`. The setting of this bit primarily affects RLE expansion in peripheral ECP Forward and Host ECP Reverse modes.

BIDIRECTIONAL CENTRONICS REGISTERS

Reading the Parallel Controls Register

Reading the Parallel Controls Register enables software to sense the state of the input signals per the current mode, plus two or three status flags.

Parallel Controls Register (PARC: DA) Read Host Mode

7	6	5	4	3	2	1	0
Busy	PError	Select	nFault	nAck	IllOp	DREQ	Idle
-	-	-	-	-	-	-	-
R	R	R	R	R	R	R	R

Bit Number	Field	R/W	Reset Values	Description
7	Busy	R		Busy
6	PError	R		PError
5	Select	R		Select
4	nFault	R		nFault
3	nAck	R		nAck

Bit Number	Field	R/W	Reset Values	Description
2	IllOp	R	0	**Illegal Operation** The controller sets IllOp when it detects an error in the protocol, for example, if it is in Peripheral mode and it detects that the host has driven P1284Active (nSelectIn) Low at a time that mandates an immediate Abort, that is, outside one of the windows in which this event indicates an organized disengagement. If status interrupts are Enabled, an interrupt is always requested when Bit 2 (IllOp) is set. Setting NewMode to 1 clears IllOp.
1	DREQ	R		**External DMA Request** DREQ is the Request presented to the DMA channels, which may or may not be programmed to service this request. If not, an interrupt can be enabled when DREQ is set.

Parallel Controls Register (PARC: DA) Read Peripheral Mode

7	6	5	4	3	2	1	0
nAutoFd	nStrobe	nSelectln	ninit	Reserved	IllOp	DREQ	Idle
-	-	-	-	-	-	-	-
R	R	R	R	R	R	R	R

Bit Number	Field	R/W	Reset Values	Description
7	nAutoFd	R		nAutoFd
6	nStrobe	R		nStrobe
5	nSelectln	R		nSelectln
4	ninit	R		ninit
3	Reserved	R		Reserved
2	IllOp	R	0	**Illegal Operation** The controller sets IllOp (Illegal Operation) when it detects an error in the protocol, for example, if it is in Peripheral mode and it detects that the host has driven P1284Active (nSelectIn) Low at a time that mandates an immediate Abort, that is, outside one of the windows in which this event indicates an organized disengagement. If status interrupts are Enabled, an interrupt is always requested when Bit 2 (IllOp) is set. Writing PARM with NewMode=1 clears IllOp.
1	DREQ	R		**DMA Request** DREQ is the Request presented to the DMA channels, which may or may not be programmed to service this request. If not, an interrupt can be enabled when DREQ is set.

Writing to the Parallel Controls Register

Writing to PARC allows the software to set and clear the output signals per the current mode.

Parallel Controls Register Write Host Mode (PARC: DA)

7	6	5	4	3	2	1	0
nAutoFd	nStrobe	nSelectln	ninit	nAutoFd	nStrobe	nSelectln	ninit
-	-	-	-	-	-	-	-
W	W	W	W	W	W	W	W

Bit Number	Field	R/W	Reset Values	Description
7	nAutoFd	W	0	1: Drive nAutoFd High.
6	nStrobe	W	0	1: Drive nStrobe High.
5	nSelectln	W	0	1: Drive nSelectln High.
4	ninit	W	0	1: Drive ninit High.
3	nAutoFd	W	0	1: Drive nAutoFd Low.
2	nStrobe	W	0	1: Drive nStrobe Low.
1	nSelectln	W	0	1: Drive nSelectln Low.
0	ninit	W	0	1: Drive ninit Low.

Parallel Controls Register Write Peripheral Mode (PARC: DA)

7	6	5	4	3	2	1	0
Busy	PError	Select	nFault	Busy	PError	Select	nFault
-	-	-	-	-	-	-	-
W	W	W	W	W	W	W	W

Bit Number	Field	R/W	Reset Values	Description
7	Busy	W	0	1: Drive Busy High.
6	PError	W	0	1: Drive PError High.
5	Select	W	0	1: Drive Select High.
4	nFault	W	0	1: Drive nFault High.
3	Busy	W	0	1: Drive Busy Low.
2	PError	W	0	1: Drive PError Low.
1	Select	W	0	1: Drive Select Low.
0	nFault	W	0	1: Drive nFault Low.

Writing to the PARC2 Register

Because there are five outputs in a Peripheral mode, another register, called `PARC2`, allows software to change the nAck line, rather than the Select line.

PARC2 Register Write Peripheral Mode (`PARC2: DB`)

7	6	5	4	3	2	1	0
Busy	PError	nAck	nFault	Busy	PError	nAck	nFault
-	-	-	-	-	-	-	-
W	W	W	W	W	W	W	W

Bit Number	Field	R/W	Reset Values	Description
7	Busy	W	0	1: Drive Busy High.
6	PError	W	0	1: Drive PError High.
5	nAck	W	0	1: Drive nAck High.
4	nFault	W	0	1: Drive nFault High.
3	Busy	W	0	1: Drive Busy Low.
2	PError	W	0	1: Drive PError Low.
1	nAck	W	0	1: Drive nAck Low.
0	nFault	W	0	1: Drive nFault Low.

Parallel Mode Register

The Parallel Mode Register includes the basic mode control of the controller.

Parallel Mode Register (`PARM: D9`)

7	6	5	4	3	0
NewMode	IdleIE	StatIE	DREQIE	Mode	
-	-	-	-	-	
W	W	W	W	W	

Bit Number	Field	R/W	Reset Values	Description
7	NewMode	W	0	**New Mode** 1: Reinitializes the state machine to the initial state for the mode called out by the Mode field. Never change the Mode without writing a `1` in this bit.
6	IdleIE	W	0	**Idle Interrupt Enable** 1: Enable interrupts when the controller sets the Idle flag. When software uses a DMA channel to provide data to the P1284 controller, it can be expected that the channel does so in a timely manner, and that an Idle condition signifies that the channel has finished transferring the block. (Software can also enable an interrupt from the DMA channel, but on the transmit side these interrupts are not well-synchronized to events on the P1284 controller.) Conversely, if software provides data, Idle may not be grounds for an interrupt. Some modes set the Idle flag when they are entered. However, such a setting of Idle never requests an interrupt.

Bit Number	Field	R/W	Reset Values	Description
5	StatIE	W	0	**Status Interrupt Enable** 1: Enable status interrupts that are described separately for each mode.
4	DREQIE	W	0	**External DMA Request Interrupt Enable** 1: Enable interrupts when the controller sets DREQ, except in those modes that set DREQ when they are entered and do not request an interrupt.
3..0	Mode	W	0	**Mode** `0000:` Non-P1284 mode `0001`: Peripheral Compatible/Negotiation mode `0010`: Peripheral Nibble mode `0011`: Peripheral Byte mode `0100`: Peripheral ECP Reverse mode `0101`: Peripheral Inactive mode `0110`: Peripheral ECP Forward mode with software RLE handling `0111`: Peripheral ECP Forward mode with hardware RLE expansion `1000`: Host Negotiation mode `1001`: Host Compatible mode `1010`: Host Nibble mode `1011`: Host Byte mode `1100`: Host ECP Forward mode `1101`: Host Reserved mode `1110`: Host ECP Reverse mode with software RLE handling `1111`: Host ECP Reverse mode with hardware RLE expansion

PIA 2 Data, Alternate PIA 2 Data, Output Holding and I/O Registers

A second output register has been added for PIA27..PIA20. Writing to either the Z80181-compatible PIA 2 Data Register (address E3) or the new Alternate PIA 2 Data Register (address EE) writes to the Output Holding Register (OHR). When the PIA27..PIA20 pins are outputs, the outputs of the OHR are the inputs to the second register, which is called the I/O register (IOR), these outputs drive the PIA27..PIA20 pins. When the pins are inputs, they are the inputs to the IOR, which can be read from the PIA 2 Data Register (address E3).

PIA 2 Data Direction Register

In non-P1284 mode, Host Negotiation mode, Reserved Modes, and in Peripheral Compatible/Negotiation mode when the host drives nSelectIn (P1284 Active) High to select negotiation, the direction of the PIA27..PIA20 pins are controlled by the PIA 2 Data Direction register, as on the Z80181. Also in these modes the IOR is loaded on every Φ clock, so that operation is virtually identical to the Z80181. In other modes the controller controls the direction of PIA27..PIA20 and when the IOR is loaded.

Time Constant Register

A Time Constant Register (PART) must be loaded by software with the smallest number of Φ clocks that equals or exceeds the critical time for the mode selected in PARM.

Time Constant Register Write (PART: DC)

7	6	5	4	0
Set IUS	cir IP	cir IUS	number of Φ clocks in critical time	
-	-	-	-	
W	W	W	W	

Bit Number	Field	R/W	Reset Values	Description
7	Set IUS	W		Set Interrupt Under Service
6	cir IP	R		cir Interrupt Pending
5	cir IUS	R		cir Interrupt Under Service
4..0	Phi clocks	R		Number of Phi clocks in critical time

The critical time is 750 ns for Host Compatible mode and 500 ns for most other modes. In the Host ECP Forward and Peripheral ECP Reverse modes, the time specified by this field is used in two ways:

- It determines how long the controller waits after requesting data without receiving data, before setting the Idle flag to indicate DMA completion.
- When RLE encoding is enabled, it determines how long the controller will wait after requesting data, to see if the next character is the same as its predecessor, before sending the previous character.

Time Constant Register Read (PART: DC)

7	6	5	4 ... 0
IUS	IP	0	number of Phi clocks in critical time
-	-	-	-
R	R	R	R

Bit Number	Field	R/W	Reset Values	Description
7	IUS	R		Interrupts Under Service
6	IP	R		Interrupts Pending
5	0	R		
4..0	Φ clocks	R		Number of Phi clocks in critical time

Reading PART yields the status of the IP and IUS bits, which are described in the Bidirectional Centronics Interface section:

Vector Register

Bits 7..1 of the Vector Register (PARV) must be loaded by software with the interrupt vector to be used for interrupts from this controller.

Vector Register (PARV: DD)

7 1	0
Interrupt Vector	Enable RCE
-	-
R/W	R/W

Bit Number	Field	R/W	Reset Values	Description
7..1	Interrupt Vector	R/W		Interrupt Vector
0	Enable RCE	R/W		Enable Run Length Encoding

The least-significant bit of PARV is a write-only bit that can be set to 1 by software to enable Run Length Encoding in the Host ECP Forward and Peripheral ECP Reverse modes. The bit always reads as 0, when reading PARV and in interrupt acknowledge cycles.

Figure 63 illustrates the Bidirectional Centronics P1284 Controller Block Diagram .

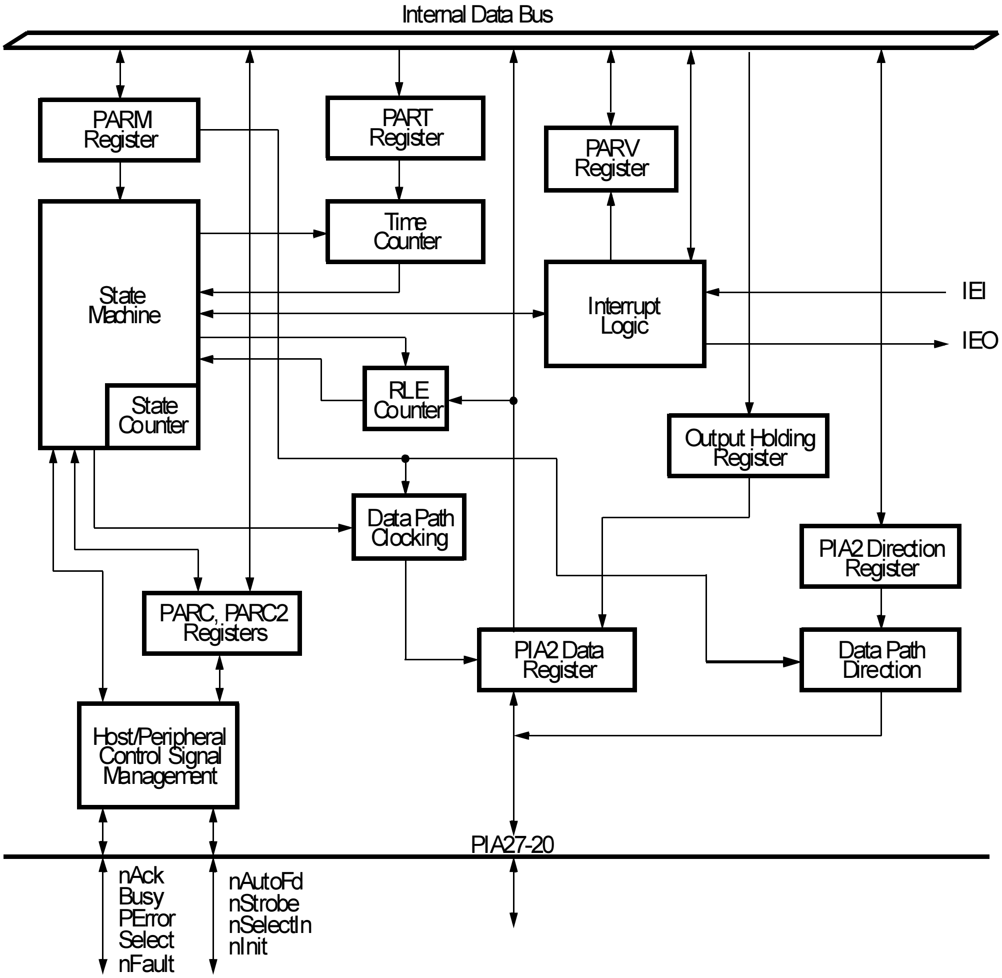


Figure 63. Bidirectional Centronics P1284 Controller Block Diagram

INTERRUPTS

As in other Zilog peripherals, the controller includes an Interrupt Pending (IP) bit and an Interrupt Under Service (IUS) bit. The controller is part of an on-chip interrupt acknowledge daisy-chain that extends from the IEI pin, through the EMSCC, CTC, and this controller in a programmable priority order, and from the lowest-priority of these devices to the IEO pin. The interrupt request from the controller is logically ORed with $\overline{\text{INT0}}$ and the interrupt requests from the ESCC and CTCs.

The controller sets its IP bit whenever any of three conditions occurs:

PARM4 is 1, and the controller sets the DREQ bit. This does not include when the controller forces the DREQ bit to 1, when software first places the controller in Peripheral Nibble, Peripheral Byte, Peripheral ECP Reverse, Host Compatible, or Host ECP Forward mode.

PARM5 is 1, and a mode-dependent status interrupt condition occurs. The following sections describe the status interrupt conditions (if any) for each mode.

PARM6 is 1, and the controller sets the Idle bit, except when the controller forces the Idle bit to 1, when software first places the controller in Peripheral Nibble, Peripheral Byte, Peripheral ECP Reverse, Host Compatible, or Host ECP Forward mode. The following sections describe when Idle is set in each mode.

When IP is set, it remains set until software writes a 1 to PARM6.

The controller begins requesting an interrupt of the processor whenever IP is set, its IEI signal from the on-chip daisy-chain is High/True, and its IUS bit is 0. When it starts requesting an interrupt, the controller continues to request until $\overline{\text{IORQ}}$ goes Low in an interrupt-acknowledge cycle, or IP is 0, or IUS is 1.

The controller drives its IEO output High, if its IEI input is High, and its IP and IUS bits are both 0. A Z80 interrupt acknowledge cycle is signalled by M1 going Low, followed by $\overline{\text{IORQ}}$ going Low. The controller, and all other devices in the daisy-chain, freeze the contribution of their IP bits to their IEO outputs while M1 is Low, which prevents new events from affecting the daisy-chain. By the time $\overline{\text{IORQ}}$ goes Low, one and only one device has its IEI pin High and its IEO pin Low. This device responds to the interrupt by providing an interrupt vector and setting its IUS bit. This controller also clears its IP bit when it responds to an interrupt acknowledge cycle.

The Interrupt Service Routine (ISR), that is initiated when the interrupt vector value identifies an interrupt from this controller, saves the processor's context and then proceeds as follows:

1. If the ISR does not allow nested interrupts, it can clear the IP and IUS bits by writing 60H, plus the critical time value to the PART register, then read the status from PARC and proceed based on that status. Near the end of the ISR it re-enables processor interrupts.
2. If the ISR allows nested interrupts, it can re-enable processor interrupts, clear IP by writing 40H plus the critical time value to the PART register, and then read the status from PARC and proceed based on that status. At the end of the ISR it clears IUS to allow further interrupts from this controller and devices lower on the daisy-chain, by writing 20H plus the critical time value to the PART register.

OPERATING MODES

The remainder of this section describes the operation of the various operating modes that can be selected.

Non-P128 Mode

The Z80185 and Z80195 default to this mode after a Reset, and this mode is compatible with the use of PIA27..PIA20 on the Z80181. The directions of PIA27..PIA20 can be controlled individually by writing to register E2, as on the Z80181. The state of outputs among PIA27..PIA20 can be set by writing to register E3, and the state of all eight pins can be sensed by reading register E3. The Busy, nAck, PError, nFault, and Select pins are tri-stated in this mode, while nStrobe, nAutoFd, nSelectIn, and nInit are inputs. There are no status interrupts in this mode.

Peripheral Inactive Mode

Peripheral Inactive mode operates identically to Non-P1284 mode as described above, except that the Busy, nAck, PError, nFault, and Select pins are outputs that can be controlled via the PARC and PARC2 registers, and status interrupts can occur in response to any edge on nAutoFd, nStrobe, nSelectIn, or nInit. This mode differs from Peripheral Compatibility/Negotiation mode with nSelectIn (P1284 Active) High, only in that the controller does not operate in Compatibility mode if nSelectIn goes Low.

Host Compatible Mode

1. Setting this mode configures PIA27..PIA20 as outputs regardless of the contents of register E2. When entering this mode, the controller sets the Idle and DREQ bits, but these settings do not request an interrupt.
2. If software, or a DMA channel, writes eight bits to the Output Holding Register (OHR) when Idle is set, the controller transfers the byte to the Input/Output Register and negates DREQ only momentarily, so as to request another byte from software or the DMA channel.
3. In this mode, the nAutoFd line is not under control of the PARC register, but rather under control of which register the software uses to write data to the OHR. Each time the controller transfers a byte from the OHR to the Input/Output Register, it sets nAutoFd High if the byte was written to address E3, and Low if the byte was written to the alternate address EE. In a DMA application all of the bytes transferred

from one output buffer will have the same state of nAutoFd, but this state can be changed from one buffer to the next by changing the I/O address used by the DMA channel. In non-DMA applications, software can set the state of nAutoFd for each character, by writing data to the two different register addresses.

4. When a data byte has been valid on PIA27..PIA20 for 750 ns (as controlled by the PART register), and the Busy and PError lines are Low and the Select, nAck, and nFault lines are High, the controller drives nStrobe Low. After the controller has held nStrobe Low for 750 ns it drives nStrobe back to High. Then it waits for 750 ns of data hold time to elapse. If software or a DMA channel has written another byte to the Output Holding Register (thus clearing DREQ) by the time this wait is satisfied, the controller transfers the byte from the Output Holding Register to the Input/Output Register, sets DREQ again, and returns to the event sequence at the start of this paragraph. Otherwise, it sets Idle and returns to the event sequence at the start of Step 2.

Status interrupts in this mode include rising and falling edges on PError, nFault, and Select.

Host Navigation Mode

Setting this mode puts PIA27..PIA20 under control of registers E2 and E3, as on the Z80181.

Software has complete control of the interface, and can either revert to Host Compatibility mode, or set one of the following Host modes, depending on how the peripheral responds to the Negotiation value(s).

Status interrupts in this mode include rising and falling edges on PtrClk (nAck), nAckReverse (PError), and nPeriphRequest (nFault). nFault is not used during actual P1284 negotiation, but is included because these events are significant during Byte and ECP mode the number of times specified in the Idle field.

Host Reserved Mode

This mode differs from Host Negotiation mode only in that there are no status interrupts in this mode.

Peripheral Compatible/Negotiation Mode

In this mode, if P1284Active (nSelectIn) is Low, the controller sets PIA27..PIA20 as inputs, regardless of the contents of register E2; when P1284Active (nSelectIn) is High, PIA27..PIA20 are under the control of registers E2 and E3. On entry to this mode, the controller sets the Idle bit, if DREQ is set from a previous mode.

If, in this mode, nStrobe goes (is) Low, P1284Active (nSelectIn) is Low, and DREQ is 0, indicating that any previous data has been taken by the processor or DMA channel, the controller captures the data on PIA27..PIA20 into the Input/Output Register, sets DREQ to notify software or the DMA channel to take the byte, drives the Busy line High, and one Φ clock later drives nAck Low. When at least 500 ns (as controlled by the PART register) have elapsed, the controller drives nAck back to High. One Φ clock later, if the CPU or DMA has taken the data and thus cleared DREQ, the controller drives Busy back to Low, otherwise it sets Idle.

Select, PError and nFault are under software control in this mode, and nAutoFd can be sensed by software, but has no other effect on operation.

In this mode, software monitors for the condition P1284Active (nSelectIn) High, and nAutoFd Low simultaneously. If software detects this state, it participates in a Negotiation process. Software reads the value on PIA27..PIA20 and set PError, nFault, XFlag, and nAck as appropriate for the data value. As long as P1284Active (nSelectIn) remains High in this mode, software is in complete control of the interface. After the host has driven nStrobe Low and then High again for an acceptable value, software reprograms the MODE field to the appropriate one of the following Peripheral modes.

Status interrupts in this mode include rising and falling edges on P1284Active (nSelectIn) and nInit, and rising and falling edges on HostBusy (nAutoFd) and HostClk (nStrobe) while P1284Active (nSelectIn) is High.

Host Nibble Mode

1. If, during Host Negotiation mode, software has placed the value `00` or `04` on the data lines, and received a positive response on Xflag (Select) and a Low on nDataAvail (nFault) at a rising edge of PtrClk (nAck), then after optionally programming a DMA channel to store data, software sets this mode.
2. For each byte in this mode, the controller performs the following activity:
   a. HostBusy (nAutoFd) goes Low and the controller waits until DREQ is cleared, indicating that the CPU or DMA has received previous data and PtrClk (nAck) is Low.
   b. The four status lines from the peripheral into the less-significant four bits of the Input/Output Register as follows:

Table 12. Nibble Mode Bit Assignments

Signal	First Data Bit	Second Data Bit
Busy	3	7
PError	2	6
Select	1	5
nFault	0	4

   c. HostBusy (nAutoFd) goes High and the controller waits until PtrClk (nAck) goes High.
   d. HostBusy (nAutoFd) goes Low and the controller waits until PtrClk (nAck) goes Low.

e. The four status lines from the peripheral to the most-significant four bits of the Input/Output Register, as shown in Table 12.

f. HostBusy (nAutoFd) goes High, the DREQ bit is `1`, and the controller waits until PtrClk (nAck) goes High.

g. If more data is available (nDataAvail (nFault) is Low), the controller returns to the beginning of Step 2. If no more data is available (nDataAvail (nFault) is High), the controller sets Idle and waits for software to program it back to Host Negotiation mode. Software can then select the next mode as described in the IEEE P1284 specification.

If Host software chooses not to receive all the data that a peripheral has available, it first disables the DMA channel, if one is in use, then waits for DREQ to be `1` and PtrClk (nAck) to be High. If nDataAvail (nFault) is Low at this point, the controller has already driven HostBusy (nAutoFd) Low to solicit the next byte. Software then performs the following tasks:

1. Program the controller back to Host Negotiation mode.
2. Read the `IOR` to get the current byte.
3. Take the next byte from the peripheral under software control.
4. After the peripheral drives nAck High after the second nibble, software can drive P1284Active (nSelectIn) Low to tell the peripheral to leave Nibble mode.

There are no status interrupts in Host Nibble mode.

Peripheral Nibble Mode

1. Software must not set this mode until there is reverse data available to send. In other words, it implements the P1284 Reverse Idle mode via software in Peripheral Compatibility/Negotiation mode. After software has driven nDataAvail (nFault), AckDataReq (PError), and Xflag (Select) all Low to signify that data is available, then driven

PtrClk (nAck) High after 500 ns, and if necessary programmed a DMA channel to provide data to send, when it sees HostBusy (nAutoFd) Low to request data, software sets this mode.

Entering this mode sets DREQ and Idle, but these settings do not request an interrupt. The PIA27..PIA20 pins remain configured for data input but are not used. Instead, four of the five control outputs are driven with the least significant and most significant four bits of the Input/Output Register, as shown in Table 12, while PtrClk (nAck) serves as a handshake/clock output. On entering this mode the hardware begins routing bits 3..0 of the `IOR` to these lines.

2. If software, or a DMA channel, writes a byte to the Output Holding Register when Idle is set, the controller immediately transfers the byte to the `IOR` and clears Idle, and negates DREQ only momentarily to request another byte from software or the DMA channel.

3. When data has been valid on the four control outputs for 500 ns (as controlled by the PART register), the controller drives the PtrClk (nAck) line Low. Then it waits for the Host to drive the HostBusy (nAutoFd) line back to High, after which it drives PtrClk (nAck) back to High, switches the four control lines to bits 7..4 of the `IOR`, and begins waiting for the host to drive HostBusy (nAutoFd) back to Low.

   a. When bits 7..4 have been valid for 500 ns and the Host has driven HostBusy (nAutoFd) Low, the controller drives PtrClk (nAck) Low again and begins waiting for the host to drive HostBusy (nAutoFd) High.

   b. When HostBusy (nAutoFd) has been driven High, the controller returns the four control outputs to the state set by software in `PARC`.

   c. If software or a DMA channel has not yet written another byte to the Output Holding Register (thus clearing DREQ), the controller sets Idle and waits for software to do so.

d. If/when software or a DMA channel has written a new byte to the OHR, the controller transfers the byte to the IOR, sets DREQ, and clears Idle if it had been set.

e. When the control outputs have been valid for 500 ns, the controller drives PtrClk (nAck) to High and waits for the host to drive HostBusy (nAutoFd) back to Low.

f. The four control lines switch back to bits 3..0 of the IOR and returns to the event sequence at the start of Step 3.

g. If there is no more data to send, when the controller sets Idle, software modifies PARC to make nDataAvail (nFault) and AckDataReq (PError) High, and then change the mode to Peripheral Compatible/Negotiation. After 500 ns, software sets PtrClk (nAck) back to High in PARC and enters Reverse Idle state.

Status interrupts in Peripheral Nibble mode include Rising and Falling edges on P1284Active (nSelectIn) and nInit. The controller sets the IllOp bit if P1284Active (nSelectIn) goes Low before it drives nAck High for the status states on the four control lines, or after the host drives HostBusy Low. Then, software immediately enters Peripheral Compatibility/Negotiation mode. If P1284Active goes Low, but IllOp stays 0, indicating that the Host negated P1284Active in a legitimate manner, software enters Peripheral Inactive mode for the duration of the return to Compatibility mode, followed by Peripheral Compatibility/Negotiation mode.

Host Byte Mode

Host Byte Mode is entered when the following events have occurred:

- The receipt of values hex 01H or 05H on PIA27..PIA20 in Host Negotiation Mode is acknowledged by the peripheral.
- The lines nDataAvail (nFault) and AckDataReq (PError) go Low to indicate data availability.

- PtrClk (nAck) goes High.

PIA27..PIA20 become inputs regardless of register E2. The Idle flag is cleared.

The controller waits 500 ns (as controlled by the PART register) before proceeding.

The controller performs the following activities:

1. For each byte, the controller drives HostBusy (nAutoFd) Low to indicate readiness for a byte from the peripheral. It waits for PtrClk (nAck) to go Low.
2. It captures the state of PIA27..PIA20 into the Input/Output Register, sets the DREQ bit to request software, or the DMA channel to take the byte.
3. It drives HostBusy (nAutoFd) High and HostClk (nStrobe) Low.
4. When software, or the DMA channel, has taken the byte (thus clearing DREQ) and the peripheral has driven PtrClk (nAck) back High, and at least 500 ns after driving HostClk (nStrobe) Low, the controller drives HostClk (nStrobe) back to High, and samples nDataAvail (nFault).
5. If HostBusy (nAutoFd) is still Low, the controller returns to the event sequence at the start.
6. Otherwise it sets the Idle flag.

In response to Idle, software enters Host Negotiation mode. Thereafter, it can set HostBusy (nAutoFd) Low, to enter Reverse Idle state, or enter Host Compatible mode as described in the IEEE P1284 specification, or conduct a new negotiation.

If software chooses not to receive all the data that a peripheral has available in this mode, it first disables the DMA channel if one is in use, and then wait for DREQ to be 1 and nAck to be 1. Then it reprograms the controller back to Host Negotiation mode, reads the last byte from the

IOR, drives HostClk (nStrobe) back to High, and then drives P1284Active (nSelectIn) Low to instruct the peripheral to leave Byte mode.

There are no status interrupts in Host Byte mode.

Peripheral Byte Mode

Software must not set this mode until there is reverse data available to send. That is, software implements the P1284 Reverse Idle mode via software in Peripheral Compatibility/Negotiation mode. The exact sequencing among PtrClk (nAck), nDataAvail (nFault), and AckDataReq (PError) differs according to whether this mode is entered directly from Negotiation or from Reverse Idle Phase, and is controlled by software. But in either case, before software sets this mode, nDataAvail (nFault) and AckDataReq (PError) is set to Low. After 500 ns, PtrClk (nAck) is set to High. When the host has driven HostBusy (nAutoFd) Low to request data, software sets Peripheral Byte mode, which sets the DREQ and Idle flags.

As long as P1284Active (nSelectIn) remains High, the controller drives PIA27-20 as outputs, regardless of the contents of register E2. When software, or a DMA channel, writes the first byte to the Output Holding Register, the controller immediately transfers the byte to the Input/Output Register, clears Idle but negates DREQ only momentarily, to request another byte from software or the DMA channel.

After each byte is transferred to the IOR, the controller performs the following tasks:

1. It waits 500 ns for data setup time (as controlled by the PART register) before driving PtrClk (nAck) Low.
2. It waits for the Host to drive HostBusy (nAutoFd) High.
3. If software or the DMA channel has not written more data to the Output Holding Register, that is, if DREQ is still set, the controller sets the Idle flag and waits for software or the DMA channel to do so.

4. If software or the DMA channel then writes data to the Output Holding Register, the controller clears DREQ and Idle.
5. When there is data in the `OHR` and DREQ is `0`, this condition guarantees that nDataAvail (nFault) and AckDataReq (PError) is held Low to indicate that more data is available.
6. The controller drives PtrClk (nAck) back to High.
7. The controller then waits for a rising edge on HostClk (nStrobe) and for the Host to drive HostBusy (nAutoFd) Low.
8. The controller transfers the byte from the `OHR` to the Output Register, sets DREQ, and returns to the event sequence at the start of this section.

While Peripheral Byte mode is in effect, software monitors the interface for two conditions:

- **Case 1:** Idle is set and no more data to send, or
- **Case 2:** P1284Active (nSelectIn) Low.

In Case #1, the software performs the following tasks:

1. Write `0` to register `E3` to keep PIA27..PIA20 outputs momentarily.
2. Set the mode back to Peripheral Compatibility, so that the interface is fully under software control.
3. Set nDataAvail (nFault) and AckDataReq (PError) High to signify no more data.
4. Wait 500 ns.
5. Set PtrClk (nAck) back to High.
6. When HostBusy returns to Low, the software sets PIA27..PIA20 back to inputs.

In Case #2, if a falling edge on P1284Active happens any time other than between a Rising edge on HostClk (nStrobe) and the next Falling edge on

HostBusy (nAutoFd), the controller sets the IllOp bit to notify software that an immediate Abort is in order, in which case software immediately enters Peripheral Compatibility/Negotiation Mode. If P1284Active goes Low, but IllOp is not set, meaning that the Host negated P1284Active in a legal manner, software enters Peripheral Inactive Mode for the duration of the Return to Compatibility Mode, and then enters Peripheral Compatibility/Negotiation Mode.

Status interrupts in Peripheral Byte Mode include rising and falling edges on P1284Active (nSelectIn) and nInit.

Host ECP Forward Mode

After a negotiation for ECP mode, host software remains in Negotiation mode so that it has complete control of the interface, until one of two situations occurs. If software has data to send, it optionally programs the DMA channel to provide the data, and then sets this mode. Alternatively, if software has no data to send and it detects that nPeriphRequest (nFault) has gone Low, indicating the peripheral is requesting reverse transfer, it sets PIA27..PIA20 as inputs, waits 500 ns, drives nReverseRequest (nInit) to Low to indicate a reverse transfer, and then sets Host ECP Reverse mode. In other words, software handles all aspects of ECP mode, other than active data transfer sequences.

Setting this mode configures PIA27..PIA20 as outputs regardless of the contents of register E2. On entry to this mode, the controller sets Idle and DREQ to request a byte from software or a DMA channel, but these settings do not cause an interrupt request.

If software or a DMA channel writes data to the Output Holding Register while the Input/Output Register is empty, the controller immediately transfers the byte to the IOR, clears Idle, and negates DREQ only momentarily, to request another byte.

The alternate address for the Output Holding Register allows software to send a channel address or an RLE count value. Such bytes are typically written by software rather than a DMA channel. Writing to the alternate address loads the OHR and clears DREQ, like writing to the primary address, but clears a ninth bit that is set when software or a DMA channel writes to the primary address. A similar ninth bit is associated with the Input/Output Register, from which it drives the HostAck (nAutoFd) line.

If the PARVO bit is set to enable Run Length Encoding, then after transferring a normal data byte (not a channel address nor an RLE count) from the OHR to the IOR and setting DREQ, the controller waits for the DMA channel or software to write another byte to the OHR, for up to the number of Φ clocks specified in the PART register. If another character is provided within this time, thus clearing DREQ, and it is neither a channel address nor an RLE count, and it is equal to the previous character in the IOR, the controller:

1. Increments its RLE counter.
2. Places the value on the RLE counter on the PIA26..PIA20 lines and a Low on PIA27, rather than the contents of the IOR.
3. Sets DREQ again.
4. Restarts the timer with the value in PART.

Unless it has just incremented the RLE counter to 127, the controller loops on the same test criteria, so as to accumulate an RLE count as defined by the P1284 specification.

One Phi clock after any of the following occurs, the controller drives HostClk (nStrobe) to low to inform the peripheral of the character, channel address, or RLE count:

1. The controller transfers a channel address or RLE count from the OHR to the IOR,
2. The controller transfers a data character from the OHR to the IOR, and PARVO is 0 so that RLE encoding is disabled,
3. PARVO is 1 so that RLE encoding is enabled, and within the time specified in PART after writing one character to the OHR, software or

a DMA channel writes another that is a channel address, or an RLE count, or differs from the preceding character,

4. PARVO is `1`, and the time defined in `PART` expires after software or a DMA channel writes one character to the `OHR`, without it writing another, or the controller increments the RLE counter to `127`.

The controller waits for the peripheral to drive PeriphAck (Busy) to High, after which the controller drives HostClk (nStrobe) back to High.

It waits for the peripheral to drive PeriphAck (Busy) back to Low. After PeriphAck (Busy) is back to Low, one of the following happens:

- If the RLE counter is non-zero and the controller has been holding its value on PIA26..PIA20, the controller clears the RLE counter to `0`, and places character in the `IOR` back on PIA27..PIA20. One clock later it drives HostClk (nStrobe) low again, and returns to the start of the preceding paragraph.
- If software or the DMA channel has written a new byte to the Output Holding Register and thus cleared DREQ, the controller transfers the byte to the IOR, sets DREQ again, and returns to the start of Step 5 above.
- If software or the DMA channel does not provide a new byte for the time indicated in the PART register after the controller sets DREQ, the controller sets the Idle flag and returns to the start of Step 3 above.

While this mode is in effect, software monitors for the condition “Idle and no more data left to send,” and/or nPeriphRequest (nFault) Low. Host software has control and may or may not honor the peripheral’s reverse request on nFault while it has data to send. When there is no more data, software can set Host Negotiation mode to have full control of the interface, and if appropriate can drive P1284Active (nSelectIn) to Low in order to terminate ECP mode or can set Host ECP Reverse mode, wait 500 ns, and drive nReverseRequest (nInit) to Low.

Status interrupts in Host ECP Forward mode include rising and falling edges on nPeriphRequest (nFault).

Peripheral ECP Forward Modes

1. After a negotiation for ECP mode, Peripheral software remains in Compatibility/Negotiation mode with P1284 Active (nSelectIn) High, so that it maintains complete control of the interface, through when it detects the host drive HostAck (nAutoFd) Low for the second time. The software then sets nAckReverse (PError) High. If software has data to send, it drives nPeriphRequest (nFault) Low at the same time, and optionally programs a DMA channel to provide the data. Whether or not it has data to send, software then sets one of the two ECP Forward modes.
2. In these modes, the controller configures PIA27..PIA20 as inputs regardless of the contents of register `E2`. On entry to one of these modes, the controller clears the Idle bit, if it is set.
3. For each byte, the controller waits for the Host to drive HostClk (nStrobe) to Low. When HostClk (nStrobe) is Low and software or the DMA channel has taken any previous byte and thus cleared DREQ, operation diverges into four cases depending on the state of HostAck (nAutoFd), the mode, the most significant bit of the data, and the state of an internal 7-bit Run-Length Encoding (`RLE`) counter.

- If HostAck (nAutoFd) is High, indicating that this byte is neither an RLE value nor a Channel Address, the controller performs the following steps:
- It captures the data from PIA27..PIA20 into the Input/Output Register, sets DREQ to request software or the DMA channel to take this byte, and drives PeriphAck (Busy) High.
- If the RLE counter is `0`, the controller waits (if necessary) for the Host to drive HostClk (nStrobe) back to High.
- It drives PeriphAck (Busy) back to Low and returns to the event sequence at the start of Step a.

- If the RLE counter is non-zero, the controller waits for software or a DMA channel to read the byte from the Input/Output Register, negates DREQ only momentarily, and decrements the RLE counter. It repeats this step until the RLE counter is 0.
- Processing continues as described at the beginning of Step 4. Thus an RLE value of “n” results in the next byte being provided to software or a DMA channel n+1 times.

4. If HostAck (nAutoFd) is Low and the most significant bit of the byte is zero (PIA27 is Low), the byte is an RLE repeat count. If the mode is Hardware RLE Expansion, the controller transfers (the seven least significant bits of) it to the RLE counter, leaves DREQ cleared, and drives PeriphAck (Busy) High. Thereafter, the controller waits for the host to drive HostClk (nStrobe) back to High, at which time it drives PeriphAck (Busy) back to Low, and returns to the event sequence at the start of Step 3. If HostAck (nAutoFd) is Low, and PIA27 is High, the byte is a channel address. In this case, or when PIA27 is Low and the mode is Software RLE Handling, the controller performs the following steps:
   a. It captures the data from PIA27..PIA20 into the Input/Output Register.
   b. It leaves DREQ cleared to keep a DMA channel from storing the byte.
   c. It sets the Idle bit, (which it does not otherwise set while in this mode).
   d. Software responds to this condition by reading the byte from the PIA2 data register E3, performs whatever else is necessary to handle the situation, and then sets Busy High.
   e. The controller clears Idle, waits (if necessary) for the host to drive HostClk (nStrobe) back to High, and then drives PeriphAck (Busy) back to Low and returns to the event sequence at the start of Step 3.

While this mode is set, if data to send becomes available, software drives nPeriphRequest (nFault) Low to alert the host of this fact. Also, software monitors the controller for either of two conditions:

- If the host drives nReverseRequest (nInit) Low in response to nPeriphRequest (nFault) Low, software drives nAckReverse (PError) Low, optionally programs a DMA channel to provide the data, and sets Peripheral ECP Reverse mode.
- If P1284Active (nSelectIn) goes Low, the controller sets the IllOp bit in PARC, if this occurs between the time the Host drives HostClk (nStrobe) Low, and when the controller subsequently drives PeriphAck (Busy) back to Low, in which case software immediately enters Peripheral Compatibility/Negotiation mode. If P1284Active goes Low, but IllOp stays 0, indicating a legal termination, software enters Peripheral Inactive mode and sequences the nAckReverse (PError), PeriphAck (Busy), PeriphClk (nAck), nPeriphRequest (nFault), and Xflag (Select) lines to leave ECP mode.

Status interrupts in Peripheral ECP Forward mode include rising and falling edges on P1284Active (nSelectIn) and nReverseRequest (nInit).

Host ECP Reverse Modes

1. In these modes the controller configures PIA27..PIA20 as inputs, regardless of the contents of register E2. On entry to one of these modes, the controller clears the Idle bit, if it had been set.
2. For each byte, the controller waits for the peripheral to drive PeriphClk (nAck) Low. When this happens, and software or the DMA channel has taken any previous byte from the Input/Output Register and thus cleared DREQ, operation diverges into four cases, depending on the state of PeriphAck (Busy), the mode, the least significant bit of the data, and the state of an internal 7-bit RLE counter.

If PeriphAck (Busy) is High, indicating that this byte is neither an RLE value nor a Channel Address, the controller performs the following tasks:

a. It captures the data from PIA27..PIA20 in the IOR, sets DREQ to notify software or the DMA channel to take the byte, and drives HostAck (nAutoFd) High.

b. If the RLE counter is 0, the controller then waits (if necessary) for the peripheral to drive PeriphClk (nAck) back to High.

c. It drives HostAck (nAutoFd) back to Low and returns to the event sequence at the start of Step 2.

d. If the RLE counter is non-zero, the controller waits for software or the DMA channel to read the byte from the IOR, negates DREQ only momentarily, and decrements the RLE counter.

e. It repeats this sequence until the RLE counter is 0, at which point it proceeds from the beginning of Step 3. Thus an RLE value of n results in the next byte being provided to software or a DMA channel n+1 times.

3. If PeriphAck (Busy) is Low, and the most significant bit of the byte is 0 (PIA27 is Low), the byte is an RLE repeat count. If the mode is Hardware RLE Expansion, the controller transfers (the seven least significant bits of) it to the RLE counter, leaves DREQ cleared, and drives HostAck (nAutoFd) High. Thereafter the controller waits for the peripheral to drive PeriphClk (nAck) back to High, at which time it drives HostAck (nAutoFd) back to Low and returns to the event sequence at the start of Step 2.

4. If PeriphAck (Busy) is Low, and the most significant bit of the byte is 1 (PIA27 is High), the byte is a channel address. In this case, or when the least significant bit is 0 but the mode is Software RLE handling, the controller performs the following tasks:

   a. It captures the data from PIA27..PIA20 in the IOR, leaves DREQ cleared to keep a DMA channel from storing the byte, and sets Idle, (which it does not otherwise set in this mode).

b. Software responds to this condition by reading the byte from the PIA2 data register E3, reprogramming a DMA channel if necessary, and performing whatever else is necessary to handle the channel address, and setting HostAck (nAutoFd) High.

c. The controller clears Idle, waits for the peripheral to drive PeriphClk (nAck) back to High, drives HostAck (nAutoFd) back to Low, and returns to the start of the event sequence.

5. If data has become available to be sent while this mode is in effect and software elects to send it, it drives nReverseRequest (nInit) to High, sets Host Negotiation mode to take full control of the interface, waits for nAckReverse (PError) to go High, and then sets PIA27..PIA20 as outputs.

6. Status interrupts in Host ECP Reverse mode include Rising and Falling edges on nPeriphRequest (nFault).

The D2.00 P1284 specification is not clear on whether nPeriphRequest carries a valid Reverse Data Available indication during Reverse ECP mode. If so, enable status interrupts during this mode; if not, disable them.

Peripheral ECP Reverse Mode

1. In this mode, as long as nReverseRequest (nInit) is Low and P1284Active (nSelectIn) is High, the controller drives the contents of the Input/Output Register onto PIA27..PIA20, regardless of the contents of the E2 register. On entry to this mode, the controller sets Idle, and sets DREQ to request data from software, or a DMA channel.

2. If software or a DMA channel writes data to the Output Holding Register while the Input/Output Register is empty, the controller immediately transfers the byte to the IOR, clears Idle, and negates DREQ only momentarily, to request another byte.

3. In this mode, an alternate address for the Output Holding Register allows software to send a channel address or an RLE count value. Such bytes are not typically written by a DMA channel. Writing to this alternate address loads the OHR and clears DREQ, the same as writing to the primary address, but clears a ninth bit that is set when software or a DMA channel writes to the primary address. A similar ninth bit is associated with the IOR, and drives the PeriphAck (Busy) line in this mode.

If the PARVO bit is set to enable Run Length Encoding, then after transferring a normal data byte (not a channel address nor an RLE count) from the OHR to the IOR and setting DREQ, the controller waits for the DMA channel or software to write another byte to the OHR, for up to the number of Φ clocks specified in the PART register. If another character is provided within this time, thus clearing DREQ, and it is neither a channel address nor an RLE count, and it is equal to the previous character in the IOR, the controller:

1. Increments its RLE counter.
2. Places the value on the RLE counter on the PIA26..PIA20 lines and a Low on PIA27, rather than the contents of the IOR.
3. Sets DREQ again.
4. Restarts the timer with the value in PART.

Unless it has just incremented the RLE counter to 127, the controller loops on the same test criteria, so as to accumulate an RLE count as defined by the P1284 specification.

One Phi clock after any of the following occurs, the controller drives PeriphClk (nAck) Low to inform the peripheral of the character, channel address, or RLE count:

- The controller transfers a channel address or RLE count from the OHR to the IOR.

- The controller transfers a data character from the OHR to the IOR, and PARVO is 0 so that RLE encoding is disabled.
- PARVO is 1 so that RLE encoding is enabled, and within the time specified in PART after writing one character to the OHR, software or a DMA channel writes another that is a channel address, or an RLE count, or differs from the preceding character.
- PARVO is 1, and the time defined in PART expires after software or a DMA channel writes one character to the OHR, without it writing another.
- The controller increments the RLE counter to 127.

The controller waits for the host to drive HostAck (nAutoFd) high, after which the controller drives PeriphClk (nAck) back to high. Then it waits for the host to drive HostAck (nAutoFd) back to low. After HostAck (nAutoFd) is back to low, one of the following occurs:

- If the RLE counter is non-zero and the controller has been holding its value on PIA26-20, the controller clears the RLE counter to 0, and places character in the IOR back on PIA27-20. One clock later it drives PeriphClk (nAck) low again, and returns to the start of the preceding paragraph.
- If software or the DMA channel has written a new byte to the Output Holding Register and thus cleared DREQ, the controller transfers the byte to the IOR, sets DREQ again.
- If software or the DMA channel does not provide a new byte for the time indicated in the PART register after the controller sets DREQ, the controller sets the Idle flag.

While this mode is in effect, software watches for the Host to drive nReverseRequest (nInit) High. Then, the software sets the mode back to Peripheral ECP Forward, waits 500 ns, and drives nAckReverse (PError) back to High before proceeding as described for Peripheral ECP Forward mode.

Status interrupts in Peripheral ECP Reverse mode include Rising and Falling edges on P1284Active (nSelectIn) and nReverseRequest (nInit). Because no legal terminations can occur during the time this mode is set, the controller sets IllOp for any falling edge on P1284Active (nSelectIn).

Enhanced Serial Communications Controller

INTRODUCTION

This element of the Z80185/Z80195 enables serial communications in a variety of modes, including asynchronous (start-stop), character-oriented synchronous modes like Bisync, and bit-oriented synchronous modes such as SDLC, HDLC, and LocalTalk®, as well as X.25, Frame Relay, and many others.

This chapter covers the Enhanced Serial Communications Controller (ESCC) channel as used in the Z80185/Z80195, with an emphasis on asynchronous and bit-oriented synchronous modes, including the LocalTalk dialect of AppleTalk®.

ESCC Features

- Transmit FIFO
- Transmit MUX
- Data Encoding and Cyclic Redundancy Check (CRC) generation
- Receive and Transmit Clock Multiplexer
- Digital Phase-Locked Loop (DPPL)
- Baud Rate Generator (BRG)
- Crystal Oscillator Amplifier
- Modem/Control Logic
- Receive Status FIFO

- Receive MUX
- CRC Checker, Data Decode and Sync Character detection
- SDLC Frame Status FIFO

THE ESCC CHANNEL

Figure 64 depicts a block diagram of the ESCC channel. The two most important elements of the channel are:

- The Transmitter
- The Receiver

The transmitter converts *parallel* data characters provided by software or a DMA channel to *serial* characters for transmission. The Receiver converts received serial data to parallel characters that can be stored in memory by software or a DMA channel.

These two elements are supported by the following:

- A Baud Rate Generator (BRG) that can divide the master PHI clock down to a clock suitable for the serial data
- A Digital Phase-Locked Loop (DPLL) that can drive such a clock from the serial receive data
- Interrupt logic that can interrupt the processor so that software can respond to important events in the ESCC.

The Receiver stores received data in an eight-character First-In First-Out memory (FIFO), from which it can be fetched by software or a DMA channel.

Transmit characters provided by software or a DMA channel are stored in a four-character Transmit FIFO until the Transmitter is ready to send them.

These two FIFOs ease the real-time response requirements for the processor and software, and help prevent *overrun* and *underrun* conditions. Figure 64 illustrates the ESCCChannel block diagram.

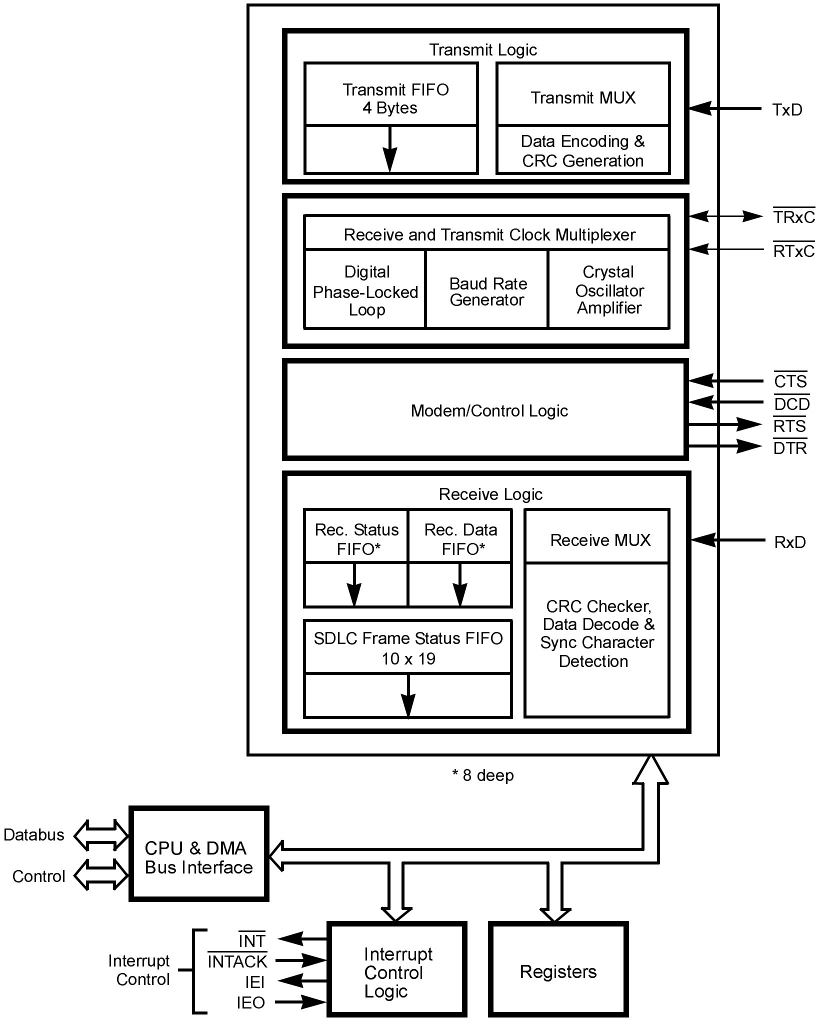


Figure 64. ESCC Channel Block Diagram

Baud Rate Generator

The Baud Rate Generator (BRG) consists of the following:

- Two 8-bit time-constant registers forming a 16-bit time constant
- A 16-bit down counter
- A flip-flop on the output so that a square wave is generated.

At start-up, the following sequence is performed:

1. The output flip-flop is set High (so that it starts in a known state)
2. The value in the time-constant register is loaded into the counter
3. The counter begins counting down.

When a count of `0` is reached, the following events occur:

1. The output of the Baud Rate Generator toggles.
2. The value in the time-constant register is re-loaded into the counter.
3. The process restarts.

The programmed time constant is contained in WR12 and WR13. Figure 65 depicts the BRG block diagram.

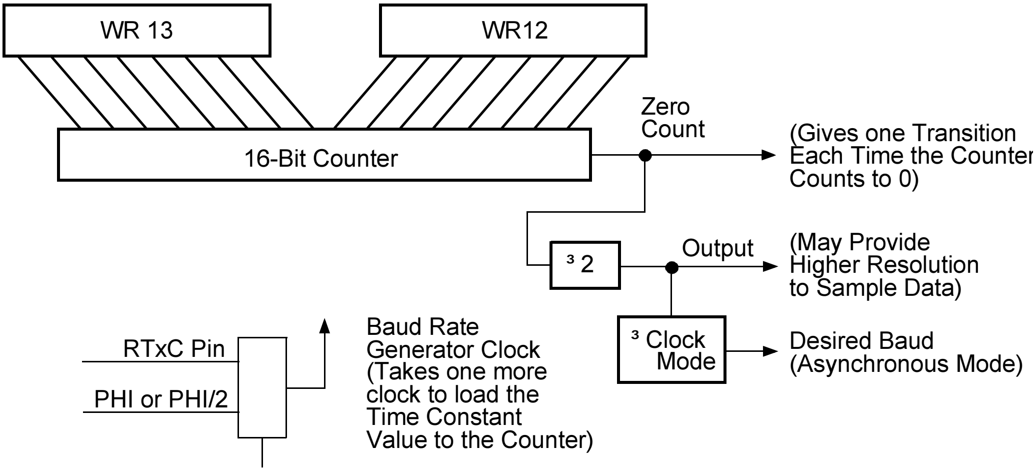


Figure 65. Baud Rate Generator Block Diagram

The time-constant can be changed at any time, but the new value does not take effect until the next load of the counter (that is, after 0 count is reached).

No attempt is made to synchronize the loading of a new time-constant with the clock used to drive the generator. When the time-constant is to be changed, the generator must be stopped first by writing WR14 Bit 0 to `0`. After loading the new time constant, the BRG can be restarted. This action ensures the loading of a correct time constant, but loading does not take place until 0 count or a reset occurs.

If neither the transmit clock nor the receive clock are programmed to come from the $\overline{\text{TRxC}}$ pin, the output of the Baud Rate Generator may be made available for external use on the $\overline{\text{TRxC}}$ pin.

The clock source for the Baud Rate Generator is selected by Bit 1 of `WR14`. When this bit is `0`, the BRG uses the signal on the $\overline{\text{RTxC}}$ pin as its clock. When this bit is `1`, the BRG is clocked by PHI or PHI/2, depending on Bit 3 in the System Configuration Register. To avoid metastable problems in the counter, this bit must be changed only while the Baud Rate Generator is disabled. Arbitrarily narrow pulses can be generated at the output of the multiplexer when this bit changes status.

The BRG is enabled when Bit 0 of `WR14` is `1`. It is disabled when `WR14` Bit 0 is `0` and after a hardware reset (but not a software reset). To prevent metastable problems when the Baud Rate Generator is first enabled, the enable bit is synchronized to the Baud Rate Generator clock. This synchronization introduces an additional delay when the Baud Rate Generator is first enabled (Figure 66). The Baud Rate Generator is disabled immediately when Bit 0 of `WR14` is `0`, because the delay is necessary only on start-up. The Baud Rate Generator can be enabled and disabled on the fly, but this delay on start-up must be taken into consideration.

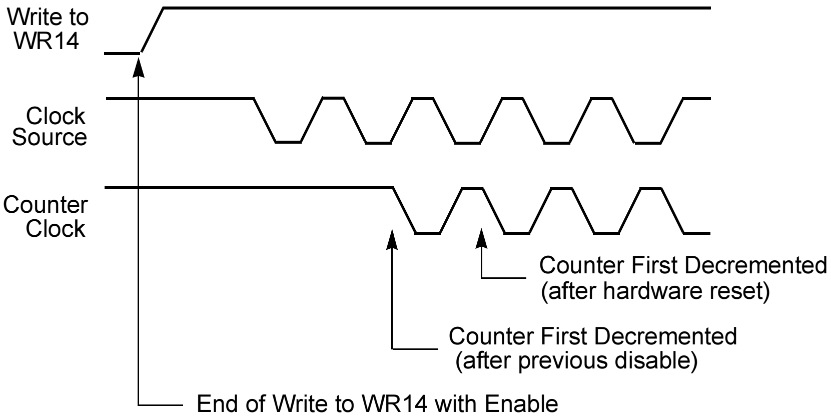

Write to
WR14
Clock
Source
Counter
Clock
Counter First Decremented
(after hardware reset)
Counter First Decremented
(after previous disable)
End of Write to WR14 with Enable

Figure 66. Baud Rate Generator Start Up

The formulas relating the baud rate and the time-constant are depicted below.

$$\text{Time Constant} = \frac{\text{Clock Frequency}}{2 \times (\text{Clock Mode}) \times (\text{Baud Rate})} - 2$$

$$\text{Baud Rate} = \frac{\text{Clock Frequency}}{2 \times (\text{Clock Mode}) \times (\text{Time Constant} + 2)}$$

In these formulas, the BRG clock frequency (PHI, PHI/2, or $\overline{\text{RTxC}}$) is in Hertz (Hz), the desired baud rate in bits/sec, Clock Mode is 1 in sync modes, 1, 16, 32 or 64 in async mode and the time constant is dimensionless. Table 13 assumes a 18.432-MHz clock, a clock mode of 16 and depicts the time constant for a number of popular baud rates.

For example:

$$\text{TC} = \frac{18.432 \times 10^6}{(2 \times 16) \times 38400} - 2 = 13$$

Table 13. Baud Rates for 18.432-MHz Clock and 16X Clock Factor

	Time Constant	
Baud Rate	**Decimal**	**Hex**
38400	13	000D
19200	28	001C
9600	58	003A
4800	118	0076
2400	238	00EE

Initializing the BRG is done in three steps. First, the time-constant is loaded into WR12 and WR13. Next, the software selects the clock source for the BRG by setting Bit 1 of WR14. Finally, the BRG is enabled by setting Bit 0 of WR14 to 1.

Data Encoding/Decoding

Data encoding allows the transmission of clock and data information over the same medium. The ESCC channel provides four different data encoding methods, selected by Bits 6 and 5 in Wr10. An example of these four encoding methods is illustrated in Figure 67. Any encoding method can be used in any X1 mode in the ESCC, asynchronous or synchronous. The data encoding selected is Active even when the transmitter or receiver is Idle or disabled.

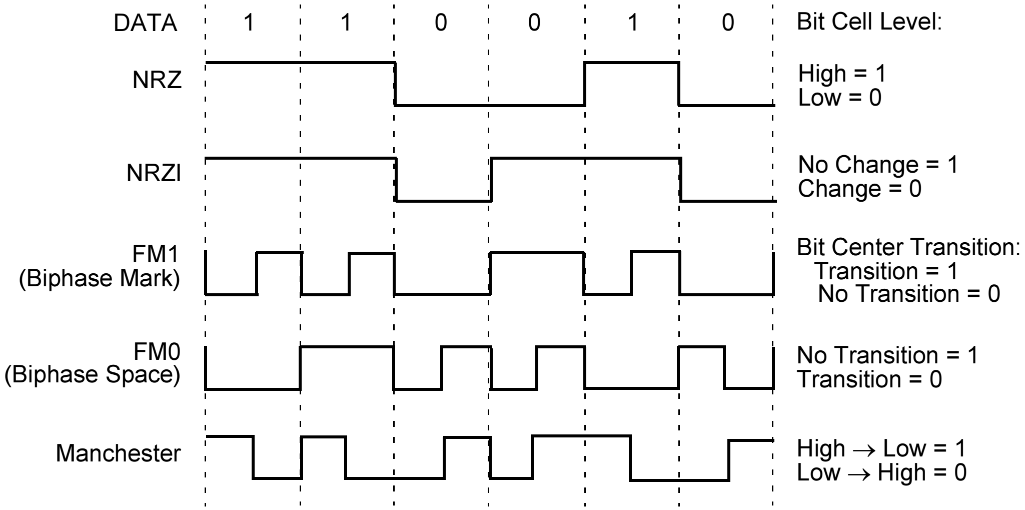


Figure 67. Data Encoding Methods

Non-Return-to-Zero Encoding

In Non-Return to Zero (NRZ) Encoding, a 1 represents a High level and a 0 represents a Low level. In this encoding method, only a minimal amount of clocking information is available in the data stream in the form of transitions on bit-cell boundaries. In an arbitrary data pattern, this information may not be sufficient to generate a clock for the data from the data itself.

Non-Return-to-Zero-Inverted Encoding

In Non-Return-to-Zero-Inverted (NRZI) Encoding, a 1 represents no change in the level and a 0 represents a change in the level. As in NRZ, only a minimal amount of clocking information is available in the data stream, in the form of transitions on bit cell boundaries. In an arbitrary data pattern this information may not be sufficient to generate a clock for the data from the data itself. In the case of SDLC, where the number of consecutive 1s in the data stream is limited, a sufficient number of transitions to generate a clock is guaranteed.

When the ESCC channel is programmed for SDLC mode with NRZI data encoding and Mark Idle (WR10 Bit 6 is 0, Bit 5 is 1, Bit 3 is 1), the TxD pin is automatically forced High when the transmitter goes to the Mark Idle state. There are several different ways for the transmitter to go into the Idle state. In each of the following cases the TxD pin is forced High when the Mark Idle condition is reached:

- Data
- CRC
- Flag and idle
- Data, flag and idle
- Data, abort (on underrun) and idle
- Data, abort (command) and idle
- Idle flag and command to idle mark.

The Force High feature is disabled when the Mark Idle bit is reset. The TxD pin is forced High on the falling edge of the TxC cycle after the falling edge of the last bit of the closing flag. Using SDLC Loop mode is independent of this feature.

This feature is used in combination with the automatic SDLC opening flag transmission feature, WR7′ Bit 0 is 1, to assure that data packets are properly formatted. Therefore, when these features are used together, it is not necessary for the CPU to issue any commands when using the Force Idle mode in combination with NRZI data encoding. If WR7′ Bit 0 is reset (0), it is necessary to reset WR10 Bit 2 (Mark Idle) to enable flag transmission before an SDLC packet is transmitted.

Bi-Phase Mark Encoding

In Bi-Phase Mark (FM1) Encoding, also known as Bi-Phase Mark, a transition is present on every bit cell boundary. An additional transition may be present in the middle of the bit-cell. In FM1, a 0 is sent as no transition in the center of the bit cell and a 1 is sent as a transition in the center of the bit cell. FM1-encoded data contains sufficient information to recover a clock from the data.

Bi-Phase Space Encoding

In Bi-Phase Space (FM0) Encoding, also known as Bi-Phase Space, a transition is present on every bit cell boundary and an additional transition may be present in the middle of the bit cell. In FM0, a 1 is sent as no transition in the center of the bit cell and a 0 is sent as a transition in the center of the bit cell. FM0-encoded data contains sufficient information to recover a clock from the data.

Manchester Encoding

Manchester (Bi-Phase Level) Encoding always produces a transition at the center of the bit cell. If the transition is Low to High, the bit is 0. If the transition is High to Low, the bit is 1. Encoding of Manchester format requires an external circuit consisting of a D flip-flop and four gates (Figure 68). The ESCC is used to decode Manchester data by using the DPLL in the FM mode and programming the receiver for NRZ data.

Data Encoding Initialization

The data encoding method is selected in the initialization procedure before the transmitter and receiver are enabled, but no other restrictions apply. In NRZ and NRZI, the receiver samples the data only on one edge, as depicted in Figure 68. However, in FM1 and FM0, the receiver samples the data on both edges. Also, as depicted in Figure 68, the transmitter defines bit cell boundaries by one edge in all cases and uses the other edge in FM1 and FM0 to create the mid-bit transition.

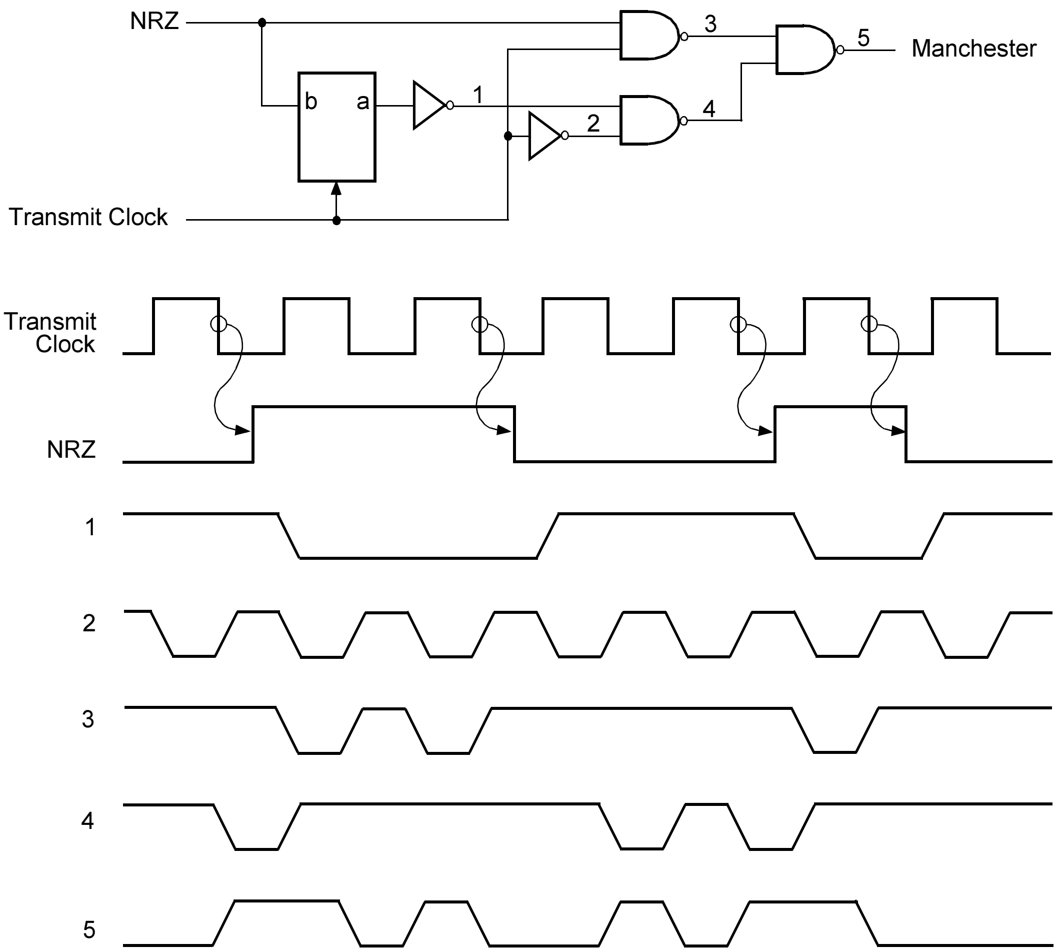


Figure 68. Manchester Encoding Circuit

Digital Phase-Locked Loop

The ESCC channel contains a Digital Phase-Locked Loop (DPLL) that can be used to recover clock information from a data stream with NRZI, FM, NRZ, or Manchester encoding. The DPLL is driven by a clock nominally at 32 (NRZI) or 16 (FM) times the data rate. The DPLL uses this clock, along with the data stream, to construct a receive clock for the data. This clock can then be used as the receive clock, the transmit clock, or both.

Figure 69 depicts a block diagram of the DPLL. It consists of a 5-bit counter, an edge detector, and a pair of output decoders. The clock for the DPLL comes from the output of a two-input multiplexer, and the two outputs go to the transmitter and receive clock multiplexers. The DPLL is controlled by seven commands encoded in `WR14` Bits 7..5.

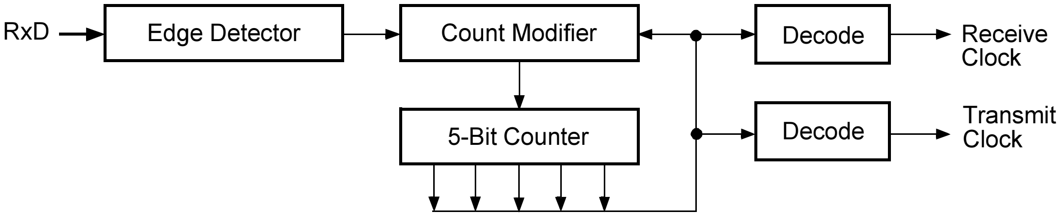


Figure 69. Digital Phase-Locked Loop Block Diagram

The clock source for the DPLL is selected by setting `WR14` Bits 7..5 with one of two values.

- Setting `WR14` Bits 7..5 to `100` selects the BRG
- Setting `WR14` Bits 7..5 to `101` selects the $\overline{\text{RTxC}}$ pin

The first command selects the Baud Rate Generator as the clock source. The second command selects the $\overline{\text{RTxC}}$ pin as the clock source.

Initialization of the DPLL must be done after the clock modes have been selected in `WR11`, and before the receiver and transmitter are enabled.

When initializing the DPLL, the clock source must be selected first, followed by the selection of the operating mode.

To avoid metastable problems in the counter, the clock source must be selected only while the DPLL is disabled, because arbitrarily narrow pulses can be generated at the output of the multiplexer when it changes status.

The DPLL can operate in one of two modes, selected by setting `WR14` Bits 7..5 to one of the following values:

- Setting `WR14` Bits 7..5 to `111` selects NRZI mode
- Setting `WR14` Bits 7..5 to `110` selects FM mode

Note: A channel or hardware reset disables the DPLL, selects the $\overline{\text{RTxC}}$ pin as the clock source for the DPLL, and places it in the NRZI mode.

As in the case of the clock source selection, the mode of operation must be changed only while the DPLL is disabled to prevent unpredictable results.

In the NRZI mode, the DPLL clock must be 32 times the data rate. In this mode, the transmit and receive clock outputs of the DPLL are identical, and the clocks are phased so that the receiver samples the data in the middle of the bit cell. In NRZI mode, the DPLL does not require a transition in every bit cell, so this mode is useful for recovering the clocking information from NRZ and NRZI data streams.

In the FM mode, the DPLL clock must be 16 times the data rate. In this mode, the transmit clock output of the DPLL lags the receive clock outputs by 90 degrees to make the transmit and receive bit cell boundaries the same, because the receiver must sample FM data at the one-quarter and three-quarters bit times.

The DPLL is enabled by issuing the Enter Search Mode command in `WR14`; that is, `WR14` Bits 7..5 are `001`. The Enter Search Mode command unlocks the counter, which is held while the DPLL is disabled, and enables the edge detector. If the DPLL is already enabled when this command is issued, the DPLL also enters Search Mode.

In Search mode, the counter is held at a specific count and no outputs are provided. The DPLL remains in this state until an edge is detected in the receive data stream. This first edge is assumed to occur on a bit cell boundary, and the DPLL begins providing an output to the receiver that properly samples the data. From this point on, the DPLL adjusts its output to remain in phase with the receive data. If the first edge that the DPLL detects does not occur on a bit cell boundary, the DPLL locks on to the receive data, but takes longer to do so.

DPLL Operation in the NRZI Mode

To operate in NRZI mode, the DPLL must be supplied with a clock that is 32 times the data rate. The DPLL uses this clock, along with the receive data, to construct receive and transmit clock outputs that are phased to properly receive and transmit data.

The DPLL divides each bit cell into four regions, and makes an adjustment to the count cycle of the 5-bit counter dependent upon the region a transition on the receive data input occurred (Figure 70).

Ordinarily, a bit cell boundary occurs between count 15 and count 16, and the DPLL output causes the data to be sampled in the middle of the bit cell. However, four different situations can occur:

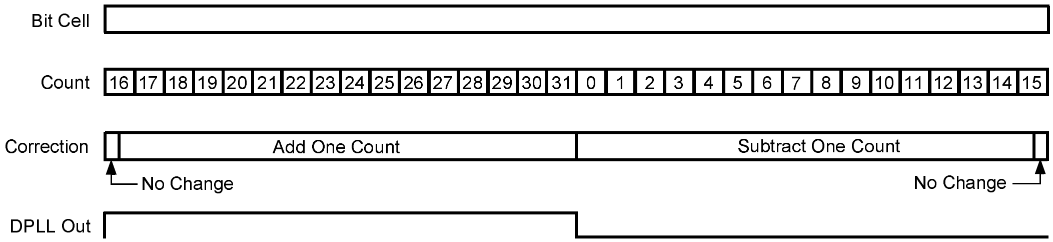


Figure 70. DPLL Operation in NRZI Mode

- If the bit cell boundary (from space to mark) occurs anywhere during the second half of count 15 or the first half of count 16, the DPLL allows the transition without making a correction to its count cycle.
- If the bit cell boundary (from space to mark) occurs between the middle of count 16 and count 31, the DPLL is sampling the data too early in the bit cell. In response to this, the DPLL extends its count by one during the next 0 to 31 counting cycle, which effectively moves the edge of the clock that samples the receive data closer to the center of the bit cell.
- If the transition occurs between count 0 and the middle of count 15, the output of the DPLL is sampling the data too late in the bit cell. To correct this, the DPLL shortens its count by one during the next 0 to 31 counting cycle, which effectively moves the edge of the clock that samples the receive data closer to the center of the bit cell.
- If the DPLL does not see any transition during a counting cycle, no adjustment is made in the following counting cycle.

If an adjustment to the counting cycle is necessary, the DPLL modifies count 5, either deleting it or doubling it. Thus, only the Low time of the DPLL output is lengthened or shortened.

While the DPLL is in Search mode, the counter remains at count 16, where the DPLL outputs are both High. The missing clock latches in the DPLL, which may be accessed in RR10, are not used in NRZI mode. An example of the DPLL in operation is illustrated in Figure 71.

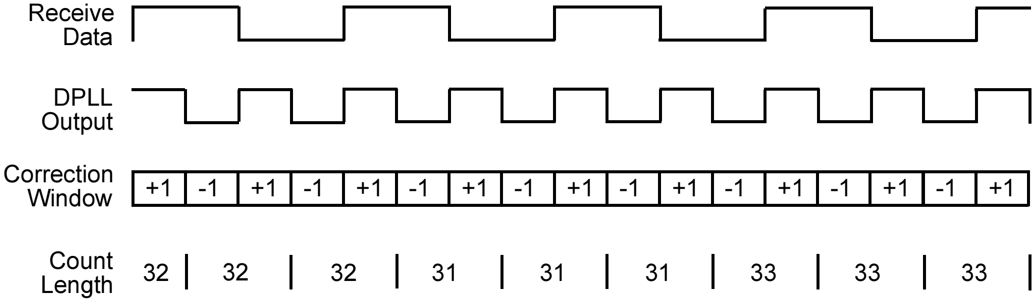


Figure 71. DPLL Operating Example, NRZI Mode

DPLL Operation in the FM Mode

To operate in FM mode, the DPLL must be supplied with a clock that is 16 times the data rate. The DPLL uses this clock, along with the receive data, to construct, receive, and transmit clock outputs that are phased to receive and transmit data properly.

In FM mode, the counter in the DPLL counts from 0 to 31, but now each cycle corresponds to 2-bit cells. To make adjustments to remain in phase with the receive data, the DPLL divides a pair of bit cells into five regions, making the adjustment to the counter dependent upon in which the region the transition on the receive data input occurred Figure 72).

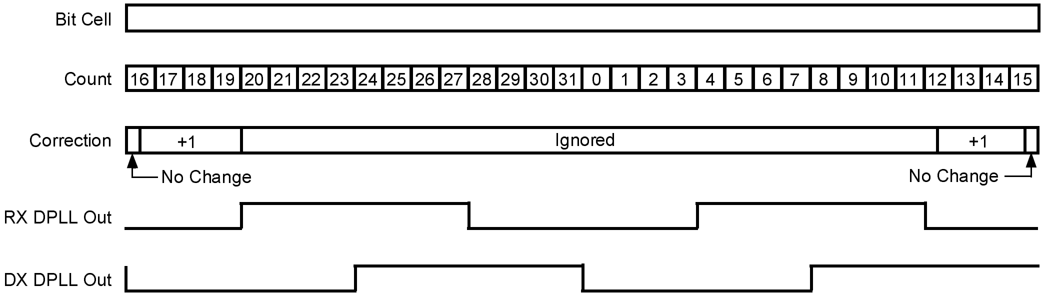


Figure 72. .DPLL Operation in FM Mode

In FM mode, the transmit clock and receive clock outputs from the DPLL are not in phase. This condition is necessary to make the transmit and receive bit cell boundaries coincide, because the receive clock must sample the data one-fourth and three-fourths of the way through the bit cell.

A bit cell boundary occurs between count 15 or count 16 and the DPLL Receive Output causes the data to be sampled at one-fourth and three-fourths of the way through the bit cell.

However, four variations can occur:

- If the bit-cell boundary (from Space to Mark) occurs anywhere during the second half of count 15 or the first half of count 16, the DPLL allows the transition without making a correction to its count cycle.
- If the bit-cell boundary (from Space to Mark) occurs between the middle of count 16 and the middle of count 19, the DPLL is sampling the data too early in the bit cell. In response, the DPLL extends its count by one during the next 0 to 31 counting cycle, which moves the receive clock edges closer to where they belong.
- Any transitions occurring between the middle of count 19 in one cycle and the middle of count 12 during the next cycle are ignored by

the DPLL. Ignoring these transitions guarantees that any data transitions in the bit cells do not cause an adjustment to the counting cycle.

- If no transition occurs between the middle of count 12 and the middle of count 19, the DPLL is not locked onto the data. When the DPLL misses an edge, the One Clock Missing bit in `RR10` Bit 7 is `1` and latched. It remains set until a Reset Missing Clock command is issued in `WR14`, or until the DPLL is disabled or programmed to enter the Search mode. At missing this one edge, the DPLL takes no other action and does not modify its count during the next counting cycle.

If the DPLL does not detect an edge between the middle of count 12 and the middle of count 19 in two successive 0 to 31 count cycles, a Line Error condition is assumed. If this condition occurs, the Two Clocks Missing bit in `RR10` (Bit 6) is `1` and latched. At the same time, the DPLL enters the Search mode.

The DPLL enters the Search mode during count 2, when both the receive clock and transmit clock outputs are Low. This action prevents any glitches on the clock outputs when Search mode is entered. While in the Search mode, no clock outputs are provided by the DPLL. The Two Clocks Missing bit in `RR10` is latched until a Reset Missing Clock command is issued in `WR14`, or until the DPLL is disabled or programmed to enter the Search mode.

While the DPLL is disabled, the transmit clock output of the DPLL may be toggled by alternately selecting FM and NRZI mode in the DPLL. The same is true of the receive clock.

While the DPLL is in Search mode, the counter remains at count 16 in which the receive output is Low and the transmit output is Low. This action provides a transmit clock under software control because the DPLL is in Search mode while it is disabled.

As in NRZI mode, if an adjustment to the counting cycle is necessary, the DPLL modifies count 5, either deleting or doubling it. If no adjustment is necessary, the count sequence proceeds.

When the DPLL is programmed to enter Search mode, only clock transitions exist on the receive data pin. If this is not true, the DPLL may attempt to lock on to the data transitions. If the DPLL does lock on to the data transitions, then the Missing Clock condition occurs because data transitions are not guaranteed every bit-cell.

To lock in the DPLL, FM0 encoding requires continuous 1s to be received when leaving Search mode. In FM1 encoding, continuous 0s are required; with Manchester-encoded data alternating 1s and 0s are required. With all of these data-encoding methods, there is at least one transition in every bit-cell. In FM mode the DPLL is designed to expect this transition.

DPLL Operation in the Manchester Mode

The ESCC channel can decode Manchester data by using the DPLL in the FM mode and programming the receiver for NRZ data. Manchester-encoded data contains a transition at the center of every bit-cell; the direction of this transition distinguishes a 1 from a 0. Therefore, for Manchester data, the DPLL must be in FM mode (WR14 command Bits 7..5 are 110), but the receiver must be set up to accept NRZ data (WR10 Bits 6..5 are 00).

Transmit Clock Counter

The Transmit Clock Counter parallels the DPLL. This counter provides a jitter-free clock source to the transmitter by dividing the DPLL clock source by the appropriate value for the programmed data encoding format (Figure 73). In FM mode (FM0 or FM1), the counter output is the input frequency divided by 16. In NRZI mode, the counter frequency is the input divided by 32. The counter output replaces the DPLL transmit clock output, available as the transmit clock source. This action has no effect on the use of the DPLL as the receive clock source.

The output of the transmit clock derived from this counter is available to the $\overline{\text{TRxC}}$ pin when the DPLL output is selected as the transmit clock source. Take care when using the ESCC channel in SDLC Loop mode with the DPLL. The SDLC Loop mode requires synchronized Tx and Rx clocks, but the ESCC's DPLL might be off-sync because of the Transmit Clock Counter. In SDLC Loop, echo the signal of the Rx DPLL out to clock the receiver and transmitter to achieve synchronization. This procedure can be programmed via Bits 1..0 in `WR11`.

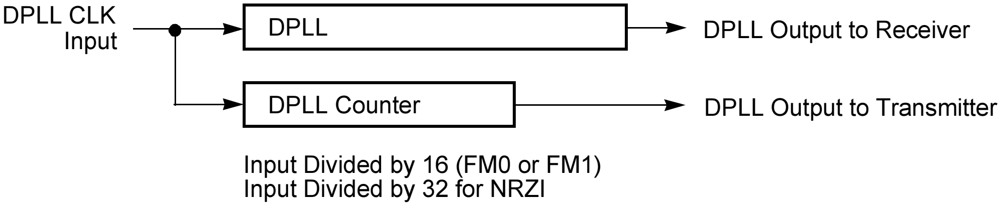

DPLL CLK
Input
DPLL
DPLL Output to Receiver
DPLL Counter
DPLL Output to Transmitter
Input Divided by 16 (FM0 or FM1)
Input Divided by 32 for NRZI

Figure 73. DPLL Transmit Clock Counter Output (ESCC Only)

Clock Selection

The ESCC channel can select several clock sources for internal and external use. Write Register 11 is the Clock Mode Control register for both the receive and transmit clocks. It determines the type of signal on the $\overline{\text{RTxC}}$ pin and the direction of the $\overline{\text{TRxC}}$ pin.

The source of the receive clock is controlled by `WR11` Bits 6..5. The receive clock may be programmed to come from the $\overline{\text{RTxC}}$ pin, the $\overline{\text{TRxC}}$ pin, the output of the Baud Rate Generator, or the receive output of the DPLL.

The source of the transmit clock is controlled by `WR11` Bits 4..3. The transmit clock may be programmed to come from the $\overline{\text{RTxC}}$ pin, the $\overline{\text{TRxC}}$ pin, the output of the Baud Rate Generator, or the transmit output of the DPLL.

Ordinarily, the $\overline{\text{TRxC}}$ pin is an input, but it can become an output if this pin has not been selected as the source for the transmitter or the receiver, and Bit 2 of `WR11` is `1`. The selection of the signal provided on the $\overline{\text{TRxC}}$

output pin is controlled by `WR11` Bits 1..0. The $\overline{\text{TRxC}}$ pin can be programmed to provide the output of the Baud Rate Generator, the receive output of the DPLL or the actual transmit clock. The option of placing the transmit clock signal on the $\overline{\text{TRxC}}$ pin when it is an output allows access to the transmit output of the DPLL.

Figure 74 illustrates a simplified schematic diagram of the circuitry used in clock multiplexing. It depicts the inputs to the multiplexer section, as well as the various signal inversions that occur in the paths to the outputs.

Selection of the clocking options may be performed anywhere in the initialization sequence, but the final values must be selected before the receiver, transmitter, Baud Rate Generator, or DPLL are enabled, to prevent problems caused by arbitrarily narrow clock signals from the multiplexers.

Also described are the edges used by the receiver, transmitter, Baud Rate Generator and DPLL to sample or send data or otherwise change state. For example, the receiver samples data on the falling edge, but because there is an inversion in the clock path between the $\overline{\text{RTxC}}$ pin and the receiver, a rising edge of the $\overline{\text{RTxC}}$ pin samples the data for the receiver.

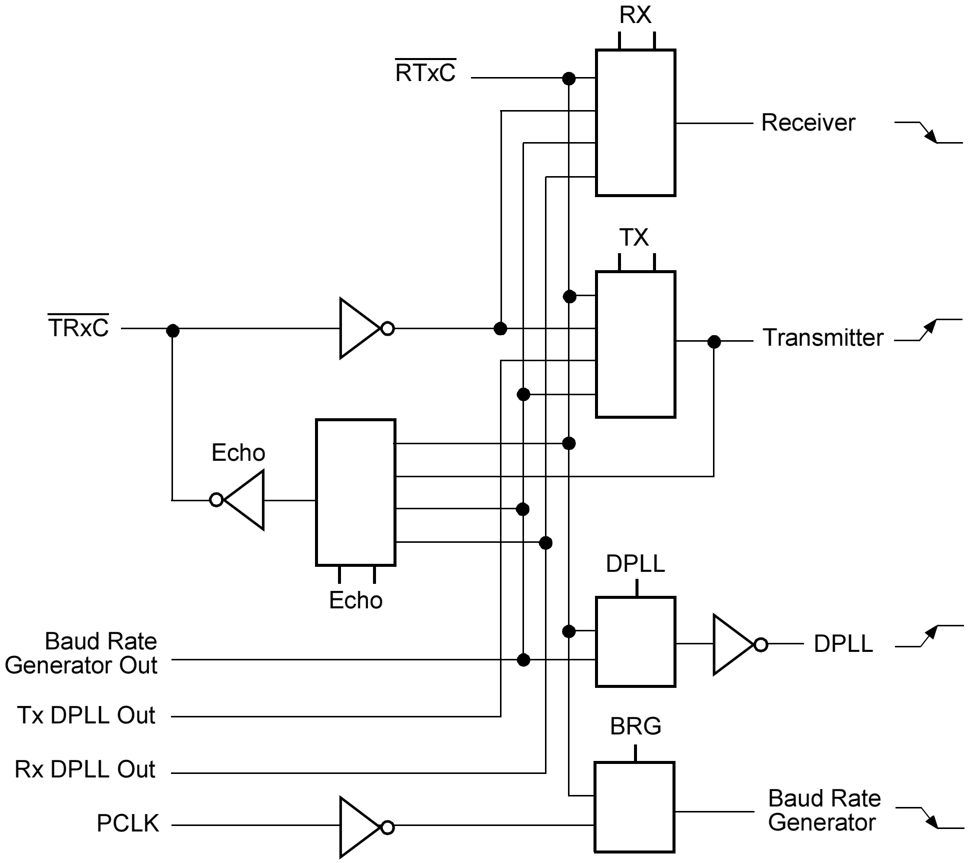


Figure 74. Clock Multiplexer

Transmit Data Path

A diagram of the transmit data path is illustrated in Figure 75. The transmitter has a 4-byte deep FIFO which is addressed through WR8. The Transmit Shift register is loaded from either WR6, WR7, or the Transmit Data (Tx FIFO) buffer. In Synchronous modes, WR6 and WR7 are programmed with the sync characters. In Monosync mode, an 8-bit or 6-bit sync character is used (WR6), whereas a 16-bit sync character is used in the Bisynchronous mode (WR6 and WR7). In bit-oriented Synchronous modes, the SDLC flag character (7EH) is programmed in WR7 and is loaded into the Transmit Shift Register at the beginning and end of each message.

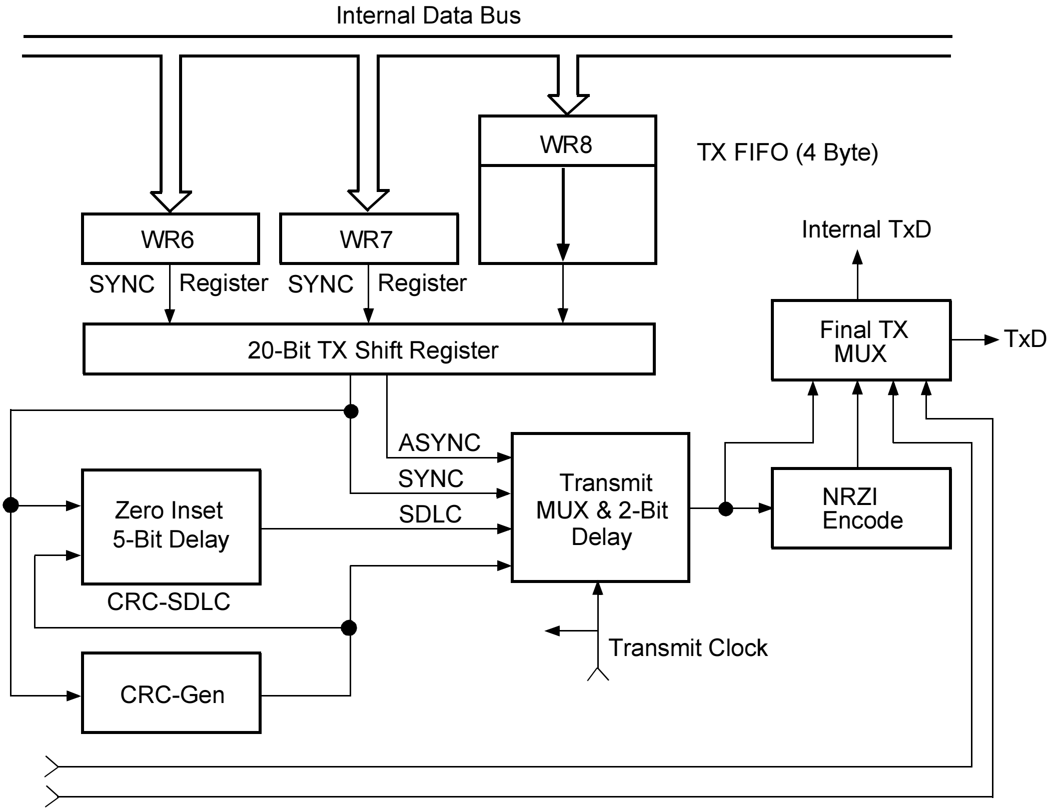


Figure 75. Transmit Data Path Block Diagram

For asynchronous data, the Transmit Shift register is formatted with Start and Stop bits along with the data; and optionally with a parity information bit. The formatted character is shifted out to the transmit multiplexer at the selected clock rate. WR6 and WR7 are not used in Asynchronous mode.

Synchronous data (except SDLC/HDLC) is shifted to the CRC generator as well as to the transmit multiplexer. SDLC/HDLC data is shifted to the CRC Generator and out through the zero-insertion logic (which is disabled while flags are being sent). A 0 is inserted in all address, control, information, and frame check fields following five contiguous 1s in the data stream. The result of the CRC generator for SDLC data is also routed through the zero-insertion logic and then to the transmit multiplexer.

Receive Data Path

The receiver has an 8-byte deep, 8-bit wide Data FIFO, Error FIFO and an 8-bit Shift Register. The receive data path is illustrated in Figure 76. For each data byte in the Receive FIFO, parity, framing, and other status information are loaded into the Error FIFO. The Error FIFO is addressed through Read Register 1

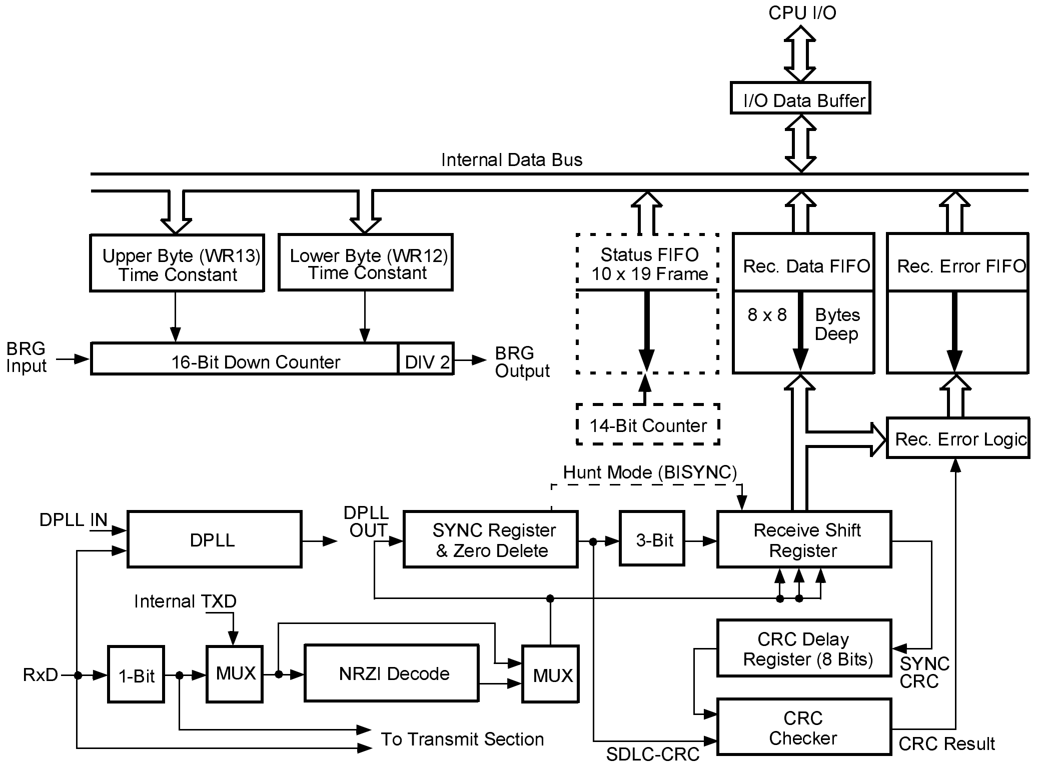


Figure 76. Receive Data Path Block Diagram

Incoming data is routed through one of several paths depending on the mode and character length. In Asynchronous mode, serial data enters the 3-bit delay if a character length of seven or eight bits is selected. If a character length of five or six bits is selected, data enters the receive shift register directly.

In Synchronous modes, the data path is determined by the phase of the receive process currently in operation. A synchronous receive operation begins with a hunt phase in which a bit pattern that matches the programmed sync characters (6-, 8-, or 16-bit) is searched.

The incoming data then passes through the Sync register and is compared to a sync character stored in `WR6` or `WR7` (depending on which mode is selected). The Monosync mode matches the sync character programmed in `WR7` and the character assembled in the Receive Sync register to establish synchronization.

Synchronization is achieved differently in the Bisync mode. Incoming data is shifted to the Receive Shift register while the next eight bits of the message are assembled in the Receive Sync register. If these two characters match the programmed characters in `WR6` and `WR7`, synchronization is established. Incoming data can then bypass the Receive Sync register and enter the 3-bit delay directly.

The SDLC mode uses the Receive Sync register to monitor the receive data stream and to perform Zero Deletion when necessary; that is, when five continuous `1`s are received, the sixth bit is inspected and deleted from the data stream if it is `0`. The seventh bit is inspected only if the sixth bit is `1`. If the seventh bit is `0`, a flag sequence has been received and the receiver is synchronized to that flag. If the seventh bit is `1`, an Abort or an End Of Poll (EOP) is recognized, depending upon the selection of either the normal SDLC mode or SDLC Loop mode.

Note: The insertion and deletion of the 0 in the SDLC data stream is transparent to the user, as it is performed after the data is written to the Transmit FIFO and before data is read from the Receive FIFO. This feature of the SDLC/HDLC protocol prevents the inadvertent sending of a Flag or Abort sequence as part of the data stream.

The same path is taken by incoming data for both SDLC and SDLC Loop modes. The reformatted data enters the 3-bit delay and is transferred to the Receive Shift register. The SDLC receive operation begins in the Hunt phase by attempting to match the assembled character in the Receive Shift

Register with the flag pattern in WR7. When the flag character is recognized, subsequent data is routed through the same path, regardless of character length.

Either the CRC-16 or CRC-SDLC (Cyclic Redundancy Check or CRC) polynomial can be used for both Monosync and Bisync modes, but only the CRC-SDLC polynomial is used for SDLC operation. The data path taken for each mode is also different. Bisync protocol is a byte-oriented operation that requires the CPU to decide whether or not a data character is to be included in CRC calculation. An 8-bit delay in all Synchronous modes except SDLC is allowed for this process. In SDLC mode, all bytes are included in the CRC calculation.

Serial Modes and Protocols

The ESCC channel can transmit and receive serial data in several major modes, or line protocols. Asynchronous mode dates back to the time when the first teletypewriters were succeeding Morse code for data communications. Async mode features a Start bit that precedes each character and a Stop bit that follows each character. As depicted in Figure 77, the Stop and Start bits have opposite polarity on the line. The Stop bit can be extended to any time-duration if the transmitting station does not have another character to send. Asynchronous receivers compensate for this temporal variability by *oversampling* the serial received data, typically 16 times per serial-data bit.

This oversampling restricts the serial data rate at which async data can be sent. The Start and Stop bits represent nonproductive overhead within the serial data stream. Character-Oriented Synchronous communications schemes are typically faster than Async modes, because characters follow each other at regular intervals. Oversampling is eliminated because there are no start and stop bits between characters.

In Character-Oriented Synchronous modes, several characters are sent together in a frame or message. The start of a frame or message is signalled by some identifiable bit pattern that does not occur between frames or messages. The length of a frame or message may be sent as part of the data near the start of the frame, or the end of a frame or message may be signalled by an identifiable bit pattern that does not occur within a frame or message. The starting and ending bit patterns are called *control* characters (not data characters).

These schemes proliferated during the 1960's as did the hardware requirements for recognizing the start- and end-of-message. Particularly acute was the problem of transparency, the ability to send any kind of data without regard for conflicts with the control characters that delimit frames.

Bit-Oriented Synchronous modes were developed in response to the complexity of character-oriented synchronous modes. Beginning with IBM’s SDLC, followed by HDLC and X.25, and more recently Frame Relay and Apple's LocalTalk, all of these schemes use the framing method depicted in Figure 78.

One unique bit sequence called a *Flag*, consisting of six consecutive `1`s between `0` bits, signals both the start and end of a frame. Transparency is guaranteed by having bit-oriented synchronous transmitters and receivers monitor the data within frames, and insert and delete extra `0` bits (respectively) to ensure that six `1`s in a row are never sent within a frame.

Such modes provide the performance of character-oriented synchronous modes without their hardware and software complexity.

The ESCC channel can communicate in all three of these modes. This document concentrates on the Async and SDLC/HDLC modes.

Asynchronous Mode

In asynchronous communications, data is transferred in the format illustrated in Figure 77.

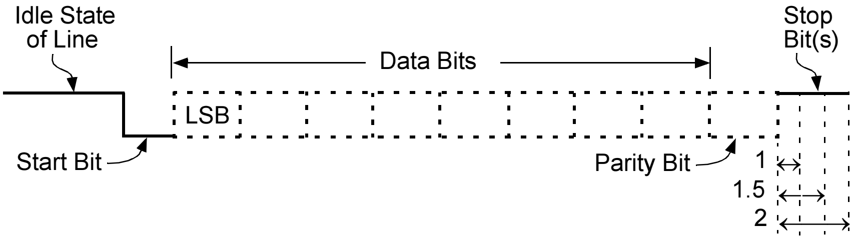


Figure 77. Asynchronous Message Format

The transmission of a character begins when the line makes a transition from the `1` state (or Mark condition) to the `0` state (or Space condition). This transition is the reference from which the character's bit cell boundaries are defined. Though the transmitter and receiver have no common clock signal, their data rates must be similar so that the receiver can sample the data near the center of each bit-cell.

The ESCC channel also supports Isochronous mode, which is the same as Asynchronous except that the clock is the same rate as the data. Isochronous mode is achieved by selecting X1 clock mode in `WR4` (Bits 7..6 are `00`). Using this mode typically requires that the transmit clock be transmitted along with the data, or that the clock be synchronized with the data.

Each character can be broken up into four parts:

- Start bit–signals the beginning of a character frame
- Data bits–typically 5..8 bits wide
- Parity bit–optional error checking mechanism
- Stop bits–provide a minimum interval between the end of one character and the beginning of the next

Generation and checking of parity is optional and is controlled by `WR4` Bits 1..0. `WR4` Bit 0 is used to enable parity. If `WR4` Bit 1 is `1`, even parity is selected. If Bit 1 is `0`, Odd parity is selected. For Even parity, the parity bit is set/reset so that the data byte plus the parity bit contains an even number of `1`s. For Odd parity, the parity bit is set/reset such that the data byte plus the parity bit contains an odd number of `1`s.

The ESCC channel supports Asynchronous mode with a number of programmable options including the number of bits per character, the minimum number of Stop bits, the clock factor, modem interface signals, and break detection and generation.

Asynchronous mode is selected by programming the desired number of stop bits in `WR4` Bits 3..2. Programming these two bits with other than `00` places both the receiver and transmitter in Asynchronous mode. In this mode, the ESCC ignores the state of `WR3` Bits 4..2, `WR4` Bits 5..4, `WR5` Bits 2..0, all of `WR6` and `WR7`, and all of `WR10` except Bits 6..5. See Table 14.

Table 14. Write Register Bits Ignored in Asynchronous Mode

Bit	7	6	5	4	3	2	1	0
WR3				x	x	x	0	
WR4			x	x				
WR5						x		x
WR6	x	x	x	x	x	x	x	x
WR7	x	x	x	x	x	x	x	x
WR10	x			x	x	x	x	x

Note: If `WR3` Bit 1 is `1` (enabling the sync character load inhibit feature), any character matching the value in `WR6` is stripped out of the incoming data stream and not put into the Receive FIFO. Because this feature applies only to synchronous formats, this bit must be reset in Asynchronous mode.

Asynchronous Transmit

Asynchronous mode is selected by specifying the number of stop bits per character in WR4 Bits 3..2. The three stop bit options are as follows:

- One
- One-and-a-half
- Two Stop bits per character

These two bits select only the minimum number of stop bits for the transmitter, as the receiver always accepts any stop bit longer than 1/2 bit time.

The number of data bits per transmitted character is controlled both by WR5 Bits 6..5 and the way the data is formatted within the Transmit FIFO. The bits in WR5 allow the option of 5, 6, 7, or 8 bits per character (Table 15). In all cases the data must be right-justified, with the unused bits being ignored except in the case of 5 bits per character. When the 5 bits per character option is selected, the data may be formatted before being written to the transmit buffer. This option allows transmission of from one to five bits per character. The formatting is described in Table 16.

Table 15. Transmit Bits per Character

Value		
Bit 6	**Bit 5**	**Bits per Character**
0	0	5 or Less
0	1	7
1	0	6
1	1	8

Note: For five or fewer bits per character selection in WR5, the following encoding is used in the data sent to the transmitter.

Table 16. Data Encoding for Five or Fewer Data Bits

Value								
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Result
1	1	1	1	0	0	0	D	Send one data bit
1	1	1	0	0	0	D	D	Send two data bits
1	1	0	0	0	D	D	D	Send three data bits
1	0	0	0	D	D	D	D	Send four data bits
0	0	0	D	D	D	D	D	Send five data bits

An additional bit, carrying parity information, may be automatically appended to every transmitted character by setting Bit 0 of WR4 to 1. This bit is sent in addition to the number of bits specified in WR4 or by Bit 1 of WR4. If this bit is 1, the transmitter sends even parity and, if 0, the parity is Odd.

The transmitter may be programmed to send a Break by setting WR5 Bit 4 to 1. The transmitter sends contiguous 0s from the first transmit clock edge after this command is issued, until the first transmit clock edge after this bit is reset. The transmit clock edges referred to here are those that define transmitted bit-cell boundaries.

As mentioned previously, the ESCC initiates the Break sequence regardless of character boundaries. Typically, the break sequence is defined as null character (all 0 data) with a Framing Error. The other party may not be able to recognize it as a Break sequence if the Send Break bit has been set in the middle of sending a non-zero character.

An additional status bit for use in Asynchronous mode is available in RR1 Bit 0. This bit, All Sent, is set when the transmitter is completely empty and any previous data or Stop bits have reached the TxD pin. The All Sent bit can be used by the processor as an indication that the transmitter may be safely disabled, or as an indication to change the signalling to the modem.

The ESCC channel may be programmed to accept a transmit clock that is 1, 16, 32, or 64 times the data rate. These transmit clock options are selected by WR4 Bits 7..6, for both the receiver and transmitter.

Note: When using Isosynchronous (X1 clock) mode, one-and-a-half Stop bits are not allowed. Select only one or two Stop bits. If some length other than one stop bit is desired in the X1 mode, only two stop bits may be used. Also, in the X1 mode, the Transmitter must send clocking information (transmit clock) along with the data in order to receive data.

There are two modem control signals associated with the transmitter, $\overline{\text{RTS}}$ and $\overline{\text{CTS}}$.

The $\overline{\text{RTS}}$ pin is an output that carries the inverted state of Bit 1 in WR5 (RTS), unless WR3 Bit 5 (Auto Enables) is set. When Auto Enables is 1, the $\overline{\text{RTS}}$ pin goes Low when the RTS bit is 1. However, when the RTS bit is 0, the $\overline{\text{RTS}}$ pin remains Low until the transmitter is completely empty and the last Stop bit has left the TxD pin. Thus, the $\overline{\text{RTS}}$ pin may be used to disable external drivers for the transmit data. The $\overline{\text{CTS}}$ pin is ordinarily an Input to RR0 Bit 5 (CTS). However, if Auto-Enabled mode is selected, this pin becomes an enable for the transmitter. That is, if Auto Enables is On and the $\overline{\text{CTS}}$ pin is High, the transmitter is disabled; the transmitter is enabled while the $\overline{\text{CTS}}$ pin is Low.

The initialization sequence for the transmitter in Asynchronous mode is to write WR4 first to select the mode. Then write WR3 and WR5 to select the various options. At this point the other registers are initialized as necessary. When complete, the transmitter may be enabled by setting WR5 Bit 3 to 1. The transmitter and receiver may be initialized at the same time.

Characters are loaded from the Transmit FIFO to the shift register where they are given a Start bit and, if selected, a Parity bit. The characters are shifted out to the TxD pin.

The ESCC can generate an interrupt or DMA request depending on the status of the Transmit FIFO. If WR7′ Bit 5 is 0, the transmit buffer empty interrupt or DMA request is asserted when the entry location of the Transmit FIFO is empty (one byte can be written). If WR7′ Bit 5 is 1, the transmit interrupt or DMA request is generated when the Transmit FIFO is completely empty (four bytes can be written). RR0 Bit 2 (TBE) is also affected by the state of WR7′ Bit 5. RR1 Bit 0 (All Sent) can be polled to determine when the last bit of transmit data has cleared the TxD pin.

The number of transmit interrupts can be minimized by setting WR7′ Bit 5 to 1 and writing four bytes to the transmitter for each transmit interrupt.

Asynchronous Mode

Asynchronous mode is selected by specifying the number of stop bits per character in WR4 Bits 3..2. This selection applies only to the transmitter, however, as the receiver accepts any Stop bit(s) longer than 1/2 bit time. If, after character assembly, the receiver samples the first Stop bit as a 0, the Framing Error bit in the receive error FIFO is set at the same time that the character is transferred to the receive data FIFO. This error bit accompanies the data to the exit location (CPU side) of the Receive FIFO, where it is a special receive condition. The Framing Error bit is not latched, so it must be read in RR1 before the accompanying data is read.

The number of bits per character is controlled by WR3 Bits 7..6. Five, six, seven or eight bits per character may be selected with these two bits. Data is right-justified with the unused bits set to 1s. An additional bit, carrying parity information, may be selected by setting WR4 Bit 0 to 1. This setting also enables parity for the transmitter. The parity sense is selected by WR4 Bit 1. If this bit is 1, the received character is checked for even parity, and if 0, the received character is checked for Odd parity. The parity bit is transferred to the receive data FIFO along with the data, if the data plus parity is eight bits or less. The parity error bit in the receive error FIFO may be programmed to cause special receive interrupts by setting WR1 Bit 2 to 1. When set, this error bit is latched and remains Active until an Error Reset command has been issued.

Because errors apply to specific characters, it is necessary that error information accompany the data to which it refers. The ESCC channel implements this process with an error FIFO in parallel with the data FIFO. The three error conditions that the receiver checks for in Asynchronous mode are:

- Framing errors –when a character's Stop bit is 0
- Parity errors–the parity bit of a character disagrees with the sense programmed in WR4
- Overrun errors–when the Receive FIFO overflows

If interrupts are not used to transfer data, the Parity Error, Framing Error, and Overrun Error bits in RR1 must be checked before the data is removed from the receive data FIFO, because reading data pops the error information stored in the Error FIFO.

The ESCC channel can be programmed to accept a receive clock that is 1, 16, 32, or 64 times the data rate. These settings are selected by WR4 Bits 7..6. The 1X mode is used when bit synchronization external to the received clock is present (that is, a clock recovery circuit, or the bit clock signal from the sender). The 1X mode is the only mode in which a data encoding method other than NRZ may be used.

The ESCC channel recognizes a Break condition upon seeing a null character (all 0s) plus a framing error. When recognizing this sequence, the Break bit in RR0 is set and remains set until a 1 is received. At this point, the break condition is no longer present. At the termination of a break, the receive data FIFO contains a single null character, which must be read and discarded. The framing error bit is not set for this character, but if odd parity has been selected, the Parity Error bit is set.

Received characters are assembled, checked for errors, and moved to the receive data FIFO. The user can program the ESCC to generate an interrupt to the CPU or to request DMA transfer when data is received.

When the ESCC channel generates the receive character available interrupt or DMA request on Receive (if enabled) and depends on WR7′ Bit 3. If this bit is 0, the receive interrupt or DMA request is generated when there is at least one character in the FIFO. If WR7′ Bit 3 is 1, the receive interrupt or DMA request is generated when there are four bytes available in the Receive FIFO. RR0 Bit 0 (the RCA bit) is set if there is at least one byte available, regardless of the status of WR7′ Bit 3.

Asynchronous Initialization

The initialization sequence for Asynchronous mode is described in Table 17. All of the ESCC's channel registers must be re-initialized after a channel or hardware reset. First, WR4 selects the mode, then WR3 and WR5 select the various options. At this point, the other registers are initialized as necessary. When all of this is complete, the receiver may be enabled by setting WR3 Bit 0 to 1.

Table 17. Initialization Sequence Asynchronous Mode

Register	Bit No.	Description
WR9	7..6	Hardware or channel reset
WR4	3..2	Select Async Mode and the number of stop bits
	1..0	Select parity
	7..6	Select clock mode
WR3	7..6	Select number of receive bits per character
	5	Select Auto Enables Mode
WR5	6..5	Select number of bits/char for transmitter
	1	Select modem control (RTS)

Character-Oriented Synchronous Modes

The Z80185/Z80195's ESCC channel supports two character-oriented synchronous modes called Monosync and Bisync. See ZiLOG's *ESCC User Manual* if you are interested in using one of these modes. The Z80185/Z80195's ESCC channel does not support the External Sync mode described in that manual, because it does not have a Sync pin.

Bit-Oriented Synchronous (SDLC/HDLC) Mode

Synchronous Data Link Control mode (SDLC) uses synchronization characters as in character oriented synchronous modes but is a bit-oriented protocol instead of a byte-oriented protocol. This protocol is also called High level Data Link Control (HDLC). The SDLC protocol uses the technique of 0 insertion to make all data transparent. All references to SDLC in this manual apply to both SDLC and HDLC.

The basic format for SDLC is the frame (Figure 78). A Frame is marked at the beginning and end by a unique flag pattern. The flags typically enclose address, control, information, and frame check fields. There are many different versions of the SDLC protocol and many do not use all of the fields.

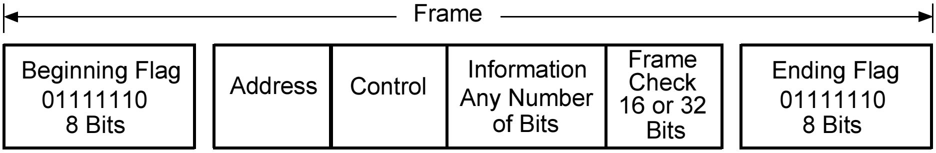


Figure 78. Typical SDLC Message Format

Frames of information are enclosed by a unique bit pattern called a flag. The flag character has a bit pattern of 01111110 (7EH). This sequence of six consecutive 1s is unique because all data between the opening and closing flags is prohibited from having more than five consecutive 1s.

The transmitter guarantees this by watching the transmit data stream and inserting a 0 after five consecutive 1s, regardless of character boundaries. In turn, the receiver searches the receive data stream for five consecutive 1s and deletes the next bit if it is a 0. Because the SDLC mode does not use characters of defined length, but rather works on a bit-by-bit basis, the 01111110 flag can be recognized at any time. Inserted and removed 0s are not included in the CRC calculation. The zero insertion and deletion is completely transparent to the user.

Because of the 0 insertion/deletion, the actual bit length of a frame on the transmission line may be longer than the number of bits presented to the HDLC transmitter plus a CRC if one is generated.

The ending flag indicates to the receiving station that the 16 or 32 bits just received constitute the frame check (CRC; also referred to as FCS or Frame Check Sequence). The ending flag can be followed by another flag, or an idle. In some SDLC protocols, a single flag may simultaneously be the ending flag of the first frame and the beginning flag of the next frame.

The ESCC channel can be programmed to check an 8-bit address field that directly follows the beginning Flag, for a programmed value. Longer addresses must be checked by software.

The control field of a SDLC frame is typically 8 bits, but can be any length. The control field is transparent to the ESCC channel and is treated as normal data by the transmit and receive logic.

The information field is not restricted in format or content and can be of any reasonable length (including 0). The Residue Code feature provides a mechanism to report any number of bits at the end of the frame that do not make up a full character. This feature allows for the data field to be an arbitrary number of bits long.

The frame check field is used to detect errors in the received address, control and information fields. The method used to verify that the received data matches the transmitted data, is called a Cyclic Redundancy Check (CRC). The ESCC channel can select from among several CRC polynomials, but in SDLC mode only the 16 bit CRC-CCITT polynomial and the 32-bit CCITT polynomials are used.

There is one other unique bit pattern in SDLC mode besides the flag sequence, the Abort or End of Poll (EOP) sequence. An Abort is a sequence of seven to thirteen consecutive 1s and is used to signal the premature termination of a frame. The EOP is a 0 followed by seven 1s, and is used in loop applications as a signal to a secondary station that it may begin transmission.

SDLC mode is selected by setting Bits 5..2 of WR4 to 1000. In addition, the flag sequence must be written to WR7. Additional control bits for SDLC mode are located in WR10 and WR7′.

SDLC Transmit

In SDLC mode, the transmitter moves characters from the four-byte transmit FIFO to the Transmit Shift register, through the zero inserter and out to the TxD pin. The insertion of 0s is completely transparent to the user. Zero insertion is done to all transmitted characters except the Flag Abort and IDLE sequences.

An SDLC frame must have the 01111110 (7EH) flag sequence transmitted before the data. This process is accomplished by programming WR7 with 7EH as part of the device initialization, enabling the transmitter, and then writing data. If the ESCC channel is programmed to send Mark Idle (WR10 Bit 3 is 1), special consideration must be taken to transmit the opening flag. Ordinarily, it is necessary to reset the WR10 Bit 3 to flag Idle, wait the length of time required to transmit eight bits, and then write data to the transmitter. It is necessary to wait the length of time required to transmit eight bits before writing data because 1s are transmitted eight at a time and all eight must leave the Transmit Shift register before a flag is loaded.

The number of bits per transmitted character is controlled by WR5 Bits 6..5 of and the way the data is formatted within the transmit buffer. The bits in WR5 allow the option of five, six, seven, or eight bits per character. In all cases, the data must be right justified, with the unused bits being ignored, except in the case of five bits per character. When five bits per character are selected, the data may be formatted before being written to the transmit buffer. This procedure allows transmission of one to five bits per character, as described in Table 16.

An additional bit, carrying parity information, is automatically appended to every transmitted character by setting WR4 Bit 0 to 1. This bit is sent in addition to the number of bits specified in WR4 or by the data format. The parity sense is selected by WR4 Bit 1.

The ESCC channel transmits address and control fields as normal data and does not automatically send any address or control information. The value programmed into WR6 can be used by the receiver to compare the one address byte in the received frame (if address search mode is enabled). WR6 is not used by the transmitter. The address is written to the transmitter as the first bytes of data in the frame.

The information field can be any number of characters long. The transmitter can interrupt the CPU when the entry location of the Transmit FIFO is empty or when the Transmit FIFO is completely empty. Or, the ESCC channel can issue a DMA request when the entry location of the Transmit FIFO is empty or when the Transmit FIFO is completely empty. RR0 Bit 2 (TBE) is set when the entry location is empty.

The character length may be changed on the fly, but the desired length must be selected before the character is loaded into the Transmit Shift register from the transmit data FIFO. The easiest way to ensure this is to write to WR5 to change the character length before writing the data to the transmit FIFO. Note that although the frames can be any length, most protocols specify the address/control field as 8-bit fields. The receiver checks the address field as 8-bit, if address search mode is enabled.

Only CRC-CCITT polynomials are used in SDLC mode. This is selected by setting WR5 Bit 2 to 0. This bit controls the selection for both the transmitter and receiver. The initial state of the generator and checker is controlled by WR10 Bit 7. This bit must be 1 so that both the generator and checker have an initial value of all 1s.

Software can preset the CRC generator by issuing the Reset Tx CRC command, which is encoded in WR0 Bits 7..6. This command must be issued while the transmitter is enabled and idling. If the CRC is used, the transmit CRC generator is enabled by setting WR5 Bit 0 to 1. The CRC is calculated on all characters between opening and closing flags, so this bit is 1 at initialization and never changed. When WR7′ Bit 1 is 1, the Auto EOM Latch reset mode is Enabled, and the CRC generator is reset automatically.

Enabling the CRC generator is not sufficient to control the transmission of the CRC. This function is controlled by the Tx Underrun/EOM bit, which may be reset by the software and set by the ESCC. When WR7′ Bit 1 is 1, the Auto EOM Reset mode is Enabled, and the Tx Underrun/EOM Latch is reset automatically.

A frame is terminated with a CRC and a flag, but the ESCC channel may be programmed to send an Abort and a flag in place of the CRC. This option allows the ESCC to abort a frame transmission in progress if the transmitter is allowed to underrun. This condition is controlled by WR10 Bit 2 (Abort/Flag on Underrun). When this bit is 1, the transmitter sends an Abort in place of the CRC when an underrun occurs. The frame is terminated with a CRC and a flag if this bit is 0.

The ESCC channel can also send an Abort because of a command from the processor. When the Send Abort command is issued in WR0, the transmitter sends eight consecutive 1s and then goes Idle. Because up to five consecutive 1s may be sent prior to the command being issued, a Send Abort causes a sequence of from eight to thirteen 1s to be transmitted. The Send Abort command also clears the transmit data FIFO.

When transmitting in SDLC mode, all data passes through the 0 inserter, which adds a delay the length of time required to transmit five bits between the Transmit Shift register and the TxD pin.

When the transmitter underruns (both the Transmit FIFO and the Transmit Shift register are empty), the state of the Tx Underrun/EOM bit determines the action taken by the ESCC channel.

If the Tx Underrun/EOM bit is `1` when the underrun occurs, the transmitter sends flags without sending the CRC. If this bit is `0` when the underrun occurs, the transmitter sends either the accumulated CRC followed by flags, or an abort followed by flags, depending on the state of the Abort/Flag on Underrun bit, `WR10` Bit 1. Table 0-1 summarizes the ESCC actions taken on Tx underrun.

The Reset Tx Underrun/EOM Latch command is encoded in `WR0` Bits 7..6.

Table 0-1. ESCC Action Taken on Tx Underrun

Tx Underrun/EOM Latch Bit	Abort/ Flag	Action Taken by ESCC on Tx Underrun
0	0	Sends CRC followed by flag
0	1	Sends Abort followed by flag
1	X	Sends flag

The ESCC channel sets the Tx Underrun/EOM latch when a CRC or abort is loaded into the shift register for transmission. This event can cause an interrupt, and the status of the Tx Underrun/EOM latch can be read in `RR0`.

The Tx Underrun/EOM latch can be reset by the software via a command encoded in `WR0` Bits 7..6. The reset can also be accomplished by setting `WR7′` Bit 1 for Auto EOM Reset mode enabled (`1`). For transmission of the CRC at the end of a frame, this command must be issued after the first character is written to the ESCC channel but before the transmitter underruns after the last character written to the ESCC channel. The command is issued immediately after the first character is written to the ESCC channel so that the Abort or CRC is sent if an underrun occurs. `WR10` Bit 2 (Abort/Flag on Underrun) is `1` at the same time as the Tx

Underrun/EOM bit is reset so that an `Abort` command is sent if the transmitter underruns. The bit is then `0` near the end of the frame to allow transmission of the CRC.

In this paragraph the term Completely Sent is defined as shifted out of the Transmit Shift register, not shifted out of the `0` inserter, which is an additional delay for the length of time required to transmit five bits. In SDLC mode, if the transmitter is disabled during transmission of a character, that character is Completely Sent. This condition applies to both data and flags. However, if the transmitter is disabled during the transmission of the CRC, the 16-bit transmission is completed. The remaining bits are from the Flag register rather than the remainder of the CRC, resulting in a CRC error.

The initialization sequence for the transmitter in SDLC mode is as follows:

1. `WR4` selects the mode.
2. `WR10` modifies it if necessary.
3. `WR7` programs the flag.
4. `WR3` and `WR5` select various options.

The other registers can now be initialized as necessary. When complete, the transmitter is enabled by setting `WR5` Bit 3 to `1`. The CRC generator may now be initialized by issuing the Reset Tx CRC Generator command in `WR0` (Bits 7..6 are `10`).

Modem Control Signals for SDLC Transmission

There are two modem control signals associated with the transmitter. The $\overline{\text{RTS}}$ pin is an output that carries the inverted state of the RTS bit (`WR5` Bit 1). The $\overline{\text{CTS}}$ pin is an input to the CTS bit in `RR0` (Bit 5). However, if Auto Enables mode is selected, this pin becomes an enable for the transmitter. If Auto Enables is On and the $\overline{\text{CTS}}$ pin is High, the transmitter is disabled. The transmitter is enabled if the $\overline{\text{CTS}}$ pin is Low.

HDLC TX Enhancements

The Transmit FIFO: The ESCC has a four-byte-deep Transmit FIFO. There are two modes of operation for the transmit interrupt and DMA request, one of which is selected by WR7′ Bit 5.

The ESCC sets WR7′ Bit 5 to 1 during a hardware or software reset. In this mode, the ESCC generates the Transmit Buffer Empty interrupt or DMA Transmit Request when the Transmit FIFO is completely empty. Interrupt-driven systems can maximize efficiency by writing four bytes for each entry into the Transmit Interrupt Service Routine (TISR), filling the Transmit FIFO without having to check any status bits. DMA-driven systems can use this mode to reassert the DMA request for more data after the first byte written to the FIFO is loaded to the Transmit Shift register. Consequently, any subsequent reassertion allows the DMA sufficient time to detect the High-to-Low edge.

If WR7′ Bit 5 is 0, the transmit buffer empty interrupt or DMA request is generated when the entry location of the FIFO is empty. Therefore, if more than one byte is required to fill the FIFO, the ESCC channel generates DMA requests until the FIFO is filled. The transmit DMA request goes Inactive after each data transfer, then goes Active again and, consequently, generates a High-to-Low edge for each byte. Edge triggered DMA must be enabled before the transmit DMA function is enabled in the ESCC to guarantee that the ESCC does not generate the edge before the DMA is ready.

CRC Takes Priority Over Data: After an Underrun/EOM (End of message) interrupt occurs, the ESCC accepts the data for the next packet without collapsing the packet. If data is written before the Tx interrupt, but after the transmitter has loaded the closing Flag into the Tx shift register, RR0 Bit 2 (TBE) is *not* set; even if the 2nd TXIP is guaranteed to set when the flag/sync pattern is loaded into the Transmit Shift register. For the detailed timing on this, refer to Figure 17 and Figure 18. It is not necessary to wait for the 2nd TXIP bit to set before writing data for the next packet, reducing overhead.

Auto EOM Reset (WR7' **Bit 1**): As described previously, the Tx Underrun/EOM Latch must be reset before the Transmit Shift register completes shifting out the last character of a frame, but after the first character has been written. The CPU issues the Reset Tx Underrun/EOM Latch command. The other method is by setting `WR7′` Bit 1, enabling the Automatic EOM Latch Reset feature. Setting this bit to `1` eliminates the need for the CPU command. In this mode, the CRC generator is automatically reset at the start of every packet. It is not necessary to reset the CRC generator prior to writing data into the ESCC. This feature is particularly valuable to a DMA-driven system wherein issuing CPU commands while the DMA is transferring data is difficult. Also, this feature is very useful if the data rate is very high and the CPU may not be able to issue the command at the correct time.

Auto Tx Flag (WR7' **Bit 0):** During idle time between frames, the transmitter sends continuous flags, but the ESCC can send Mark Idle under program control. By setting the Mark/Flag idle bit (`WR10` Bit 3) to `1`, the transmitter sends continuous `1`s in place of the Idle flags. The closing flag is always transmitted even when this mode is selected. If `WR7'` Bit 0 is `1`, an opening flag is transmitted automatically and it is not necessary for the CPU to turn the Mark Idle feature On and Off between frames.

Note: When this mode in not in effect (`WR7'` Bit 0 is `0`), the Mark/Flag Idle bit is cleared to `0`, allowing a flag to be transmitted before data is written to the transmit buffer. Exercise care when performing this procedure because the continuous `1`s are transmitted eight at a time and all eight must leave the Transmit Shift register. This procedure allows a flag to be loaded into the Transmit Shift register before the first data is written to the Transmit FIFO.

Auto RTS Deactivation (WR7' Bit 2): Some applications require toggling the modem signal to indicate the end of the packet.

If this feature is enabled by setting `WR7′` Bit 2, and when `WR5` Bit 1 is reset during the transmission of a SDLC frame, the deassertion of the $\overline{\text{RTS}}$ pin is delayed until the last bit of the closing flag clears the TxD pin. The $\overline{\text{RTS}}$ pin is de-asserted after the rising edge of the transmit clock cycle on which the last bit of the closing flag is transmitted. This action implies that the ESCC is programmed for Flag on Underrun (`WR10` Bit 2 is `1`) for the $\overline{\text{RTS}}$ pin to de-assert at the end of the frame. (Otherwise, the deassertion occurs when the next flag is transmitted). This feature works independently of the programmed Transmitter Idle state. In Synchronous modes other than SDLC, the $\overline{\text{RTS}}$ pin immediately follows the state programmed into `WR5` Bit 1. If the $\overline{\text{RTS}}$ pin is connected to $\overline{\text{CTS}}$ or $\overline{\text{DCD}}$, it can be used to generate an external status interrupt when a frame is completely transmitted.

NRZI Forced High After Closing Flag: With SDLC, NRZI and Mark Idle, the TxD pin is automatically forced High on the falling edge of the TxC of the last bit of the closing flag, and then the transmitter goes to the Mark Idle state.

There are several different ways for a transmitter to go into the Idle state. In each of the following cases, the TxD pin is forced High when the Mark Idle condition is reached:

- Data, CRC (2 bytes), Flag and Idle
- Data, Flag and Idle
- Data, Abort (on underrun) and Idle
- Data, Abort (by command) and Idle
- Idle, Flag and command to Mark Idle.

The Force High feature is disabled when the Mark Idle bit is `0` (programmed as Mark Idle). This feature is used in combination with the automatic SDLC opening flag transmission feature, `WR7′` Bit 0 is `1`, to assure that data packets are properly formatted. When these features are used together, it is not necessary for the CPU to issue any commands after sending a closing flag in combination with NRZI data encoding. If `WR7′`

Bit 0 is reset (0), software must reset the Mark Idle bit (WR10 Bit 3) to enable flag transmission before a SDLC packet is transmitted.

SDLC Receive

The receiver always searches the receive data stream for flag characters in SDLC mode. Ordinarily, the receiver transfers all received data between flags to the receive data FIFO. However, if the receiver is in Hunt mode, no data is received. The receiver is in Hunt mode when first enabled, or the receiver is placed in Hunt mode by the software issuing the Enter Hunt Mode command in WR3. This bit (Bit 4) is a command, and writing a 0 to it has no effect. The Hunt status of the receiver is reported by the Sync/Hunt bit in RR0 (Bit 4).

Sync/Hunt is one of the possible sources of external/status interrupts, with both transitions causing an interrupt. This condition is true even if the Sync/Hunt bit is set as a result of the software issuing the Enter Hunt mode command.

The receiver automatically enters Hunt mode if an Abort command is received. The receiver always searches the receive data stream for flags, and automatically enters Hunt Mode when an Abort command is received.

The first byte in an SDLC frame may be the address of the secondary station for which the frame is intended, or part of such an address. The receiver provides two options for handling this address.

- If the Address Search Mode bit (WR3 Bit 2) is 0, the address recognition logic is disabled and all received frames are transferred to the receive data FIFO. In this mode the software must perform any address recognition.
- If the Address Search Mode bit is 1, only those frames whose address matches the address programmed in WR6 or the global address (all 1s) are transferred to the receive data FIFO.

The address comparison is across all eight bits of WR6 if the Sync Character Load inhibit bit (WR3 Bit 1) in is 0. The comparison may be modified so that only the four most significant bits of WR6 match the received address. This mode is selected by setting the Sync Character Load Inhibit bit to 1. In this mode, however, the address field is still eight bits wide. The address field is transferred to the receive data FIFO in the same manner as data. It is not treated differently than data.

The number of bits per character is controlled by WR3 Bits 7..6. Five, six, seven, or eight bits per character may be selected using these two bits. The data is right-justified in each byte. The ESCC takes a snapshot of the receive data stream at the appropriate times, so the unused bits in the receive buffer are the bits following the character.

An additional parity bit is selected by setting WR4 Bit 6 to 1. This action also enables parity in the transmitter. The parity sense is selected by WR4 Bit 1. Parity is not used in most SDLC protocols.

The character length can be changed at any time before the new number of bits have been assembled by the receiver.

Most bit-oriented protocols allow an arbitrary number of bits between opening and closing flags. The ESCC channel allows for this condition by providing three bits of Residue Code in RR1. These bits indicate which bits in the last three bytes transferred from the receive data FIFO by the processor are actually valid data bits (and not part of the frame check sequence or CRC). Table 18 gives the definitions of the different codes for the four character length options. The valid data bits are right-justified, that is, if the number of valid bits given by the table is less than the character length, then the bits that are valid are the right-most or least significant bits. The Residue Code is only valid at the time when the End of Frame bit in RR1 (Bit 7) is 1.

Table 18. Residue Codes

Residue Code			Bits in Last Byte				Bits in Second Last Byte				Bits in Third Last Byte			
0	1	2	8B/C	7B/C	6B/C	5B/C	8B/C	7B/C	6B/C	5B/C	8B/C	7B/C	6B/C	5B/C
0	0	1	0	0	0	0	3	1	0	0	8	7	5	2
0	1	0	0	0	0	0	4	2	0	0	8	7	6	3
0	1	1	0	0	0	0	5	3	1	0	8	7	6	4
1	0	0	0	0	0	0	6	4	2	0	8	7	6	5
1	0	1	0	0	0	0	7	5	3	1	8	7	6	5
1	1	0	0	0	0		8	6	4		8	7	6	
1	1	1	1	0			8	7			8	7		
0	0	0	2				8				8			

As indicated in the table, these bits allow the processor to determine those bits in the information (and not CRC) field. This information enables transparent retransmission of the received frame. The Residue Code bits do not go through a FIFO; they change in RR1 when the last character of the frame is loaded into the receive data FIFO. If there are any characters already in the receive data FIFO, the Residue Code is updated before they are read by the processor.

Either a 16-bit or 32-bit CRC-CCITT polynomial can be used for CRC calculations in SDLC mode; the generator and checker can be preset to all 1s. The CRC-CCITT polynomial is selected by setting WR5 Bit 2 to 0. WR10 Bit 7 controls the preset value. This bit must be 1 so the generator and checker are preset to 1s.

Because the CRC is inverted before transmission, the receiver checks the CRC result for the value 0001110100001111. The ESCC channel presets the CRC checker whenever the receiver is in Hunt mode or whenever a flag is received, so a CRC reset command is not necessary. However, the CRC checker must be preset by issuing the Reset CRC Checker command in WR0.

The CRC checker is automatically enabled for all data between the opening and closing flags in SDLC mode, and the Rx CRC Enable bit (WR3 Bit 3) is ignored. The result of the CRC calculation for the entire frame is valid in RR1 only when accompanied by the End of Frame bit set in RR1 (Bit 7). At all other times, software ignores the CRC Error bit in RR1 (Bit 6).

On the ESCC channel, an enhancement has been made allowing the 2nd byte of the CRC to be received completely. This feature is useful when the application requires the 2nd CRC byte as data.

A frame is terminated by a closing flag. When the ESCC channel recognizes this flag:

1. The contents of the Receive Shift register are transferred to the receive data FIFO.
2. The Residue Code is latched, the CRC Error bit is latched in the status FIFO, and the End of Frame bit is set in the receive status FIFO.

The End of Frame bit, upon reaching the exit location of the FIFO, causes a Special Receive Condition. The processor may then read RR1 to determine the result of the CRC calculation as well as the Residue Code. If either the Rx Interrupt on Special Condition Only or the Rx Interrupt on First Character or Special Condition modes are selected, the processor must issue an Error Reset command in WR0 to unlock the Receive FIFO.

In addition to searching the data stream for flags, the receiver also watches for seven consecutive 1s, which is the Abort condition. The presence of seven consecutive 1s is reported in the Break/Abort bit in RR0 (Bit 7). This condition is one of the possible external/status interrupts, so transitions of this status may be programmed to cause interrupts. At receipt of an Abort command the receiver is forced into Hunt mode where it looks for flags.

The Hunt status is also a possible external/status condition whose transition may be programmed to cause an interrupt. The transitions of these two bits occur very close together, but either one or two external/

status interrupts may result. The Abort condition is terminated when a 0 is received, such as the leading 0 of a subsequent flag. The receiver does not leave Hunt mode until a flag has been received, so two discrete external/status conditions occur at the end of an Abort. An Abort command received in the middle of a frame terminates the frame reception, but not in an orderly manner because the character being assembled is lost.

Two modem control signals associated with the receiver are available in SDLC mode:

- The $\overline{\text{DTR}}$ pin carries an inverted state of the DTR bit (WR5 Bit 7)
- The $\overline{\text{DCD}}$ pin is ordinarily a simple input to the DCD bit in RR0 (Bit 3)

However, if the Auto Enables mode is selected by setting WR3 Bit 5 to 1, this pin becomes an enable for the receiver. That is, if Auto Enables is On and the $\overline{\text{DCD}}$ pin is High, the receiver is disabled. While the $\overline{\text{DCD}}$ pin is Low, the receiver is enabled.

SDLC Initialization. The initialization sequence for SDLC mode is the following:

- Write WR4 to select SDLC mode
- Write WR3 and WR5 to select the various options
- Write WR7 to program a flag
- Write WR6 for the receive address

At this point the other registers are initialized as necessary. When complete, the receiver is enabled by setting WR3 Bit 0 to 1. Table 19 summarizes the SDLC Initialization process.

Table 19. Initializing in SDLC Mode

	Bit Number								
Reg	7	6	5	4	3	2	1	0	Description
WR4	0	0	1	0	0	0	0	0	Select X1 clock, SDLC mode, enable Sync mode

Notes:

1. The receiver searches for synchronization when it is in Hunt mode. In this mode, the receiver is Idle except for searching the data stream for a flag match.
2. When the receiver detects a flag match it achieves synchronization and interprets the following non-flag byte as the address field.
3. The SYNC/HUNT bit in RR0 reports the Hunt Status, and an interrupt is generated at transitions between the Hunt state and the Sync state.

Table 19. Initializing in SDLC Mode

	Bit Number								
Reg	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**	**Description**
WR3	r	x	0	1	1	1	0	0	*rx* is the number of Rx bits/char, no Auto Enable, enter Hunt, Enable Rx CRC
WR5	d	t	x	0	0	0	r	1	*d* is the inverse of DTR pin, *tx* is the number of Tx bits/char, use SDLC CRC, *r* is the inverse state of $\overline{\text{RTS}}$ pin, CRC enable
WR7	0	1	1	1	1	1	1	0	SDLC Flag
WR6	x	x	x	x	x	x	x	x	Receive secondary address
WR15	x	x	x	x	x	x	x	1	Enable access to WR7′
WR7’	0	1	1	d	1	r	1	1	Enable extended read, Tx INT on FIFO empty, *d* is Request Timing mode, Rx INT on 4 char, *r* is RTS deactivation, auto EOM reset, auto flag tx
WR10	1	d	e	0	i	a	0	0	CRC preset to 1s, *de* is decoding, *i* is idle line, *a* is Abort/CRC
WR3	r	x	0	1	1	1	0	1	Enable Receiver
WR5	d	t	x	0	1	0	r	1	Enable Transmitter
WR0	1	0	0	0	0	0	0	0	Reset CRC generator

Notes:

1. The receiver searches for synchronization when it is in Hunt mode. In this mode, the receiver is Idle except for searching the data stream for a flag match.
2. When the receiver detects a flag match it achieves synchronization and interprets the following non-flag byte as the address field.
3. The SYNC/HUNT bit in RR0 reports the Hunt Status, and an interrupt is generated at transitions between the Hunt state and the Sync state.

SDLC FRAME STATUS FIFO

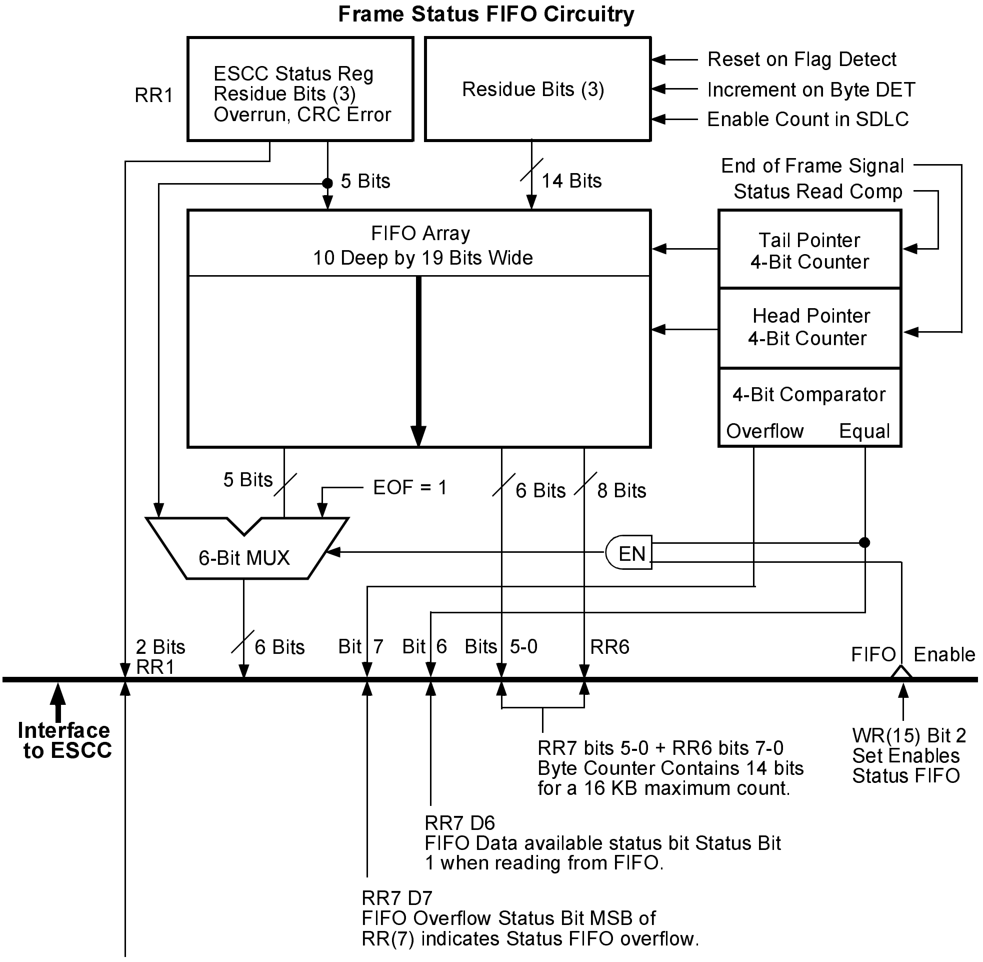
The ability to receive high-speed back-to-back SDLC frames is maximized by a 10-bit deep by 19-bit wide status FIFO. When enabled (`WR15` Bit 2 is `1`), the FIFO provides a DMA the ability to continue to transfer data into memory so that the CPU can examine the frames later. For each SDLC frame, a 14-bit byte count and five status/error bits are stored. The byte count and status bits are accessed through Read Registers 6 and 7. Read Registers 6 and 7 are only accessible when the SDLC FIFO is enabled. The 10x19 status FIFO is separate from the 8-byte Receive Data FIFO.

When this enhancement is enabled, the status in Read Register 1 (`RR1`) and the byte count for each SDLC frame is stored in the 10 x 19 bit status FIFO. This procedure enables the DMA controller to transfer the next frame into memory while the CPU verifies the message was properly received.

In summary, data is received, assembled, and loaded into the 8-byte FIFO before being transferred to memory by the DMA controller. When a flag is received at the end of an SDLC frame, the frame byte count from the 14-bit counter and five status bits are loaded into the status FIFO for verification by the CPU. The CRC checker is automatically reset in preparation for the next frame. Because the byte count and status are saved for each frame, the message integrity can be verified at a later time. Status information for up to 10 frames can be stored before a status FIFO overrun occurs.

If a frame is terminated with an Abort command, the byte count is loaded to the status FIFO and the counter reset for the next frame.

For a better understanding of details of the FIFO operation, refer to the block diagram in Figure 16.



In SDLC Mode the following definitions apply:

– All Sent and Parity Error bits bypass the MUX and always come from the ESCC Status Register.

– EOF is 1 whenever reading from the FIFO.

Figure 16. SDLC Frame Status FIFO

Enable/Disable. The frame status FIFO is enabled when WR15 Bit 2 is set and the receiver is in SDLC/HDLC mode. Otherwise, the status register contents bypass the FIFO and go directly to the bus interface (the FIFO pointer logic is reset either when disabled or via a channel or Power-On Reset). The FIFO mode is disabled on power-up (WR15 Bit 2 is 0 on reset). The status of the FIFO Enable signal can be obtained by reading RR15 Bit 2. If the FIFO is enabled, the bit is 1; otherwise, it is reset (0).

Read Operation. When WR15 Bit 2 is set and the FIFO is not empty, the next read from any of status register RR1 or the additional registers RR7 and RR6 is from the FIFO. Reading status register RR1 causes one location of the FIFO to be emptied, so RR1 is read after reading the byte count. Before the FIFO underflows, it is disabled. In this case, the multiplexer is switched to enable status to read directly from the status register, and reads from RR7 and RR6 are undefined. RR7 Bit 6 (FIFO Data Available) is used to determine if status data is coming from the FIFO or directly from the status register, because it is 1 whenever the FIFO is not empty.

Not all status bits are stored in the FIFO. The All Sent, Parity, and EOF bits bypass the FIFO. The status bits sent through the FIFO are Residue Bits (3), Overrun, and CRC Error.

The sequence for proper operation of the byte count and FIFO logic is to read the register in the following order: RR7, RR6, and RR1 (reading RR6 is optional). Additional logic prevents the FIFO from being desynchronized by multiple reads from RR1. The read from RR7 latches the FIFO empty/full status bit (bit 6) and steers the status multiplexer to read from the receiver instead of the status FIFO (because the status FIFO is empty). The read from RR1 allows an entry to be read from the FIFO (if the FIFO was empty, logic was added to prevent a FIFO underflow condition).

Write Operation. When the end of an SDLC frame (EOF) has been received and the FIFO is enabled, the contents of the status and byte-count registers are loaded into the FIFO. The EOF signal is used to advance the FIFO. If the FIFO overflows, RR7 Bit 7 (FIFO Overflow) is 1 to indicate the overflow. This bit and the FIFO control logic is reset by disabling and re-enabling the FIFO control bit (WR15 Bit 2). For details of FIFO control timing during an SDLC frame, refer to Figure 17.

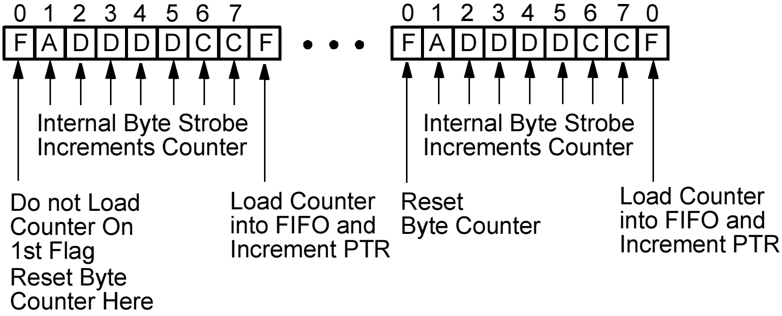

0 1 2 3 4 5 6 7
F A D D D D C C F
• • •
0 1 2 3 4 5 6 7 0
F A D D D D C C F
Internal Byte Strobe Increments Counter
Internal Byte Strobe Increments Counter
Do not Load Counter On 1st Flag
Reset Byte Counter Here
Load Counter into FIFO and Increment PTR
Reset Byte Counter
Load Counter into FIFO and Increment PTR

Figure 17. SDLC Byte Counting Detail

SDLC Status FIFO Anti-Lock Feature. When the Frame Status FIFO is enabled and the ESCC is programmed for Special Receive Condition Only (WR1 Bits 4..3 are 11), the data FIFO is not locked when a character with End of Frame status is read. When a character with the EOF status reaches the top of the FIFO, an interrupt with a vector for receive data is generated. The command Reset Highest IUS must be issued at the end of the Interrupt Service Routine regardless of whether an interrupt acknowledge cycle had been executed (hardware or software). This process enables a DMA to complete a transfer of the received frame to memory and then inform the CPU that a frame has been completed without locking the FIFO. In the Receive Interrupt on Special Condition Only mode the interrupt vector for receive data is not used; it is used to indicate that the last byte of a frame has been read from the Receive FIFO. This process eliminates reading the frame status (CRC and other status is stored in the status FIFO with the frame byte count).

When a character with a Special Receive Condition other than EOF is received (receive overrun, or parity), a Special Receive Condition Interrupt is generated after the character is read from the FIFO. The Receive FIFO is locked until the Error Reset command is issued.

SDLC Loop Mode

The ESCC supports SDLC Loop mode in addition to normal SDLC. In an SDLC Loop, there is a primary controller that manages the message traffic flow on the loop and one or more secondary stations. In SDLC Loop mode, an ESCC channel operating in regular SDLC mode can act as the primary controller.

A secondary station in an SDLC Loop is always listening to the messages being sent around the loop, and passes these messages to the rest of the loop by retransmitting them with a one-bit-time delay.

The secondary station can place its own message on the loop only at specific times. The controller signals that secondary stations may transmit messages by sending a special character, called an End of Poll (EOP), around the loop. The EOP character is a 0 followed by seven 1s.

When a secondary station has a message to transmit and recognizes an EOP on the line, it changes the last binary 1 of the EOP to a 0 before transmission, turning the EOP into a flag pattern. The secondary station now places its message on the loop and terminates its message with an EOP. Any secondary stations further down the loop with messages to transmit can append their messages to the message of the first secondary station by the same process.

All secondary stations without messages to send merely echo the incoming messages.

SDLC Loop mode is quite similar to normal SDLC mode except that two additional control bits are used. Writing a 1 to the WR10 Bit 1 (Loop Mode) configures the ESCC for Loop mode. Writing a 1 to the Go Active on Poll bit (Bit 4) in the same register normally causes the ESCC to change the next EOP into a flag and then begin transmitting on loop. However, when the ESCC first goes On-Loop it uses the first EOP as a signal to insert the one-bit delay, and does not begin transmitting until it receives the second EOP. There are also two additional status bits in RR10, the On-Loop bit (Bit 1) and the Loop-Sending bit (Bit 4).

There are also restrictions as to when and how a secondary station physically becomes part of the loop.

A secondary station that has just powered up must monitor the loop, without the one bit-time delay, until it recognizes an EOP. When an EOP is recognized the one-bit-time delay is switched on. This does not disturb the loop because the line is marking idle between the time that the controller sends the EOP and the time that it receives the EOP back. The secondary station that has gone on-loop cannot place a message on the loop until the next time that an EOP is issued by the controller. A secondary station goes off loop in a similar manner. When given a command to go off-loop, the secondary station waits until the next EOP to remove the one-bit-time delay.

To operate the ESCC channel in SDLC Loop mode, first program it as for normal SDLC. Then select Loop mode by writing the appropriate control word in WR10.

The ESCC channel waits for the EOP so that it can go On-Loop. While waiting for the EOP, the ESCC ties TxD to RxD with only the internal gate delays in the signal path. When the first EOP is recognized by the ESCC, the Break/Abort/EOP bit is set in RR0, generating an External/Status interrupt (if so enabled). At the same time, the On-Loop bit in RR10 is 1 to indicate that the ESCC is indeed On-Loop, and a one-bit time delay is inserted in the TxD to the RxD path.

The ESCC is now On-Loop but cannot transmit a message until a flag and the next EOP are received. The requirement that a flag be received ensures that the ESCC cannot erroneously send messages until the controller ends the current polling sequence and starts another one.

If the software needs to transmit a message, it must set WR10 (Bit 4), the Go Active On Poll bit. If this bit is set when the EOP is detected, the ESCC channel changes the EOP to a flag and starts sending another flag. The EOP is reported in RR0 Bit 7, the Break/Abort/EOP bit and the CPU writes its data bytes to the Tx FIFO, just as in normal SDLC frame transmission. When the frame is complete and CRC has been sent, the transmitter closes with a flag and reverts to One Bit Delay mode. The last 0 of the flag, along with the marking line echoed from the RxD pin, form an EOP for secondary stations further down the loop.

The ESCC sets R10 Bit 4 to 1 to indicate that the ESCC channel is actually transmitting a message.

If the Go Active On Poll bit (Bit 4) is not set at the time the EOP passes by, the ESCC channel cannot send a message until a flag (terminating the current polling sequence) and another EOP are received.

If SDLC Loop mode is deselected, the ESCC channel is designed to exit from the loop gracefully. When the SDLC Loop mode is deselected by writing to WR10, the ESCC channel waits until the next polling cycle to remove the one-bit time delay.

If a polling cycle is in progress at the time the command is written, the ESCC channel finishes sending any message that it is transmitting, ends with an EOP, and disconnects TxD from RxD. If no message was in progress, the ESCC channel immediately disconnects TxD from RxD.

When the ESCC channel is not sending on the loop, exit from the loop by setting WR10 Bit 1, the Loop Mode bit to 0. At the same time, write the WR10 Bits 3..2, the Mark/Flag Idle and Abort/Flag on Underrun bits with the desired values. The ESCC channel reverts to normal SDLC operation as soon as an EOP is received, or immediately if the receiver is already in Hunt mode because of the receipt of an EOP.

To ensure loop operation after the ESCC channel goes off -loop, and until external relays take the ESCC channel completely out of the loop, the ESCC channel must be programmed for Mark Idle instead of Flag Idle. When the ESCC channel goes off-loop, the On-Loop bit is reset.

Note: With NRZI encoding, removing the stations from the loop (removing the one-bit time delay) may cause problems further down the loop because of extraneous transitions on the line. The ESCC channel avoids this problem by making transparent adjustments at the end of each frame it sends in response to an EOP. A response frame from the ESCC channel is terminated by a flag and EOP. Normally, the flag and the EOP share a `0`, but if sharing causes the RxD and TxD pins to have opposite polarity after the EOP, the ESCC channel adds another `0` between the flag and the EOP. This process causes an extra line transition so that RxD and TxD are identical after the EOP is sent. The extra `0` is completely transparent because the flag and the EOP no longer share a `0`. All that a loop exit needs, is the removal of the one-bit delay.

The ESCC channel allows the user the option of using NRZI in SDLC Loop mode by programming `WR10`. With NRZI encoding, the outputs of secondary stations in the loop are inverted from their inputs because of messages that they have transmitted.

SDLC Loop Initialization

The processor must program `WR4` first to select SDLC mode, and then `WR10` to select the CRC preset value and program the Mark/Flag Idle bit. The Go Active On Poll and Loop Mode bits in `WR10` (bits 4 and 1) must not be set to `1` yet. The flag is written in `WR7` and the various options are selected in `WR3` and `WR5`. At this point, the other registers are initialized as necessary (Table 20).

Table 20. SDLC Loop Mode Initialization

				Bit Number					
Reg	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**	**Description**
WR4	0	0	1	0	0	0	0	0	Select X1 clock, SDLC mode, enable sync mode *rx* is the number of Rx bits/char, no auto enable, enter Hunt, Enable Rx CRC, Address Search, No sync character load inhibit
WR3	r	x	0	1	1	1	0	0	
WR5	d	t	x	0	0	0	r	1	*d* is the inverse of DTR pin, *tx* is the number of Tx bits/char, use SDLC CRC, *r* is the inverse state of /RTS pin, CRC enable SDLC Flag
WR7	0	1	1	1	1	1	1	0	
WR6	x	x	x	x	x	x	x	x	Receiver secondary address
WR15	x	x	x	x	x	x	x	1	Enable access to new register Enable extended read, Tx INT on FIFO empty, *d* is the REQUEST timing mode, RxINT on 4 char, *r* is the RTS deactivation, auto EOM reset, Auto Flag Tx
WR7’	0	1	1	d	1	r	1	1	
WR10	c	d	e	1	i	0	1	0	Enable Loop Mode, Go Active On Poll, *c* is CRC preset, *de* is the data encoding method, *i* is the Idle line
WR3	r	x	0	1	1	1	0	1	Enable Receiver
WR5	d	t	x	0	1	0	r	1	Enable Transmitter
WR0	1	0	0	0	0	0	0	0	Reset CRC generator

`WR10` Bit 1 (the Loop Mode bit) is `1`. When complete, the transmitter is enabled by setting `WR5` Bit 3 of to `1`. Now that the transmitter is enabled, the CRC generator is initialized by issuing the Reset Tx CRC Generator command in `WR0` (Bits 7..6 are `10`). The receiver is enabled by setting the `WR10` Bit 4 (the Go Active On Poll bit) to `1`.

The ESCC channel goes On-Loop when seven consecutive 1s are received, and signals this condition by setting RR10 Bit 1 (the On-Loop bit). The seven consecutive 1s sets the Break/Abort and Sync/Hunt bits (Bits 7 and 4) in RR0 as well. When the ESCC is On-Loop, the Go Active On Poll bit (WR10 Bit 4) must be 0 until a message is to be transmitted on the loop.

To transmit a message on the loop, the Go Active On Poll bit must be 1. At this point, the processor may either write the first character to the transmit buffer and wait for a transmit buffer empty condition, or wait for the Break/Abort and Sync/Hunt bits (Bits 7 and 4) to be set in RR0 and the Loop Sending bit (Bit 4) to be set in RR10 before writing the first data to the transmitter.

The Go Active On Poll bit (WR10 Bit 4) must be 0 after the transition of the frame has begun. To go Off-Loop, the processor must set the Go Active On Poll bit to 0 and then wait for the Loop Sending bit (Bit 4) in RR10 to be 0.

At this point, the Loop Mode bit (bit 1) in WR10 is 0 to request an orderly exit from the loop. The ESCC exits SDLC Loop mode when seven consecutive 1s have been received; at the same time the Break/Abort and Hunt bits (Bits 7 and 4) in RR0 are 1, and the On Loop bit in RR10 is 0.

LocalTalk (AppleTalk) Mode

- To enter this mode, initialize the part as described for normal SDLC mode, with the following additional settings:
- WR4 Bits 5..2 are 1000 (SDLC)
- WR5 Bit 1 is 0 (RTS false)
- WR7′ Bit 2 is 1 (auto RTS deactivation)
- WR10 Bit 3 is 1 (Mark Idle)

Then, before sending the first frame, write a `1` to `WR5` Bit 4, which in other modes is the Send Break command bit. On this revision of the ESCC, this action places the Transmitter in a special LocalTalk mode, in which:

1. When software or a DMA channel writes the first data character of a new frame into the TxFIFO, the transmitter sends three Flags, asserting RTS during the first bit of the first Flag, then negating RTS for four bits to create a coding violation, then asserting RTS again for the balance of the Flags, before it sends the first character.
2. When a frame is over, the transmitter delays the TXIP that would otherwise be set as the last byte of the CRC was going out, until after it has sent the CRC, a closing Flag, and 16 bits of mark Idle.

These two procedures by the transmitter greatly ease the software burden required to transmit a LocalTalk (AppleTalk) frame.

REGISTER ADDRESSING

The ESCC channel responds to two addresses in the Z80185/Z80195's I/O space.

Address `E8H` is called ESCC Control, and is the address used by software for all accesses except reading and writing serial data.

Because the ESCC channel includes more than two registers, software must access most of its registers by first writing a register address to the ESCC Control address, and then reading or writing the same address. Most of the ESCC channel's registers are not symmetrical between reading and writing, and so ESCC Write Registers 0..15 and Read Registers 0..15 are discussed separately.

Address `E9H` is called ESCC Data. Software can read this address to obtain serially received data, and can write this address to provide data to be serially transmitted. The ESCC can be used with the Z80185/Z80195's DMA channels; if so, `00E9H` is the I/O address to use in programming the DMA channel(s).

Of the 16 Write registers and 16 Read registers in the ESCC channel, only Write Register 0 and Read Register 0 can be accessed at the ESCC Control address after a Reset. To enable software to read or write the other registers, `WR0` includes an *indirect register address* field.

Bits 7..6 and 5..3 of `WR0` enable software to write certain commands to the ESCC channel; in each field the all-`0` value is a *null command* which can be used when the purpose of writing `WR0` is to define a register address for a subsequent read or write operation.

Bit 3 of `WR0` is shared between command Bits 5..4 and register address Bit 2..0. Any of `WR1`..7 or `RR1`..7 can be selected for subsequent writing or reading, in the same `WR0` write with one of the commands `010`..`111` in Bits 5..3. But `WR8`..`15` and `RR8`..`15` can only be selected by writing their register number in Bits 3..0 of `WR0`, with `00` in Bits 5..4. Another way to describe this process is that the value `001` in Bits 5..3 is a Point High command that selects one of registers 8..15 for the next read or write.

When software writes a register address 1..15 to `WR0`, the next time it reads or writes the ESCC Command address, the ESCC channel writes or reads the register selected by the address in `WR0`. Thereafter, software clears the register address in `WR0` to `0`, so that the next read or write of the ESCC Command address again accesses `RR0` or `WR0`.

The ESCC channel's indirect register addressing represents a danger with interrupt-driven software. Whenever an interrupt can lead to code that accesses the ESCC channel, software must prevent such an interrupt between an output instruction (for example, OUT or OUT0) that writes a register address to `WR0`, and the following input or output instruction that reads or writes the register. Precede the first OUT or OUT0 with a `DI` instruction, and follow the subsequent IN or OUT with an `EI` instruction.

INTERRUPTS

The ESCC channel contains three sources of interrupts:

- The receiver
- The transmitter
- External/Status conditions

In addition, there are several conditions that may cause these interrupts. Figure 18 illustrates the different conditions for each interrupt source. Receive interrupts have the highest priority, and External/Status interrupts the lowest.

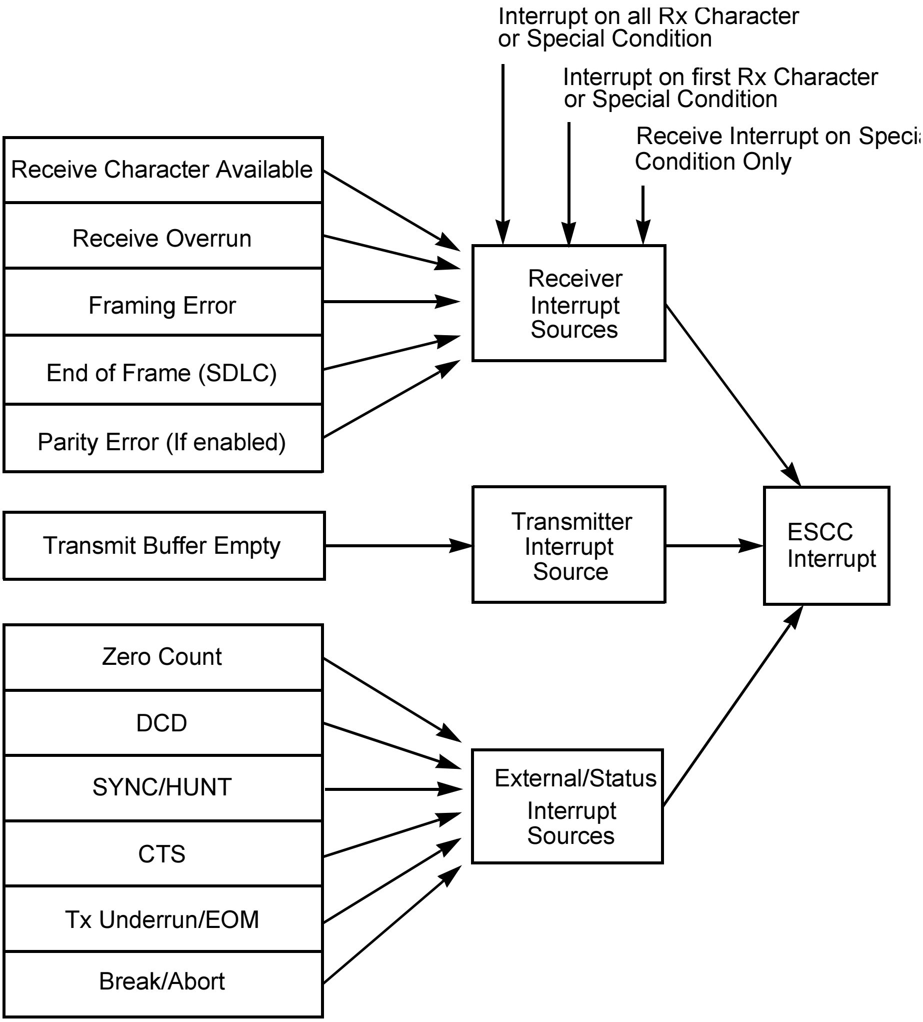


Figure 18. ESCC Interrupt Sources

The Receive Interrupt Request can be caused by a Receive Character Available or a Special Condition. When the Receive Character Available Interrupt is generated is dependent on `WR7'` Bit 3. If `WR7'` Bit 3 is `0`, the Receive Character Available Interrupt is generated when one character has been loaded into the FIFO and is ready to be read. If `WR7'` Bit 3 is `1`, the Receive Character Available Interrupt is generated when four bytes are available to be read in the receive data FIFO. The programmed value of `WR7'` Bit 3 also affects how receive DMA requests are generated.

Note: In SDLC mode, enabling the SDLC Status FIFO affects how Receive Interrupts are generated. If this feature is used, read the section on the SDLC Anti-Lock Feature.

The special conditions are Receive FIFO overrun, CRC/framing error, end of frame, and parity. If parity is included as a special condition, it is dependent on `WR1` Bit 2. The Special Condition status can be read from `RR1`.

The transmit interrupt request has only one source and is dependent on `WR7'` Bit 5. If `WR7'` Bit 5 is `0`, it is set when the transmit buffer becomes completely empty. If `WR7'` Bit 5 is `1`, the transmit interrupt is generated when the entry location of the FIFO is empty. In both cases the transmit interrupt is not set until after the first character is written to the ESCC channel.

The External/status interrupts have several sources which may be individually enabled in `WR15`. The sources are Zero Count, $\overline{\text{DCD}}$, Sync/Hunt, $\overline{\text{CTS}}$, transmitter underrun/EOM and Break/Abort.

Interrupt Control

In addition to the MIE bit that enables or disables all ESCC channel interrupts, each source of interrupt has three control/status bits associated with it. They are the Interrupt Enable (IE), Interrupt Pending (IP), and

Interrupt-Under-Service (IUS). Figure 19 illustrates the ESCC channel interrupt structure.

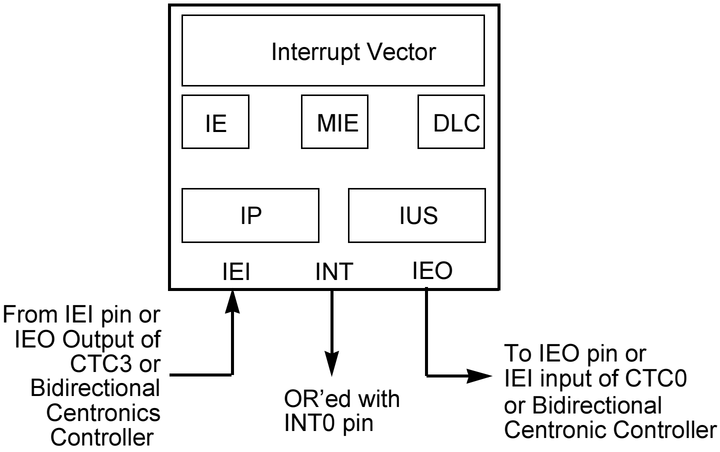

Interrupt Vector
IE
MIE
DLC
IP
IUS
IEI
INT
IEO
From IEI pin or IEO Output of CTC3 or Bidirectional Centronics Controller
OR'ed with INT0 pin
To IEO pin or IEI input of CTC0 or Bidirectional Centronic Controller

Figure 19. Peripheral Interrupt Structure

Figure 20 illustrates the internal daisy chaining in the ESCC channel. Lower priority devices on the daisy chain can be prevented from requesting interrupts using the Disable Lower Chain bit in WR9 Bit 2.

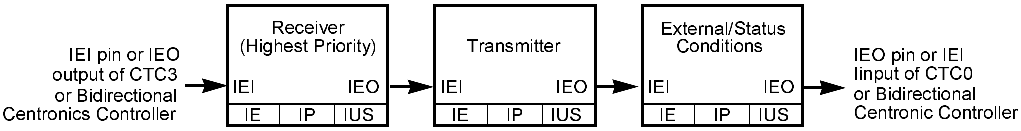

IEI pin or IEO output of CTC3 or Bidirectional Centronics Controller
Receiver (Highest Priority)
IEI
IEO
IE
IP
IUS
Transmitter
IEI
IEO
IE
IP
IUS
External/Status Conditions
IEI
IEO
IE
IP
IUS
IEO pin or IEI Iinput of CTC0 or Bidirectional Centronic Controller

Figure 20. Internal Priority Resolution

Master Interrupt Enable Bit

The Master Interrupt Enable (MIE) bit, WR9 Bit 3, must be set to enable the ESCC channel to generate interrupts. The MIE bit must be set after initializing the ESCC channel registers and enabling the individual interrupts. The ESCC requests an interrupt by asserting an internal signal, processed with an OR instruction, with the $\overline{\text{INT0}}$ pin and similar requests from the CTCs and Bidirectional Centronics Controller, when detecting that one of the enabled interrupt conditions has occurred.

Interrupt Enable Bit

The Interrupt Enable (IE) bits control interrupt requests from each interrupt source on the ESCC. If the IE bit is 1 for an interrupt source, that source may generate an interrupt request, providing all of the necessary conditions are met. If the IE bit is reset, no interrupt request can be generated by that source.

The transmit interrupt IE bit is WR1 Bit 1. The receive interrupt IE bits are WR1 Bits 4..3. The external status interrupts are individually enabled in WR15 with the master external status interrupt enable being WR1 Bit 0. The MIE bit, WR9 Bit 3, must be set (1) for any interrupt to occur.

Interrupt Pending Bit

The Interrupt Pending (IP) bit for a given source of interrupt is set by the presence of an interrupt condition. It is reset directly by the processor, or indirectly by some action that the processor may take. If the corresponding IE bit is not set, the IP for that source of interrupt never set. The IP bits can be read using RR3 as depicted below.

7	6	5	4	3		1	0
Reserved		RXIP	TXIP	Ext/StatIP	Reserved		
R		R	R	R	R		
–		0	0	0	–		

Interrupt-Under-Service Bit

The Interrupt-Under-Service (IUS) bits are set during an interrupt acknowledge cycle for the highest priority IP. The IUS bits can be set by either a hardware acknowledge cycle with the $\overline{\text{INTACK}}$ pin asserted, or if software writes WR9 Bit 5 as 1 and then reads RR2.

The IUS bits control the operation of the internal and daisy-chain. Within the ESCC channel, the internal daisy chain links the three sources of interrupt in a fixed order. If an internal IUS bit is set (1), all lower priority interrupt requests are masked off. During an interrupt acknowledge cycle the IP bits are also gated into the daisy chain. This process ensures that the highest priority IP is selected to have its IUS bit set. At the end of an interrupt service routine, the processor must issue a Reset Highest IUS command in WR0 to re-enable lower priority interrupts. This is the only way, short of a software or hardware reset, that an IUS bit may be reset.

Note: It is not necessary to issue the Reset Highest IUS command in the interrupt service routine, if no hardware acknowledge or software acknowledge cycle is executed. The only exception is when the SDLC Frame Status FIFO is enabled and Receive Interrupt On Special Condition Only is used. See the section “SDLC Frame Status FIFO” on page 287 for more details on this mode.

Disable Lower Chain Bit

The Disable Lower Chain (DLC) bit, WR9 Bit 2, is used to disable all peripherals in a lower position on the external daisy chain. If DLC is 1, the ESCC channel’s IEO output is driven Low and prevents lower-priority devices from generating an interrupt request. The IUS bit, when set, has the same effect, but is not controllable through software.

Daisy-Chain Resolution

The three sources of interrupt in the ESCC channel are prioritized in a fixed order via a daisy chain; these sources then form part of a larger daisy chain that can include high-priority external devices connected to the IEI pin, the on-chip CTCs and Bidirectional Centronics Controller; and low-priority external devices connection to the IEO pin. The receiver, transmitter, and External/Status interrupts are prioritized in that order. The ESCC channel's interrupt request is controlled by the IP bits and the IEI input, among other things. A flowchart of the interrupt sequence for the ESCC channel is illustrated in Figure 21.

The internal daisy chain links the three sources of interrupt in a fixed order. While an IUS bit is set, all lower-priority interrupt requests are masked off, thus preventing lower-priority interrupts, but allow higher-priority interrupts to occur. During an interrupt acknowledge cycle the IP bits are gated into the daisy chain. This action ensures that the highest priority IP is selected to set IUS. The MIE bit, `WR9` Bit 3, when reset (`0`), disables all interrupt requests.

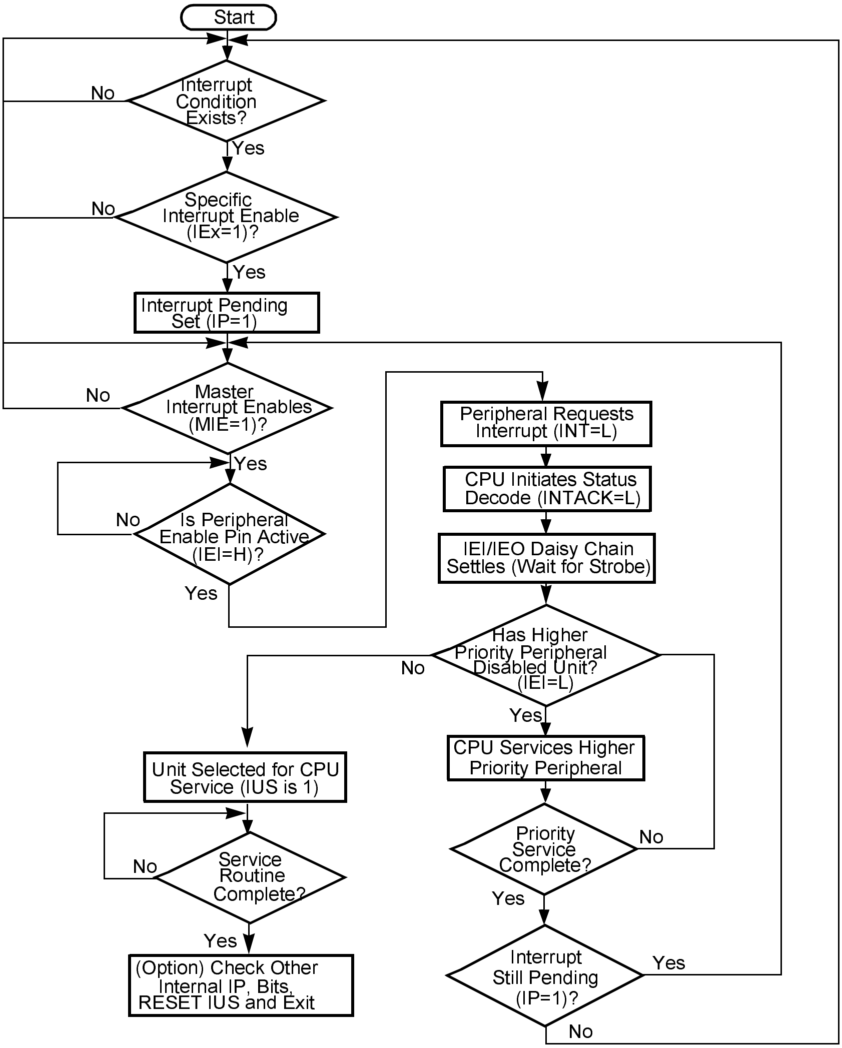

Start
Interrupt Condition Exists?
No
Yes
Specific Interrupt Enable (IEx=1)?
No
Yes
Interrupt Pending Set (IP=1)
Master Interrupt Enables (MIE=1)?
No
Yes
Is Peripheral Enable Pin Active (IEI=H)?
No
Yes
Peripheral Requests Interrupt (INT=L)
CPU Initiates Status Decode (INTACK=L)
IEI/IEO Daisy Chain Settles (Wait for Strobe)
Has Higher Priority Peripheral Disabled Unit? (IEI=L)
No
Yes
Unit Selected for CPU Service (IUS is 1)
Service Routine Complete?
No
Yes
(Option) Check Other Internal IP, Bits, RESET IUS and Exit
CPU Services Higher Priority Peripheral
Priority Service Complete?
No
Yes
Interrupt Still Pending (IP=1)?
Yes
No

Figure 21. Interrupt Flow Chart, For Each Interrupt Source

External Daisy-Chain Operations

The ESCC channel generates an interrupt request only if such requests are enabled (MIE is 1) and all of the following conditions occur:

- An IP bit and the corresponding IE bit are both set
- No higher priority IUS bit is set
- No higher priority interrupt is being serviced (IEI is High)
- No interrupt acknowledge transaction is taking place

The ESCC channel does not drive IEO Low when it requests an interrupt, but IEO instead continues to follow IEI until an interrupt acknowledge transaction occurs. Some time after the interrupt request, the processor initiates an Interrupt Acknowledge transaction. Between the time the ESCC channel recognizes that an Interrupt Acknowledge cycle is in progress and the time during the acknowledge cycle that the processor requests an interrupt vector, the IEI/IEO daisy chain settles. Any peripheral in the daisy chain having an Interrupt Pending (IP is 1) or an Interrupt Under Service (IUS is 1) holds its IEO line Low and all others make IEO follow IEI.

When the processor requests an interrupt vector, only the highest priority interrupt source with a pending interrupt (IP is 1) has its IEI input High, its IE bit 1, and its IUS bit 0. This is the interrupt source being acknowledged, and at this point it sets its IUS bit to 1. If its NV bit is 0, the ESCC channel identifies itself by placing the interrupt vector from WR2 on the data bus. If the NV bit is 1, the ESCC channel does not drive the data bus. If the VIS bit in the ESCC is 1, the vector also contains status information, encoded as described in Table 21, which further describes the nature of the ESCC interrupt.

Table 21. Interrupt Vector Modification

V2 V5	V1 V6	Status High/Status Low is 0 Status High/Status Low is 1
0	0	Transmit Buffer Empty
0	1	External/Status Change
1	0	Receive Character Available
1	1	Special Receive Condition

If the VIS bit is `0`, the vector held in `WR2` is returned without modification. If the ESCC channel is programmed to include status information in the vector, this status may be encoded and placed in either bits 2–1 or in bits 5–6. This operation is selected by programming the Status High/Status Low bit in `WR9`. At the end of the interrupt service routine, the processor issues the Reset Highest IUS command to unlock the daisy chain and allow lower-priority interrupt requests. The IP must be reset during the interrupt service routine, either directly by command or indirectly through some action taken by the processor.

The external daisy chain may be controlled by the DLC bit in `WR9`. This bit, when set, forces IEO Low, disabling all lower-priority devices

Interrupt Acknowledge

After the ESCC channel requests an interrupt, the CPU can respond with a hardware acknowledge cycle. After enough time has elapsed to allow the daisy chain to settle, the ESCC channel is requesting an interrupt and its IEI input is High, it sets the IUS bit for the highest priority IP. If the No Vector bit is reset (`WR9` bit 1 is `0`), the ESCC channel then places the interrupt vector on the data bus during a read.

To speed the interrupt response time, the ESCC channel can modify 2 bits in the vector to indicate the source of the interrupt. To include the status, the VIS bit, `WR9` bit 0, must be set (`1`). The service routine must then clear the interrupting condition. For example, writing a character to the FIFO clears the transmit buffer empty IP.

After the interrupting condition is cleared, the routine can read RR3 to determine if any other IP's are set and take the appropriate action to clear them. At the end of the interrupt routine, a Reset IUS command (WR0) must be issued to unlock the daisy chain and allow lower-priority interrupt requests. This is the only way, short of a software or hardware reset, that an IUS bit can be reset.

The Receiver Interrupt

Receive interrupts include Receive Character Available and Special Receive Condition. The Special Receive Condition can be subdivided into Receive Overrun, Framing Error (Asynchronous) or End of Frame (SDLC). In addition, a Parity Error can be a Special Receive condition.

As depicted in Figure 22, the Receive Interrupt mode is controlled by three bits in WR1. Bits 4–3 select the interrupt mode; while Bit 2 determines whether a Parity Error is a Special Receive Condition. WR7′ Bit 3 affects the receive interrupt operation mode as well. To view the register contents, refer to "Special Receive Condition Status Register (RR1)" on page 386,.

	4 3	2		
	RxINT Mode	ParitySplRx		
	W	W		

Figure 22. Write Register 1 Receive Interrupt Mode Control

If WR7′ Bit 3 is 0, a receive interrupt is generated when one byte is available in the FIFO. This mode is selected after reset. Systems with a long interrupt-response time can use this mode to generate an interrupt when one byte is received, but still allow up to seven more bytes to be received without an overrun error. By polling the Receive Character Available bit, RR0 Bit 0, and reading all available data to empty the FIFO before exiting the interrupt service routine, the frequency of interrupts can be minimized.

If WR7′ Bit 3 is 1, the ESCC channel generates an interrupt when there are four bytes in the Receive FIFO or when a special condition is received. This allows the CPU not to be interrupted until at least four bytes can be read from the FIFO, thereby minimizing the frequency of receive interrupts. If four or more bytes remain in the FIFO when the Reset Highest IUS command is issued at the end of the service routine, another receive interrupt is generated.

When a special receive condition is detected, an interrupt is generated immediately. This feature is intended to be used with the Interrupt On All Receive Characters and Special Condition mode. This is especially useful in SDLC mode because the characters are contiguous and the reception of the closing flag immediately generates a special receive interrupt. The generation of receive interrupts is described in the following two cases:

Case 1: Four Bytes Received with No Errors. A receive character available interrupt is triggered when the at least four bytes in the receive data FIFO are available and no special conditions have been detected. Therefore, the interrupt service routine can read four bytes from the data FIFO without having to read RR1 to check for error conditions.

Case 2: Data Received with Error Conditions. When any of the four oldest bytes in the receive error FIFO indicate an error has been detected, a Special Receive condition interrupt is triggered without waiting for the byte to reach the top of the FIFO. In this case, the interrupt service routine must read RR1 first before reading each data byte to determine which byte has the special receive condition and then take the appropriate action. because, in this mode, the status must be checked before the data is read, the data FIFO is not locked and the Error Reset command is not necessary.

WR7′ Bit 3 must be 0 when using Interrupt on First Character and Special Condition or Interrupt on Special Condition Only. See the description for Interrupt on All Characters or Special Condition mode for more details on this feature.

Note: The Receive Character Available Status bit, RR0 Bit 0, indicates whether at least one byte is available in the Receive FIFO, independent of WR7′ Bit 3. Therefore, this bit can be polled at any time for status if there is data in the Receive FIFO.

Receive Interrupts Disabled

This mode prevents the receiver from requesting an interrupt. It is used in a polled environment where either the status bits in RR0 or the modified vector in RR2 is read. Although the receiver interrupts are disabled, the interrupt logic can still be used to provide status.

When these bits indicate that a received character has reached the exit location of the FIFO, the status in RR1 must be checked and then the data must be read. If status is to be checked, it must be done before the data is read, because the act of reading the data pops both the data and error FIFOs.

Receive Interrupt on First Character or Special Condition

This mode is designed for use with DMA transfers of the receive characters. The processor is interrupted when the ESCC channel receives the first character of a block of data. It reads the character and then turns control over to a DMA device to transfer the remaining characters. After this mode is selected, the first character received, or the first character already stored in the FIFO, sets the receiver IP. This IP is reset when this character is removed from the Rx FIFO.

No further receive interrupts occur until the processor issues an Enable Interrupt on Next Receive Character command in WR0 or until a special receive condition occurs. The correct sequence of events when using this mode is to first select the mode and wait for the receive character available interrupt. When the interrupt occurs, the processor must read the character and then enable the DMA to transfer the remaining characters. WR7′ Bit 3 must be reset to 0 in this mode.

A special receive condition interrupt may occur any time after the first character is received, but is guaranteed to occur after the character having the special condition has been read. The status is not lost in this case, however, because the FIFO is locked by the special condition. In the interrupt service routine, the processor must read RR1 to obtain the status, and may read the data again if necessary. The FIFO is unlocked by issuing an Error Reset command in WR0. If the special condition was End-of-Frame, the processor must now issue the Enable Interrupt on Next Receive Character command to prepare for the next frame. The first character interrupt and special condition interrupt are distinguished by the status included in the interrupt vector. In all other respects they are identical, including sharing the IP and IUS bits.

Interrupt on All Receive Characters or Special Condition

This mode is designed for an interrupt driven system. In this mode, with WR7′ Bit 3 set to 0 the receiver sets the receive IP when a received character is shifted into the exit location of the FIFO. This occurs whether or not it has a special receive condition. This includes characters already in the FIFO when this mode is selected. In this mode of operation the IP is reset when the character is removed from the FIFO, so if the processor requires status for any characters, this status must be read before the data is removed from the FIFO.

With `WR7'` Bit 3 set to `1`, four bytes are accumulated in the Receive FIFO before an interrupt is generated (IP is set), and reset when the number of the characters in the FIFO is less than four.

The special receive conditions are identical to those previously mentioned, and as before, the only difference between a Receive Character Available interrupt and a Special Receive Condition interrupt is the status encoded in the vector. In this mode a special receive condition does not lock the receive data FIFO so that the service routine must read the status in `RR1` before reading the data.

At moderate to high data rates where the interrupt overhead is significant, time can usually be saved by checking for another character before exiting the service routine. This technique eliminates the interrupt acknowledge and the status processing, saving time, but care must be exercised because this receive character must be checked for special receive conditions before it is removed from the Rx FIFO.

Receive Interrupt on Special Conditions

This mode is designed for use when a DMA transfers all receive characters between memory and the ESCC channel. In this mode, only receive characters with special conditions causes the receive IP to be set. All other characters are assumed to be transferred via DMA. No special initialization sequence is needed in this mode. Usually, the DMA is initialized and enabled, then this mode is selected in the ESCC channel.

A special receive condition interrupt may occur at any time after this mode is selected, but the logic guarantees that the interrupt does not occur until after the character with the special condition has been read from the ESCC. The special condition locks the FIFO so that the status is valid when read in the interrupt service routine, and it guarantees that the DMA does not transfer any more characters until the special condition has been serviced.

In the service routine, the processor must read RR1 to obtain the status and unlock the FIFO by issuing an Error Reset command. DMA transfer of the receive characters then resumes. Figure 23 depicts the special conditions interrupt service routine.

Note: Special Receive Condition interrupts are generated after the character is read from the FIFO, not when the special condition is first detected. This sequence is performed so that when using receive interrupt on First or Special Condition, the first data character can be read out of the data FIFO without checking the status first.

If a Special Condition interrupted the CPU when first detected, it is necessary to read RR1 before each byte in the FIFO to determine which byte had the special condition. Therefore, by not generating the interrupt until after the byte has been read and then locking the FIFO, only one status read is necessary.

A DMA can be used to do all data transfers (otherwise, it is necessary to disable the DMA to allow the CPU to read the status on each byte). Consequently, because the special condition locks the FIFO to preserve the status, it is necessary to issue the Error Reset command to unlock it. Only the exit location of the FIFO is locked, allowing more data to be received into the other bytes of the Receive FIFO.

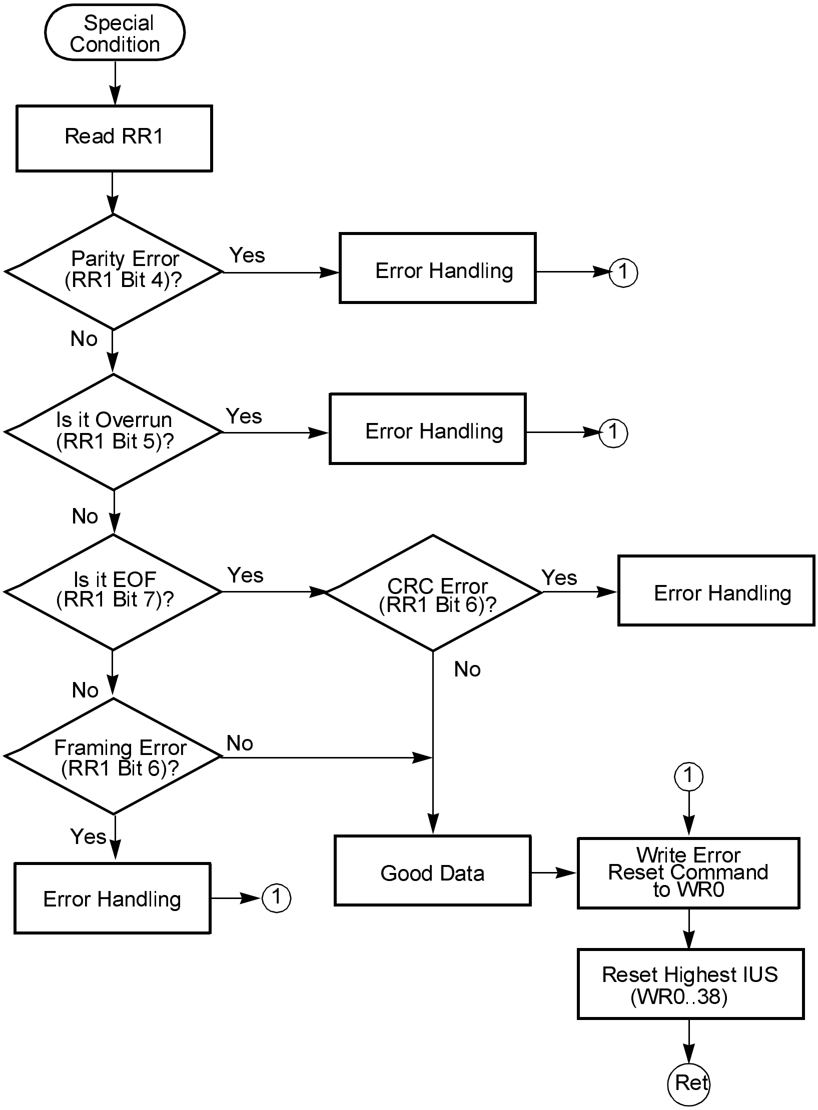


Figure 23. Special Conditions Interrupt Service Flow

Transmit Interrupts and Transmit Buffer Empty Bit

Transmit interrupts are controlled by the Transmit Interrupt Enable bit (Bit 1) in WR1.

The ESCC has two modes of transmit interrupt generation, which are controlled by Bit 5 of WR7′. One transmit mode generates interrupts when the entry location (the location to which the CPU writes data) of the Transmit FIFO is empty. This allows the ESCC transmit interrupt request to be tailored to system requirements for the frequency of interrupts and the interrupt response time. On the other hand, the Transmit Buffer Empty (TBE) bit on the ESCC responds the same way in each mode; the bit is set when the entry location of the Transmit FIFO is empty. The TBE bit is not directly related to the transmit interrupt status nor the state of WR7′ Bit 5.

When WR7′ Bit 5 is 1 (the default case), the ESCC channel generates a transmit interrupt when the Transmit FIFO becomes completely empty. The transmit interrupt occurs when the data in the exit location of the Transmit FIFO loads into the Transmit Shift Register and the Transmit FIFO becomes completely empty. This mode minimizes the frequency of transmit interrupts because the interrupt service routines can write 4 bytes to the Transmit FIFO each time it is entered.

The TBE bit, RR0 Bit 2, is set (1) whenever the entry location of the Transmit FIFO becomes empty. The TBE bit is reset (0) when the entry location becomes full. The TBE bit means *Transmit Buffer Not Full*, as it is set when the entry location of the Transmit FIFO becomes empty. This bit may be polled at any time to determine if a byte can be written to the FIFO. Figure 24 depicts when the TBE bit is set. WR7′ Bit 5 is 1 by a hardware or channel reset.

When WR7′ Bit 5 is 0, the TXIP bit is set when the entry location of the Transmit FIFO becomes empty. The ESCC transmits interrupts when there are 3 or fewer bytes in the FIFO, and continues to do so until the FIFO is filled. When WR7′ Bit 5 is 0, the transmit interrupt is reset momentarily when data is loaded into the entry location of the Transmit FIFO.

Transmit interrupt is not requested when the entry location of the Transmit FIFO is filled. The transmit interrupt is generated when the data is pushed down the FIFO and the entry location becomes empty (approximately one PCLK time). Figure 25 depicts when the transmit interrupts is requested. Again, the TBE bit is not dependent on the state of WR7′ Bit 5 nor the transmit interrupt status, and responds exactly the same way as mentioned above. Figure 25 illustrates when the TBE bit is set.

Note: When WR7′ Bit 5 is 0, multiple interrupts can be generated to fill the FIFO. To avoid multiple interrupts, software can poll the TBE bit (RR0 Bit 2) after writing each byte.

While transmit interrupts are enabled, the transmitter sets the TXIP when the transmit buffer reaches the condition programmed in WR7′ Bit 5. This means that the transmit buffer must have been written to before the TXIP is set.

The TXIP is reset either by writing data to the transmit buffer or by issuing the Reset Tx Int Pending command in WR0. Ordinarily, the response to a transmit interrupt is to write more data to the Tx FIFO; however, if it is the end of the frame, the Reset Tx Int Pending command is used to reset the TXIP and clear the interrupt. For example, at the end of a frame or block of data where the CRC is to be sent next, the Reset Tx Int Pending command must be issued after the last byte of data has been written to the Tx FIFO.

In synchronous modes, one other condition can cause the TXIP to be set. This occurs at the end of a transmission after the CRC is sent. When the last bit of the CRC has cleared the Transmit Shift Register and the flag or sync character is loaded into the Transmit Shift Register, the transmitter sets the TXIP. Data for the new frame or block to be transmitted may be written at this time. In this particular case, the Transmit Buffer Empty bit (RR0 Bit 2) and the TXIP are set.

The CRC at the end of the frame has priority over the data for the next frame. The CRC bytes are guaranteed to be sent, even if the data for the next packet is written before the second transmit interrupt, but after the EOM/Underrun condition. This CRC priority helps increase the system throughput because there is no waiting for the second transmit interrupt.

The ESCC latches the transmit interrupt when the CRC is loaded into the Transmit Shift Register even if the transmit interrupt, due to the last data byte, is not yet reset. Therefore, the end of a synchronous frame is guaranteed to generate two transmit interrupts even if a Reset Tx Int Pending command for the data created interrupt is issued after (Time A in Figure 24) the CRC interrupt had occurred. In this case, two reset TxInt Pending commands are required. The TXIP is latched if the EOM latch has been reset before the end of the frame.

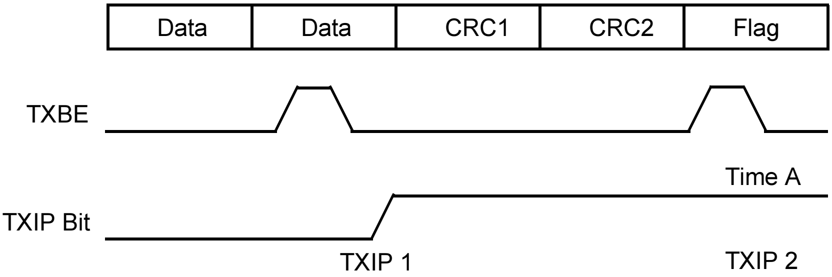


Figure 24. TXIP Latching on the ESCC

Transmit Interrupt and Tx Underrun/EOM Bit in Synchronous Modes

As described in the section above, when a frame-ending Underrun occurs, the transmitter always sends the CRC for one frame before the data for the next. After the Underrun/EOM (End Of Message) interrupt, the Tx FIFO accepts the data for the next packet without collapsing the two packets.

If data is written during the time period described above, the TBE bit (Bit 2 of RR0) is not set even if the second TXIP is guaranteed to set when the flag/sync pattern loads into the Transmit Shift Register. Therefore, there is no need to wait for the second TXIP to be set before writing data for the next packet. Figure 25 illustrates this process.

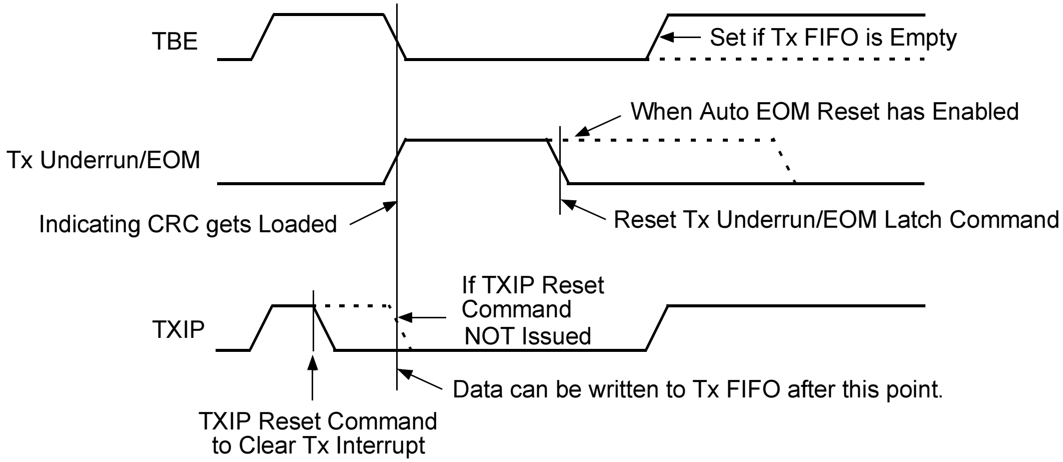

TBE
Set if Tx FIFO is Empty
When Auto EOM Reset has Enabled
Tx Underrun/EOM
Indicating CRC gets Loaded
Reset Tx Underrun/EOM Latch Command
If TXIP Reset Command NOT Issued
TXIP
Data can be written to Tx FIFO after this point.
TXIP Reset Command to Clear Tx Interrupt

Figure 25. Operation of TBE, Tx Underrun/EOM and TXIP

An example flowchart for processing the end of packet is depicted in Figure 26. In this chart, TXIP and Underrun/EOM INT can be processed by interrupts or by polling the registers. This flowchart does not include the procedures for interrupt handling, such as saving/restoring of registers to be used in the Interrupt Service Routine (ISR), the Reset IUS command, or the Return From Interrupt sequence.

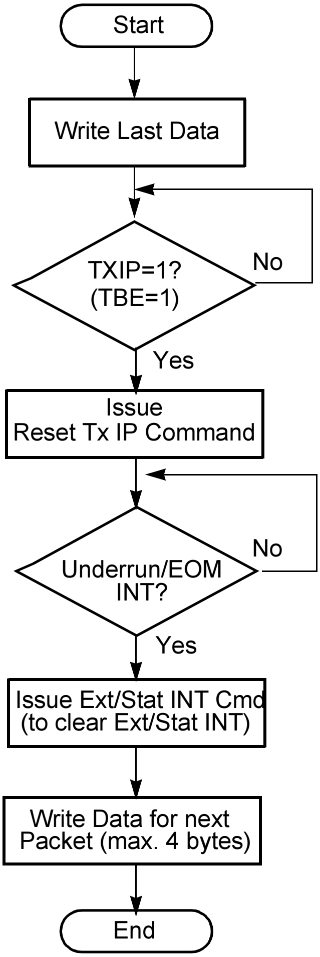


Figure 26. Flow Chart Example of Processing an End-of-Packet

External/Status Interrupts

Each channel has six external/status interrupt conditions: BRG Zero Count, Data Carrier Detect, Sync/Hunt, Clear to Send, Tx Underrun/EOM, and Break/Abort. The master enable for external/status interrupts is Bit 0 of WR1, and the individual enable bits are in WR15. Individual enable bits control whether or not a latch is present in the path from the source of the interrupt to the corresponding status bit in RR0. If the individual enable is 0, then RR0 reflects the current unlatched status, and if the individual enable is 1, then RR0 reflects the latched status.

The latches for the external/status interrupts are not independent. Rather, they all close at the same time as a result of a state change in one of the sources of enabled external/status interrupts. This process is described schematically in Figure 27.

The External/Status IP is set by the closing of the latches and remains set as long as they are closed. To determine which condition(s) require service when an external/status interrupt is received, the processor keeps an image of RR0 in memory and update this image each time it executes the external/status service routine.

Thus, a read of RR0 returns the current status for any bits whose individual enable is 0, and either the current state or the latched state of the remainder of the bits. To guarantee the current status, the processor issues a Reset External/Status interrupts command in WR0 to open the latches. The External/Status IP is set by the closing of the latches and remains set as long as they are closed. If the master enable for the External/Status interrupts is not set, the IP is never set, even though the latches may be present in the signal paths and working as described.

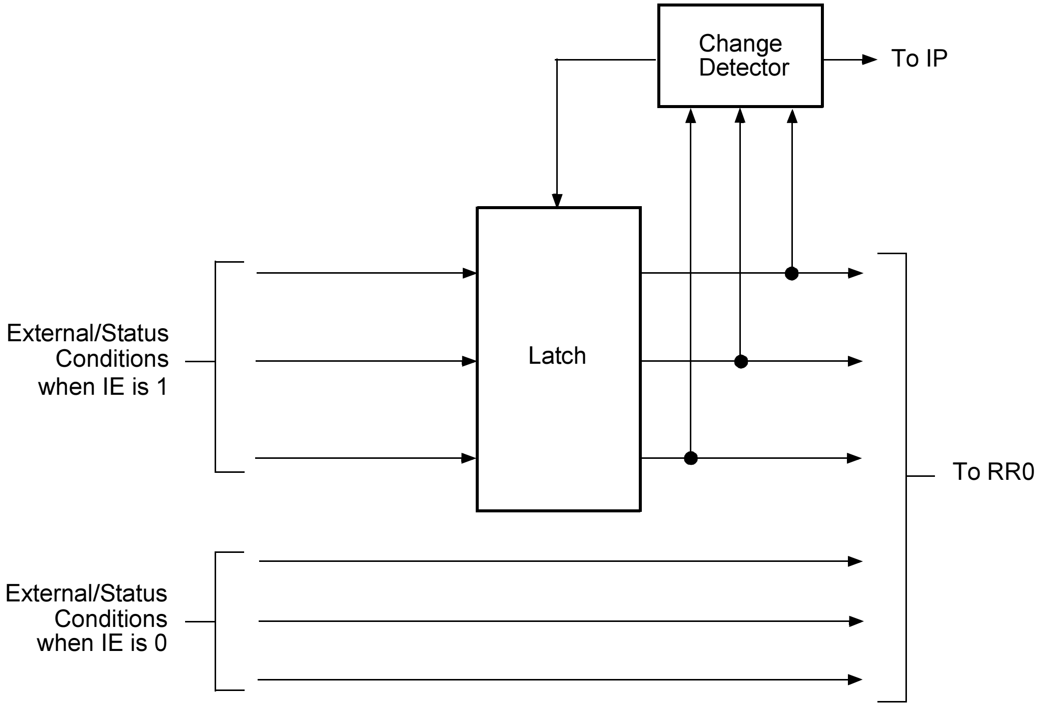

Change Detector
To IP
Latch
External/Status Conditions when IE is 1
External/Status Conditions when IE is 0
To RR0

Figure 27. RR0 External/Status Interrupt Operation

Because the latches close on the current status, but give no indication of change, the processor must maintain a copy of RR0 in memory. When the ESCC channel generates an External/Status Interrupt, the processor reads RR0 and determine which condition changed state and take appropriate action. Software then updates the copy of RR0 in memory and the Reset External/Status Interrupt command. Care must be taken in writing the interrupt service routine for the External/Status interrupts because it is possible for more than one status condition to change state at the same time. All of the latched bits in RR0 must be compared to the copy of RR0 in memory. If none have changed and the ZC interrupt is enabled, the Zero Count condition caused the interrupt.

The contents of RR0 are latched while reading this register. This prevents the contents of RR0 from changing while the read cycle is active.

The operation of the individual enable bits in WR15 for each of the six sources of External/Status interrupts is identical, but subtle differences exist in the operation of each source of interrupt. The six sources are:

- Break/Abort
- Underrun/EOM
- CTS
- DCD
- Sync/Hunt
- Zero Count

The Break/Abort, Underrun/EOM, and Zero Count conditions are internal to the ESCC, while Sync/Hunt, CTS and DCD are external signals. In the following discussions, each source is assumed to be enabled so that the latches are present and the External/Status interrupts are enabled. The External/Status IP is set (1) while the latches are closed and the state of the signal is reflected immediately in RR0 if the latches are open.

Break/Abort

The Break/Abort status is used in asynchronous and SDLC modes, but is always 0 in synchronous modes other than SDLC. In asynchronous modes, this bit is set when a break sequence (null character plus framing error) is detected in the receive data stream, and remains set as long as 0s continue to be received. This bit is reset when a 1 is received. A single null character is left in the Receive FIFO each time that the break condition is terminated. This character must be read and discarded.

In SDLC mode, this bit is set by the detection of an abort sequence which is seven or more contiguous 1s in the receive data stream. The bit is reset when a 0 is received. A received abort forces the receiver into Hunt mode, which is also an External/Status condition. Though these two bits change state at roughly the same time, it is possible that two External/Status Interrupts may be generated as a result.

The Break/Abort bit is unique in that both transitions are guaranteed to cause the latches to close, even if another External/Status interrupt is pending at the time these transitions occur. This guarantees that a Break or Abort are caught. This bit is undetermined after reset.

Transmit Underrun/EOM

The Transmit Underrun/EOM bit is used in synchronous modes to control the transmission of the CRC. This bit is reset by issuing the Reset Transmit Underrun/EOM command in WR0. However, this transition does not cause the latches to close; this occurs only when the bit is set. To inform the processor of this fact, the transmitter sets this bit when the CRC is loaded into the Transmit Shift Register. This bit is also set if the processor issues the Send Abort command in WR0. This bit is always set (1) in Asynchronous mode.

In SDLC mode this interrupt indicates when more data can be written to the Transmit FIFO. When this interrupt is used in this way, the Automatic SDLC Flag Transmission feature must be enabled (WR7′ Bit 0 is 1). The Transmit FIFO, which promotes the transmission of back to back frames.

CTS/DCD

The CTS bit reports the state of the $\overline{\text{CTS}}$ input, and the DCD bit reports the status of the $\overline{\text{DCD}}$ input. Both bits latch on either input transition. In both cases, after the Reset External/Status Interrupt command is issued, if the latches are closed, they remain closed if there is any odd number of transitions on an input; they open if there is an even number of transitions on the input.

Zero Count

The Zero Count bit is 1 when the counter in the Baud Rate Generator reaches a count of 0 and is 0 when the counter is reloaded. The latches are closed only when this bit is 1. The status in RR0 always reflects the current status. While the Zero Count IE bit (WR15 Bit 1) is reset, this bit is forced to 0.

Sync/Hunt

In Synchronous modes other than SDLC, Sync/Hunt reports the Hunt state of the receiver. Hunt mode is entered when the processor issues the Enter Hunt command in WR3. This command forces the receiver to search for a sync character match in the receive data stream. Because both transitions of the Hunt bit close the latches, issuing this command causes an External/Status interrupt. The receiver resets this bit to 0 when character synchronization has been achieved, causing the latches to again be closed.

In these synchronous modes, the receiver does not re-enter the Hunt mode automatically; only the Enter Hunt command sets this bit. In SDLC mode this bit is also set by the Enter Hunt command, but the receiver automatically enters the Hunt mode if an Abort sequence is received. The receiver leaves Hunt upon receipt of a flag sequence. Both transitions of the Hunt bit cause the latches to be closed. In SDLC mode, the receiver automatically synchronizes on Flag characters. The receiver is in Hunt mode when it is enabled, so the Enter Hunt command is seldom needed.

External/Status Interrupt Handling

If careful attention is paid to details, the interrupt service routine for External/Status interrupts is straightforward. To determine which bit or bits changed state, the routine must first read RR0 and compare it to a copy from memory. For each changed bit, the appropriate action must be taken and the copy in memory updated. The service routine must close with two Reset External/Status interrupt commands to reopen the latches. The copy of RR0 in memory must always have the Zero Count bit 0, because this is the state of the bit after the Reset External/Status interrupts command at the end of the service routine. When the processor issues the Reset Transmit Underrun/EOM latch command in WR0, the Transmit Underrun/EOM bit in the copy of RR0 in memory must be reset because this transition does not cause an interrupt.

Write Registers

Command Register (WR0)

WR0 is the Indirect Register and the Command Register.

7 6	5 3	2 0
CRC Reset Codes	Command Codes	Register Selection
W	W	W
0	0	0

Bit Number	Field	R/W	Reset Value	Description
7..6	CRC Reset Codes	W	0	**Null Command** 00: This command has no effect on the ESCC channel and is used when a write to WR0 is necessary for some reason other than one of the Reset commands in the field. **Reset Receive CRC Checker Command** 01: This command is used to initialize the receive CRC circuitry. It is necessary in synchronous modes (except SDLC) if the Enter Hunt Mode command in Write Register 3 is not issued between received messages. Any action that disables the receiver initializes the CRC circuitry. Resetting the Receive CRC Checker command is accomplished automatically in SDLC mode. **Reset Transmit CRC Generator Command** 10: This command initializes the CRC generator. It is usually issued in the initialization routine and after the CRC has been transmitted. A Channel Reset does not initialize the generator and this command is not issued until after the transmitter has been enabled in the initialization routine. This command is not needed if Auto EOM Reset mode is enabled (WR7′ Bit 1 is 1).

Bit Number	Field	R/W	Reset Value	Description
7..6 (Cont.)				**Reset Transmit Underrun/EOM Latch Command** 11: This command controls the transmission of CRC at the end of transmission (EOM). If this latch has been reset, and a transmit underrun occurs, the transmitter automatically appends CRC to the message. In SDLC mode with Abort on Underrun selected, the transmitters sends an abort and Flag on underrun if the TX Underrun/EOM latch has been reset. At the start of the CRC transmission, the Tx Underrun/EOM latch is set. The Reset command can be issued at any time during a message. If the transmitter is disabled, this command does not reset the latch. However, if no External Status interrupt is pending, or if a Reset External Status interrupt command accompanies this command while the transmitter is disabled, an External/Status interrupt is generated with the Tx Underrun/EOM bit reset in RR0.
5..3	Comman d Codes	W	0	**Null Command** 000: The Null command has no effect on the ESCC channel. **Point High Command** 001: This command effectively adds eight to the Register Pointer (Bits 2..0), allowing WR8 through WR15 to be accessed. The Point High command and the Register Pointer bits are written simultaneously.

Bit Number	Field	R/W	Reset Value	Description
				Reset External/Status Interrupts Command 100: After an External/Status interrupt (a change on a modem line or a break condition, for example), the status bits in RR0 are latched. This command re-enables the bits and allows interrupts to occur again as a result of a status change. Latching the status bits captures short pulses until the CPU has time to read the change. The ESCC channel contains simple queueing logic associated with most of the external status bits in RR0. If another External/Status condition changes while a previous condition is still pending (the Reset External/Status Interrupt command has not yet been issued) and this condition persists until after the command is issued, this second change causes another External/Status interrupt. However, if this second status change does not persist (there are two transitions), another interrupt is not generated. Exceptions to this rule are detailed in the RR0 description.
5..3 (Cont.)				**Send Abort Command** 011: This command is used in SDLC mode to transmit a sequence of eight to thirteen 1s. This command always empties the transmit buffer and sets Tx Underrun/EOM bit in RR0.
				Enable Interrupt On Next Rx Character Command 100: If the Interrupt on First Received Character mode is selected, this command is used to reactivate that mode after each message is received. The next character to enter the Receive FIFO causes a Receive interrupt. Alternatively, previously-stored data in the FIFO causes a Receive interrupt.

Bit Number	Field	R/W	Reset Value	Description
				Reset Tx Interrupt Pending Command 101: This command is used in cases where there are no more characters to be sent; for example, at the end of a message. This command prevents further transmit interrupts until after the next character has been loaded into the transmit buffer or until the CRC has been sent. This command is necessary to prevent the transmitter from requesting an interrupt when the transmit buffer becomes empty (with Transmit Interrupt Enabled). **Error Reset Command** 110: This command resets the error bits in `RR1`. If the Interrupt on First Rx Character or Interrupt on Special Condition mode is selected and a special condition exists, the data with the special condition is held in the Receive FIFO until this command is issued. If either of these modes is selected and this command is issued before the data has been read from the Receive FIFO, the data is lost. **Reset Highest IUS Command** 111: This command resets the highest priority Interrupt Under Service (IUS) bit, allowing lower-priority conditions to request interrupts. This command allows the use of the internal daisy chain (even in systems without an external daisy chain) and must be the last operation in an interrupt service routine.
2..0	Register Selection Code	W	0	These three bits select `WR0..7`. In conjunction with the Point High command, `WR8..15` are selected.

Transmit/Receive Interrupt and Data Transfer Mode Definition Register (WR1)

Write Register 1 controls the various interrupt and Wait/Request modes.

7 5	4	2	1	0
WAIT/DMA Request Enable	Receive Interrupt Modes	Parity Is Special Condition	Transmitter Interrupt Enable	External/ Status Master Interrupt Enable
W	W	W	W	W
0	0	X	0	0

Bit Number	Field	R/W	Reset Value	Description
7..5	WAIT/ DMA Request Enable	W	0	**WAIT/DMA Request Enable** This bit enables the Wait/Request function in conjunction with the Request/Wait Function Select bit (bit 6). These three bits control whether the ESCC channel waits the processor when it requests a data transfer that the ESCC channel cannot do. They also determine whether software can use the DMA channels to transfer data to or from the ESCC. 0xx: No Wait states are generated for accesses to the ESCC. The DMA channels cannot be used to transfer data to or from the ESCC. 10x: The ESCC channel generates Wait states to stop the processor if software reads from an Empty Rx FIFO or writes to a full Tx FIFO. When an Rx byte arrives or the Tx FIFO has room for a Tx byte, the WAIT state is removed. DMA channels cannot be used for ESCC data. 110: Reserved. Do not program this value. 111: Program this value if a DMA channel is used for Rx and/or Tx data. No WAIT states are generated by the ESCC channel.
4..3	Receive Interrupt Modes	W	0	**Receive Interrupt Modes** 00: This mode prevents the receiver from requesting an interrupt. It is normally used in a polled environment where either the status bits in RR0 or the modified vector in RR2 are monitored to initiate a service routine. Although receiver interrupts are disabled, a special condition can still provide a unique vector status in RR2.

Bit Number	Field	R/W	Reset Value	Description
		W		**Receive Interrupt on First Character or Special Condition** 01: The receiver requests an interrupt in this mode on the first available character (or stored FIFO character) or on a special condition. Sync characters, stripped from the message stream, do not cause interrupts. Special receive conditions are: receiver overrun, framing error, end of frame, or parity error (if selected). If a special receive condition occurs, the data containing the error is stored in the Receive FIFO until an Error Reset command is issued by the CPU. This mode is usually selected when a Block Transfer mode is used. In this interrupt mode, a pending special receive condition remains set until either an error Reset command, a channel or hardware reset, or until receive interrupts are disabled. The Receive Interrupt on First Character or Special Condition mode can be re-enabled by the Enable Rx Interrupt on Next Character command in `WR0`. See the description of `WR7′` on how this function can be changed.
	Interrupt on All Receive Characters or Special Condition			**Interrupt on All Receive Characters or Special Condition** 10: This mode allows an interrupt for every character received (or character in the Receive FIFO) and provides a unique vector when a special condition exists. The Receiver Overrun bit and the Parity Error bit in `RR1` are two special conditions that are latched. These two bits are reset by the Error Reset command. Receiver overrun is always a special receive condition, and parity can be programmed to be a special condition. Data characters with special receive conditions are not held in this mode as they are in the other receive interrupt modes.

Bit Number	Field	R/W	Reset Value	Description
4..3 (Cont.	Receive Interrupt On Special Condition			**Receive Interrupt On Special Condition** 11: This mode allows the receiver to interrupt only on characters with a Special Receive Condition. When an interrupt occurs, the data containing the error is held in the Receive FIFO until an Error Reset command is issued. When using this mode in conjunction with a DMA, the DMA is initialized and enabled before any characters have been received by the ESCC. This eliminates the time-critical section of code required in the Receive Interrupt on First Character or Special Condition mode. Hence, all data can be transferred via the DMA so that the CPU need not handle the first received character as a special case. In SDLC mode, if the SDLC Frame Status FIFO is enabled and an EOF is received, an interrupt with vector for receive data available is generated and the Receive FIFO is not locked.
2	Parity Is Special Condition	W	0	**Parity Is Special Condition** A special condition modifies the status of the interrupt vector stored in WR2. During an interrupt acknowledge cycle, this vector can be placed on the data bus. 1: Any received characters with parity not matching the sense programmed in WR4 creates a Special Receive Condition. 0: If parity is disabled (WR4), this bit is ignored.

Bit Number	Field	R/W	Reset Value	Description
1	Transmitter Interrupt Enable	W	0	**Transmitter Interrupt Enable** If this bit is 1, the transmitter requests an interrupt whenever the transmit FIFO reaches the degree of emptiness selected by WR7′ bit 5. 1: The transmitter interrupt is enabled. 0: The transmitter interrupt is disabled.
0	External/ Status Master Interrupt Enable	W	0	**External/Status Master Interrupt Enable** This bit is the master enable for External/Status interrupts including the $\overline{\text{DCD}}$ and $\overline{\text{CTS}}$ pins, break, abort, the beginning of CRC transmission when the Transmit/Underrun/ EOM latch is set, or when the counter in the Baud Rate Generator reaches 0. WR15 contains the individual enable bits for each of these sources of External/Status interrupts. This bit is reset by a channel or hardware reset. 1: External/Status master interrupt is enabled. 0: External/Status master interrupt is disabled.

Interrupt Vector Register (WR2)

WR2 is the interrupt vector register.

7	0
Interrupt Vector	
W	
X	

Bit Number	Field	R/W	Reset Value	Description
7..0	Interrupt Vector	W	X	**Interrupt Vector** The interrupt vector can be modified by status information. This is controlled by the Vector Includes Status (VIS) and the Status High/Status Low bits in WR9. 1: Interrupt Vector is enabled. 0: Interrupt Vector is disabled.

Receive Parameters and Control Register (WR3)

This register contains the control bits and parameters for the receiver logic. With the Extended Read option enabled this register may be read as RR9

7	6	5	4	3	2	1	0
Receiver Bits/Character		Auto Enable	Enter Hunt Mode	Receive CRC Enable	Address Search Mode	Sync Character Load Inhibit	Receiver Enable
W		W	W	W	W	W	W
X		X	X	X	X	X	0

Bit Number	Field	R/W	Reset Value	Description
7..6	Receiver Bits/ Character	W	X	**Receiver Bits/Character** The state of these two bits determines the number of bits to be assembled as a character in the received serial data stream. The number of bits per character can be changed while a character is being assembled, but only before the number of bits currently programmed is reached. Unused bits in the Received Data Register (`RR8`) are `1` in asynchronous modes. In Synchronous and SDLC modes, the receiver transfers an 8-bit section of the serial data stream to the Receive FIFO at the appropriate time. 00: Rx 5 Bits/Character 01: Rx 7 Bits/Character 10: Rx 6 Bits/Character 11: Rx 8 Bits/Character
5	Auto Enable	W	X	**Auto Enable** This bit programs the function for both the $\overline{\text{DCD}}$ and $\overline{\text{CTS}}$ pins. 1: $\overline{\text{CTS}}$ becomes the transmitter enable and $\overline{\text{DCD}}$ becomes the receiver enable when this bit is `1`. However, the Receiver Enable and Transmit Enable bits must be set before the $\overline{\text{DCD}}$ and $\overline{\text{CTS}}$ pins, respectively, can be used in this manner. 0: The $\overline{\text{DCD}}$ and $\overline{\text{CTS}}$ pins are inputs to the corresponding status bits in Read Register 0. The state of $\overline{\text{DCD}}$ is ignored in the Local Loopback mode. The state of $\overline{\text{CTS}}$ is ignored in both Auto Echo and Local Loopback modes.

Bit Number	Field	R/W	Reset Value	Description
4	Enter Hunt Mode	W	X	**Enter Hunt Mode** This command forces the comparison of sync characters or flags to assembled receive characters for the purpose of synchronization. Whenever a flag or sync character is matched, the Sync/Hunt bit in RR0 is reset and, if External/ Status Interrupt Enable is set, an interrupt is requested. The receiver automatically enters the Hunt mode when an abort condition is received or when the receiver is enabled (except in asynchronous modes). 1: Enter Hunt mode 0: Do not enter Hunt mode
3	Receiver CRC Enable	W	X	**Receiver Cyclic Redundancy Check Enable** This bit is used to initiate CRC calculation at the beginning of the last byte transferred from the Receiver Shift register to the Receive FIFO. This operation occurs independently of the number of bytes in the Receive FIFO. When a particular byte is to be excluded from the CRC calculation, this bit must be reset before the next byte is transferred to the Receive FIFO. If this feature is used, care must be taken to ensure that eight bits per character is selected in the receiver because of an inherent delay from the Receive Shift register to the CRC checker. This bit is internally 1 in SDLC mode and the receiver calculates the CRC on all bits except 0s inserted between the opening and closing flags. This bit is ignored in asynchronous modes. 1: Receiver CRC enabled 0: Receiver CRC disabled

Bit Number	Field	R/W	Reset Value	Description
2	Address Search Mode	W	X	**Address Search Mode (SDLC)** Setting this bit in SDLC mode causes messages with addresses not matching the address programmed in WR6 to be ignored. No receiver interrupts occur in this mode unless there is an address match. The address that the receiver attempts to match is unique (1 in 256) or multiple (16 in 256), depending on the state of Sync Character Load Inhibit bit. Address FFH is always recognized as a global address. The Address Search mode bit is ignored in all modes except SDLC. 1: Enter Address Search Mode 0: Do not enter Address Search Mode

Bit Number	Field	R/W	Reset Value	Description
1	Sync Character Load Inhibit	W	X	**Sync Character Load Inhibit** If this bit is 1 in any mode except SDLC, the receiver compares the byte in WR6 with the byte about to be stored in the FIFO, and it inhibits this load if the bytes are equal. This sequence also occurs in the asynchronous mode if the received character matches the contents of WR6.) The receiver does not calculate the CRC on bytes stripped from the data stream in this manner. The comparison for detecting these embedded syncs is always performed 8 bits wide, so this feature cannot be used with 6-bit Sync characters in Monosync mode, which are selected by setting Bit 1 of WR10 to 1. The address recognition logic of the receiver is modified in SDLC mode if this bit is 1, that is, only the four most significant bits of WR6 must match the receiver address. This procedure allows the ESCC channel to receive frames from up to 16 separate sources without programming WR6 for each source (if each station address has the four most significant bits in common). The address field in the frame is still eight bits long. Address FFH is always recognized as a global address. The bit is ignored in SDLC mode if Address Search mode has not been selected. 1: Sync Character Load Inhibit is enabled. 0: Sync Character Load Inhibit is disabled.
0	Receiver Enable	W	0	**Receiver Enable** When this bit is 1, receiver operation begins. Set this bit only after all other receiver parameters are established and the receiver is completely initialized. This bit is reset by writing a 0 to it or by a channel or hardware reset command, and it disables the receiver. 1: Receiver Enable is ON. 0: Receiver Enable is OFF.

Transmit/Receive Miscellaneous Parameters and Modes Register (WR4)

WR4 contains the control bits for both the receiver and the transmitter. These bits must be set in the transmit and receiver initialization routine before issuing the contents of WR1, WR3, WR6, and WR7. With the Extended Read option enabled, this register is read as RR4.

7	6	5	4	3	2	1	0
Clock Rate		Sync Mode Selection		Stop Bits Selection		Parity Even/Odd Select	Parity Enable
W		W		W		W	W
X		X		X	1	X	X

Bit Number	Field	R/W	Reset Value	Description
7..6	Clock Rate	W	X	**Clock Rate Bits** These bits specify the multiplier between the clock and data rates. In synchronous modes, the 1X mode is forced internally and these bits are ignored unless External Sync mode has been selected. 00**:** 1X Mode. The clock rate and data rate are the same. 01**:** 16X Mode. The clock rate is 16 times the data rate. 10**:** 32X Mode. The clock rate is 32 times the data rate. 11**:** 64X Mode. The clock rate is 64 times the data rate.

Bit Number	Field	R/W	Reset Value	Description
5..4	Sync Mode Selection	W	X	**Sync Mode Selection Bits** These two bits select the various options for character synchronization. They are ignored unless synchronous modes are selected in the stop bits field of this register. 00**:** Monosync Mode. In this mode, the receiver achieves character synchronization by matching the character stored in WR7 with an identical character in the received data stream. The transmitter uses the character stored in WR6 as a time fill. The sync character is either six or eight bits, depending on the state of the 6-bit/8-bit sync bit in WR10. If the Sync Character Load Inhibit bit is set, the receiver strips the contents of WR6 from the data stream if received within character boundaries.
5..4 (Cont.)				01**:** Bisync Mode. The concatenation of WR7 with WR6 is used for receiver synchronization and as a time fill by the transmitter. The sync character is 12 or 16 bits in the receiver, depending on the state of the 6-bit/8-bit sync bit in WR10. The transmitted character is always 16 bits. 10**:** SDLC Mode. In this mode, SDLC is selected and requires a Flag (01111110) to be written to WR7. If Address Search mode is selected, the receiver address field must be written to WR6. The SDLC CRC polynomial must be selected in WR5. 11**:** Reserved. Do not program.

Bit Number	Field	R/W	Reset Value	Description
3..2	Stop Bits Selection	W	X, 1	**Stop Bits Selection** These bits determine the number of stop bits added to each asynchronous character that is transmitted. The receiver always checks for one stop bit in Asynchronous mode. A special mode specifies that a Synchronous mode is to be selected. Bit 2 is always 1 by a channel or hardware reset. 00**:** Synchronous Modes Enable. This bit combination selects one of the synchronous modes specified by Bits 7..4 of this register and forces the 1X Clock mode internally. 01**:** 1 Stop Bit/Character. This bit selects Asynchronous mode with one stop bit per character. 10**:** 1 1/2 Stop Bits/Character. These bits select Asynchronous mode with 1-1/2 stop bits per character. This mode cannot be not used with the 1X clock mode. 11**:** 2 Stop Bits/Character. These bits select Asynchronous mode with two stop bits per transmitted character and checks for one received stop bit.
1	Parity Even/ Odd Select Bit	W	X	**Parity Even/Odd Select Bit** This bit determines whether parity is checked as even or odd. This bit is ignored if the Parity Enable bit is not set. 1: Even parity 0: Odd parity.
0	Parity Enable	W	X	**Parity Enable** When this bit is set, an additional bit position beyond those specified in the bits/character control is added to the transmitted data and is checked in the receive data. The Received Parity bit is transferred to the CPU as part of the data unless eight bits per character is selected in the receiver. 1: Additional bits added 0: Additional bits not added

Transmit Parameters and Controls (WR5)

WR5 contains control bits that affect the operation of the transmitter. Bit 2 affects both the transmitter and the receiver. With the Extended Read option enabled, this register can be read as RR5.

7	6	5	4	3	2	1	0
Data Terminal Ready Control Bit	Transmit Bits/Character		Send Break Control Bit	Transmit Enable	SDLC/ CRC-16 Polynomial Select Bit	Request To Send Control Bit	Transmit CRC Enable
W	W		W	W	W	W	W
0	X		0	0	0	0	X

Bit Number	Field	R/W	Reset Value	Description
7	Data Terminal Ready Control Bit	W	0	**Data Terminal Ready Control Bit** This is the control bit for the $\overline{\text{DTR}}$ pin. When set, $\overline{\text{DTR}}$ is Low; when reset, $\overline{\text{DTR}}$ is High. This bit is reset by a channel or hardware reset. 1: $\overline{\text{DTR}}$ is Low 0: $\overline{\text{DTR}}$ is High
6..5	Transmit Bits/ Character Select Bits 1 and 0	W	X	**Transmit Bits/Character Select Bits 1 and 0** These bits control the number of bits in each byte transferred to the transmit buffer. Bits sent must be right justified; The least significant bits are sent first. The Five Or Less mode allows transmission of one to five bits per character. For five or fewer bits per character, the data character must be formatted as described in Table 22. In the Six or Seven Bits/Character modes, unused data bits are ignored. For five or less bits per character selection in `WR5` the coding used in the data sent to the transmitter is described in Table 22. 00: 5 or less bits/character 00: 7 bits/character 00: 6 bits/character 00: 8 bits/character

Bit Number	Field	R/W	Reset Value	Description
4	Send Break Control Bit	W	0	**Send Break Control Bit** If this bit is 1 after setting exactly the following combination of register bits: 1. WR4 Bits 5..2 are 1000 (SDLC) 2. WR5 Bit 1 is 0 (RTS false) 3. WR7′ Bit 2 is 1 (auto RTS deactivation) 4. WR10 Bit 3 is 1 (mark idle) then the Transmitter enters, and operates in, a special LocalTalk (AppleTalk) mode as described in an earlier section. In all other cases, if this bit is 1, the transmitter forces the TxD output to send continuous 0s beginning with the following transmit clock, regardless of any data being transmitted at the time. This bit functions whether or not the transmitter is enabled. When reset, TxD continues to send the contents of the Transmit Shift register, which might be syncs, data, or all 1s. If this bit is set while in the X21 mode (Monosync and Loop mode selected) and character synchronization is achieved in the receiver, this bit is automatically reset and the transmitter begins sending syncs or data. This bit is also reset by a channel or hardware reset. 1: TxD output sends continuous 0s 0: TxD sends the contents of the Transmit Shift register
3	Transmit Enable	W	0	**Transmit Enable** Data is not transmitted until this bit is set. When this bit is 1, the TxD output sends continuous 1s unless Auto Echo mode or SDLC Loop mode is selected. If this bit is reset after transmission starts, the transmission of data or sync characters is completed. If the transmitter is disabled during the transmission of a CRC character, sync or flag characters are sent instead of CRC. This bit is reset by a channel or hardware reset. 1: TxD output sends continuous 1s 0: Transmission of data or sync characters is completed

Bit Number	Field	R/W	Reset Value	Description
2	SDLC/ CRC-16 Polynomial Select Bit	W	0	**SDLC/CRC-16 Polynomial Select Bit** This bit selects the CRC polynomial used by both the transmitter and receiver. When set, the CRC-16 polynomial is used; when reset, the SDLC polynomial is used. The SDLC/ CRC polynomial must be selected when SDLC mode is selected. The CRC generator and checker can be preset to all `0`s or all `1`s, depending on the state of the Preset 1/Preset 0 bit in `WR10`. 1: CRC-16 polynomial enabled 0: SDLC polynomial enabled

Bit Number	Field	R/W	Reset Value	Description
1	Request To Send Control Bit	W	0	**Request To Send Control Bit** This bit is the control bit for the $\overline{\text{RTS}}$ pin. When the RTS bit is set, the $\overline{\text{RTS}}$ pin goes Low; when reset, $\overline{\text{RTS}}$ goes High. When Auto Enable is set in asynchronous mode, the $\overline{\text{RTS}}$ pin immediately goes Low when the RTS bit is set. However, when the RTS bit is reset, the $\overline{\text{RTS}}$ pin remains Low until the transmitter is completely empty and the last stop bit has left the TxD pin. In synchronous modes, the $\overline{\text{RTS}}$ pin directly follows the state of this bit, except in SDLC mode under specific conditions. In SDLC mode, if Flag On Underrun bit (`WR10`, bit 2) is set, this bit in `WR5` is reset, and Bit 2 in `WR7'` is set, the $\overline{\text{RTS}}$ pin de-asserts automatically at the last bit of the closing flag triggered by the rising edge of the Tx clock. This bit is reset by a channel or hardware reset. 1: RTS is Low 0: RTS is High
0	Transmit CRC Enable	W	X	**Transmit CRC Enable** This bit determines whether or not the CRC is calculated on each transmit character. If this bit is set at the time the character is loaded from the transmit buffer to the Transmit Shift register, the CRC is calculated on that character. The CRC is not automatically sent unless this bit is set when the transmit underrun occurs. 1: Transmit CRC enabled 0: Transmit CRC disabled

Table 22. Data Encoding for Five or Less Bits/Character in WR5, Bits 6 and 5

7	6	5	4	3	2	1	0	
1	1	1	1	0	0	0	Data	Sends one data bit
1	1	1	0	0	0	Data		Sends two data bits
1	1	0	0	0	Data			Sends three data bits
1	0	0	0	Data				Sends four data bits
0	0	0	Data					Sends five data bits

Character or SDLC Address Field Register (WR6)

WR6 is programmed to contain the transmit sync character in the Monosync mode, or the first byte of a 16-bit Sync character in the Bisync mode. WR6 is not used in asynchronous modes. In the SDLC mode, it is programmed to contain the value used to compare against the address field of the SDLC Frame. In SDLC mode, the transmitter does not automatically transmit the station address at the beginning of a response frame. Because this register is used in different modes, the tables below describe the fields used in each mode.

7							0
Sync 7..0							
W	W	W	W	W	W	W	W
X	X	X	X	X	X	X	X

Bit Number	Field	R/W	Reset Value	Description
7..0		W	X	Sync7..0

Bit Number	Field	R/W	Reset Value	Description
7..6	Not used	W	X	Sync1..0
5..0		W	X	Sync5..0

Bit Number	Field	R/W	Reset Value	Description
7..0		W	X	First sync character

Bit Number	Field	R/W	Reset Value	Description
7..4		W	X	Sync3.. Sync0
3..0		W	X	1

Bit Number	Field	R/W	Reset Value	Description
7..4		W	0	ADR7..ADR4
3..0		W	0	x

Sync Character or SDLC Flag Register (WR7)

WR7 is programmed to contain the receive sync character in the Monosync mode, a second byte (the last eight bits) of a 16-bit sync character in the Bisync mode or a Flag character (01111110) in the SDLC modes. WR7 is not used in Asynchronous mode. Because this register is used in different modes, the tables below describe the fields used in each mode.

7	0
Sync 7..0	
W	
X	

Bit			Reset	
Number	Field	R/W	Value	Description
7..0		W	X	Sync7..0

Bit			Reset	
Number	Field	R/W	Value	Description
7..2		W	X	Sync5..0
1..0		W	X	x

Bit			Reset	
Number	Field	R/W	Value	Description
7..0		W	X	Sync15..81

Bit Number	Field	R/W	Reset Value	Description
7..0		W	X	Sync11..4

Bit Number	Field	R/W	Reset Value	Description
7		W	X	0
6		W	X	1
5		W	X	1
4		W	X	1
3		W	X	1
2		W	X	1
1		W	X	1
0		W	X	0

Write Register 7 Prime

Write Register 7 Prime (`WR7'`) is located at the same address as Write Register 7. This register is written to by setting Bit 0 of `WR15` to `1`. Refer to the description in the section on `WR15`. Bit 5 is set after a reset. All other bits reset to `0`.

7	6	5	4	3	2	1	0
Reserved	Extended Read Enable	Transmit FIFO Interrupt Level	Tx Request Timing	Receive FIFO Interrupt Level	Auto $\overline{\text{RTS}}$ Pin Deactivation	Automatic EOM Reset	Automatic Tx SDLC Flag
	W	W	W	W	W	W	W
	0	1	0	0	0	0	0

Bit Number	Field	R/W	Reset Value	Description
7	Reserved			**Reserved.** Must be 0.
6	Extended Read Enable	W	0	**Extended Read Enable** Setting this bit enables the reading of `WR3`, `WR4`, `WR5`, `WR7'` and `WR10`. When this feature is enabled, these registers can be accessed by reading `RR9`, `RR4`, `RR5`, `RR14`, and `RR11`, respectively. 1: Reading of `WR3`, `WR4`, `WR5`, `WR7'` and `WR10` enabled. 0: Reading of `WR3`, `WR4`, `WR5`, `WR7'` and `WR10` disabled.

Bit Number	Field	R/W	Reset Value	Description
5	Transmit FIFO Interrupt Level		1	**Transmit FIFO Interrupt Level** A Transmit DMA request is asserted when the Transmit FIFO is completely empty if bit is set. The request is asserted when the entry location of the Transmit FIFO is empty if the Transmit FIFO Interrupt Level bit is reset (0). 1: If this bit is 1, the transmit buffer empty interrupt is generated when the Transmit FIFO is completely empty. 0: If this bit is 0, the transmit buffer empty interrupt is generated when the entry location of the Transmit FIFO is empty.
4	Tx Request Timing	W	1	**Tx Request Timing** Software must always set this bit to 1 to ensure that the timing on the Transmit DMA Request is the same as the timing on the Receive DMA request.
3	Receive FIFO Interrupt Level	W	0	**Receive FIFO Interrupt Level** 1: If this bit is 1 and Receive Interrupt on All Characters and Special Conditions is enabled, the Receive Character Available interrupt is triggered when the Rx FIFO is half full; that is, when four byte slots of the Rx FIFO are empty. However, if any character has a special condition, a special condition interrupt is generated when the character is loaded into the Receive FIFO. Therefore, the special condition interrupt service routine must read RR1 before reading the data to determine which bytes have special conditions. 0: If this bit is 0, the ESCC channel generates the receive character available interrupt on every received character, regardless of Special Receive Conditions.

Bit Number	Field	R/W	Reset Value	Description
2	Auto $\overline{\text{RTS}}$ Pin Deactivation	W	0	**Auto $\overline{\text{RTS}}$ Pin Deactivation** This bit controls the timing of the de-assertion of the $\overline{\text{RTS}}$ pin. 1: If the ESCC is programmed for SDLC mode, Flag-On-Underrun (WR10 Bit 2 is 0), this bit is set, and the RTS bit is reset, $\overline{\text{RTS}}$ is de-asserted automatically at the last bit of the closing flag, triggered by the rising edge of the Transmit Clock. 0: If this bit is reset, the $\overline{\text{RTS}}$ pin follows the state programmed in WR5 bit 1.
1	Automatic EOM Reset	W	0	**Automatic EOM Reset** If this bit is set, the ESCC channel automatically resets the Tx Underrun/EOM latch and presets the transmit CRC generator to its programmed preset state (per values set in WR5 Bit 2 and WR10 Bit 7). Therefore, it is not necessary to issue the Reset Tx Underrun/EOM latch command when this feature is enabled. If this bit is reset, the command is required between the time the first character of a frame is written to the TxFIFO, and when the end-of-frame underrun occurs. 1: Automatic EOM Reset is enabled. 0: Automatic EOM Reset is disabled.
0	Automatic Tx SDLC Flag	W	0	**Automatic Tx SDLC Flag** If this bit is set, the ESCC automatically transmits an SDLC flag before transmitting data. This procedure removes the requirement to reset the Mark Idle bit (WR10 Bit 3) before writing data to the transmitter, or having to enable the transmitter before writing data to the Transmit FIFO. Also, this feature enables a transmit data write before enabling the transmitter If this bit is reset, operation is identical to that of the SCC. 1: Automatic TX SDLC Flag is enabled. 0: Automatic TX SDLC Flag is disabled.

Transmit Buffer Register (WR8)

WR8 is the transmit buffer register.

Master Interrupt Control Register (WR9)

WR9 is the Master Interrupt Control register and contains the Reset command bits. The Interrupt control bits can be programmed at the same time as the Reset command.

7	6	5	4	3	2	1	0
Reset Command Bits		Software INTACK Enable	Status High/Low	Master Interrupt Enable	Disable Lower Chain Control Bit	No Vector Select Bit	Vector Includes Status Control Bit
W		W	W	W	W	W	W
1		0	0	0	0	X	X

Bit Number	Field	R/W	Reset Value	Description
7..6	Reset Comman d Bits		1	**Reset Command Bits** Together, these bits select one of the reset commands for the ESCC channel. Setting bit 7 to 1 disables both the receiver and the transmitter in the channel; forces TxD marking, forces the modem control signals High, resets all IPs and IUSs and disables all interrupts. Four extra PCLK cycles must be allowed beyond the usual cycle time after any of the reset commands is issued before any additional commands or controls are written to the ESCC channel. 00**: Null Command.** This command has no effect. It is used when a write to WR9 is necessary for some reason other than a Reset command. 01: **Reserved.** Do not program. 10**: Channel Reset Command.** Issuing this command causes a channel reset to be performed. 11: **Force Hardware Reset Command.** The effects of this command are identical to those of a hardware reset, except that the Shift Right/Shift Left bit is not changed and the MIE, Status High/Status Low and DLC bits take the programmed values that accompany this command.
5	Soft-ware Interrupt Acknowl edge Control Bit	W	0	**Software Interrupt Acknowledge Control Bit** If Bit 5 is set, reading Read Register 2 (RR2) results in an interrupt acknowledge cycle being executed internally. Like a hardware INTACK cycle, a software acknowledge causes the INT pin to return High, the IEO pin to go Low, and sets the IUS latch for the highest priority interrupt pending.

<table>
<tr><th>Bit
Number</th><th>Field</th><th>R/W</th><th>Reset
Value</th><th>Description</th></tr>
<tr><td>4</td><td>Status High/ Status Low Control Bit</td><td>W</td><td>0</td><td><b>Status High/Status Low Control Bit</b>
This bit controls which vector bits the ESCC channel modifies to indicate status.
1: The ESCC channel modifies bits 5 and 6 according to the table below.
0: The ESCC channel modifies bits 2 and 1. This bit controls status in both the vector returned during an interrupt acknowledge cycle and the status in RR2. This bit is reset by a hardware reset
<table>
<tr><th>V2</th><th>V1</th><th></th></tr>
<tr><th>V5</th><th>V6</th><th>Status High/Low</th></tr>
<tr><td>0</td><td>0</td><td>Transmit Buffer Empty</td></tr>
<tr><td>0</td><td>1</td><td>External/Status Change</td></tr>
<tr><td>1</td><td>0</td><td>Receive Char. Available</td></tr>
<tr><td>1</td><td>1</td><td>Special Receive Condition</td></tr>
</table></td></tr>
<tr><td>3</td><td>Master Interrupt Enable</td><td>W</td><td>0</td><td><b>Master Interrupt Enable</b>
This bit is 1 to globally enable interrupts, and cleared to 0 to disable interrupts. Clearing this bit to 0 forces the IEO pin to follow the state of the IEI pin unless there is an IUS bit set in the ESCC channel. This bit is reset by a hardware reset.
1: Interrupts enabled
0: Interrupts disabled</td></tr>
<tr><td>2</td><td>Disable Lower Chain Control Bit</td><td>W</td><td>0</td><td><b>Disable Lower Chain Control Bit</b>
The Disable Lower Chain bit is used by the CPU to control the interrupt daisy chain. Setting this bit to 1 forces the IEO output of the ESCC channel Low, preventing lower priority devices on the daisy chain from requesting interrupts. This bit is reset by a hardware reset.
1: IEO output of the ESCC channel is Low
0: IEO output of the ESCC channel is High</td></tr>
</table>

Bit Number	Field	R/W	Reset Value	Description
1	No Vector Select Bit	W	X	**No Vector Select Bit** The No Vector bit controls whether or not the ESCC channel responds to an interrupt acknowledge cycle, by placing a vector on the data bus if the ESCC channel is the highest priority device requesting an interrupt. If this bit is set, no vector is returned; that is, AD7..AD0 remains 3-stated during an interrupt acknowledge cycle, even if the ESCC channel is the highest priority device requesting an interrupt. 1: No Vector is returned 0: Vector returned
0	Vector Includes Status Control Bit	W	X	**Vector Includes Status Control Bit** The Vector Includes Status Bit controls whether or not the ESCC channel includes status information in the vector it places on the bus in response to an interrupt acknowledge cycle. If this bit is set, the vector returned is variable, with the variable field depending on the highest priority IP that is set. The table located in Bit 4, above, describes the encoding of the status information. This bit is ignored if the No Vector (NV) bit is set. 1: Includes status information 0: Do not include status information

Miscellaneous Tx/Rx Control Bits Register (WR10)

WR10 contains miscellaneous control bits for both the receiver and the transmitter. With the Extended Read option enabled, this register may be read as RR11.

7	6	5	4	3	2	1	0
CRC Presets 1s and 0s	Data Encoding Select		Go Active On Poll Control	Mark/Flag Idle Control	Abort/Flag On Underrun Select	Loop Mode Control	6 Bit/8 Bit Sync Select
W	W		W	W	W	W	W
0	0		0	0	0	0	0

Bit Number	Field	R/W	Reset Value	Description
7	CRC Presets 1s/0s	W	0	**CRC Presets 1s/0s** This bit specifies the initialized condition of the receive CRC checker and the transmit CRC generator. If this bit is 1, the CRC generator and checker are preset to all 1s. If this bit is 0, the CRC generator and checker are preset to all 0s. Either option can be selected with either CRC polynomial. In SDLC mode, the transmitted CRC is inverted before transmission, and the received CRC is checked against the bit pattern 0001110100001111. This bit is reset by a channel or hardware reset. This bit is ignored in Asynchronous mode. 1: The CRC generator and checker are preset to all 1s. 0: The CRC generator and checker are preset to all 0s
6–5	Data Encoding Select Bits	W	0	**Data Encoding Select Bits** These bits control the coding method used for both the transmitter and the receiver, as described in the table below. All of the clocking options are available for all coding methods. The DPLL in the ESCC is useful for recovering clocking information in NRZI and FM modes. Any coding method can be used in X1 mode. A hardware reset forces NRZ mode. Waveforms for the various modes is illustrated in Figure 28. 00: NRZ 01: NRZI 10: FM1 (transition equals 1) 11: FM0 (transition equals 0)

Bit Number	Field	R/W	Reset Value	Description
4	Go Active On Poll Control Bit	W	0	**Go Active On Poll Control Bit** When Loop mode is first selected during SDLC operation, the ESCC channel connects RxD to TxD with only gate delays in the path. The ESCC channel does not go on-loop and insert the 1-bit delay between RxD and TxD until this bit has been set and an EOP received. When the ESCC channel is on-loop, the transmitter does not go Active unless this bit is set at the time an EOP is received. The ESCC channel examines this bit whenever the transmitter is Active in SDLC Loop mode and is sending a flag. If this bit is set at the time the flag is leaving the Transmit Shift register, another flag or data byte (if the transmit buffer is full) is transmitted. If the Go Active On Poll bit is not set at this time, the transmitter finishes sending the flag and reverts to the 1-Bit Delay mode. Thus, to transmit only one response frame, this bit must be reset after the first data byte is sent to the transmitter, but before the CRC has been transmitted. If the bit is not reset before the CRC is transmitted, extra flags are sent, slowing down response time on the loop. If this bit is reset before the first data for a frame is written, the ESCC channel completes the transmission of the present flag and reverts to the 1-Bit Delay mode. After gaining control of the loop, the ESCC channel is not able to transmit again until a flag and another EOP are received. Set this bit only upon receipt of a poll frame, to ensure that the ESCC channel does not go on-loop without the CPU noticing it. In synchronous modes other than SDLC with the Loop Mode bit set, this bit is set before the transmitter goes active in response to a received sync character. This bit is always ignored in Asynchronous mode and Synchronous modes unless the Loop Mode bit is set. This bit is reset by a channel or hardware reset. 1: Go Active On Poll Control Bit enabled. 0: Go Active On Poll Control Bit disabled.

Bit Number	Field	R/W	Reset Value	Description
3	Mark/ Flag Idle Line Control Bit	W	0	**Mark/Flag Idle Line Control Bit** This bit affects only SDLC operation and is used to control the idle line condition. If this bit is 1, the transmitter sends continuous 1s after the closing flag of a frame. The Idle line condition is selected byte by byte, that is, either a flag or eight 1s are transmitted. If this bit is 0, the transmitter send flags as an idle line. The primary station in an SDLC loop must be programmed for Mark Idle to create the EOP sequence. Mark Idle must be deselected at the beginning of a frame before the first data is written to the ESCC channel, so that an opening flag is transmitted. This bit is ignored in Loop mode, but the programmed value takes effect upon exiting the Loop mode. This bit is reset by a channel or hardware reset. With the Automatic TX SDLC Flag mode enabled (WR7′ Bit 0 is 1), this bit can be left as Mark Idle. The transmitter sends an opening flag automatically, as well as sending a closing flag followed by mark idle after the frame transmission is completed. 1: Transmitter sends continuous 1s after the closing flag of a frame. 0: Sends flags as an Idle line

Bit Number	Field	R/W	Reset Value	Description
2	Abort/ Flag On Underrun Select Bit	W	0	**Abort/Flag On Underrun Select Bit** This bit affects only SDLC operation and is used to control how the transmitter responds to a Transmit Underrun Condition. If this bit is 1 and an underrun occurs, the transmitter sends an abort and a flag instead of a CRC. If this bit is reset, the transmitter sends a CRC on a transmit underrun. At the beginning of this 16-bit transmission, the Transmit Underrun/EOM bit is set, causing an External/Status interrupt. The CPU uses this status, along with the byte count from memory or the DMA, to determine whether the frame must be retransmitted. To start the next frame, a Transmit Buffer Empty interrupt occurs at the end of this 16-bit transmission. If both this bit and the Mark/Flag Idle bit are 1, all 1s are transmitted after the transmit underrun. Set this bit after the first byte of data for a frame is sent to the ESCC channel and reset immediately after the last byte of data, terminating the frame properly with CRC and a flag. This bit is ignored in Loop mode, but the programmed value is active upon exiting Loop mode. This bit is reset by a channel or hardware reset. 1: Transmitter send an abort and a flag 0: Transmitter sends a CRC
1	Loop Mode Control Bit	W	0	**Loop Mode Control Bit** In SDLC mode, the initial set condition of this bit forces the ESCC channel to connect TxD to RxD and to begin searching the incoming data stream so that it can go on loop. All bits pertinent to SDLC mode operation in other registers must be set before this mode is selected. The transmitter and receiver must not be enabled until after this mode has been selected. As soon as the Go Active On Poll bit is set and an EOP is received, the ESCC channel goes on-loop. If this bit is reset after the ESCC channel goes on-loop, the device waits for the next EOP to go off-loop.

Bit Number	Field	R/W	Reset Value	Description
1 (Cont.)				In synchronous modes, the ESCC channel uses this bit, along with the Go Active On Poll bit, to synchronize the transmitter to the receiver. The receiver must not be enabled until after this mode is selected. The TxD pin is held marking when this mode is selected unless a break condition is programmed. The receiver waits for a sync character to be received and then enables the transmitter on a character boundary. The break condition, if programmed, is removed. This mode works properly with sync characters of 6, 8, or 16 bits. This bit is ignored in Asynchronous mode and is reset by a channel or hardware reset.
0	6-Bit/ 8-Bit Sync Select Bit	W	0	**6-Bit/8-Bit Sync Select Bit** This bit is used to select a special case of synchronous modes. If this bit is 1 in Monosync mode, the receiver and transmitter sync characters are 6 bits long instead of the usual 8. If this bit is set 1 in Bisync mode, the received sync is 12 bits and the transmitter sync character remains 16 bits long. This bit is ignored in SDLC and Asynchronous modes. This bit is reset by a channel or hardware reset. 1: In Monosync mode, Rx and Tx sync characters are 6 bits long. In Bisync mode, the Rx bits are12 bits: The Tx bits are 16 0: In Monosync mode, Rx and Tx sync characters are 8 bits long

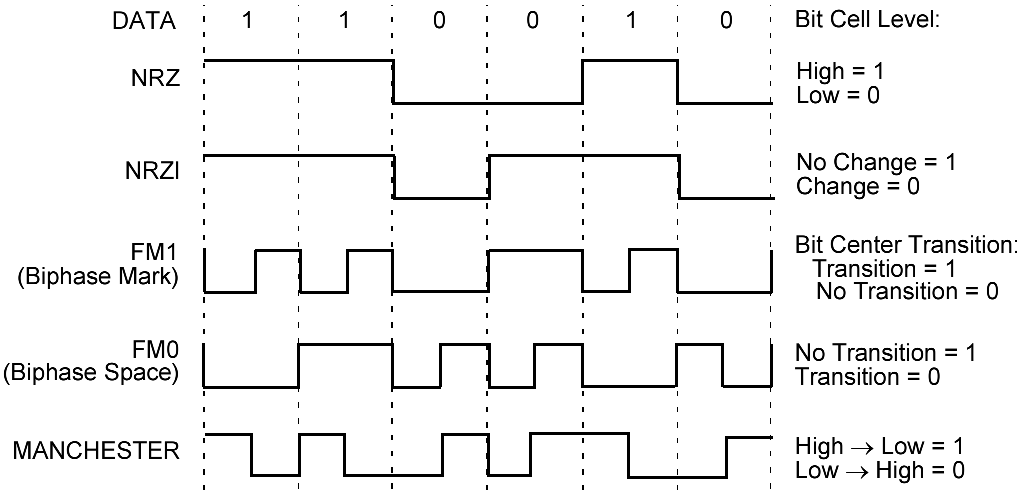


Figure 28. Data Encoding Waveforms

Clock Mode Control Register (WR11)

`WR11` is the Clock Mode Control register. The bits in this register control the sources of receive and transmit clocks, the type of signal on the $\overline{\text{RTxC}}$ pin, and the direction of the $\overline{\text{TRxC}}$ pin.For additional information, refer to the section "Clock Selection" on page 253.

7	6	5	4	3	2	1	0
Reserved	Receiver Clock		Transmit Clock		TRxC Pin	$\overline{\text{TRxC}}$ Output	
	W		W		W	W	
	0	0	0	1	0	0	0

Bit Number	Field	R/W	Reset Value	Description
7	Reserved			**Reserved**. Must be 0
6–5	Receiver Clock Select Bits	W	0	**Receiver Clock Select Bits** These bits determine the source of the receive clock as described in the table below. They do not interfere with any of the modes of operation in the ESCC channel, but simply control a multiplexer that selects the internal receive clock. A hardware reset forces the receive clock to come from the $\overline{\text{RTxC}}$ pin. 00: $\overline{\text{RTxC}}$ Pin 01: $\overline{\text{TRxC}}$ Pin 10: BRG Output 11: DPLL Output
4–3	Transmit Clock Select Bits	W	0	**Transmit Clock Select Bits** These bits determine the source of the transmit clock as described in the table below. They do not interfere with any of the modes of operation of the ESCC channel, but simply control a multiplexer that selects the internal transmit clock. The DPLL output that is used to feed the transmitter in FM modes lags the output of the DPLL used by the receiver by 90 degrees. This makes the received and transmitted bit cells occur simultaneously, neglecting delays. A hardware reset selects the $\overline{\text{TRxC}}$ pin as the source of the transmit clocks. 00: $\overline{\text{RTxC}}$ Pin 01: $\overline{\text{TRxC}}$ Pin 10: BRG Output 11: DPLL Output

Bit Number	Field	R/W	Reset Value	Description
2	TRxC Pin O/I Control Bit	W	0	**TRxC Pin O/I Control Bit** This bit determines the direction of the $\overline{\text{TRxC}}$ pin. If this bit is 1, the $\overline{\text{TRxC}}$ pin is an Output and carries the signal selected by Bits 1 and 0 of this register. However, if either the receive or the transmit clock is programmed to come from the $\overline{\text{TRxC}}$ pin, $\overline{\text{TRxC}}$ is an Input, regardless of the state of this bit. The $\overline{\text{TRxC}}$ pin is also an input if this bit is 0. A hardware reset forces this bit to 0. 1: $\overline{\text{TRxC}}$ pin is an Output. 0: $\overline{\text{TRxC}}$ pin is an Input.
1..0	$\overline{\text{TRxC}}$ Output Source Select Bits	W	0	**TRxC Output Source Select Bits** These bits determine the signal to be echoed out of the ESCC channel via the $\overline{\text{TRxC}}$ pin as given in the table below. No signal is produced if $\overline{\text{TRxC}}$ has been programmed as the source of either the receive or the transmit clock. If $\overline{\text{TRxC}}$ O/I (bit 2) is 0, these bits are ignored. The DPLL signal that is echoed is the DPLL signal used by the receiver. Hardware reset sets the reserved value of these bits. 00: Reserved 01: Transmit Clock 10: BRG Output 11: DPLL Output (receive)

Lower Byte of Baud Rate Generator Time Constant (WR12)

`WR12` contains the less significant byte of the time constant for the Baud Rate Generator. The time constant can be changed at any time, but the new value does not take effect until the next time the time constant is loaded into the down counter. No attempt is made to synchronize the loading of the time constant into `WR12` and `WR13` with the clock driving the down counter. For this reason, it is advisable to disable the Baud Rate Generator while the new time constant is loaded into `WR12` and `WR13`. This action prevents a load of the down counter between the writing of the upper and lower bytes of the time constant.

The formula for determining the appropriate time constant (Tc) for a given bit rate is described below, with the desired rate in bits per second. Clock Mode in the formula is `1` for divide-by-one modes or 16, 32, or 64 for asynchronous modes. This formula is used because the counter decrements from N down to 0 plus 1 cycle for reloading the time constant. This information is then fed to a toggle flip-flop to make the output a square wave.

$$\text{Tc} = \frac{\text{BRG Clock Frequency}}{2 \times (\text{Desired rate}) \times (\text{Clock mode})} - 2$$

7	0
Time Constant 7..0	
W	
X	

Bit Number	Field	R/W	Reset Value	Description	
7		W	X	TC7	Lower Byte of Time Constant
6		W	X	TC6	
5		W	X	TC5	
4		W	X	TC4	
3		W	X	TC3	
2		W	X	TC2	
1		W	X	TC1	
0		W	X	TC0	

Upper Byte of Baud Rate Generator Time Constant (WR13)

`WR13` contains the more significant byte of the time constant for the Baud Rate Generator.

7 … 0
Time Constant 15..8
W
X

Bit Number	Field	R/W	Reset Value	Description	
7		W	X	TC8	Upper Byte of Time Constant
6		W	X	TC9	
5		W	X	TC10	
4		W	X	TC11	
3		W	X	TC12	
2		W	X	TC13	
1		W	X	TC14	
0		W	X	TC15	

Miscellaneous Control Bits Register (WR14)

WR14 contains miscellaneous control bits.

7		5	4	3	2	1	0
	DPLL		Local Loop Back	Auto Echo	Reserved	BRG Source	BRG Enable
	W		W	W	W	W	W
X	X	1	1	0	0	0	0

Bit Number	Field	R/W	Reset Value	Description
7..5		W	X, X, 1	**Digital Phase-Locked Loop Command Bits** These three bits encode the eight commands for the Digital Phase-Locked Loop. A channel or hardware reset disables the DPLL, resets the missing clock latches, sets the source to the $\overline{\text{RTxC}}$ pin and selects NRZI mode. The Enter Search Mode command enables the DPLL after a reset. 000**:** Null Command. This command has no effect on the DPLL. 001**:** Enter Search Mode Command. Issuing this command causes the DPLL to enter the Search mode, where the DPLL searches for a locking edge in the incoming data stream. The action taken by the DPLL upon receipt of this command depends on the operating mode of the DPLL.

Bit Number	Field	R/W	Reset Value	Description
7..5 (Cont.)				In NRZI mode, the output of the DPLL is High while the DPLL is waiting for an edge in the incoming data stream. After the Search mode is entered, the first edge the DPLL sees is assumed to be a valid data edge, and the DPLL begins the clock recovery operation from that point. The DPLL clock rate must be 32x the data rate in NRZI mode. When leaving the Search mode, the first sampling edge of the DPLL occurs 16 of these 32x clocks after the first data edge, and the second sampling occurs 48 of these 32x clocks after the first data edge. Beyond this point, the DPLL begins normal operation, adjusting the output to remain in sync with the incoming data. In FM mode, the output of the DPLL is Low while the DPLL is waiting for an edge in the incoming data stream. The first edge the DPLL detects is assumed to be a valid clock edge. For this to be the case, the line must contain only clock edges; that is, with FM1 encoding, the line must be continuous 0s. With FM0 encoding the line must be continuous 1s, whereas Manchester encoding requires alternating 1s and 0s on the line. The DPLL clock rate must be 16 times the data rate in FM mode. The DPLL output causes the receiver to sample the data stream in the nominal center of the two halves of the bit to decide whether the data is a 1 or a 0. After this command is issued, as in NRZI mode, the DPLL starts sampling immediately after the first edge is detected. (In FM mode, the DPLL examines the clock edge of every other half-bit cell to decide what correction must be made to remain in sync.) If the DPLL does not see an edge during the expected window, the one clock missing bit in RR10 is set. If the DPLL does not see an edge after two successive attempts, the two clocks missing bits in RR10 is set and the DPLL automatically enters the Search mode. This command resets both Clock Missing latches.

Bit Number	Field	R/W	Reset Value	Description
				010**: Reset Clock Missing Command.** Issuing this command disables the DPLL, resets the clock missing latches in RR10, and forces a continuous Search mode state. 011**: Disable DPLL Command.** Issuing this command disables the DPLL, resets the clock missing latches in RR10, and forces a continuous Search mode state. 100**: Set Source to BRG Command.** Issuing this command forces the clock for the DPLL as the output of the BRG. 101**: Set Source to RTxC Command.** Issuing the command selects the clock for the DPLL as the RTxC pin. This mode is selected by a channel or hardware reset. 110**: Set FM Mode Command.** This command conditions the DPLL to operate in the FM mode and is used to recover the clock from FM or Manchester-Encoded data. (Manchester is decoded by placing the receiver in NRZ mode while the DPLL is in FM mode.) 111**: Set NRZI Mode Command.** Issuing this command conditions the DPLL to operate in the NRZI mode. This mode is selected by a hardware or channel reset.
4	Local Loopback Select Bit		1	**Local Loopback Select Bit** Setting this bit to 1 selects the Local Loopback mode of operation. In this mode, the internal transmitted data is routed back to the receiver, and to the TxD pin. The /CTS and /DCD inputs are ignored as enables in Local Loopback mode, even if auto enable is selected. (If so programmed, transitions on these inputs still cause interrupts.) This mode works with any Transmit/Receive mode except Loop mode. For meaningful results, the frequency of the transmit and receive clocks must be the same. This bit is reset by a channel or hardware reset. 1: Selects the Local Loopback mode. 0: Does not select Local Loopback mode.

Bit Number	Field	R/W	Reset Value	Description
3	Auto Echo Select Bit		0	**Auto Echo Select Bit** Setting this bit to `1` selects the Auto Echo mode of operation. In this mode, the TxD pin is connected to RxD as in Local Loopback mode, but the receiver still listens to the RxD input. Transmitted data is never seen inside or outside the ESCC in this mode, and $\overline{CTS}$ is ignored as a transmit enable. This bit is reset by a channel or hardware reset. 1: Selects the Auto Echo mode. 0: Does not select the Auto Echo mode.
2	Reserved			**Reserved**. Must be `0`.
1	Baud Rate Generator Source Select Bit		0	**Baud Rate Generator Source Select Bi** This bit selects the source of the clock for the Baud Rate Generator. Hardware reset clears this bit to `0`, selecting the $\overline{RTxC}$ pin as the clock source for the BRG. 1: The clock for the Baud Rate Generator is PHI or PHI/2, depending on bit 3 of the System Configuration Register. 0: The Baud Rate Generator clock comes from the $\overline{RTxC}$ pin.
0	Baud Rate Generator Enable		0	**Baud Rate Generator Enable** This bit controls the operation of the BRG. This bit allows the command to be synchronized. However, when the bit is switched to `0`, disabling is immediate. This bit is reset by a hardware reset. 0: The counter in the BRG is disabled for counting when this bit is switched from `0` to `1`. 1: The counter in the BRG is enabled. The change is not reflected by the output of the BRG for two counts of the counter.

External/Status Interrupt Control Register (WR15)

`WR15` is the External/Status Source Control register. If the External/Status interrupts are enabled as a group via `WR1`, bits in this register control which External/Status conditions cause an interrupt. Only the External/Status conditions that occur after the controlling bit is 1 cause an interrupt. This condition is true, even if an External/Status condition is pending at the time the bit is set.

7	6	5	4	3	2	1	0
Break/ Abort Interrupt Enable	Transmit Underrun/ EOM Interrupt Enable	Clear to Send Interrupt Enable	Sync/Hunt Interrupt Enable	Data Carrier Detect Interrupt Enable	Status FIFO Enable Control	Zero Count Interrupt Enable	Select Write Register WR7 Prime
W	W	W	W	W	W	W	W
1	1	1	1	1	0	0	0

Bit Number	Field	R/W	Reset Value	Description
7	Break/ Abort Interrupt Enable	W	1	**Break/Abort Interrupt Enable** This bit is set by a channel or hardware reset. 1: A change in the Break/Abort status of the receiver causes an External/Status interrupt. 0: No External/Status interrupt is generated
6	Transmit Underrun/EOM Interrupt Enable	W	1	**Transmit Underrun/EOM Interrupt Enable** This bit is reset by a channel or hardware reset. 1: A change of state by the Tx Underrun/EOM latch in the transmitter causes an External/Status interrupt. 0: No External/Status interrupt is generated
5	Clear to Send Interrupt Enable	W	1	**Clear to Send Interrupt Enable** This bit is set by a channel or hardware reset. 1: A change of state on the $\overline{CTS}$ pin causes an External/Status Interrupt. 0: No External/Status interrupt is generated

Bit			Reset	
Number	**Field**	**R/W**	**Value**	**Description**
4	Sync/ Hunt Interrupt Enable	W	1	**Sync/Hunt Interrupt Enable** This bit is set by a channel or hardware reset. 1: A change of state on the $\overline{\text{SYNC}}$ pin causes an External/ Status interrupt in Asynchronous mode, and a change of state in the Hunt bit in the receiver causes and External/ Status interrupt in synchronous modes. 0: No External/Status interrupt is generated
3	Data Carrier Detect Interrupt Enable	W	1	**Data Carrier Detect Interrupt Enable** 1: A change of state on the $\overline{\text{DCD}}$ pin causes an External/ Status interrupt. This bit is set by a channel or hardware reset. 0: No External/Status interrupt is generated
2	Status FIFO Enable Control Bit	W	0	**Status FIFO Enable Control Bit** This bit is reset to 0 by a channel or hardware reset. For details on this function, refer to the section SDLC Frame Status FIFO. 1: In the SDLC/HDLC Mode, five bits of status (from Read Register 1, including: Residue, Overrun, and CRC Error) and fourteen bits of byte count are held in the Status FIFO until read. Status information for up to ten frames can be stored. 0: If this bit is reset or if the ESCC channel is not in the SDLC/HDLC Mode, the FIFO is not operational and status information read reflects the current status only.

Bit Number	Field	R/W	Reset Value	Description
1	Zero Count Interrupt Enable	W	0	**Zero Count Interrupt Enable** This bit is reset by a channel or hardware reset. 1: An External/Status interrupt is generated whenever the counter in the Baud Rate Generator reaches 0. 0: No External/Status interrupt is generated
0	Select Write Register WR7 Prime	W	0	**Select Write Register WR7'** This bit is cleared to 0 by a hardware or software reset. If the extended read option is enabled, WR7′ can be read as RR14. 1: Writes to the WR7 address are made to WR7′. 0: Writes to the WR7 address are made to WR7.

Read Registers

The status of these registers is continually changing and depends on the mode of communication, received and transmitted data, and the manner in which this data is transferred to and from the CPU. The following description details the bit assignments for each register.

Transmit/Receive Buffer Status and External Status Register (RR0)

Read Register 0 (RR0) contains the status of the receive and transmit FIFOs. RR0 also contains the status bits for the six sources of External/ Status interrupts. The ESCC channel latches the contents of RR0 during read transactions for this register. The latch is released on the rising edge of $\overline{\text{RD}}$.

7	6	5	4	3	2	1	0
Break/ Abort Status	Transmit Underrun/ EOM Status	Clear to Send Pin Status	Sync/Hunt Status	Data Carrier Detect Status	Tx Buffer Empty Status	Zero Count Status	Rx Character Available
R	R	R	R	R	R	R	R
X	1	X	X	X	1	0	0

Bit Number	Field	R/W	Reset Value	Description
7	Break/ Abort Status	R	X	**Break/Abort Status** This character must be read and discarded. In SDLC mode, this bit is set by the detection of an Abort sequence (seven or more `1`s), then reset automatically at the termination of the Abort sequence. In either case, if the Break/Abort IE bit is set, an External/ Status interrupt is initiated. Unlike the remainder of the External/Status bits, both transitions are guaranteed to cause an External/Status interrupt, even if another External/Status interrupt is pending at the time these transitions occur. This procedure is necessary because Abort or Break conditions may not persist. 1: In the Asynchronous mode, this bit is set when a Break sequence (null character plus framing error) is detected in the receive data stream. 0: This bit is reset when the sequence is terminated, leaving a single null character in the Receive FIFO.

Bit Number	Field	R/W	Reset Value	Description
6	Transmit Underrun/EOM Status	R	1	**Transmit Underrun/EOM Status** This bit is set by a channel or hardware reset when the transmitter is disabled or a Send Abort command is issued. This bit is reset by the reset Tx Underrun/EOM Latch command in `WR0`. When the Transmit Underrun occurs, this bit is set and causes an External/Status interrupt (if the Tx Underrun/EOM IE bit is set). Only the 0-to-1 transition of this bit causes an interrupt. This bit is always `1` in Asynchronous mode, unless a Reset Tx Underrun/EOM Latch command has been erroneously issued. In this case, the Send Abort command can be used to set the bit to `1` and at the same time cause an External/Status interrupt. 1: This value is determined by a channel or hardware reset when the transmitter is disabled or a Send Abort command is issued. 0: This value is determined by the reset Tx Underrun/EOM Latch command in `WR0`
5	Clear to Send Pin Status	R	X	**Clear to Send Pin Status** If the CTS IE bit in `WR15` is set, this bit indicates the state of the $\overline{\text{CTS}}$ pin (the last time any of the enabled External/Status bits changed) while no interrupt is pending, latches the state of the $\overline{\text{CTS}}$ pin and generates an External/Status interrupt. Any odd number of transitions on the $\overline{\text{CTS}}$ pin while another External/Status on Interrupt is pending, also causes another External/Status interrupt condition. If the CTS IE bit is reset, this bit reports the current unlatched state of the $\overline{\text{CTS}}$ pin. 1: Indicates the state of the $\overline{\text{CTS}}$ pin the last time the External/Status bits changed. 0: Indicates the current unlatched state of the $\overline{\text{CTS}}$ pin.

Bit Number	Field	R/W	Reset Value	Description
4	Sync/ Hunt Status	R	X	**Sync/Hunt Status** In the Monosync and Bisync Receive modes, the Sync/Hunt Status bit is initially set to 1 by the Enter Hunt Mode command. The Sync/Hunt Status bit is reset when the ESCC established character synchronization. Both transitions cause External/Status interrupts if the Sync/Hunt IE bit is set. When the CPU detects the end of message or the loss of character synchronization, the Enter Hunt Mode command must be issued to set the Sync/Hunt bit and cause an External/Status interrupt. In the SDLC modes, the Sync/Hunt bit is initially set by the Enter Hunt Mode command or when the receiver is disabled. It is reset when the opening flag of the first frame is detected by the receiver. An External/Status interrupt is also generated if the Sync/Hunt IE bit is set. Unlike the Monosync and Bisync modes, after the Sync/Hunt bit is reset in SDLC mode, it does not need to be set when the end of the frame is detected. The receiver automatically maintains synchronization. The only way the Sync/Hunt bit is set again is by the Enter Hunt Mode command or by disabling the receiver. 1: The default value determined by the Enter Hunt Mode command. 0: The value determined the ESCC establishes character synchronization

Bit Number	Field	R/W	Reset Value	Description
3	Data Carrier Detect Status	R	X	**Data Carrier Detect Status** If the DCD IE bit in `WR15` is set, this bit indicates the state of the $\overline{\text{DCD}}$ pin the last time any of the enabled External/ Status bits changed. Any transition on the $\overline{\text{DCD}}$ pin, while no interrupt is pending, latches the state of the $\overline{\text{DCD}}$ pin and generates an External/Status interrupt. If the DCD IE is reset, this bit merely reports the current, unlatched state of the $\overline{\text{DCD}}$ pin. 1: Indicates the state of the DCD pin after External/Status bit change. 0: Indicates the current, unlatched state of the DCD pin.
2	Tx Buffer Empty Status	R	1	**Tx Buffer Empty Status** This bit is `1` when the transmit FIFO is empty. It is reset while the CRC is sent in a synchronous or SDLC mode and while the transmit FIFO is not empty. The bit is reset when a character is loaded into the transmit FIFO. The status of this bit is not related to the Transmit Interrupt Status or the state of `WR7′` Bit 5, but it indicates the status of the entry location of the Transmit FIFO and whether more data can be written. This bit is `1` after a hardware or channel reset. 1: Transmit FIFO is empty. 0: Transmit FIFO is not empty.

Bit Number	Field	R/W	Reset Value	Description
1	Zero Count Status	R	0	**Zero Count Status** If the Zero Count Interrupt Enable bit is set in WR15, this bit is 1 while the counter in the Baud Rate Generator is at the count of 0. If there is no other External/Status interrupt condition pending at the time this bit is set, an External/ Status interrupt is generated. However, if there is another External/Status interrupt pending at this time, no interrupt is initiated until interrupt service is complete. If the Zero Count condition does not persist beyond the end of the interrupt service routine, no interrupt is generated. This bit is not latched High, even though the other External/ Status latches close as a result of the Low-to-High transition on ZC. The interrupt routine must check the other External/ Status conditions for changes. If none changed, ZC was the source. In polled applications, check the IP bit in RR3 for a status change and then proceed as in the interrupt service routine. 1: An External/Status interrupt is generated. 0: No External/Status interrupt is generated.
0	Receive Character Available	R	0	**Receive Character Available** A channel or hardware reset empties the receive data FIFO. The status of this bit is independent of WR7′ Bit 3. 1: At least one character is available in the receive data FIFO. 0: The receive data FIFO is completely empty.

Special Receive Condition Status Register (RR1)

RR1 contains the Special Receive Condition status bits and the residue codes for the l-field in SDLC mode. If the SDLC Status FIFO is enabled, some of these bits reflect the oldest entry in the FIFO.

7	6	5	4	3		1	0
End of Frame	CRC Frequency Error	Rx Overrun Error	Parity Error	Residue Code			All Sent
R	R	R	R	R			R
0	0	0	0	0	1	1	X

Bit Number	Field	R/W	Reset Value	Description
7	End of Frame (SDLC) Status	R	0	**End of Frame (SDLC) Status** This bit is used only in SDLC mode and indicates that a valid closing flag has been received and that the CRC Error bit and residue codes are valid. This bit is reset by issuing the Error Reset command. It is also updated by the first character of the following frame. This bit is reset in any mode other than SDLC.
6	CRC/ Framing Error Status	R	0	**CRC/Framing Error Status** If a framing error occurs in Asynchronous mode, this bit is set (and not latched) for the receive character in which the framing error occurred. Detection of a framing error adds an additional one-half bit to the character time so that the framing error is not interpreted as a new Start bit. In Synchronous and SDLC modes, this bit indicates the result of comparing the CRC checker to the appropriate check value. This bit is reset by issuing an Error Reset command, but the bit is never latched. Therefore, it is always updated when the next character is received. When used for CRC error status in Synchronous or SDLC modes, this bit is usually set because most bit combinations, except for a correctly completed message, result in a non-zero CRC. If the Status FIFO is enabled (refer to the descriptions for WR15 Bit 2 and RR7 Bits 7..6), this bit reflects the status stored at the exit location of the Status FIFO.

Bit Number	Field	R/W	Reset Value	Description
5	Receiver Overrun Error Status	R	0	**Receiver Overrun Error Status** This bit indicates that the Receive FIFO has overflowed. Only the character that has been written over is flagged with this error. When that character is read, the Error condition is latched until reset by the Error Reset command. Also, a Special Receive Condition vector is returned, caused by the overrun characters and all subsequent characters received until the Error Reset command is issued. If the Status FIFO is enabled (refer to the descriptions for `WR15` Bit 2 and `RR7` Bits 7..6), this bit reflects the status stored at the exit location of the Status FIFO.
4	Parity Error Status	R	0	**Parity Error Status** When parity is enabled, this bit is set for characters whose parity does not match the programmed sense (even/odd). This bit is latched so that once an error occurs, it remains set until the Error Reset command is issued. If the parity in Special Condition bit is set, a parity error causes a Special Receive Condition vector to be returned on the character containing the error and on all subsequent characters until the Error Reset command is issued.

Bit Number	Field	R/W	Reset Value	Description
3..1	Residue Code	R	1, 1, X	**Residue Code** In those cases in SDLC mode where the received I-Field is not an integral multiple of the character length, these three bits indicate the length of the I-Field and are meaningful only for the transfer in which the end of frame bit is set. This field is set to `011` by a channel or hardware reset and is forced to this state in Asynchronous mode. These three bits can leave this state only if SDLC is selected and a character is received. Figure 29 depicts the meaning of the Residue Codes for 5 through 8 bit characters. If the Status FIFO is enabled (refer to the descriptions for `WR15`, Bit 2 and `RR7`, Bits 7..6), these bits reflect the status stored at the exit location of the Status FIFO. I-Field bits are right-justified in all cases. Figure 30 describes the residue codes for no residue, that is, when the I-Field boundary lies on a character boundary
0	All Sent Status	R	X	**All Sent Status** In Asynchronous mode, this bit is set when all characters have completely cleared the transmitter pins. Most modems contain additional delays in the data path, which requires the modem control signals to remain active until after the data has cleared both the transmitter and the modem. This bit is always set in synchronous and SDLC modes

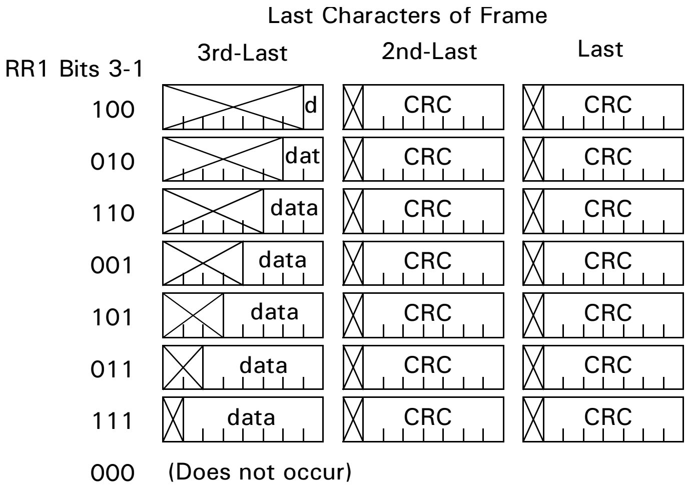


Figure 29. Residue Values vs. Last Characters of Frame, 7 Bits/Character

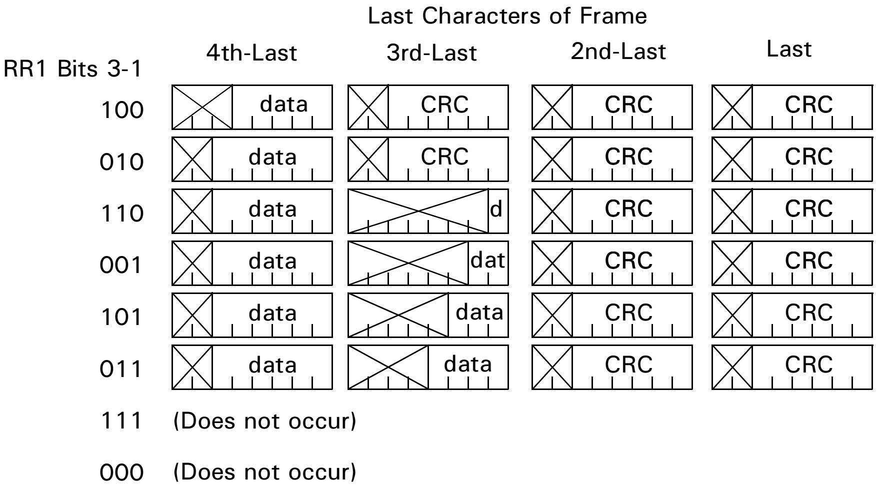


Figure 30. Residue Values vs. Last Characters of Frame, 6 Bits/Character

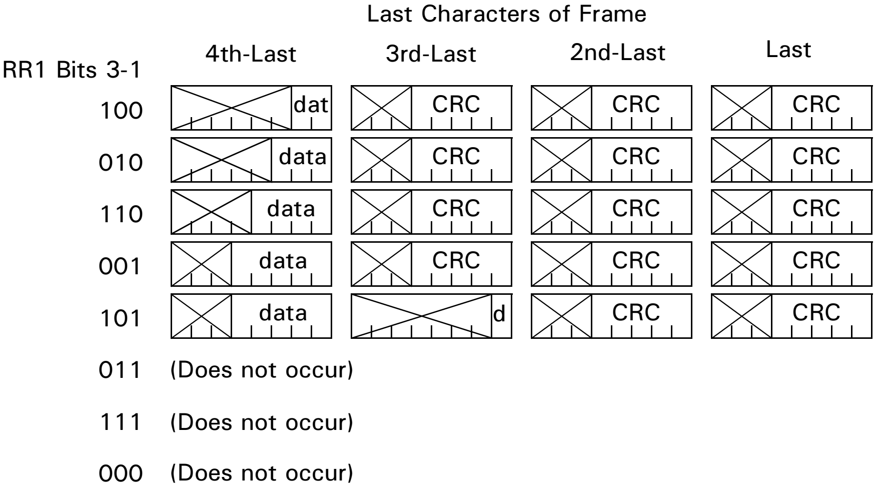


Figure 31. Residue Values vs. Last Characters of Frame, 5 Bits/Character

Interrupt Vector Status Register (RR2)

RR2 reflects the interrupt vector written into WR2, but includes status information in bits 1 and 2 or in bits 6 and 5, depending on the state of the Status High/Status Low bit in WR9 and independent of the state of the VIS bit in WR9. The vector is modified according to the table located in the explanation of the VIS bit in WR9. If no interrupts are pending, both of these status bits are 1.

Interrupt Pending Register (RR3)

RR3 is the Interrupt Pending register. The status of each of the Interrupt Pending bits in the ESCC channel is reported in this register. Unused bits are always returned as 0.

7	6	5	4	3	0
Reserved		RXRIP	TXIP	External/ Status IP	Reserved
		R	R	R	

Transmit/Receive Miscellaneous Parameters and Modes Register (RR4)

RR4 reflects the contents of WR4 provided the Extended Read option is enabled. Otherwise, this register returns an image of RR0.

Transmit Parameters and Controls Register (RR5)

RR5 reflects the contents of WR5 provided the Extended Read option is enabled. Otherwise, this register returns an image of RR1.

Least Significant Byte of Byte Count (RR6)

RR6 contains the least significant byte of the frame Byte Count (BC) that is currently at the top of the Status FIFO. This register is readable only if the FIFO is enabled; that is, WR15 Bit 2 is 1. Otherwise, this register is an image of RR2.

7	0
Byte Count 7..0	
R	
X	

Most Significant Byte of Byte Count Register (RR7)

RR7 contains the most significant six bits of the frame byte count that is currently at the top of the Status FIFO. Bit 7 is the FIFO Overflow Status and Bit 6 is the FIFO Data Available Status. This register is readable only if the FIFO is enabled; that is, WR15 Bit 2 is 1. Otherwise this register is an image of RR3. For the FIFO and byte count logic to operate, the registers must be read in the following order: RR7, RR6, RR1.

If the FIFO overflows, the FIFO and the FIFO Overflow Status bit are cleared by disabling and then re-enabling the FIFO through the FIFO control bit (WR15, Bit 2).

7	6	5	4	3	2	1	0
FIFO Overflow	FIFO Data Available	Byte Count 13..8					
R	R	R	R	R	R	R	R
X	X	X	X	X	X	X	X

Bit Number	Field	R/W	Reset Value	Description
7	FIFO Overflow Status	R	X	**FIFO Overflow Status** 1: FIFO has Overflowed. 0: Normal operation.
6	FIFO Data Available	R	X	**FIFOData Available** 1: Status Reads from FIFO. Status reads come from FIFO (FIFO is not empty). 0: Status Reads from ESCC. Status reads bypass FIFO (because FIFO is empty).
5	BC13		X	
4	BC12		X	
3	BC11		X	
2	BC10		X	
1	BC9		X	
0	BC8		X	

Receive Data Register (RR8)

RR8 is the Receive Data register.

Interrupt Vector Register (RR9)

RR9 reflects the contents of WR3 provided the Extended Read option has been enabled. Otherwise, it returns an image of RR13.

Miscellaneous Status Register (RR10)

RR10 contains some miscellaneous status bits. Unused bits are always 0.

7	6	5	4	3	1	0
1 Clock Missed	2 Clocks Misseed	Reserved	LoopSend	Reserved	On Loop	Reserved
R	R		R		R	
0	X		0		0	

Bit Number	Field	R/W	Reset Value	Description
7	One Clock Missing Status	R	0	**One Clock Missing Status** While operating in an FM mode, the DPLL sets this bit to 1 when it does not detect a clock edge on the incoming lines in the window where it is expected. This bit is latched until reset by a Reset Missing Clock or Enter Search Mode command in WR14. In the NRZI mode of operation and while the DPLL is disabled, this bit is always 0. 1: DPLL has not detected a clock edge on the incoming lines 0: This value is the default value of this bit.
6	Two Clocks Missing Status	R	X	**Two Clocks Missing Status** While operating in an FM mode, the DPLL sets this bit to 1 when it does not detect a clock edge in two successive bits. At the same time the DPLL enters the Search mode. This bit is latched until reset by a Reset Missing Clock or Enter Search Mode command in WR14 Bits 5..7. In the NRZI mode of operation and while the DPLL is disabled, this bit is always 0. 1: DPLL has not detected a clock edge in two successive bits. 0: This value is the default value of this bit.
5	Reserved			**Reserved.** Must be 0.

Bit Number	Field	R/W	Reset Value	Description
4	Loop Sending Status		0	**Loop Sending Status** This bit is 1 in SDLC Loop mode while the transmitter is in control of the Loop, that is, while the ESCC channel is actively transmitting on the Loop. This bit is reset (0) at all other times. This bit can be polled in SDLC mode to determine when the closing flag has been sent. 1: Transmitter is in control of the Loop in SDLC Loop mode. 0: This value is the default value of this bit.
3..2	Reserved			**Reserved.** Must be 0.
1	On Loop Status		0	**On Loop Status** This bit is 1 while the ESCC channel is actually On Loop in SDLC Loop mode. This bit is 1 in the X21 mode (Loop mode selected while in monosync) when the transmitter goes active. This bit is 0 at all other times. This bit can also be polled in SDLC mode to determine when the closing flag has been sent. 1: ESCC is On Loop in SDLC Loop mode. 0: This value is the default value of this bit.
0	Reserved			**Reserved.** Must be 0.

Miscellaneous Tx/Rx Control Bits Register (RR11)

RR11 reflects the contents of WR10 provided the Extended Read option has been enabled. Otherwise, this register returns an image of RR15.

Lower Byte of Baud Rate Generator Time Constant Register (RR12)

`RR12` returns the value stored in `WR12`, the lower byte of the time constant for the BRG..

7	0
Time Constant 7..0	
R	
X	

Bit Number	Field	R/W	Reset Value	Description	
7		R	X	TC7	Lower Byte of Time Constant
6		R	X	TC6	
5		R	X	TC5	
4		R	X	TC4	
3		R	X	TC3	
2		R	X	TC2	
1		R	X	TC1	
0		R	X	TC0	

Upper Byte of Baud Rate Generator Time Constant Register (RR13)

RR13 returns the value stored in WR13, the upper byte of the time constant for the BRG.

7	0
Time Constant 15..8	
	R
	X

Bit Number	Field	R/W	Reset Value	Description	
7		R	X	TC8	Upper Byte of Time Constant
6		R	X	TC9	
5		R	X	TC10	
4		R	X	TC11	
3		R	X	TC12	
2		R	X	TC13	
1		R	X	TC14	
0		R	X	TC15	

Contents of Write Register 7 Prime Register (RR14)

RR14 reflects the contents of WR7′ provided the Extended Read option is enabled. Otherwise, this register returns an image of RR10.

External/Status Interrupt Control Register (RR15)

`RR15` reflects the value stored in `WR15`, the External/Status IE bits. The two reserved bits must be `0`. Refer to “External/Status Interrupt Control Register (WR15)” on page 376.

7	6	5	4	3	2	1	0
Break/ Abort Interrupt Enable	Transmit Underrun/ EOM Interrupt Enable	Clear to Send Interrupt Enable	Sync/Hunt Interrupt Enable	Data Carrier Detect Interrupt Enable	Reserved	Zero Count Interrupt Enable	Reserved
W	W	W	W	W		W	
1	1	1	1	1		0	

Instruction Set

INTRODUCTION

This chapter describes the Z80185/Z80195 processor instruction set. To minimize the number of pages and redundant information, instructions differ only where the operands reside together. After introductory information, instructions are in alphabetical order. Each description includes the following:

- The top line of each includes the assembly language mnemonic and operands. These items identify the instruction. One instruction description may cover many different instructions. Operands are often given in alphabetic code that stands for many possible values.
- The Operation section indicates the function of the instruction in a symbolic form.
- The Format section indicates how the instruction is structured in memory. For instructions with more than one type of operand, separate formats are given for each type.
- The Description section contains a textural explanation of the instruction's use and the processing that occurs at executions.
- Timing indicates the number of machine cycles (M cycles) and minimum number of clock ticks (T states) the processor requires to execute the instruction. For instructions with more than one type of operand, separate timings may be given for each type.

The number of M cycles indicates only the number of read and write bus cycles that occur during the instruction fetching and execution. Internal idle times are not included in the number of M cycles. The minimum number of T states required to execute instructions are given separately for the Z80 and Z80185, and each total is followed by a breakdown of the number of clocks required for each M cycle and internal idle period (*n idle*). The Z80 timings are included for reference only. In some Z80 T-state breakdowns, short internal idle times may not be broken out from the M cycle that they precede or follow. I/O instructions for T states are given separately for registers in the Z180 core (those which are typically addressed 0-3FH) and for all other on-chip and off-chip registers, indicating that the processor automatically inserts one wait state.

- Flags indicate if and how the instruction affects the Flag register F'.
- An example is included for most instruction types.

OPERAND CODES

Table 23. Operand

Operand	Description
(aa)	(mn), (IX+d), (IY+d), or the 8-bit value of memory at the address in BC, DE, or HL.
cc	Condition code Z, NZ, C, NC, PE, PO, P, or M
cc'	Condition code Z, NZ, C, or NC
d	Signed 8-bit displacement, from an index register or the Program Counter (PC)
ee	16-Bit register BC, DE, HL, SP, IX, or IY
(IX+d), (IY+d)	The 8-bit value of memory at the address calculated by adding the value of index register IX or IY and the signed 8-bit displacement d given in the instruction
m	r or (HL) or (IX+d) or (IY+d)
mn	16-Bit immediate or literal address or value given in the instruction
(mn)	8-Bit value of memory at an address given in the instruction
n	8-Bit immediate or literal value given in the instruction

Table 23. Operand

Operand	Description
pp	16-Bit register BC, DE, HL, IX, IY, or the concatenation of the Accumulator A as the MSB and the Flag F as the LSB
qq	16-Bit register BC, DE, HL, or the concatenation of the Accumulator A as the MSB and the Flag F as the LSB
r	8-Bit register A, B, C, D, E, H, or L
rr	16-Bit register HL, IX, or IY
s	r or n or (HL) or (IX+d) or (IY+d)
ss	16-Bit register BC, DE, HL, or the SP
tt	Similar to ss, except that the encoded value that corresponds to HL indicates the destination register HL, IX, or IY

Z80 STATUS INDICATOR FLAGS

The Flag registers (F and F′) supply information to the user about the state of the Z80 at any given time. Figure 32 depicts the bit positions for each flag.

7	6	5	4	3	2	1	0
S	Z	X	N	X	P/V	N	C

Figure 32. Flags Bit Positions

Symbol	Field Name
C	Carry Flag
N	Add/Subtract
P/V	Parity/Overflow Flag
H	Half Carry Flag
Z	Zero Flag
S	Sign Flag
X	Not Used

Each of the two flag registers has 6 bits of status information that are set or cleared by CPU operations. (Bits 3 and 5 are not used.) Four of these bits (C, P/V, Z, and S) may be tested for use with conditional `JUMP`, `CALL`, or `RETURN` instructions. Two flags may not be tested (H, N) and are used for BCD arithmetic.

Carry Flag

The Carry Flag (C) is set or cleared depending on the operation performed. For `ADD` instructions that generate a Carry, and `SUB` instructions that generate a Borrow, the Carry Flag sets. The Carry Flag is reset by an `ADD` that does not generate a Carry, and by a `SUB` that does not generate a Borrow. This saved Carry facilitates software routines for extended precision arithmetic. Also, the `DAA` instruction sets the Carry Flag if the conditions for making the decimal adjustment are met.

For instructions `RLA`, `RRA`, `RL`, and `RR`, the Carry bit is used as a link between the least significant byte (LSB) and most significant byte (MSB) for any register or memory location. During instructions `RLCA`, `RLC`, and `SLA`, the Carry contains the last value shifted out of Bit 7 of any register or memory location. During instructions `RRCA`, `RRC`, `SRA`, and `SRL`, the Carry contains the last value shifted out of Bit `0` of any register or memory location.

For the logical instructions AND, OR, and XOR, the Carry is reset.

The Carry Flag can also be set by the Set Carry Flag (SCF) and complemented by the Compliment Carry Flag (CCF) instructions.

Add/Subtract Flag

The Add/Subtract Flag (N) is used by the Decimal Adjust Accumulator instruction (DAA) to distinguish between ADD and SUB instructions. For ADD instructions, N is cleared to 0. For SUB instructions, N is set to 1.

Parity/Overflow Flag

The Parity/Overflow Flag (O/V) is set to a specific state depending on the operation performed.

For arithmetic operations, this flag indicates an Overflow condition when the result in the Accumulator is greater than the maximum possible number (+127) or is less than the minimum possible number (–128). This Overflow condition is determined by examining the sign bits of the operands.

For addition, operands with different signs never cause Overflow. When adding operands with like signs and the result has a different sign, the Overflow Flag is set, for example:

```
+120   =  0111  1000
+105   =  0110  1001
_____     _________________
+225   =  1110  0001  (-95)    SUM
```

The two numbers added together resulted in a number that exceeds +127 and the two positive operands have resulted in a negative number (–95), which is incorrect. The Overflow Flag is therefore set.

For subtraction, Overflow can occur for operands of unlike signs. Operands of like sign never cause Overflow, for example:

```
     +127  0111   1111  MINUEND
(-)   -64  1100   0000  SUBTRAHEND
     +191  1011   1111  DIFFERENCE
```

The minuend sign has changed from a Positive to a negative, giving an incorrect difference. Overflow is therefore set.

Another method for identifying an Overflow is to observe the Carry to and out of the sign bit. If there is a Carry in and no Carry out, or if there is no Carry in and a Carry out, then Overflow has occurred.

This flag is also used with logical operations and rotate instructions to indicate the resulting parity is Even. The number of `1` bits in a byte are counted. If the total is Odd, ODD parity is flagged (P is `0`). If the total is Even, EVEN parity is flagged (P is s`1`).

During search instructions (`CPI`, `CPIR`, `CPD`, `CPDR`) and block transfer instructions (`LDI`, `LDIR`, `LDD`, `LDDR`), the P/V Flag monitors the state of the Byte Count Register (`BC`). When decrementing, if the byte counter decrements to `0`, the flag is cleared to `0`, otherwise the flag is set to`1`.

During `LD A, I` and `LD A, R` instructions, the P/V Flag is set with the value of the interrupt enable flip-flop (`IEF2`) for storage or testing.

When inputting a byte from an I/O device with an `IN r`, (`C`), or `INO r`, (`N`) instruction, the P/V Flag is adjusted to indicate the data parity.

Half-Carry Flag

The Half-Carry Flag (H) is set (`1`) or cleared (`0`) depending on the Carry and Borrow status between Bits 3 and 4 of an 8-bit arithmetic operation. This flag is used by the Decimal Adjust Accumulator instruction (`DAA`) to correct the result of a packed BCD add or subtract operation. The H Flag is set (`1`) or cleared (`0`) according to the following table:

H Flag	Add	Subtract
1	A Carry occurs from Bit 3 to Bit 4	A Borrow from Bit 4 occurs
0	No Carry occurs from Bit 3 to Bit 4	No Borrow from Bit 4 occurs

Zero Flag

The Zero Flag (Z) is set (1) or cleared (0) if the result generated by the execution of certain instructions is 0.

For 8-bit arithmetic and logical operations, the Z flag is set to 1 if the resulting byte in the Accumulator is 0. If the byte is not 0, the Z flag is 0 or 1.

For compare (Search) instructions, the Z flag is set to 1 if the value in the Accumulator is equal to the value in the memory location indicated by the value of the Register pair HL.

When testing a bit in a register or memory location, the Z flag contains the complemented state of the indicated bit (see “Bit b, s”).

When inputting or outputting a byte between a memory location and an I/O device (INI, IND, OUTI, and OUTD), if the result of decrementing the B Register is 0, the Z flag is 1, otherwise the Z flag is 0. Also for byte inputs from I/O devices using IN r, (C) or IN0 r, (n), the Z flag is set to indicate a 0-byte input.

Sign Flag

In some instructions, the Sign Flag (S) stores the state of the most significant bit of the result. When the Z80 performs arithmetic operations on signed numbers, the binary twos-complement notation is used to represent and process numeric information. A positive number is identified by a 0 in Bit 7. A negative number is identified by a 1. The binary equivalent of the magnitude of a positive number is stored in bits 0 to 6 for a total range of from 0 to 127. A negative number is represented by the twos complement of the equivalent positive number. The total range for negative numbers is from –1 to –128.

When inputting a byte from an I/O device to a register using an IN r,(C) or INO r,(n) instruction, the S Flag indicates the state of Bit 7.

INSTRUCTION SUMMARY

Table 24. Instruction Summary

Name	Assembler	Description
Add with Carry (8 bit)	ADC A,s	A ¨ A + s + carry
Add with Carry (16 bit)	ADC HL,ss	HL ¨ HL + ss + carry
Add (8 bit)	ADD A,s	A ¨ A + s
Add (16 bit)	ADD rr,tt	rr ¨ rr + tt
Logical And	AND A,s	A ¨ A and s
Bit Test	BIT b,m	Set Z flag per bit
Call Subroutine Conditionally	CALL cc,mn	If cc, call subroutine at address mn
Call Subroutine	CALL mn	call subroutine at address mn
Complement Carry Flag	CCF	carry ¨ not carry
Compare	CP A,s	Compare Accumulator
Compare and Decrement	CPD	Compare A vs. (HL), Decrement HL, B
Compare, Decrement, Repeat	CPDR	Scan memory block for match with A
Compare and Increment	CPI	Compare A vs. (HL), Increment HL, B
Compare, Increment, Repeat	CPIR	Scan memory block for match with A
Complement	CPL	Ones Complement Accumulator

Table 24. Instruction Summary (continued)

Name	Assembler	Description
Decimal Adjust Accumulator	DAA	Corrects decimal add or subtract
Decrement (16 bit)	DEC ee	ee ¨ ee - 1
Decrement (8 bit)	DEC m	m ¨ m - 1
Disable Interrupts	DI	Clears IEF1 and IEF2 flags
Decrement, Jump if Non-Zero	DJNZ d	Decrement B, jump if not zero
Enable Interrupts	EI	Sets IEF! and IEF2 flags
Exchange Accumulator/Flags	EX AF,AF'	Exchange AF with their alternates
Exchange DE and HL	EX DE,HL	DE ´ HL
Exchange with Top of Stack	EX (SP),rr	Exchange rr with top stack entry
Exchange Register Banks	EXX	Exchange BC, DE, HL with alternates
Halt	HALT	Halts processor execution
Interrupt Mode	IM n	Set INT0 interrupt mode 0, 1, or 2
Input	IN A,(n)	Input to A from port (A,n)
Input	IN r,(C)	Input to Register r from port (C) or (BC)
Input from Page 0	IN0 r,(n)	Input to Register r from port (0,n)
Increment (16 bit)	INC ee	ee ¨ ee + 1
Increment (8 bit)	INC m	m ¨ m + 1
Input and Decrement	IND	(HL) ¨ port (C), HL ¨ HL - 1, B ¨ B - 1
Input, Decrement, Repeat	INDR	Repeat IND until B = 0
Input and Increment	INI	(HL) ¨ port (C), HL ¨ HL + 1, B ¨ B - 1
Input, Increment, Repeat	INIR	Repeat INI until B = 0
Jump via Register	JP (rr)	PC ¨ (rr)

Table 24. Instruction Summary (continued)

Name	Assembler	Description
Jump Conditionally	JP cc,mn	If cc, PC ¨ mn
Jump Relative Conditionally	JR cc',d	If cc’, PC ¨ PC ± d
Jump	JP mn	PC ¨ mn
Jump Relative	JR d	PC ¨ PC ± d
Load to Memory (8 bit)	LD (aa),A	Store A at (aa)
Load to Memory (16 bit)	LD (mn),ee	Store ee at (mn)
Load from Memory (8 bit)	LD A,(aa)	Load A from (aa)
Load from Register I	LD A,I	A ¨ I
Load from Register R	LD A,R	A ¨ R
Load Immediate (16 bit)	LD ee,mn	Load ee with immediate value mn
Load from Memory (16 bit)	LD ee,(mn)	Load ee from (mn)
Load to Register I	LD I,A	I ¨ A
Load Immediate (8 bit)	LD m,n	Load m with immediate value m
Load from Register (8 bit)	LD m,r	Store r to m
Load to Register R	LD R,A	R ¨ A
Load to Register (8 bit)	LD r,s	Load r from s
Load Stack Pointer	LD SP,rr	SP ¨ rr
Load and Decrement	LDD	(DE)¨(HL), DE¨DE-1, HL¨HL-1, BC¨BC-1
Load, Decrement, Repeat	LDDR	Repeat LDD until BC = 0
Load and Increment	LDI	(DE)¨(HL), DE¨DE+1, HL¨HL+1, BC¨BC-1
Load, Increment, Repeat	LDIR	Repeat LDI until BC = 0
Multiply	MLT ss	ss ¨ ss_H times ss_L

Table 24. Instruction Summary (continued)

Name	Assembler	Description
Negate	NEG	Twos complement accumulator
No Operation	NOP	Do nothing
Inclusive Or	OR A,s	A ¨ A or s
Output and Decrement (page 0)	OTDM	Port (0,C)¨(HL), HL¨HL-1, C¨C-1, B¨B-1
Output, Decrement, Repeat (page 0)	OTDMR	Repeat OTDM until B = 0
Output, Decrement, Repeat	OTDR	Repeat OUTD until B = 0
Output and Increment (page 0)	OTIM	Port (0,C)¨(HL), HL¨HL+1, C¨C+1, B¨B-1
Output, Increment, Repeat (page 0)	OTIMR	Repeat OTIM until B = 0
Output, Decrement, Repeat	OTIR	Repeat OUTI until B = 0
Output	OUT (C),r	Port (C) or (BC) ¨ r
Output	OUT (n),A	Port (n) ¨ A
Output to Page 0	OUT0 (n),r	Port (0,n) ¨ r
Output and Decrement	OUTD	Port (C)¨(HL), HL¨HL-1, B¨B-1
Output and Increment	OUTI	Port (C)¨(HL), HL¨HL+1, B¨B-1
Pop from Stack	POP pp	pp ¨ (SP), SP ¨ SP+2
Push onto Stack	PUSH pp	SP ¨ SP-2, SP ¨ pp
Reset Bit	RES b,m	Bit b of m ¨ 0
Return	RET	PC ¨ (SP), SP ¨ SP+2
Return Conditionally	RET cc	If cc, {PC ¨ (SP), SP ¨ SP+2}
Return from Interrupt	RETI	As RET, recognized by Z80 peripherals
Return from NMI	RETN	IEF1 ¨ IEF2, PC ¨ (SP), SP ¨ SP+2

Table 24. Instruction Summary (continued)

Name	Assembler	Description
Rotate Left	RL m	Rotate m Left through carry
Rotate Left Accumulator	RLA	Rotate A Left through carry
Rotate Left Circular	RLC m	Rotate m Left into (not thru) carry
Rotate Left Circular Accumulator	RLCA	Rotate A Left into (not thru) carry
Rotate Left Decimal	RLD	Rotate bits 3-0 of A Left with (HL)
Rotate Right	RR m	Rotate m Right through carry
Rotate Right Accumulator	RRA	Rotate A Right through carry
Rotate Right Circular	RRC m	Rotate m Right into (not thru) carry
Rotate Right Circular Accumulator	RRCA	Rotate A Right into (not thru) carry
Rotate Right Decimal	RRD	Rotate bits 3-0 of A Right with (HL)
Restart	RST p	CALL to fixed location 00H, 08H,..., 38H
Subtract with Carry (8 bit)	SBC A,s	A ¨ A - s - carry
Subtract with Carry (16 bit)	SBC HL,ss	HL ¨ HL - ss - carry
Set Carry Flag	SCF	Carry ¨ 1
Set Bit	SET b,m	Bit b of m ¨ 1
Shift Left Arithmetic	SLA m	Shift m left with incoming 0
Sleep	SLP	Draws less power than Halt
Shift Right Arithmetic	SRA m	Shift m Right, replicating sign bit
Shift Right Logical	SRL m	Shift m Right with incoming 0
Subtract	SUB A,s	A ¨ A - s

Table 24. Instruction Summary (continued)

Name	Assembler	Description
Test	TST A,s	Set Z flag per A and s
Test I/O	TSTIO n	Set Z flag per port (C) and n
Exclusive Or	XOR A,s	A ¨ A xor s

ADC A, S — Add with Carry (8 Bit)

Operation

A ← A + s + CY

Format

ADC A, r	1	0	0	0	1	← r →			
ADC A, n	1	1	0	0	1	1	1	0	CE
	← n →								
ADC A, (HL)	1	0	0	0	1	1	1	0	8E
ADC A, (IX+d)	1	1	0	1	1	1	0	1	DD
	1	0	0	0	1	1	1	0	8E
	← d →								
ADC A, (IY+d)	1	1	1	1	1	1	0	1	FD
	1	0	0	0	1	1	1	0	8E
	← d →								

Description

The s operand and the C Flag are added to the contents of the Accumulator (A). The result is stored in A, and the flags are set as described below. The s can be any of a Register r, an immediate value n in the instruction, a memory location selected by the contents of Register pair HL, or a memory location selected by the sum of the contents of an index Register IX or IY and a signed 8-bit displacement d. In the register form, r selects the source register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
ADC A, r	1	4	4 (3+ 1 int)
ADC A, n	2	7 (4, 3)	6 (3, 3)
ADC A, (HL)	2	7 (4, 3)	6 (3, 3)
ADC A, (IX+d)	4	19 (4, 4, 3, 5 int, c 3)	14 (3, 3, 3, 2 int, 3)
ADC A, (IY+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)

Flags

Flags	Description
S	Set when the result is Negative; reset otherwise
Z	Set when the result is 0; reset otherwise
H	Set when a Carry occurs from Bit 3; reset otherwise
P/V	Set when an Overflow occurs; reset otherwise
N	Reset
C	Set when a Carry occurs from Bit 7; reset otherwise

Example

The Accumulator value is `1`. The C Flag is set. The Register pair `HL` value is `6666H`, and address `6666H` value is `1.` At `ADC A`, `(HL)`, the Accumulator value is `27H`.

Add with Carry (16 Bit) ADC HL, SS

Operation

`HL ← HL + ss + CY`

Format

1	1	1	0	1	1	0	1	ED
0	1	s	s	1	0	1	0	

Description

The contents of Register pair `ss` (any of Register pairs BC, DE, HL, or SP) and the C Flag are added to the contents of Register pair HL, the result is stored in HL, and the flags are set as described below. Operand `ss` is specified as follows in the assembled object code:

Register Pair	**Hex Value (ss)**
BC	00
DE	01
HL	10
SP	11

Timing

M Cycles	Z80 T States	Z18x T States
2	15 (4, 4, 7 int)	10 (3, 3, 4 int)

Flags

Flags	Description
S	Set when the result is Negative; reset otherwise
Z	Set when the result is 0; reset otherwise
R	Set when a Carry out of Bit 11 occurs; reset otherwise
P/V	Set when an Overflow occurs; reset otherwise
N	Reset
C	Set when a Carry from Bit 15 occurs; reset otherwise

Example

Register pair BC value is 2222H. Register pair HL value is 5437H and the C Flag is set. At ADC HL, BC, the value of HL is 765AH.

Add (8 Bit) ADD A,S

Operation

A ← A + s

Format

ADD A,r	1	0	0	0	0	← r →			
ADD A, n	1	1	0	0	0	1	1	0	C6
	← n →								
ADD A, (HL)	1	0	0	0	0	1	1	0	86
ADD A, (IX+d)	1	1	0	1	1	1	0	1	DD
	1	0	0	0	0	1	1	0	86
	← d →								
ADD A, (IY+d)	1	1	1	1	1	1	0	1	FD
	1	0	0	0	0	1	1	0	86
	← d →								

Description

The s operand is added to the value in the Accumulator (A), the result is stored in A, and the flags are set as described below. The s can be any of a Register r, an immediate value n in the instruction, a memory location selected by the contents of Register pair HL, or a memory location selected by the sum of the contents of an index Register IX or IY and a signed 8-bit displacement d. In the register form, r selects a source register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
ADD A,r	1	4	4 (3, 1 int)
ADD A,n	2	7 (4, 3)	6 (3, 3)
ADD A,(HL)	2	7 (4, 3)	6 (3, 3)
ADD A,(IX+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)
ADD A,(IY+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)

Flags

Flag	Description
S	Set if the result is Negative, reset otherwise
Z	Set if the result is 0; reset otherwise
H	Set if a Carry from Bit 3 occurs; reset otherwise
P/V	Set if an Overflow occurs; reset otherwise

N	Reset
C	Set if a Carry from Bit 7 occurs; reset otherwise

Example

The Accumulator (A) value is 11H. The Index Register IY value is 1000H, and memory location 1005H value is 22H. At ADD A, (IY + 5), the value of A is 33H, and the S, Z, H, P/V, and C flags all reset to 0.

ADD RR,TT Call Subroutine Conditionally

Operation

`rr ← rr + tt`

Format

ADD HL, tt

0	0	t	t	1	0	0	1

ADD IX, tt

1	1	0	1	1	1	0	1	DD
0	0	t	t	1	0	0	1	

ADD IY, tt

1	1	1	1	1	1	0	1	FD
0	0	t	t	1	0	0	1	

Description

The contents of 16-bit Register `tt` are added to the contents of 16-bit Register `rr`, the result is stored in `rr`, and the `C` Flag indicates a Carry out. The destination Register `rr` can be Register pair `HL` or an index Register `IX` or `IY`. The source Register `tt` is encoded as follows:

Register	**Hex Value (tt)**
BC	00
DE	01
Same as rr (HL, IX, or IY)	10
SP	11

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
ADD HL, tt	1	11 (4, 7 int)	7 (3, 4 int)
ADD IX, tt	2	15 (4, 4, 7 int)	10 (3, 3, 4 int)
ADD IY, tt	2	15 (4, 4, 7 int)	10 (3, 3, 4 int)

Flags

Flags	Description
S	Not affected
Z	Not affected
H	Set if a Carry out of Bit 11 occurs; reset otherwise
P/V	Not affected
N	Reset
C	Set if a Carry out of Bit 15 occurs; reset otherwise

Example

The Register pair `BC` value is `0102H`. The `IX` value is `1234H`. At `ADD IX, BC`, the `IX` value is `1336H`, and the C Flag is cleared to `0`.

AND A,S

Logical And

Operation

A ←A and s

Format

AND A, r	1	0	1	0	0	← r →			
AND A, n	1	1	1	0	0	1	1	0	E6
	← n →								
AND A, (HL)	1	0	1	0	0	1	1	0	A6
AND A, (IX+d)	1	1	0	1	1	1	0	1	DD
	1	0	1	0	0	1	1	0	A6
	← d →								
AND A, (IY+d)	1	1	1	1	1	1	0	1	FD
	1	0	1	0	0	1	1	0	A6
	← d →								

Description

The s operand and the value in the Accumulator (A) are combined using a logical AND quantify. The result is stored in A, and the flags are set as described below. The s can be any of a Register r, an immediate value n in the instruction, a memory location selected by the contents of Register pair HL, or a memory location selected by the sum of the contents of an index Register IX or IY and a signed 8-bit displacement d. In the register form, r selects a source register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z180x States
AND A, r	1	4	4 (3 + 1 int)
AND A, n	2	7 (4, 3)	6 (3, 3)
AND A, (HL)	2	7 (4, 3)	6 (3, 3)
AND A, (IX+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)
AND A, (IX+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)

Flags

Flags	Description
S	Set if the result is Negative; reset otherwise
Z	Set if the result is 0; reset otherwise
H	Set
P/V	Set if resulting parity is Even; reset otherwise

AND A,S   Logical And

Flags	Description
N	Reset
C	Reset

Example

The B register value is 7BH. The Accumulator value is C3H. At AND A, B, the Accumulator value is 43H.

Bit Test **BIT B,M**

Operation

```
Z flag ← NOT (bit b of m)
```

Format

BIT b, r	1	1	0	0	1	0	1	1	CB
	0	1	← b →			← r →			
BIT b, (HL)	1	1	0	0	1	0	1	1	CB
	0	1	← b →			1	1	0	
BIT b, (IX+d)	1	1	0	1	1	1	0	1	DD
	1	1	0	0	1	0	1	1	CB
	← d →								
	0	1	← b →			1	1	0	
BIT b, (IY+d)	1	1	1	1	1	1	0	1	FD
	1	1	0	0	1	0	1	1	CB
	← d →								
	0	1	← b →			1	1	0	

Description

The Z flag is set if Bit `b` of operand `m` is `0`, or cleared if the Bit is `1`. The `b` can be `0` for the least significant bit through Bit 7. The `m` can be a Register `r`, a memory location selected by the contents of Register pair HL, or a memory location selected by the sum of the contents of an index Register IX or IY and a signed 8-bit displacement d. In the register form, r selects the register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
BIT b, r	2	8 (4, 4)	6 (3, 3)
BIT b, (HL)	3	12 (4, 4, 4)	9 (3, 3, 3)
BIT b, (IX+d)	5	20 (4, 4, 3, 5, 4)	15 (3, 3, 3, 3, 3)
BIT b, (IY+d)	5	20 (4, 4, 3, 5, 4)	15 (3, 3, 3, 3, 3)

Flags

Flags	Description
S	Unknown
Z	Set if the specified Bit is 0; reset otherwise
H	Set
P/V	Unknown

N	Reset
C	Not affected

Example

The IY value is 1000H. The memory location 0FF0H value is BFH. At BIT 6, (IY - 16), the Z flag is set to 1. For any of the other seven bits in the byte, the Z flag is cleared to 0.

CALL CC,MN

Call Subroutine Conditionally

Operation

IF cc true: (SP-1)←PC_H, (SP-2)←PC_L, SP←SP-2, PC←mn

Format

1	1	←cc→			1	0	0
←n→							
←m→							

Note: The n value in the assembled object code above is the less significant byte of the 2-byte memory address.

Description

If condition cc is True, this instruction pushes the current contents of the PC to the top of the stack, then loads the value mn to PC, pointing to the address in memory where the first opcode of a subroutine is fetched. (At the end of the subroutine, a RET instruction can be used to return to the original program flow by popping the top of the stack back to PC.) If condition cc is False, the PC is incremented as usual, and the program continues with the next sequential instruction. The stack push is accomplished by first decrementing the current contents of the SP, loading the high-order byte of the PC contents to the memory address now pointed to by SP, then decrementing SP again, and loading the low-order byte of the PC contents to the top of the stack. Because this is a 3-byte instruction, the PC is incremented by three before the push is executed. Condition cc is programmed as one of eight values that correspond to condition bits in the Flag Register. These are defined in the table below:

Hex Value (cc)	Condition	Relevant Flag
000	NZ - Nonzero	Z
001	Z - 0	Z
010	NC - No Carry	C
011	C - Carry	C
100	PO - Parity Odd	P/V
101	PE - Parity Even	P/V
110	P - Sign Positive	S
111	M - Sign Negative	S

Timing

If cc is True:

Z80 M Cycles	Z80 T States	Z180x M Cycles	Z180x T States
5	17 (4, 3, 4, 3, 3)	5	16 (3, 3, 3, 1 int, 3, 3)

If cc is False:

Z80 M Cycles	Z80 T States	Z180x M Cycles	Z180x T States
3	10 (4, 3, 3)	2	6 (3, 3)

Example

The C Flag is reset. The value of the PC is 1A. The value of the SP is 3002H, and the instruction CALL NC, 2135H, is located in 1A47H through 1A49H. At CALL, the value of memory address 3001H is 1AH.

CALL CC,MN

Call Subroutine Conditionally

The value of address 3000H is 4AH. The value of the SP is 3000H, and the value of the PC is 2135H, pointing to the address of the first opcode of the subroutine executed.

Call Subroutine CALL MN

Operation

$(SP\text{-}1) \leftarrow PC_H,\ (SP\text{-}2) \leftarrow PC_L,\ SP \leftarrow (SP)\text{-}2,\ PC \leftarrow mn$

Format

1	1	0	0	1	1	0	0	CD
←			n				→	
←			m				→	

Note: The n value in the assembled object code above is the less significant byte of the 2-byte memory address.

Description

The current value of the PC is pushed to the top of the stack. The value mn is then loaded to the PC pointing to the address in memory where the first opcode of a subroutine is fetched. (At the end of the subroutine, a RET instruction can be used to return to the original program flow by popping the top of the stack back to the PC.) The PUSH is accomplished by first decrementing the current value of the SP, loading the high-order byte of the PC value to the memory address now pointed to by the SP; then decrementing SP again, and loading the low-order byte of the PC value to the top of stack. Because this is a 3-byte instruction, the PC increments by 3 before the PUSH is executed.

Timing

M Cycles	Z80 T States	Z180x T States
5	17 (4, 3, 4, 3, 3)	16 (3, 3, 3, 1 int, 3, 3)

Example

The value of the PC is 1A47H. The value of the SP is 3002H, and the instruction CALL 2135H is located at addresses 1A47H through 1A49H. After the execution, the value of memory address 3001H is 1AH. The value of address 3000H is 4AH. The value of the SP is 3000H, and the value of the PC is 2135H, pointing to the address of the first opcode of the subroutine to be executed.

Complement Carry Flag CCF

Operation

```
CY ← /CY
```

Format

0	0	1	1	1	1	1	1	3F

Description

The Carry Flag is inverted.

Timing

M Cycles	Z80 T States	Z18x T States
1	4	3

Flags

Flags	Description
S	Not affected
Z	Not affected
H	Previous Carry is copied
P/V	Not affected
N	Reset
C	Complemented (inverted)

CP A,S

Compare

Operation

A - s

Format

CP A,r	1	0	1	1	1	←	r	→	

CP A, n	1	1	1	1	1	1	1	0	FE
	←			n				→	

CP A, (HL)	1	0	1	1	1	1	1	0	BE

CP A, (IX+d)	1	1	0	1	1	1	0	1	DD
	1	0	1	1	1	1	1	0	BE
	←			d				→	

CP A, (IY+d)	1	1	1	1	1	1	0	1	FD
	1	0	1	1	1	1	1	0	BE
	←			d				→	

Description

The operand s is subtracted from the value of the Accumulator (A), the result is discarded and A is not affected, but the result is indicated in the flags, which are set as described below. The s can be any of a Register r, an immediate value n in the instruction, a memory location selected by the value of Register pair `HL`, or a memory location selected by the sum of the value of an index Register `IX` or `IY` and a signed 8-bit displacement `d`. In the register form, `r` selects a source register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
CP A, r	1	4	4 (3 + 1 int)
CP A, n	2	7 (4, 3)	6 (3, 3)
CP A, (HL)	2	7 (4, 3)	6 (3, 3)
CP A, (IX+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)
CP A,(IY+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)

Flags

Flags	Description
S	Set if the result is Negative; reset otherwise
Z	Set if the result is 0; reset otherwise
H	Set if Borrow from Bit 4; reset otherwise
P/V	Set if an Overflow occurs; reset otherwise

CP A,S Compare

Flags	Description
S	Set if the result is Negative; reset otherwise
N	Set
C	Set if Borrow; reset otherwise

Example

The Accumulator value is `63H`. The Register pair `HL` value is `6000H`. Memory location `6000H` value is `60H`. The instruction `CP A, (HL)` results in the following flag settings:

Compare and Decrement CPD

Operation

A -(HL), HL←HL -1, BC ←BC -1

Format

1	1	1	0	1	1	0	1	ED
1	0	1	0	1	0	0	1	A9

Description

The value of the memory location addressed by HL is compared with the value of the Accumulator. If the bytes are equal, the Z flag is set. HL and the Byte Counter (Register pair BC) are decremented, and the P/V Flag is set to indicate if BC has been decremented to 0.

Timing

M Cycles	Z80 T States	Z180x T States
3	16 (4, 4, 3, 5 int)	12 (3, 3, 3, 3 int)

Flags

Flags	Description
S	Set if A-(HL) is Negative; reset otherwise
Z	Set if A is (HL); reset otherwise
H	Set if Borrow from Bit 4; reset otherwise
P/V	Reset if BC is 0000; set otherwise
N	Set
C	Not Affected

Example

The Register pair HL value is 1111H. The memory location 1111H is 3BH. The Accumulator is 3BH, and BC is 0001H. At CPD, the value of BC is 0000. The value of HL is 1110H. The Z flag is set, and the P/V Flag resets. There is no effect on the value of the Accumulator or address 1111H.

Compare, Decrement, Repeat CPDR

Operation

A-(HL), HL ← HL -1, BC ← BC -1, repeat if no match and BC nonzero

Format

1	1	1	0	1	1	0	1	ED
1	0	1	1	1	0	0	1	B9

Description

The value of the memory location addressed by HL is compared with the value of the Accumulator. If the bytes are equal, the Z flag is set. The HL and BC (Byte Counter) Register pairs are decremented, and the P/V Flag is set to indicate if BC has been decremented to 0. If decrementing causes the BC to go to 0 or if A is HL, the instruction is terminated. If BC is not 0 and A is not HL, the instruction is repeated. Interrupts are recognized and refresh cycles may be executed after each data transfer. If BC is set to 0 prior to instruction execution, the instruction loops through 64 KB if no match is found.

Timing

For each repetition with BC is not 0 and A is not HL:

M Cycles	Z80 T States	Z180x T States
3	21 (4, 4, 3, 10 int)	14 (3, 3, 3, 5 int)

Where BC is 0 or A is (HL):

CPDR Compare, Decrement, Repeat

M Cycles	Z80 T States	Z180x T States
3	16 (4, 4, 3, 5 int)	12 (3, 3, 3, 3 int)

Flags

Flags	Description
S	Set if A-(HL) is Negative; reset otherwise
Z	Set if A is (HL); reset otherwise
H	Set if a Borrow from Bit 4 occurs; reset otherwise
P/V	Reset (0) if BC is 0000; set otherwise
N	Set
C	Not affected

Example

The Register pair `HL` value is `1118H`. The Accumulator value is `F3H`. The Register pair `BC` value is `0007H`, and memory locations values are:

Location	Value
1118H	52H
1117H	00H
1116H	F3H

At `CPDR`, the value of `HL` is `1115H`. The value of Register pair `BC` is `0004H`, and the P/V and Z flags are both set to `1`.

Compare and Increment CPI

Operation

```
A- (HL), HL ← HL +1, BC ←BC -1
```

Format

1	1	1	0	1	1	0	1	ED
1	0	1	0	0	0	0	1	A1

Description

The value of the memory location addressed by the HL Register is compared with the value of the Accumulator. If the bytes are equal, the Z flag is set. Then HL is incremented, the Byte Counter (Register pair BC) is decremented, and the P/V Bit is set to indicate if BC has decremented to 0.

Timing

M Cycles	Z80 T States	Z180x T States
3	16 (4, 4, 3, 5 int)	12 (3, 3, 3, 3 int)

Flags

Flags	Description
S	Set if A-(HL) is Negative; reset otherwise
Z	Set if A is (HL); reset otherwise
H	Set if a Borrow from Bit 4 occurs; reset otherwise
P/V	Reset if BC is 0000; set otherwise
N	Set
C	Not affected

Example

The Register pair HL value is 1111H. The memory location 1111H value is 3BH. The Accumulator value is 3BH, and Register pair BC value is 0001H. At CPI, the BC value is 0000H. The Register pair HL value is 1112H. The Z flag is set, and the P/V Flag resets to 0. Neither the Accumulator or memory location 1111H are affected.

Compare, Increment, Repeat CPIR

Operation

```
A-(HL), HL←HL+1, BC←BC-1, repeat if no match and BC
nonzero
```

Format

1	1	1	0	1	1	0	1	ED
1	0	1	1	0	0	0	1	B1

Description

The value of the memory location addressed by HL is compared with the value of the Accumulator. If the bytes are equal, the Z flag is set. HL is incremented and the Byte Counter (Register pair BC) is decremented. The P/V Flag is set to indicate if BC has been decremented to 0. If decrementing causes the BC to go to 0 or if A is HL, the instruction is terminated. If BC is not 0 and A is not HL, the instruction is repeated. Interrupts are recognized and refresh cycles may be executed after each data transfer. If BC is set to 0 before instruction execution, the instruction loops through 64 KB if no match is found.

Timing

For each repetition where BC is 0 and A is HL:

M Cycles	Z80 T States	Z18x T States
3	21 (4, 4, 3, 5, 10 int)	14 (3, 3, 3, 5 int)

Where BC is 0 or A is HL:

M Cycles	Z80 T States	Z18x T States
3	16 (4, 4, 3, 5 int)	12 (3, 3, 3, 3 int)

Flags

Flags	Description
S	Set if A-(HL) is Negative; reset otherwise
Z	Set if A is (HL); reset otherwise
H	Set if a Borrow from Bit 4 occurs; reset otherwise
P/V	Reset (0) if BC is 0000; set otherwise
N	Set
C	Not affected

Example

The Register pair HL value is 1111H. The Accumulator value is F3H. The BC value is 0007H, and memory locations values are:

Location	Value
1111H	52H
1112H	00H
1112H	F3H

At CPIR, the value of Register pair HL is 1114H. The Register pair BC value is 0004H, and the P/V and Z flags are both set.

CPL Complement

Operation

A← /A

Format

0	0	1	0	1	1	1	1	2F

Description

The value of the Accumulator (Register A) is inverted (1s complemented).

Timing

M Cycles	Z80 T States	Z180x T States
1	4	3

Flags

Flags	Description
S	Not affected
Z	Not affected
H	Set
P/V	Not affected
N	Set
C	Not affected

Example

The value of the Accumulator is D4H:

CPL Complement

1	0	1	1	0	1	0	0

At `CPL`, the Accumulator value is `4BH`:

0	1	0	0	1	0	1	1

Decmal Adjust Accumulator DAA

Operation

`Decimal Adjust Accumulator`

Format

0	0	1	0	0	1	1	1	27

Description

This instruction conditionally adjusts the Accumulator after addition and subtraction of BCD values. For addition (`ADD, ADC, INC`) or subtraction (`SUB, SBC, DEC, NEG`), Table 25 indicates the operation performed:

Table 25. DAA Addition/Subtraction Operation

Operation	C Before DAA	Hex Value In Upper Digit (Bit 7-4)	H Before DAA	Hex Value In Lower Digit (Bit 3-0)	Number Added To Byte	C After DAA	H After DAA
	0	0-9	0	0-9	00	0	0
	0	0-8	0	A-F	06	0	1
	0	0-9	1	0-3	06	0	0
ADD	0	A-F	0	0-9	60	1	0
ADC	0	9-F	0	A-F	66	1	1
INC	0	A-F	1	0-3	66	1	0
	1	0-2	0	0-9	60	1	0
	1	0-2	0	A-F	66	1	1

DAA Decmal Adjust Accumulator

Table 25. DAA Addition/Subtraction Operation

Operation	C Before DAA	Hex Value In Upper Digit (Bit 7-4)	H Before DAA	Hex Value In Lower Digit (Bit 3-0)	Number Added To Byte	C After DAA	H After DAA
	1	0-3	1	0-3	66	1	0
SUB	0	0-9	0	0-9	00	0	0
SBC	0	0-8	1	6-F	FA	0	1
DEC	1	7-F	0	0-9	A0	1	0
NEG	1	6-F	1	6-F	9A	1	1

Timing

M Cycles	Z80 T States	Z18x T States
1	4	4 (3 + 1 int)

Flags

Flags	Description
S	Set if the most significant bit of the Accumulator is 1 after the operation; reset otherwise
Z	Set if the Accumulator is 0 after the operation; reset otherwise
H	See instruction
P/V	Set if the Accumulator has Even parity after the operation; reset otherwise
N	Not affected
C	See “Description” (above)

Example

The15 (BCD) and 27 (BCD) are added, and the decimal arithmetic result is:

```
  15
 +27
-----
  42
```

But when the binary representations are added in the Accumulator according to standard binary arithmetic,

```
    0001    0101
+   0010    0111
--------  -------
    0011    1100   = 3C
```

The DAA instruction adjusts this result so that the correct BCD representation is calculated:

```
   0011   1100
 + 0000     0110
--------  -------
   0100     0010  = 42
```

DEC EE Decrement (16 Bit)

Operation

```
ee ← ee - 1
```

Format

DEC ss	0	0	s	s	1	0	1	1	
DEC IX	1	1	0	1	1	1	0	1	DD
	0	0	1	0	1	0	1	1	2B
DEC IY	1	1	1	1	1	1	0	1	FD
	0	0	1	0	1	0	1	1	2B

Description

The 16-bit Register, `ee`, is decremented by `1`. The flags are not affected. The `ee` may be any of the Register pairs `BC`, `DE`, or `HL`, the `SP`, or an index Register `IX` or `IY`. In the first form shown above, `ss` is encoded as follows:

Register	Hex Value (ss)
BC	00
DE	01
HL	10
SP	11

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
DEC ss	1	6 (4, 2 int)	4 (3, 1 int)
DEC IX	2	10 (4, 4, 2 int)	7 (3, 3, 1 int)
DEC IY	2	10 (4, 4, 2 int)	7 (3, 3, 1 int)

Example

The Register pair BC value is 0. The C Flag is 0. At DEC BC, the BC value is FFFFH, and the C Flag remains 0.

DEC M — Decrement (8 Bit)

Operation

m←m - 1

Format

DEC r	0	0	← r →			1	0	1	
DEC (HL)	0	0	1	1	0	1	0	1	35
DEC (IX+d)	1	1	0	1	1	1	0	1	DD
	0	0	1	1	0	1	0	1	35
	← d →								
DEC (IY+d)	1	1	1	1	1	1	0	1	FD
	0	0	1	1	0	1	0	1	35
	← d →								

Description

The 8-bit value in the `m` operand is decremented by one, and the flags (except `C`) are set as described below. The `m` may be a Register `r`, a memory location selected by the value of Register pair `HL`, or a memory location selected by the sum of the contents of an index Register `IX` or `IY` and a signed 8-bit displacement `d`. In the register form, `r` selects the register as follows:

Decrement (8 Bit) DEC M

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
DEC r	1	4	4 (3 + 1 int)
DEC (HL)	3	11 (4, 4, 3)	10 (3, 3, 1 int, 3)
DEC (IX+d)	5	23 (4, 4, 3, 5 int, 4, 3)	18 (3, 3, 3, 2 int, 3, 1 int, 3)
DEC (IY+d)	5	23 (4, 4, 3, 5 int, 4, 3)	18 (3, 3, 3, 2 int, 3, 1 int, 3)

Flags

Flags	Description
S	Set if the result is Negative; reset otherwise
Z	Set if the result is 0; reset otherwise
H	Set if a Borrow from Bit 4 occurs; reset otherwise
P/V	Set if 80H to 7FH; reset otherwise
N	Set
C	Not affected

Example

The D Register contains byte 2AH. At DEC D, the value of Register D is 29H.

Disable Interrupt DI

Operation

`IFF ← 0`

Format

1	1	1	1	0	0	1	1	F3

Description

The `DI` disables maskable interrupts by resetting the interrupt enable flip-flops (`IFF1` and `IFF2`). This instruction disables maskable interrupts during the execution.

Timing

M Cycles	Z80 T States	Z180x T States
1	4	3

Example

When the CPU executes the instruction `DI`, maskable interrupts are disabled until they are subsequently re-enabled by an `EI` instruction. The CPU does not respond to an Interrupt Request (`INT`) signal until that time.

DJNZ D

Decrement, Jump if Non-Zero

Operation

`Decrement and Jump if nonzero`

Format

0	0	0	1	0	0	0	0	10
←			d-2				→	

Description

This instruction is similar to the conditional jump instructions except that a register value determines branching. The `B` Register is decremented and if a nonzero value remains, the value of the displacement `d` is added to the `PC`. The next instruction is fetched from the location designated by the new value of the `PC`. Measured from the address of the instruction opcode, the jump has a range of `-126` to `+129` bytes. The assembler automatically adjusts for the twice incremented `PC`.

If decrementing leaves `B` with a `0` value, the next instruction executed is taken from the location following this instruction.

Timing

Register `B` is not `0`:

M Cycles	Z80 T States	Z18x T States
2	13 (5 ,3, 5 int)	9 (3, 1 int, 3, 2 int)

Register `B` is `0`:

M Cycles	Z80 T States	Z18x T States
2	8 (5 ,3)	7 (3, 1 int, 3)

Example

A typical software routine is used to demonstrate the use of the DJNZ instruction. This routine moves a line from an input buffer (INBUF) to an output buffer (OUTBUF). It moves the bytes until it finds a CR, or until it has moved 80 bytes, whichever occurs first.

```
LD               B, 80       Set up counter

LD               HL, Inbuf   Set up pointers

LD               DE, Outbuf

LOOP:

LD               A, (HL)     Get next byte from input
buffer

LD               DE), A      Store in output buffer
CP               0DH         Is it a CR?
JRZ, DONE                    Yes finished
INC              HL          Increment pointers
INC              DE
DJNZ LOOP
Loop back if 80 bytes have not been moved

DONE:
```

EI Enable Interrupts

Operation

`IFF ← 1`

Format

1	1	1	1	1	0	1	1	FB

Description

The enable interrupt instruction sets both interrupt enable flip flops (`IFFI` and `IFF2`) to `1` allowing recognition of any maskable interrupt. During the execution and the following instruction, maskable interrupts are disabled.

Timing

M Cycles	Z80 T States	Z18x T States
1	4	3

Example

The CPU executes instructions `EI RETI`, and maskable interrupts are enabled at the `RETI`.

Exchange Acacumulator/Flags

EX AF,AF’

Operation

AF ↔ AF

Format

0	0	0	0	1	0	0	0	08

Description

The value of the Register pairs AF and AF′ are exchanged. Register pair AF′ consists of registers A and F′.

Timing

M Cycles	Z80 T States	Z18x T States
1	4	4 (3, 1 int)

Flags

All

Example

The value of Register pair AF is 9900H, and the value of Register pair AF′ is 5944H. After the instruction EX AF, AF′, the value of AF is 5944H, and the value of AF′ is 9900H.

EX DE,HL

Exchange DE and HL

Operation

DE ↔ HL

Format

1	1	1	0	1	0	1	1	EB

Description

The contents of Register pairs DE and HL are exchanged. No flags are affected.

Timing

M Cycles	Z80 T States	Z18x T States
1	4	3

Example

The content of Register pair DE is 2822H. The content of Register pair HL is 499AH. After the instruction EX DE, HL, the content of Register pair DE is 499AH and the content of Register pair HL is 2822H.

Exchange with Top of Stack — EX (SP),RR

Operation

rr_L ↔ (SP), rr_H ↔ (SP+1

Format

EX (SP), HL	1	1	1	0	0	0	1	1	E3
EX (SP), IX	1	1	0	1	1	1	0	1	DD
	1	1	1	0	0	0	1	1	E3
EX (SP), IY	1	1	1	1	1	1	0	1	FD
	1	1	1	0	0	0	1	1	E3

Description

The contents of 16-bit Register `rr` are exchanged with the two bytes at the top of the Stack (the address in the `SP` and the next higher one) in memory. Neither `SP` or any flags are affected. The `rr` may be the Register pair `HL` or an index Register `IX` or `IY`.

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
EX (SP), HL	5	19 (4, 3, 4, 3, 5)	16 (3, 3, 3, 1 int, 3, 3)
EX (SP), IX	6	23 (4, 4, 3, 4, 3, 5)	19 (3, 3, 3, 3, 1 int, 3, 3)
EX (SP), IY	6	23 (4, 4, 3, 4, 3, 5)	19 (3, 3, 3, 3, 1 int, 3, 3)

Example

The Register pair HL value is 7012H. The SP value is 8856H. The memory location 8856H value is 11H, and location 8857H value is 22H. At EX (SP), HL, the HL value is 2211H. The memory location 8856H value is 12H. Location 8857H value is 70H, and SP value remains 8856H.

Exchange Register Banks EXX

Operation

(BC) ↔ (BC′), (DE) ↔ (DE′), (HL) ↔ (HL′)

Format

1	1	0	1	1	0	0	1	D9

Description

Each value in Register pairs BC, DE, and HL is exchanged with the 2-byte value in BC′, DE′, and HL′ respectively. No flags are affected.

Timing

M Cycles	Z80 T States	Z18x T States
1	4	3

Example

The value of Register pairs BC, DE, and HL are 445AH, 3DA2H, and 8859H, respectively. The value of Register pairs BC′, DE′, and HL′ are 0988H, 9300H, and 00E7H respectively. After the instruction EXX, the value of the Register pairs are as follows: BC is 0988H; DE is 9300H; HL is 00E7H; BC′ is 445AH; DE′ is 3DA2H; and HL′ is 8859H.

HALT Halt

Operation

`Halt the processor`

Format

0	1	1	1	0	1	1	0	76

Description

The `HALT` instruction suspends CPU operation until a subsequent interrupt or reset is received. While in the `HALT` state, the processor executes `NOPs` to maintain memory refresh logic.

Timing

M Cycles	Z80 T States	Z18x T States
1	(Indefinite) 4 Minimum	(Indefinite) 3 Minimum

Interrupt Mode IM N

Operation

```
Set interrupt mode n (0 to 2)
```

Format

1	1	1	0	1	1	0	1	ED
0	1	0	i	i	0	0	0	

Description

The processor is placed in interrupt mode n. The n may be 0 to 2.

In interrupt mode 0, an interrupting device can insert any instruction to the data bus for execution by the CPU. The first byte of the instruction is read during the interrupt acknowledge sequence. For multi-byte instructions, special external hardware is required because bytes after the first are read by normal memory read cycles. In Interrupt Mode 1, the processor responds to all interrupts similar to an RST 38H instruction. In Interrupt Mode 2, the processor responds to interrupts by fetching an interrupt vector from the interrupting device. It then reads a byte from memory at the address having the vector as the LSB and the contents of the I Register as the MSB, then loads the byte to the LSB of the PC. It reads a second byte from memory at the next higher address, and loads that byte to the MSB of PC. Finally, the processor starts execution of the interrupt service routine at the address fetched from memory.

When n is 0 to 2, it is encoded to ii in the instruction as shown below.

Mode n	ii
0	00
1	10
2	11

IM N

Interrupt Mode

Timing

M Cycles	Z80 T States	Z18x T States
2	8 (4, 4)	6 (3, 3)

Input IN A,(N)

Operation

`A ← (n)`

Format

1	1	0	1	1	0	1	1	DB
←			n				→	

Description

The operand `n` is placed on the bottom half (`A0` through `A7`) of the address bus to select the I/O device at one of 256 possible ports. The contents of the Accumulator appear on the top half (`A8` through `A15`) of the address bus. Then one byte from the selected port is placed on the data bus and written to the Accumulator (Register A) in the CPU. No flags are affected.

Timing

M Cycles	Z80 T States	Z180 Register T States	Z18x Other Register T States
3	11 (4, 3, 4)	9 (3, 3, 3)	10 (3, 3, 4)

Example

The value of the Accumulator is `23H`. The byte `7BH` is available at the peripheral device mapped to I/O port address `2301H`. At `IN A, (01H)` the Accumulator value is `7BH`. The peripheral device ignores `A8` through `A15`, and the above statement reads `at the peripheral device mapped to I/O port address 01H.`

IN A,(N) Input

Operation

`r ← (C)`

Format

1	1	1	0	1	1	0	1	ED
0	1	←	r	→	0	0	0	

Description

The contents of Register C are placed on the bottom half (`A0` through `A7`) of the address bus to select the I/O device at one of 256 possible ports. The contents of Register B are placed on the top half (`A8` through `A15`) of the address bus. Then one byte from the selected port is placed on the data bus and written to Register `r` in the CPU. Register `r` identifies any of the CPU registers shown in the following table. The flags are set to indicate the input data.

Register	**Hex Value (r)**
B	000
C	001
D	010
E	011
H	100
L	101
None	Flags are set as shown below
A	111

Timing

M Cycles	Z80 T States	Z180 Register T States	Z18x Other Register T States
3	12 (4, 4, 4)	9 (3, 3, 3)	10 (3, 3, 4)

Flags

Flags	Description
S	Set if Bit 7 of the input data is 1; reset otherwise
Z	Set if all 8 bits of the input data are 0; reset otherwise
H	Reset
P/V	Set if the parity of the input data is Even; reset otherwise
N	Reset
C	Not affected

Example

The value of Register `C` is `07`. The value of Register `B` is `10H`, and the byte `7BH` is available at the peripheral device mapped to I/O port address `1007H`. At `IN D, (C)`, Register `D` value is `7BH`. The peripheral device ignores `A8` through `A15`, and the above statement reads `at the peripheral device mapped to I/O port address 07H.`

IN0 R,(N) Input from Page 0

Operation

`r ← (0, n)`

Format

1	1	1	0	1	1	0	1	ED
0	0	← r →			0	0	0	
← n →								

Description

The operand `n` is placed on `A0` through `A7` with `0` on `A8` through `A15` to select an input port. The contents of the selected port are then read to the register selected by `r` as shown in the following table. The flags are affected as shown below.

Register	**Hex Value (r)**
B	000
C	001
D	010
E	011
H	100
L	101
(none)	110 (flags remain set as shown below)
A	111

Input from Page 0 IN0 R,(N)

Timing

M Cycles	T States (180 Reg)	T States (Other 18x)
4	12 (3, 3, 3, 3)	13 (3, 3, 3, 4)

Flags

Flags	Description
S	Set if Bit 7 of the input data is 1; reset otherwise
Z	Set if all 8 bits of the input data are 0; reset otherwise
H	Reset
P/V	Set if the parity of the input data is Even; reset otherwise
N	Reset
C	Not affected

Example

The value of the 8018x processor Input/Output Control Register (`IOCR`) is `1FH`. The 16-bit I/O address of the `IOCR` value is `003FH`. At `IN0 E (3FH)`, the `E` Register value is `1FH`. When the value of the `A` Register is nonzero, the execution of `IN A, (3FH)` does *not* return the value of the `IOCR`.

INC EE

Increment (16 Bit)

Operation

`ee ← ee + 1`

Format

INC ss	0	0	s	s	0	0	1	1	
INC IX	1	1	0	1	1	1	0	1	DD
	0	0	1	0	0	0	1	1	23
INC IY	1	1	1	1	1	1	0	1	FD
	0	0	1	0	0	0	1	1	23

Description

The16-bit Register `ee` is incremented by `1`. The flags are not affected. The `ee` may be any of the Register pairs `BC`, `DE`, or `HL`, the `SP`, or an index Register `IX` or `IY`. In the first form shown above, `ss` is encoded as follows:

Register	**Hex Value (ss)**
BC	00
DE	01
HL	10
SP	11

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
INC ss	1	6 (4, 2 int)	4 (3, 1 int)
INC IX	2	10 (4, 4, 2 int)	7 (3, 3, 1 int)
INC IY	2	10 (4, 4, 2 int)	7 (3, 3, 1 int)

Example

The `SP` value is `FFFFH`. The `C` Flag value is `0`. At `INC SP,` the SP value is `0000`. The C Flag remains `0`.

INC EE

Increment (8 Bit)

Operation

m ← m + 1

Format

INC r	0	0	← r →			1	0	0	
INC (HL)	0	0	1	1	0	1	0	0	34
INC (IX+d)	1	1	0	1	1	1	0	1	DA
	0	0	1	1	0	1	0	0	34
	← d →								
INC (IY+d)	1	1	1	1	1	1	0	1	FD
	0	0	1	1	0	1	0	0	34
	← d →								

Description

The 8-bit value in the `m` operand is incremented by one, and the flags (except C) are set as described below. The `m` can be a Register `r`, a memory location selected by the value of Register pair `HL`, or a memory location selected by the sum of the contents of an index Register `IX` or `IY` and a signed 8-bit displacement `d`. In the register form, `r` selects the register as follows:

Increment (8 Bit) INC EE

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
INC r	1	4	4 (3, 1 int)
INC (HL)	3	11 (4, 4, 3)	10 (3, 3, 1 int, 3)
INC (IX+d)	5	23 (4, 4, 3, 5 int, 4, 3)	18 (3, 3, 3, 2 int, 3, 1 int, 3)
INC (IY+d)	5	23 (4, 4, 3, 5 int, 4, 3)	18 (3, 3, 3, 2 int, 3, 1 int, 3)

Flags

Flags	Description
S	Set if the result is Negative; reset otherwise
Z	Set if the result is 0; reset otherwise
H	Set if a Carry from Bit 3 occurs; reset otherwise
P/V	Set if 7FH to 80H; reset otherwise
N	Reset
C	Not affected

Example

The IY value is 2020H. The memory location 2030H value is 34H. At INC (IY + 10H), memory location 2030H value is 35H.

Input and Decrement IND

Operation

`(HL) ← (C), B ← B - 1, HL ← HL - 1`

Format

1	1	1	0	1	1	0	1	ED
1	0	1	0	1	0	1	0	AA

Description

The contents of Register `C` are placed on the bottom half (`A0` through `A7`) of the address bus to select the I/O device at one of 256 possible ports. Register `B` may be used as a byte counter, and its contents are placed on the top half (`A8` through `A15`) of the address bus. One byte is read from the selected port and is captured by the CPU. The value of `HL` is placed on the address bus, and the input byte is written to the corresponding memory location. Registers `B` and `HL` are decremented.

Note: This instruction is not completely compatible with peripheral devices that decode `A15-8` as part of a 16-bit I/O address.

Timing

M Cycles	Z80 T States	Z180 Register T States	Z18x Other Register T States
4	16 (4, 5, 3, 4)	12 (3, 3, 3, 3)	13 (3, 3, 4, 3)

Flags

Flags	Description
S	Unknown
Z	Set if B is 0; reset otherwise
H	Unknown
P/V	Unknown
N	Set
C	Not affected

Example

The C Register value is 07H. The B Register value is 10H. The Register pair HL value is 1000H. The byte 7BH is available at the peripheral device mapped to I/O port address 07H. At IND, memory location 1000H value is 7BH, HL value is 0FFFH, and Register B value is 0FH.

Input, Decrement, Repeat INDR

Operation

`(HL)←(C), B←B-1, HL←HL-1, repeat while B is not 0`

Format

1	1	1	0	1	1	0	1	ED
1	0	1	1	1	0	1	0	BA

Description

The contents of Register C are placed on the bottom half (A0 through A7) of the address bus to select the I/O device at one of 256 possible ports. Register B is used as a byte counter, and its contents are placed on the top half (A8 through A15) of the address bus. Then one byte is read from the selected port and is captured by the CPU. The contents of HL are placed on the address bus and the input byte is written to the corresponding memory location. Finally, HL and B are decremented. If decrementing causes B to go to 0, the instruction is terminated. If B is not 0, the instruction is repeated. Interrupts are recognized and refresh cycles may be executed after each data transfer. If B is set to 0 prior to instruction the execution, 256 bytes of data are input.

Note: This instruction is not completely compatible with peripheral devices that decode A15-8 as part of a 16-bit I/O address.

Timing

For each byte, B is not 0:

M Cycles	Z80 T States	Z180 Register T States	Z18x Other Register T States
4	21 (4, 5, 3, 4, 5 int)	14 (3, 3, 3, 3, 2 int)	15 (3, 3, 4, 3, 2 int)

For the last byte, B is 0:

M Cycles	Z80 T States	Z180 Register T States	Z18x Other Register T States
4	16 (4, 5, 3, 4)	12 (3, 3, 3, 3)	13 (3, 3, 4, 3)

Flags

Flags	Description
S	Unknown
Z	Set
H	Unknown
P/V	Unknown
N	Set
C	Not affected

Input, Decrement, Repeat INDR

Example

Register C value is 07H. Register B value is 03H. The Register pair HL value is 1000H. The following sequence of bytes are available at the peripheral device mapped to I/O port address 07H:

```
51H (first)
A9H
03H (third)
```

At INDR, the HL value is 0FFDH, Register B value is 0, and memory location values are:

Location	Value
0FFEH	03H
0FFFH	A9H
1000H	51H

INI — Input and Increment

Operation

```
(HL)←(C), B←B -1, HL←HL +1
```

Format

1	1	1	0	1	1	0	1	ED
1	0	1	0	0	0	1	0	A2

Description

The contents of Register C are placed on the bottom half (A0 through A7) of the address bus to select the I/O device at one of 256 possible ports. Register B may be used as a byte counter, and its contents are placed on the top half (A8 through A15) of the address bus. Then one byte is read from the selected port and is captured by the CPU. The contents of HL are then placed on the address bus and the input byte is written to the corresponding location of memory. Finally, B is decremented and Register pair HL is incremented.

Note: This instruction is not completely compatible with peripheral devices that decode A15-8 as part of a 16-bit I/O address.

Timing

M Cycles	Z80 T States	Z180 Register T States	Z18x Other Register T States
4	16 (4, 5, 3, 4)	12 (3, 3, 3, 3)	13 (3, 3, 4, 3)

Input and Increment INI

Flags

Flags	Description
S	Unknown
Z	Set if B is 0; reset otherwise
H	Unknown
P/V	Unknown
N	Set
C	Not affected

Example

The value of Register C is 07H. The value of Register B is 10H. The value of HL is 1000H, and the byte 7BH is available at the peripheral device mapped to I/O port address 07H. At INI, memory location 1000H value is 7BH, HL value is 1001H, and Register B value is 0FH.

INIR Input and Increment

Operation

(HL) ←(C), B ←B -1, HL ←HL +1, repeat while B is not 0

Format

1	1	1	0	1	1	0	1	ED
1	0	1	1	0	0	1	0	B2

Description

The contents of Register C are placed on the bottom half (A0 through A7) of the address bus to select the I/O device at one of 256 possible ports. Register B is used as a byte counter, and its contents are placed on the top half (A8 through A15) of the address bus. Then one byte is read from the selected port and is captured by the CPU. The contents of HL is placed on the address bus and the input byte is written to the corresponding location of memory. Then HL is incremented and B is decremented. If decrementing causes B to go to 0, the instruction is terminated. If B is not 0, the instruction is repeated. Interrupts are recognized and refresh cycles execute after each data transfer. If B is set to 0 prior to instruction the execution, 256 bytes of data are input.

Note: This instruction is not completely compatible with peripheral devices that decode A15-8 as part of a 16-bit I/O address.

Timing

For each byte while B is not 0:

M Cycles	Z80 T States	Z180 Register T States	Z18x Other Register T States
4	21 (4, 5, 3, 4, 5 int)	14 (3, 3, 3, 3, 2 int)	15 (3, 3, 4, 3, 2 int)

For the last byte, B is 0:

M Cycles	Z80 T States	Z180 Register T States	Z18x Other Register T States
4	16 (4, 5, 3, 4)	12 (3, 3, 3, 3)	13 (3, 3, 4, 3)

Flags

The value of Register C is 07H. The value of Register B is 03H. The value of Register pair HL is 1000H, and the following sequence of bytes are available at the peripheral device mapped to I/O port address 07H:

```
51H (first)
A9H
03H (third)
```

At INIR, the value of HL is 1003H. The value of Register B is 0. Memory location values are:

Location	Value
1000H	51H
1001H	A9H
1002H	03H

JP (RR) Jump via Register

Operation

PC ← rr

Format

JP (HL)	1	1	1	0	1	0	0	1	E9
JP (IX)	1	1	0	1	1	1	0	1	DD
	1	1	1	0	1	0	0	1	E9
JP (IY)	1	1	1	1	1	1	0	1	FD
	1	1	1	0	1	0	0	1	E9

Description

The contents of 16-bit Register rr are loaded to the PC, the next instruction then starts at that address. No flags are affected. The rr can be the Register pair HL or an index Register IX or IY.

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
JP (HL)	1	4	3
JP (IX)	2	8 (4, 4)	6 (3, 3)
JP (IY)	2	8 (4, 4)	6 (3, 3)

Example

The Register pair `HL` value is `7404H`. At `JP (HL)`, the processor starts fetching the next instruction from memory location `7404H`.

JP CC,MN

Jump Conditionally

Operation

`IF cc is true, PC ←mn else continue`

Format

1	1	←cc→			0	1	0
←n→							
←m→							

Note: The n value above is the low order byte of the 2-byte memory address.

Description

If condition cc is True, the instruction loads the value mn to the PC, and the program continues with the instruction beginning at address mn. If condition cc is False, the PC is incremented as usual, and the program continues with the next instruction. Condition cc is programmed as one of eight that correspond to condition bits in the Flag Register:

Hex Value (cc)	Condition	Relevant Flag
000	NZ - Nonzero	Z
001	Z - 0	Z
010	NC - No Carry	C
011	C - Carry	C
100	PO - Parity Odd	P/V
101	PE - Parity Even	P/V
110	P - Sign Positive	S
111	M - Sign Negative	S

Timing

Jump

Z80 M Cycles	Z80 T States	Z18x M Cycles	Z18x T States
3	10 (4, 3, 3)	3	9 (3, 3, 3)

No jump

Z80 M Cycles	Z80 T States	Z18x M Cycles	Z18x T States
3	10 (4, 3, 3)	2	6 (3, 3)

Example

The `C` Flag is set. At `JP C, 1520H,` the `PC` value is `1520H`.

JR CC’,D — Jump Relative Conditionally

Operation

```
If cc true, PC ←PC ±d else continue
```

Format

0	0	1	c	c	0	0	0
← d →							

Description

If the flag condition selected by `cc'` is True, the 8-bit signed displacement `d` is added to the `PC`, which by this time is incremented to the next instruction address. If the selected condition is False, the execution continues with the next instruction. No flags are affected. The `cc` is encoded to express the following flag conditions:

Hex Value (cc)	Condition
00	NZ - Nonzero
01	Z - 0
10	NC - No Carry
11	C - Carry

Timing

If the condition is met and a jump occurs:

M Cycles	Z80 T States	Z18x T States
2	12 (4, 3, 5 int)	8 (3, 3, 2 int)

If the condition is not met, and no jump occurs:

M Cycles	Z80 T States	Z18x T States
2	7 (4, 3)	6 (3, 3)

Example

Because the assembler symbol $ refers to the first byte of the current instruction, the instruction JR C, $ - 4 is assembled as 38H FAH. (The assembled d byte is always 2 less than a displacement from $ in the source. FAH is -6 is two less than the -4 given in the source operand.) When C is set and the preceding instruction has its opcode byte located at location 480H, the execution of the instruction results in a branch to 47CH.

JP MN Jump

Operation

PC ← mn

Format

<table>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>C3</td></tr>
<tr><td colspan="8">← n →</td><td></td></tr>
<tr><td colspan="8">← m →</td><td></td></tr>
</table>

Note: The n value above is the low order byte of the two-byte address.

Description

Operand mn is loaded to Register pair PC. The next instruction is fetched from the location designated by the new contents of the PC.

Timing

M Cycles	Z80 T States	Z18x T States
3	10 (4, 3, 3)	9 (3, 3, 3)

Jump Relative JR D

Operation

PC ← PC ±d

Format

0	0	0	1	1	0	0	0	18
←d-2→								

Description

The value of the displacement d is added to the PC and the next instruction is fetched from the location designated by the new contents of the PC. Measured from the address of the instruction Op Code, the jump has a range of -126 to +129 bytes. The assembler automatically adjusts for the twice-incremented PC.

Timing

M Cycles	Z80 T States	Z18x T States
2	12 (4, 3, 5 int)	8 (3, 3, 2 int)

Example

To jump forward five locations from address 480, use this assembly language statement

JR $+5. The resulting object code and final PC value is:

JR D Jump Relative

Location	Instruction
480	18
481	03
482	–
483	–
484	–
485	← PC after jump

Load to Memory (8 Bit) LD (AA),A

Operation

(aa) ← A

Format

LD (BC), A	0	0	0	0	0	0	1	0	02
LD (DE), A	0	0	0	1	0	0	1	0	12
LD (HL), A	0	1	1	1	0	1	1	1	77
LD (mn), A	0	0	1	1	0	0	1	0	32
	← n →								
	← m →								
LD (IX+d), A	1	1	0	1	1	1	0	1	DD
	0	1	1	1	0	1	1	1	77
	← d →								
LD (IY+d), A	1	1	1	1	1	1	0	1	FD
	0	1	1	1	0	1	1	1	77
	← d →								

Description

The value in the Accumulator A is stored to the memory location selected by aa. The flags are not affected. The memory address aa can be the contents of any of the Register pairs BC, DE, or HL, a direct address mn in the instruction, or the sum of the contents of an index Register IX or IY and a signed 8-bit displacement d.

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
LD (BC), A	2	7 (4, 3)	7 (3, 1 int, 3)
LD (DE), A	2	7 (4, 3)	7 (3, 1 int, 3)
LD (HL), A	2	7 (4, 3)	7 (3, 1 int, 3)
LD (mn), A	4	13 (4, 3, 3, 3)	13 (3, 3, 3, 1 int, 3)
LD (IX+d), A	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)
LD (IY+d), A	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)

Example

The Accumulator value is D7H. The instruction LD (3141H), A results in D7H written to memory location 3141H.

Load to Memory (16 Bit)

LD (MN),EE

Operation

$(mn) \leftarrow ee_L, (mn+1) \leftarrow ee_H$

Format

LD (mn), HL

0	0	1	0	0	0	1	0	22
←				n			→	
←				m			→	

LD (mn), ss

0	0	1	1	1	0	1	0	ED
0	1	s	s	0	0	1	1	
←				n			→	
←				m			→	

LD (mn), IX

1	1	0	1	1	1	0	1	DD
0	0	1	0	0	0	1	0	22
←				n			→	
←				m			→	

LD (mn), IY

1	1	1	1	1	1	0	1	FD
0	0	1	0	0	0	1	0	22
←				n			→	
←				m			→	

LD (MN),EE Load to Memory (16 Bit)

Description

The contents of the less-significant half of 16-bit Register ee are stored in memory location mn, and the contents of the more-significant half are stored in location mn+1. The flags are not affected. The ee can be any of the Register pairs BC, DE, or HL, the SP, or an index Register IX or IY. The n is the least-significant byte of the memory address. In the second form shown above, ss is encoded as follows:

Register	Hex Value (ss)
BC	00
DE	01
HL	10 (The first form shown above is preferred)
SP	11

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
LD (mn), HL	5	16 (4, 3, 3, 3, 3)	15 (3, 3, 3, 3, 3)
LD (mn), ss	6	20 (4, 4, 3, 3, 3, 3)	19 (3, 3, 3, 3, 1 int, 3, 3)
LD (mn), IX	6	20 (4, 4, 3, 3, 3, 3)	19 (3, 3, 3, 3, 1 int, 3, 3)
LD (mn), IY	6	20 (4, 4, 3, 3, 3, 3)	19 (3, 3, 3, 3, 1 int, 3, 3)

Example

The SP value is CDE0H. At LD (6789H), the SP memory location 6789H value is E0H and location 678AH value is CDH.

Load from Memory (8 Bit) LD A,(AA)

Operation

A ← (aa)

Format

LD A, (BC)	0	0	0	0	1	0	1	0	0A
LD A, (DE)	0	0	0	1	1	0	1	0	1A
LD A, (HL)	0	1	1	1	1	1	1	0	7E
LD A, (mn)	0	0	1	1	1	0	1	0	3A
	← n →								
	← m →								
LD A, (IX+d)	1	1	0	1	1	1	0	1	DD
	0	1	1	1	1	1	1	0	7E
	← d →								
LD A, (IY+d)	1	1	1	1	1	1	0	1	FD
	0	1	1	1	1	1	1	0	7E
	← d →								

Description

The value in the memory location selected by aa is loaded to the Accumulator A. The flags are not affected. The memory address aa can be the contents of any of the Register pairs BC, DE, or HL, a direct address mn in the instruction, or the sum of the contents of an index Register IX or IY and a signed 8-bit displacement d.

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
LD A, (BC)	2	7 (4, 3)	6 (3, 3)
LD A, (DE)	2	7 (4, 3)	6 (3, 3)
LD A, (HL)	2	7 (4, 3)	6 (3, 3)
LD A, (mn)	4	13 (4, 3, 3, 3)	12 (3, 3, 3, 3)
LD A, (IX+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)
LD A, (IY+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)

Example

Index Register IY value is 25AFH. The instruction LD A, (IY + 19H) instructs the processor to calculate the address 25AFH + 19H, which is 25C8H. When the memory location 25C8H value is 39H, the result is 39H loads to the Accumulator A.

Load from Register I

LD A,I

Operation

A ← I

Format

1	1	1	0	1	1	0	1	ED
0	1	0	1	0	1	1	1	57

Description

The contents of the Interrupt Vector Register (I) are loaded to the Accumulator, and the P/V Flag is set to the state of the interrupt enable Flag IEF2.

Timing

M Cycles	Z80 T States	Z18x T States
2	9 (4, 5)	6 (3, 3)

Flags

Flags	Description
S	Set if Register I is Negative; reset otherwise
Z	Set if Register I is 0; reset otherwise
H	Reset
P/V	The value of IFF2
N	Reset
C	Not affected

Note: If an interrupt occurs during the execution, the Parity Flag is 0.

LD A,R — Load from Register R

Operation

A ← R

Format

1	1	1	0	1	1	0	1	ED
0	1	0	1	1	1	1	1	5F

Description

The contents of Memory Refresh Register (R) are loaded to the Accumulator, and the P/V Flag is set to the state of the interrupt enable Flag IEF2.

Timing

M Cycles	Z80 T States	Z18x T States
2	9 (4, 5)	6 (3, 3)

Flags

Flags	Description
S	Set if the value of Register R is Negative; reset otherwise
Z	Set if the value of Register R is 0; reset otherwise
H	Reset
P/V	The value of IFF2
N	Reset
C	Not affected

Note: If an interrupt occurs during the execution, the Parity Flag is 0.

Load Immediate (16 Bit) LD EE,MN

Operation

`ee ← mn`

Format

LD ss, mn

0	0	s	s	0	0	0	1
←			n				→
←			m				→

LD IX, mn

1	1	0	1	1	1	0	1	DD
0	0	1	0	0	0	0	1	21
←			n				→	
←			m				→	

LD IY, mn

1	1	1	1	1	1	0	1	FD
0	0	1	0	0	0	0	1	21
←			n				→	
←			m				→	

Description

The 16-bit immediate value `mn` in the instruction is loaded to Register `ee`. The flags are not affected. The `ee` can be any of the Register pairs `BC`, `DE`, or `HL`, the `SP`, or an index Register `IX` or `IY`. The `n` is loaded to the

LD EE,MN

Load Immediate (16 Bit)

less-significant half of the register. In the first form shown above, ss is encoded as follows:

Register	Hex Value (ss)
BC	00
DE	01
HL	10
SP	11

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
LD ss, mn	3	10 (4, 3, 3)	9 (3, 3, 3)
LD IX, mn	4	14 (4, 4, 3, 3)	12 (3, 3, 3, 3)
LD IY, mn	4	14 (4, 4, 3, 3)	12 (3, 3, 3, 3)

Example

At LD SP, 7FFEH, the SP value is 7FFEH.

Load from Memory (16 Bit) LD EE,(MN)

Operation

$ee_L \leftarrow (mn)$, $ee_H \leftarrow (mn+1)$

Format

LD HL, (mn)	0	0	1	0	1	0	1	0	2A
	←			n				→	
	←			m				→	

LD ss, (mn)	1	1	1	0	1	1	0	1	ED
	0	1	s	s	1	0	1	1	
	←			n				→	
	←			m				→	

LD IX (mn)	1	1	0	1	1	1	0	1	DD
	0	0	1	0	1	0	1	0	2A
	←			n				→	
	←			m				→	

LD IY, (mn)	1	1	1	1	1	1	0	1	FD
	0	0	1	0	1	0	1	0	2A
	←			n				→	
	←			m				→	

LD EE,(MN) Load from Memory (16 Bit)

Description

The contents of memory location mn are loaded to the less-significant half of 16-bit Register ee, and the contents of location mn+1 are loaded to the more-significant half. The flags are not affected. The ee can be any of the Register pairs BC, DE, or HL, the SP, or an index Register IX or IY. The n following the opcode(s) is the less-significant byte of the memory address. In the second form shown above, ss is encoded as follows:

Register	Hex Value (ss)
BC	00
DE	01
HL	10 (The first form shown above is preferred.)
SP	11

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
LD HL, (mn)	5	16 (4, 3, 3, 3, 3)	15 (3, 3, 3, 3, 3)
LD ss, (mn)	6	20 (4, 4, 3, 3, 3, 3)	18 (3, 3, 3, 3, 3, 3)
LD IX, (mn)	6	20 (4, 4, 3, 3, 3, 3)	18 (3, 3, 3, 3, 3, 3)
LD IY, (mn)	6	20 (4, 4, 3, 3, 3, 3)	18 (3, 3, 3, 3, 3, 3)

Example

Memory location 6789H value is 34H, 3. Location 678AH value is 12H. At LD BC, (6789H), the B value is 12H and C value is 34H.

Load to Register I — LD I,A

Operation

`I ←A`

Format

1	1	1	0	1	1	0	1	ED
0	1	0	0	0	1	1	1	47

Description

The contents of the Accumulator are loaded to the Interrupt Control Vector Register (`I`). No flags are affected.

Timing

M Cycles	Z80 T States	Z18x T States
2	9 (4, 5)	6 (3, 3)

LD M,N

Load Immediate (8 Bit)

Operation

m ← n

Format

LD r, n	0	0	← r →			1	1	0	
	← n →								
LD (HL), n	0	0	1	1	0	1	1	0	36
	← n →								
LD (IX+d), n	1	1	0	1	1	1	0	1	DD
	0	0	1	1	0	1	1	0	36
	← d →								
	← n →								
LD (IY+d), n	1	1	1	1	1	1	0	1	FD
	0	0	1	1	0	1	1	0	36
	← d →								
	← n →								

Description

The immediate value n is loaded to the m operand. The flags are not affected. The m can be any of a destination Register r, a memory location selected by the contents of Register pair HL, or a memory location selected by the sum of the contents of an index Register IX or IY and a signed 8-bit displacement d. In the register form, r selects a destination register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
LD r, n	2	7 (4, 3)	6 (3, 3)
LD (HL), n	3	10 (4, 3, 3)	9 (3, 3, 3)
LD (IX+d), n	5	19 (4, 4, 3, 3, 2 int, 3)	15 (3, 3, 3, 3, 3)
LD (IY+d), n	5	19 (4, 4, 3, 3, 2 int, 3)	15 (3, 3, 3, 3, 3)

Example

The value of Register pair HL is 4444H. The instruction LD (HL), 28H loads 28H to memory location 4444H.

LD M,R

Load from Register (8 Bit)

Operation

`m ← r`

Format

LD r, r	0	1	← r →			← r →			
LD (HL), r	0	1	1	1	0	← r →			
LD (IX+d), r	1	1	0	1	1	1	0	1	DD
	0	1	1	1	0	← r →			
	← d →								
LD (IY+d), r	1	1	1	1	1	1	0	1	FD
	0	1	1	1	0	← r →			
	← d →								

Description

The value in the source Register `r` is loaded to the `m` operand. The flags are not affected. The `m` can be any of a destination Register `r'`, a memory location selected by the contents of Register pair `HL`, or a memory location selected by the sum of the contents of an index Register `IX` or `IY` and a signed 8-bit displacement `d`. The `r` selects a source register, and in the register-to-register form `r` selects a destination register as follows:

Load from Register (8 Bit) LD M,R

Register	Hex Value (r, r')
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
LD r', r	1	4	4 (3, 1 int)
LD (HL), r	2	7 (4, 3)	7 (3, 1 int, 3)
LD (IX+d), r	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)
LD (IY+d), r	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)

Example

The `C` Register value is `1CH`. The `IX` value is `3100H`. At execution of the instruction `(IX + 6), C,` the memory location `3106H` value is `1CH`.

LD R,A

Load to Register R

Operation

R ← A

Format

1	1	1	0	1	1	0	1	ED
0	1	0	0	1	1	1	1	4F

Description

The value of the Accumulator is loaded to the Memory Refresh Register R. No flags are affected.

Timing

M Cycles	Z80 T States	Z18x T States
2	9 (4, 5)	6 (3, 3)

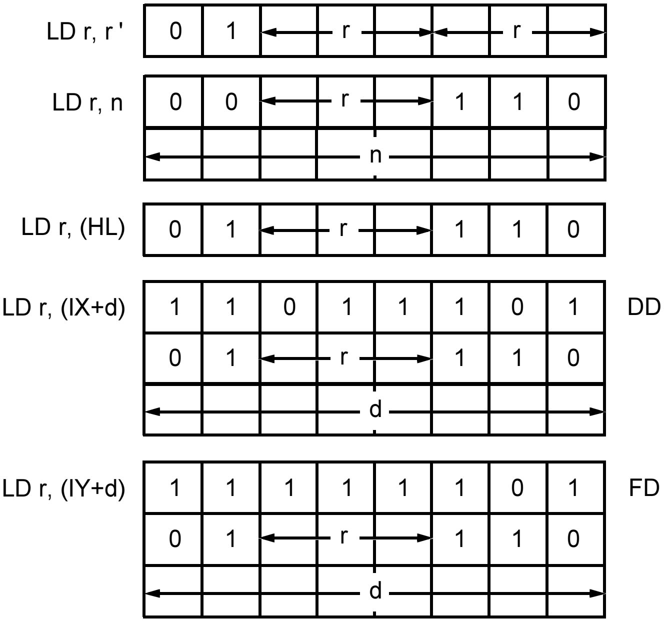
Load to Register (8 Bit)

LD R,S

Operation

r ← s

Format



Description

The value of the s operand is loaded to the destination Register r. The flags are not affected. The s can be any of a source Register r, an immediate value n in the instruction, a memory location selected by the value of HL, or a memory location selected by the sum of the value of an index Register IX or IY and a signed 8-bit displacement d. The r selects a destination register, and in the register-to-register form r selects a source register, as follows:

Register	Hex Value (r, r')
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
LD r, r'	1	4	4 (3, 1 int)
LD r, n	2	7 (4, 3)	6 (3, 3)
LD r, (HL)	2	7 (4, 3)	6 (3, 3)
LD r, (IX+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)
LD r, (IY+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)

Example

The index Register IY value is 25AFH. The instruction D B, (IY + 19H) makes the processor calculate the address 25AFH + 19H, which is 25C8H. When memory location 25C8H value is 39H, the result is 39H loads to the B Register.

Load Stack Pointer LD SP,RR

Operation

SP ← rr

Format

LD SP, HL	1	1	1	1	1	0	0	1	F9

LD SP, IX	1	1	0	1	1	1	0	1	DD
	1	1	1	1	1	0	0	1	F9

LD SP, IY	1	1	1	1	1	1	0	1	FD
	1	1	1	1	1	0	0	1	F9

Description

The value of the 16-bit Register `rr` loads to the `SP`. The flags are not affected. The `rr` can be Register pair `HL` or an index Register `IX` or `IY`.

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
LD SP, HL	1	6 (4, 2 int)	4 (3, 1 int)
LD SP, IX	2	10 (4, 4, 2 int)	7 (3, 3, 1 int)
LD SP, IY	2	10 (4, 4, 2 int)	7 (3, 3, 1 int)

Example

The IX value is AB04H. At execution of the instruction LD SP, the IX Register pair SP value is AB04H.

Load and Decrement LDD

Operation

(DE) ← (HL), DE ← DE–1, HL ← HL–1, BC ← BC –1

Format

1	1	1	0	1	1	0	1	ED
1	0	1	0	1	0	0	0	A8

Description

This instruction transfers a byte of data from the memory location addressed by the value of Register pair HL to the memory location addressed by the value of the DE Register pair. Then contents of both Register pairs and the BC are decremented. The P/V Flag is set to indicate if BC is decremented to 0.

Timing

M Cycles	Z80 T States	Z18x T States
4	16 (4, 4, 3, 5)	12 (3, 3, 3, 3)

Flags

Flag	Value
S	Not affected
Z	Not affected
H	Reset
P/V	Reset if BC is 0000; set otherwise
N	Reset
C	Not affected

Example

The Register pair `HL` value is `1111`. The memory location `1111H` value is the byte `88H`. The `DE` Register pair value is `2222H`. The memory location `2222H` value is byte `66H`, and the `BC` Register pair value is `0007H`. At execution of the instruction `LDD,` the following values are in register pairs and memory addresses:

Register Pair	Memory Address
HL	1110H
(1111H)	88H
DE	2221H
(2222H)	88H
BC	0006H

Load, Decrement, Repeat LDDR

Operation

```
(DE)←(HL), DE←DE–1, HL ←HL–1, BC ← BC–1; repeat
until (BC); is 0
```

Format

1	1	1	0	1	1	0	1	ED
1	0	1	1	1	0	0	0	B8

Description

This instruction transfers a byte of data from the memory location addressed by the value of Register pair HL, to the memory location addressed by the value of the DE Register pair. Then both of these registers and the BC (Byte Counter) are decremented. When decrementing causes BC to go to 0, the instruction is terminated. When BC is not 0, the instruction is repeated. Interrupts are recognized and two refresh cycles are executed after each data transfer. When BC is set to 0 prior to instruction execution, the instruction loops through 64 KB.

Timing

For each repetition while BC is not 0:

M Cycles	Z80 T States	Z18x T States
4	21 (4, 4, 3, 5, 5 int)	14 (3, 3, 3, 3, 2 int)

While BC is 0:

M Cycles	Z80 T States	Z18x T States
4	16 (4, 4, 3, 5)	12 (3, 3, 3, 3)

Flags

Flag	Value
S	Not Affected
Z	Not Affected
H	Reset
P/V	Reset
N	Reset

Example

The Register pair HL value is 1114H, DE value is 2225H, BC value is 0003H, and memory location values are:

Register Pair	Memory Address	Register Pair	Memory Address
(1114H)	A5H	(2225H)	C5H
(1113H)	36H	(2224H)	59H
(1112H)	88H	(2223H)	66H

At execution of the instruction LDDR, the value of Register pairs and memory locations are:

Register Pair	Memory Address
HL	1111H
DE	2222H
BC	0000H

Register Pair	Memory Address	Register Pair	Memory Address
(1114H)	A5H	(2225H)	A5H
(1113H)	36H	(2224H)	36H
(1112H)	88H	(2223H)	88H

Load and Increment LDI

Operation

(DE) ← (HL), DE ← DE + 1, HL ← HL + 1, BC ← BC –1

Format

1	1	1	0	1	1	0	1	ED
1	0	1	0	0	0	0	0	A0

Description

A byte of data is transferred from the memory location addressed by the value of Register pair HL, to the memory location addressed by the value of DE. Then both these register pairs are incremented, BC is decremented, and the P/V Flag indicates if BC decremented to 0.

Timing

M Cycles	Z80 T States	Z18x T States
4	16 (4, 4, 3, 5)	12 (3, 3, 3, 3)

Flags

Flag	Value
S	Not affected
Z	Not affected
H	Reset
P/V	Reset if BC is 0000; set otherwise
N	Reset
C	Not affected

Example

The Register pair HL value is 1111H. The memory location 1111H value is 88H. The DE value is 2222H. The memory location 2222H value is 66H, and BC value is 0007H. At execution of the instruction LDI, the values are:

Register Pair	Memory Address
HL	1112H
(1111H)	88H
DE	2223H
(2222H)	88H
BC	0006H

Load, Increment, Repeat LDIR

Operation

```
(DE) ← (HL), DE ← DE + 1, HL ← HL + 1, BC ← BC -1; repeat
until (BC) is 0
```

Format

1	1	1	0	1	1	0	1	ED
1	0	1	1	0	0	0	0	B0

Description

This instruction transfers a byte of data from the memory location addressed by the value of Register pair HL to the memory location addressed by DE. Then both these register pairs are incremented and BC is decremented. When decrementing causes the BC to go to 0, the instruction is terminated. When BC is not 0, the instruction is repeated. Interrupts are recognized and two refresh cycles are executed after each data transfer. When BC is set to 0 prior to instruction execution, the instruction loops through 64 KB.

Timing

For each repetition while BC is not 0:

M Cycles	Z80 T States	Z18x T States
4	21 (4, 4, 3, 5, 5 int)	14 (3, 3, 3, 3, 2 int)

When BC is 0:

M Cycles	Z80 T States	Z18x T States
4	16 (4, 4, 3, 5)	12 (3, 3, 3, 3)

Flags

Flag	Value
S	Not affected
Z	Not affected
H	Reset
P/V	Reset
N	Reset
C	Not affected

Example

The Register pair `HL` value is `1111H`. The `DE` value is `2222H`. `BC` is `0003H`. Memory locations values are:

Register Pair	Memory Address	Register Pair	Memory Address
(1111H)	88H	(2222H)	66H
(1112H)	36H	(2223H)	59H
(1113H)	A5H	(2224H)	C5H

At execution of the instruction `LDIR`, the value of register pairs and memory locations are:

Register Pair	Memory Address
HL	1114H
DE	2225H
BC	0000H

Register Pair	Memory Address	Register Pair	Memory Address
(1111H)	88H	(2222H)	88H
(1112H)	36H	(2223H)	36H
(1113H)	A5H	(2224H)	A5H

Multiply **MLT SS**

Operation

$ss \leftarrow ss_L * ss_H$

Format

1	1	1	0	1	1	0	1	ED
0	1	s	s	1	1	0	0	

Description

The two 8-bit halves of a register pair are multiplied and the product is stored in the same register pair. Multiplication is unsigned, and the flags are not affected. The `ss` is encoded as follows:

Register	Hex Value (ss)
BC	00
DE	01
HL	10
SP	11

Timing

M Cycles	Z80 T States	Z18x T States
2	NA	17 (3, 3, 11 int)

Example

The `H` value is `08H,` and `L` value is `AAH`. At execution of the instruction `MLT HL,` Register pair `HL` value is `0550H`. (In decimal, this is 8 * 170, or 1360.)

NEG Negate

Operation

A ← – A

Format

1	1	1	0	1	1	0	1	ED
0	1	0	0	0	1	0	0	44

Description

The value of the Accumulator are negated (twos complement). This is the same as subtracting the value of the Accumulator from `0`. The `80H` is left unchanged.

Timing

M Cycles	Z80 T States	Z18x T States
2	8 (4, 4)	6 (3, 3)

Flags

Flag	Value
S	Set if the result is Negative; reset otherwise
Z	Set if the result is 0; reset otherwise
H	Set if Borrow from Bit 4; reset otherwise
P/V	Set if Accumulator is 80H before operation; reset otherwise
N	Set
C	Reset if Accumulator is 00H before operation; set otherwise

Example

The value of the Accumulator is `98 (-10410).`

1	0	0	1	1	0	0	0

At execution of the instruction `NEG`, the Accumulator value is `68H` `(10410).`

0	1	1	0	1	0	0	0

NOP — No Operation

Operation

None

Format

0	0	0	0	0	0	0	0	00

Description

The CPU performs no operation during this machine cycle.

Timing

M Cycles	Z80 T States	Z18x T States
1	4	3

Inclusive Or

OR A,S

Operation

A ← A OR s

Format

OR A, r	1	0	1	1	0	← r →			
OR A, n	1	1	1	1	0	1	1	0	F6
	← n →								
OR A, (HL)	1	0	1	1	0	1	1	0	B6
OR A, (IX+d)	1	1	0	1	1	1	0	1	DD
	1	0	1	1	0	1	1	0	B6
	← d →								
OR A, (IY+d)	1	1	1	1	1	1	0	1	FD
	1	0	1	1	0	1	1	0	B6
	← d →								

Description

A logical OR operation is performed using the s operand and the value in the Accumulator A. The result is stored in A, and the flags are set as described below. The s can be any of a Register r, an immediate value n in the instruction, a memory location selected by the value of Register pair HL, or a memory location selected by the sum of the value of an index register, IX or IY, and a signed 8-bit displacement d. In the register form, r selects a source register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
OR A, r	1	4	4 (3 + 1 int)
OR A, n	2	7 (4, 3)	6 (3, 3)
OR A, (HL)	2	7 (4, 3)	6 (3, 3)
OR A, (IX+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)
OR A, (IY+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)

Flags

Flag	Value
S	Set if the result is Negative; reset otherwise
Z	Set if the result is 0; reset otherwise
H	Reset
P/V	Set if the resulting parity is Even; reset otherwise
N	Reset
C	Reset

Example

The H Register value is 48H. The Accumulator value is 12H. At execution of OR A, H, the Accumulator value is 5AH.

ODTM Output and Decrement (page 0)

Operation

`(0, (C)) ← (HL), HL ← HL - 1, C ← C - 1, B ←B-1`

Format

1	1	1	0	1	1	0	1	ED
1	0	0	0	1	0	1	1	8B

Description

A byte is read from memory at the address in Register pair HL. It is written to the output port selected by placing the value of C on A7-A0 and zeroes on A15-A8 (if any). Then the memory address in Register pair HL and the I/O address in C are both decremented, a byte count in B is decremented, and the Z flag is set to indicate if B has been decremented to 0. Other flags are affected as shown below.

Timing

M Cycles	T States (180 reg)	T States (other 18x)
4	14 (3, 3, 3, 3, 2 int)	15 (3, 3, 3, 4, 2 int)

Output and Decrement (page 0) ODTM

Flags

Flag	Value
S	Set if new (B) Negative; reset otherwise
Z	Set if new (B) 0; reset otherwise
H	Set if Borrow from Bit 4 of B; reset otherwise
P/V	Set if parity of new (B) is Even; cleared to 0 otherwise
N	Set if Bit 7 of data byte is 1; reset otherwise
C	Set if B 0 to FF; reset otherwise

Example

The Register pair HL value is 4200H. C value is 80H. B value is 10H. Memory location 4200H value is 81H. At execution of the instruction OTDM, 81H is written to output port 0080H. The value of Register pair HL is 41FFH. C is 7FH. B is 0FH. The flags are set as follows:

Flag	Value
S	0
Z	0
C	0
H	1
P/V	1
N	1

OTDMR Output, Decremen, Repeat (page 0)

Operation

`(0, (C)) ← (HL), HL ← HL –1, C ← C –1, B ← B – 1, repeat until B is 0`

Format

1	1	1	0	1	1	0	1	ED
1	0	0	1	1	0	1	1	9B

Description

A byte is read from memory at the address in Register pair HL, and then written to the output port selected by placing the value of C on A7-A0 and zeroes on A15-A8 (if any). Then the memory address in Register pair HL and the I/O address in C are both decremented, and a byte count in B is decremented. These steps are repeated until B is decremented to 0. An initial B value of 0 causes 256 bytes to transfer from memory to output ports. The flags are affected as shown below.

Interrupts may occur after any cycle, and the instruction continues transparently on return from the interrupt service routine. This instruction in useful for initializing a block of output registers, such as the Z80180 core registers.

Timing

	M Cycles	T States (180 reg)	T States (other 18x)
B is Nonzero	4	16 (3, 3, 3, 3, 4 int)	17 (3, 3, 3, 4, 4 int)
New (B) 0	4	14 (3, 3, 3, 3, 2 int)	15 (3, 3, 3, 4, 2 int)

Output, Decremen, Repeat (page 0) OTDMR

Flags

Flag	Value
S	Reset
Z	Set
H	Reset
P/V	Set
N	Set if Bit 7 of last data byte is 1; reset otherwise
C	Reset

Example

The Register pair HL value is 4200H. C value is 80H. B value is 3. Memory locations 41FEH to 4200H value is 13H 00H 81H. At execution of the instruction OTDMR, the value 81H is written to output port 0080H. The value 00 is written to port 007FH. The value 13H is written to port 007EH. Register pair HL value is 41FDH. C value is 7DH, and B value is 0.

OTDR Output, Decremen, Repeat

Operation

(C) ← (HL), B ← B – 1, HL ← HL -1, repeat until B is 0

Format

1	1	1	0	1	1	0	1	ED
1	0	1	1	1	0	1	1	BB

Description

The value of Register pair HL are placed on the address bus to select a location in memory. The byte read from this memory location is temporarily stored in the CPU. When the byte counter (B) is decremented, the value of Register C are placed on the bottom half (A7 through A0) of the address bus to select the I/O device at one of 256 possible ports. Register B may be used as a byte counter, and its decremented value is placed on the top half (A15 through A8) of the address bus. The byte read from memory is placed on the data bus and written to the selected peripheral device. Then Register pair HL is decremented. When the decremented B Register is not 0, the instruction is repeated. When B has gone to 0, the instruction is terminated. Interrupts are recognized and refresh cycles may be executed after each data transfer. When B is 0 prior to instruction execution, the instruction outputs 256 bytes of data.

Note: This instruction is not completely compatible with peripheral devices that decode A15-8 as part of a 16-bit I/O address.

Output, Decremen, Repeat OTDR

Timing

When B is not 0:

M Cycles	Z80 T States	Z180 Register T States	Z18x Other Register T States
4	21 (4, 5, 3, 4, 5 int)	14 (3, 3, 3, 3, 2 int)	15 (3, 3, 3, 4, 2 int)

When B is 0:

M Cycles	Z80 T States	Z180 Register T States	Z18x Other Register T States
4	16 (4, 5, 3, 4)	12 (3, 3, 3, 3)	13 (3, 3, 3, 4)

Flags

Flag	Value
S	Unknown
Z	Set
H	Unknown
P/V	Unknown
N	Set
C	Not affected

Example

The value of Register C is 07H. The value of Register B is 03H. The value of Register pair HL is 1000H, and memory location values are:

Register	Hex Value (r)
0FFEH	51H
0FFFH	A9H
1000H	03H

At the execution of OTDR, the value of Register pair HL is 0FFDH. Register B is 0. Three bytes are written to the peripheral device mapped to I/O port address 07H in the following sequence:

```
03H (first)
A9H
51H (third)
```

Output and Incremen (page 0) OTIM

Operation

(0, (C)) ← (HL), HL ←HL +1, C ← C+1, B ←B -1

Format

1	1	1	0	1	1	0	1	ED
1	0	0	0	0	0	1	1	83

Description

A byte is read from memory at the address in Register pair HL, and then written to the output port selected by placing the value of C on A7-A0 and zeroes on A15-A8 (if any). Then the memory address in Register pair HL and the I/O address in C are both incremented, Register B is decremented, and the Z flag is set to indicate if B is decremented to 0. Other flags are affected as shown below.

Timing

M Cycles	T States (180 Reg)	T States (Other 18x)
4	14 (3, 3, 3, 3, 2 int)	15 (3, 3, 3, 4, 2 int)

OTIM Output and Incremen (page 0)

Flags

Flag	Value
S	Set if new (B) is Negative; reset otherwise
Z	Set if new (B) is 0; reset otherwise
H	Set if Borrow from Bit 4 of B; reset otherwise
P/V	Set if parity of new (B) is Even; cleared to 0 otherwise
N	Set if Bit 7 of data byte is 1; reset otherwise
C	Set if B is 0 to FF; reset otherwise

Example

Register pair HL value is 4200H. C value is 80H. B value is 10H. Memory location 4200H value is 81H. At execution of the instruction OTDM, the value 81H is written to output port 0080H. Register pair HL value is 4201H. C value is 81H. B value is 0FH. The flags are set as follows:

Flag	Value
S	0
Z	0
C	0
H	1
P/V	1
N	1

Output, Incremen, Repeat (page 0) OTIMR

Operation

```
(0, (C)) ← (HL), HL ←HL+1, C ← C+1, B ←B - 1, repeat
until B is 0
```

Format

1	1	1	0	1	1	0	1	ED
1	0	0	1	0	0	1	1	93

Description

A byte is read from memory at the address in Register pair HL, and then written to the output port selected by placing the value of C on A7-A0 and zeroes on A15-A8 (if any). Then the memory address in Register pair HL and the I/O address in C are both incremented, and B is decremented. These steps are repeated until B decrements to 0. An initial B value of 0 causes 256 bytes to transfer from memory to output ports. The flags are affected as shown below.

Interrupts may occur after any cycle, and the instruction continues transparently at return from the interrupt service routine. This instruction in useful for initializing a block of output registers, such as the Z80180 core registers.

OTIMR Output, Incremen, Repeat (page 0)

Timing

	M Cycles	T States (180 reg)	T States (other 18x)
While B is nonzero	4	16 (3, 3, 3, 3, 4 int)	17 (3, 3, 3, 4, 4 int)
New (B) 0	4	14 (3, 3, 3, 3, 2 int)	15 (3, 3, 3, 4, 2 int)

Flags

Flag	Value
S	Reset
Z	Set
H	Reset
P/V	Set
N	Set if Bit 7 of last data byte is 1; reset otherwise
C	Reset

Example

The Register pair `HL` value is `4200H`. `C` value is `80H`. `B` value is `3`. Memory locations `4200H` to `4202H` value is `13H 00H 81H`. At execution of the instruction, `OTIMR`, the value `13H` is written to output port `0080H`. The value `00` is written to port `0081H`. The value `81H` is written to port `0082H`. Register pair `HL` value is `4203H`. `C` value is `83H`, and `B` value is `0`.

Output, Decremen, Repeat OTIR

Operation

B ← B – 1, (C) ← (HL), HL ← HL + 1, repeat until B is 0

Format

1	1	1	0	1	1	0	1	ED
1	0	1	1	0	0	1	1	B3

Description

The value of Register pair HL is placed on the address bus to select a location in memory. The byte contained in this memory location is read and temporarily stored in the CPU. Then, after B is decremented, the value of Register C is placed on the bottom half (A7 through A0) of the address bus to select the I/O device at one of 256 possible ports. The decremented value of Register B is placed on the top half (A15 through A8) of the address bus. The byte read from memory is placed on the data bus and written to the selected peripheral device. Register pair HL is incremented. When the decremented B Register is not 0, the instruction is repeated. When B is 0, the instruction terminates. Interrupts are recognized and refresh cycles may be executed after each data transfer. When B is 0 prior to instruction execution, the instruction outputs 256 bytes of data.

Note: This instruction is not completely compatible with peripheral devices that decode A15-8 as part of a 16-bit I/O address.

Timing

When B is not 0:

M Cycles	Z80 T States	Z180 Register T States	Z18x Other Register T States
4	21 (4, 5, 3, 4, 5 int)	14 (3, 3, 3, 3, 2 int)	15 (3, 3, 3, 4, 2 int)

When B is 0:

M Cycles	Z80 T States	Z180 Register T States	Z18x Other Register T States
4	16 (4, 5, 3, 4)	12 (3, 3, 3, 3)	13 (3, 3, 3, 4)

Flags

Flag	Value
S	Unknown
Z	Set
H	Unknown
P/V	Unknown
N	Set
C	Not affected

Example

Register C value is 07H. Register B value is 03H. Register pair HL value is 1000H. Memory locations have the following value:

Location	Contents
1000H	51H
1001H	A9H
1002H	03H

At execution of OTIR, Register pair HL value is 1003H. Register B value is 0. These bytes are written to the peripheral device mapped to I/O port address 07H in the following sequence:

```
51H (first)
A9H
03H (third)
```

OUT (C),R

Output

Operation

(C) ← r

Format

1	1	1	0	1	1	0	1	ED
0	1	←	r	→	0	0	1	

Description

The value of Register `C` are placed on the bottom half (`A7` through `A0`) of the address bus to select the I/O device at one of 256 possible ports. The value of Register `B` is placed on the top half (`A15` through `A8`) of the address bus. Then the byte contained in Register `r` is placed on the data bus and written to the selected peripheral device. Register `r` identifies any of the CPU registers shown in the following table:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

M Cycles	Z80 T States	Z180 Register T States	Z18x Other Register T States
4	12 (4, 4, 4)	10 (3, 3, 1 int, 3)	11 (3, 3, 1 int, 4)

Example

The value of Register `C` is `01H`. The value of Register `B` is `02H`. The value of Register `D` is `5AH`. At execution of `OUT (C),D`, the byte `5AH` is written to the peripheral device mapped to I/O port address `0201H`. When the peripheral device ignores `A15` through `A8`, the statement above should read `the peripheral device mapped to I/O port address 01H`.

OUT (N),A — Output

Operation

(n) ← A

Format

1	1	0	1	0	0	1	1	D3
← n →								

Description

The operand n is placed on the bottom half (A7 through A0) of the address bus to select the I/O device at one of 256 possible ports. The value of the Accumulator (A) appear on the top half (A15 through A8) of the address bus. Then the byte contained in the Accumulator is placed on the data bus and written to the selected peripheral device.

Note: This instruction is not completely compatible with peripherals that decode A15 through A8 as part of a 16-bit I/O address.

Timing

M Cycles	Z80 T States	Z180 Register T States	Z18x Other Register T States
3	11 (4, 3, 4)	10 (3, 3, 1 int, 3)	11 (3, 3, 1 int, 4)

Example

The value of the Accumulator is 23H. At execution of OUT (01H), A, the byte 23H is written to the peripheral device mapped to I/O port address 01H.

Output to Page 0 OUT0 (N),R

Operation

`(0, n) ← r`

Format

1	1	1	0	1	1	0	1	ED
0	0	←	r	→	0	0	1	
←				n			→	

Description

The operand `n` is placed on `A7` through `A0` with `0` on `A15` through `A8` to select an output port. The value of the register selected by `r`, as shown in the following table, are then written to the selected output port (if any). The flags are not affected.

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

M Cycles	T States (180 Register)	T States (other 18x Registers)
4	13 (3, 3, 3, 3, 1 int)	14 (3, 3, 3, 4, 1 int)

Example

The value of Input/Output Control Register (IOCR) is `1FH`. The 16-bit I/O address of its `ASCI0` transmit data register is `0006H`, and the value of A is `41H`. At execution of `OUT0 (6), A`, the byte `41H` is written to `ASCI0` for serial transmission. Under the same conditions, execution of `OUT (6), A,` does **not** write the byte to `ASCI0`. Instead, it is written to 16-bit I/O address `4106H`.

Output and Decrements OUTD

Operation

```
B← B–1, (C) ← (HL), HL ←HL– 1
```

Format

1	1	1	0	1	1	0	1	ED
1	0	1	0	1	0	1	1	AB

Description

The value of Register pair HL is placed on the address bus to select a location in memory. The byte contained in this memory location is read and temporarily stored in the CPU. When Register B is decremented, the value of Register C is placed on the bottom half (A7 through A0) of the address bus to select the I/O device at one of 256 possible ports. The decremented value of Register B is placed on the top half (A8 through A15) of the address bus. Next the byte to be output is placed on the data bus and written to the selected peripheral device. The Register pair HL is decremented.

Note: This instruction is not completely compatible with peripheral devices that decode A15 through A8 as part of a 16-bit I/O address.

Timing

M Cycles	Z80 T States	Z180 Register T States	Z18x Other Register T States
4	16 (4, 5, 3, 4)	12 (3, 3, 3, 3)	13 (3, 3, 3, 4)

Flags

Flag	Value
S	Unknown
Z	Set if B is 1 is 0; reset otherwise
H	Unknown
P/V	Unknown
N	Set
C	Not affected

Example

The value of Register `C` is `07H`. The value of Register `B` is `10H`. The value of Register pair `HL` is `1000H`. The value of memory location `1000H` is `59H`. At execution of `OUTD`, Register `B` is `0FH`. Register pair `HL` is `0FFFH`. The byte `59H` is written to the peripheral device mapped to I/O port address `07H`.

Output and Increment OUTD

Operation

`B← B– 1, (C)←(HL), HL←HL +1`

Format

1	1	1	0	1	1	0	1	ED
1	0	1	0	0	0	1	1	A3

Description

The value of Register pair `HL` are placed on the address bus to select a location in memory. The byte contained in this memory location is temporarily stored in the CPU. After Register `B` is decremented, the value of Register `C` are placed on the bottom half (`A7` through `A0`) of the address bus to the select the I/O device at one of 256 possible ports. The decremented value of Register `B` is placed on the top half (`A15` through `A8`) of the address bus. The byte to be output is placed on the data bus and written to the selected peripheral device. Register pair `HL` is incremented.

Note: This instruction is not completely compatible with peripheral devices that decode `A15` through `A8` as part of a 16-bit I/O address.

Timing

M Cycles	Z80 T States	Z180 Register T States	Z18x Other Register T States
4	16 (4, 5, 3, 4)	12 (3, 3, 3, 3)	13 (3, 3, 3, 4)

OUTD Output and Increment

Flags

Flag	Value
S	Unknown
Z	Set if B 1 is 0; reset otherwise
H	Unknown
P/V	Unknown
N	Set
C	Not affected

Example

The value of Register `C` is `07H`. The value of Register B is `10H`. The value of Register pair `HL` is `1000H`. The value of memory address `1000H` is `59H`. At execution of OUTI, Register `B` is `0FH`. Register pair `HL` is `1001H`. Byte `59H` is written to the peripheral device mapped to I/O port address `07H`.

Push from Stack — POP PP

Operation

$pp_L \leftarrow (SP),\ pp_H \leftarrow (SP+1),\ SP \leftarrow SP+2$

Format

POP qq	1	1	z	z	0	0	0	1	
POP IX	1	1	0	1	1	1	0	1	DD
	1	1	1	0	0	0	0	1	E1
POP IY	1	1	1	1	1	1	0	1	FD
	1	1	1	0	0	0	0	1	E1

Description

Two bytes are popped from the stack in external memory, to 16-bit Register pp. First a byte is read from memory at the address in the SP, and is loaded to the less significant half of pp. Then SP is incremented by one, another byte is read from memory at the new value of SP, and is loaded to the more-significant half of pp. SP is incremented again. No flags are affected unless the operand is AF. There is no checking for stack underflow.

Register pp can have the value AF (the accumulator A as most significant byte, flags as least significant byte), any of the Register pairs BC, DE, or

HL, or an index Register IX or IY. In the first form shown above, qq is encoded as follows:

Register	Hex Value (qq)
BC	00
DE	01
HL	10
AF	11

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
POP qq	3	10 (4, 3, 3)	9 (3, 3, 3)
POP IX	4	14 (4, 4, 3, 3)	12 (3, 3, 3, 3)
POP IY	4	14 (4, 4, 3, 3)	12 (3, 3, 3, 3)

Flags

Flags are set only if the operand is AF.

Example

The SP value is 1000H. Memory location 1000H value is 55H. Location 1001H value is 33H. At execution of POP IX, the IX Register value is 3355H. SP value is 1002H.

Operation

(SP-1) ← ppH, (SP-2) ← ppL, SP ← SP-2

Format

PUSH qq	1	1	q	q	0	1	0	1	

PUSH IX	1	1	0	1	1	1	0	1	DD
	1	1	1	0	0	1	0	1	E5

PUSH IY	1	1	1	1	1	1	0	1	FD
	1	1	1	0	0	1	0	1	E5

Description

The value of the 16-bit Register pp are pushed to the stack in external memory. First, the value of the SP are decremented by 1, and the value of the more-significant byte of Register pp are stored in memory at the new value of SP. Then SP is decremented again, and the value of the less-significant byte of the Register pp are stored in memory at the new value of SP. No flags are affected and there is no checking for stack Overflow.

The pp can be AF (the accumulator A as MSB, flags as LSB), any of the Register pairs BC, DE, or HL, or an index Register IX or IY. In the first form shown above, qq is encoded as follows:

Register	Hex Value (qq)
BC	00
DE	01
HL	10
AF	11

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
PUSH qq	3	11 (4, 1 int, 3, 3)	11 (3, 2 int, 3, 3)
PUSH IX	4	15 (4, 4, 1 int, 3, 3)	14 (3, 3, 2 int, 3, 3)
PUSH IY	4	15 (4, 4, 1 int, 3, 3)	14 (3, 3, 2 int, 3, 3)

Example

The Accumulator A value is 22H. The flags contain 11H. The SP value is 1007H. At execution of PUSH AF, memory address 1006H value is 22H. Address 1005H value is 11H. SP value is 1005H.

Operation

```
m ← m and not (2 ^b)
```

Format

RES b, r	1	1	0	0	1	0	1	1	CB
	1	0	← b →			← r →			

RES b, (HL)	1	1	0	0	1	0	1	1	CB
	1	0	← b →			1	1	0	

RES b, (IX+d)	1	1	0	1	1	1	0	1	DD
	1	1	0	0	1	0	1	1	CB
	← d →								
	1	0	← b →			1	1	0	

RES b, (IY+d)	1	1	1	1	1	1	0	1	FD
	1	1	0	0	1	0	1	1	CB
	← d →								
	1	0	← b →			1	1	0	

Description

Bit b in operand m is reset to 0. No flags are affected. The b can be 0 for the least significant bit through Bit 7 for the most significant bit. The m can be a Register r, a memory location selected by the value of Register pair HL, or a memory location selected by the sum of the value of an index Register IX or IY and a signed 8-bit displacement d. In the register form, r selects the register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
RES r	4	8 (4, 4)	7 (3, 3, 1 int)
RES (HL)	4	15 (4, 4, 4, 3)	12 (3, 3, 3, 1 int, 3)
RES (IX+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)
RES (IY+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)

Example

At execution of RES 6, D, Bit 6 in Register D resets. Bit 6 has the value of 40H.

Operation

```
PCL ← (SP), PCH ←(SP+1), SP ← SP +2
```

Format

1	1	0	0	1	0	0	1	C9

Description

The byte at the memory location specified by the value of the SP is isolated to the low order 8 bits of the PC. SP is then incremented and the byte at the memory location specified by the new value of SP is fetched and loaded to the higher order 8 bits of PC. Finally, SP is incremented again. This instruction is normally used to return to the calling sequence at the completion of a routine entered by a CALL instruction.

Timing

M Cycles	Z80 T States	Z18x T States
3	10 (4, 3, 3)	9 (3, 3, 3)

Example

The value of the Program Counter is 3535H. The value of the SP is 2000H. The value of memory location 2000H is B5H. The value of memory location 2001H is 18H. At execution of RET, the value of the SP is 2002H. The value of PC is 18B5H, pointing to the address of the next program opcode fetched.

Operation

```
If cc true: PCL ← (SP), PCH ← (SP+1), SP ← SP + 2
```

Format

1	1	←	cc	→	0	0	0

Description

If condition cc is True (1), the byte at the memory location specified by the value of the SP is loaded to the low order eight bits of the PC. SP is then incremented and the byte at the memory location specified by the new value of SP are loaded to the high-order 8 bits of PC. SP is incremented again. The next opcode following this instruction is fetched from the memory location specified by the PC. This instruction is normally used to return to the calling sequence at the completion of a routine entered by a CALL instruction. When condition cc is False, PC is incremented as usual, and the program continues with the next instruction. Condition cc is programmed as one of eight values which correspond to condition bits in the Flag Register.

Hex Value (cc)	Condition	Relevant Flag
000	NZ–Nonzero	Z
001	Z–0	Z
010	NC–No Carry	C
011	C–Carry	C
100	PO–Parity Odd	P/V
101	PE–Parity Even	P/V
110	P–Sign Positive	S
111	M–Sign Negative	S

Timing

When cc is True (1):

M Cycles	Z80 T States	Z18x T States
3	11 (5, 3, 3)	10 (3, 1 int, 3, 3)

When cc is False (0):

M Cycles	Z80 T States	Z18x T States
1	5	5 (3, 2 int)

Example

The the S Flag is set, the value of the PC is 3535H. The value of SP is 2000H. The value of memory location 2000H is B5H. The value of memory location 2001H is 18H. At execution of RET M, the value of the SP is 2002H. The value of the PC is 18B5H, pointing to the address of the next program opcode fetched.

Operation

Return from Interrupt

Format

AA	1	1	1	0	1	1	0	1	ED
AA	0	1	0	0	1	1	0	1	4D

Description

This instruction is used at the end of an interrupt service routine to:

1. Restore the value of the PC (analogous to the RET instruction)

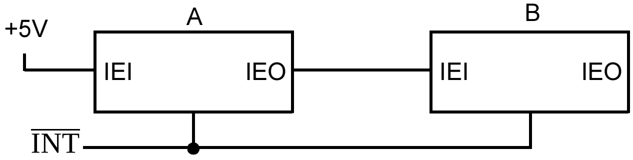
2. To signal a Z80 family I/O device that the interrupt routine is complete. The RETI instruction facilitates the nesting of interrupts, which allows higher priority devices to temporarily suspend service of lower priority service routines. This instruction does not enable interrupts, which were disabled when the interrupt routine was entered. Before doing the RETI instruction, the enable interrupt instruction (EI) must be executed to allow recognition of interrupts after completion of the RETI.

Timing

Z80		Z195 (OMCR7 (M1E) = 1)		Z18x (Otherwise)	
M Cycles	**T States**	**M Cycles**	**T States**	**M Cycles**	**T States**
4	14 (4,4,3,3)	4	13 (3,3,1 int,3,3)	6	22 (3,3,3 int,3,1 int,3,3,3)

Example

Two interrupting devices, A and B, are connected in a daisy chain configuration with A, which has a higher priority than B.



Device B generates an interrupt and is acknowledged. The interrupt enable out, `IEO`, of B goes Low, blocking any lower priority devices from interrupting while B is serviced. Device A generates an interrupt, suspending service of B. (The `IEO` of A goes Low, indicating that a higher priority device is serviced). The A routine is completed and a `RETI` is issued, clearing to (`0`) the A6 IUS Bit and setting the `IEO` of A back to High. A second `RETI` is issued on completion of the B, which sets the `IEO` of B back to High.

Operation

Return from nonmusical interrupt

Format

AA	1	1	1	0	1	1	0	1	ED
AA	0	1	0	0	0	1	0	1	45

Description

This instruction is used at the end of a non-maskable interrupt service routine to restore the value of the PC (analogous to the RET instruction). The state of IFF2 is copied back to IFF1, enabling maskable interrupts following the RETN (if they were enabled before the non-maskable interrupt).

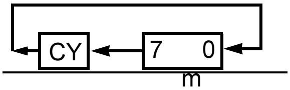
Timing

M Cycles	Z80 T States	Z18x T States
4	14 (4, 4, 3, 3)	12 (3, 3, 3, 3)

Example

The value of SP is 1000H. The value of PC is 1A45H. When a nonmusical interrupt (NMI) signal is received, the CPU restarts at memory address 0066H. That is, the current PC value of 1A45H is pushed to the external stack at addresses 0FFFH and 0FFEH, high order-byte first. Address 0066H is loaded to the PC. That address begins an interrupt service routine ending with a RETN instruction. At execution of RETN, the former PC value is popped off the external memory stack to PC, low-order first, resulting in a SP value again of 1000H. The program continues where it left off with an opcode fetch from address 1A45H.

Operation



Format

RL r	1	1	0	0	1	0	1	1	CB
	0	0	0	1	0	← r →			
RL (HL)	1	1	0	0	1	0	1	1	CB
	0	0	0	1	0	1	1	0	16
RL (IX+d)	1	1	0	1	1	1	0	1	DD
	1	1	0	0	1	0	1	1	CB
	← d →								
	0	0	0	1	0	1	1	0	16
RL (IY+d)	1	1	1	1	1	1	0	1	FD
	1	1	0	0	1	0	1	1	CB
	← d →								
	0	0	0	1	0	1	1	0	16

Description

The value of the `m` operand is rotated left one bit position. The content of Bit 7 is loaded to the Carry Flag. The previous content of the Carry Flag is loaded to Bit `0`. Other flags are set as described below. The `m` can be a Register `r`, a memory location selected by the value of Register pair `HL`, or a memory location selected by the sum of the value of an index Register IX or IY and a signed 8-bit displacement d. In the register form, r selects the register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
RL r	2	8 (4, 4)	7 (3, 3, 1 int)
RL (HL)	4	15 (4, 4, 4, 3)	13 (3, 3, 3, 1 int, 3)
RL (IX+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 30
RL (IY+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)

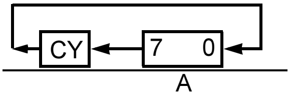
Flags

Flag	Value
S	Set if the result is Negative; reset otherwise
Z	Set if the result is 0; reset otherwise
H	Reset
P/V	Set if resulting parity is Even; reset otherwise
N	Reset
C	Data from Bit 7 of source register

Example

The Carry Flag is 0. Register D value is 8FH. At execution of RL D, the value of the Carry Flag is 1. The value of Register D is 1EH.

Operation



Format

AA	0	0	0	1	0	1	1	1	17

Description

The value of the Accumulator (A) is rotated left one bit position through the Carry Flag. The previous content of the Carry Flag is loaded to Bit 0.

Timing

M Cycles	Z80 T States	Z18x T States
1	4	3

Flags

Flag	Value
S	Not affected
Z	Not affected
H	Reset
P/V	Not affected
N	Reset
C	Data from Bit 7 of Accumulator

Example

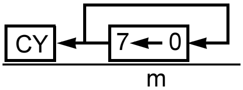
The value of the Accumulator and the Carry Flag is

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

At execution of RLA, the value of the Accumulator and the Carry Flag is

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

Operation



Format

RLC r	1	1	0	0	1	0	1	1	CB
	0	0	0	0	0	← r →			

RLC (HL)	1	1	0	0	1	0	1	1	CB
	0	0	0	0	0	1	1	0	06

RLC (IX+d)	1	1	0	1	1	1	0	1	DD
	1	1	0	0	1	0	1	1	CB
	← d →								
	0	0	0	0	0	1	1	0	06

RLC (IY+d)	1	1	1	1	1	1	0	1	FD
	1	1	0	0	1	0	1	1	CB
	← d →								
	0	0	0	0	0	1	1	0	06

Description

The value of the `m` operand are rotated left by one bit. The former Bit 7 is copied to both Bit `0` and the C Flag. Other flags are set as described below. The `m` can be a Register `r`, a memory location selected by the value of Register pair `HL`, or a memory location selected by the sum of the value of an index Register IX or IY and a signed 8-bit displacement d. In the register form, r selects the register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
RLC r	2	8 (4, 4)	7 (3, 3, 1 int)
RLC (HL)	4	15 (4, 4, 4, 3)	13 (3, 3, 3, 1 int, 3)
RLC (IX+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)
RLC (IY+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)

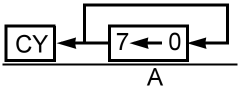
Flags

Flag	Value
S	Set if new Bit 7 is 1, reset otherwise
Z	Set if bits 7-0 are all 0; reset otherwise
H	Reset
P/V	Set if resulting parity is Even; reset otherwise
N	Reset
C	Former Bit 7 (same as new Bit 0)

Example

The value of Register pair IY is 1000H. The value of memory location 0FFEH is 88H. At execution of INC (IY - 2), the Carry Flag value is 1. The value of memory location 0FFEH is 11H.

Operation



Format

AA	0	0	0	0	0	1	1	1	07

Description

The value of the Accumulator (Register A) are rotated left one bit position. The sign bit (Bit 7) is copied to the Carry Flag and also to Bit 0. Bit 0 is the least significant bit.

Timing

M Cycles	Z80 T States	Z18x T States
1	4	3

Flags

Flag	Value
S	Not affected
Z	Not affected
H	Reset
P/V	Not affected
N	Reset
C	Data from Bit 7 of Accumulator

Example

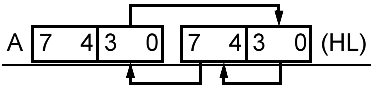
The value of the Accumulator is

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

At Execution of RLCA, the value of the Accumulator and Carry Flag is

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

Operation



Format

1	1	1	0	1	1	0	1	ED
0	1	1	0	1	1	1	1	6F

Description

The value of the low-order four bits (bits 3, 2, 1, and 0) of the memory location (HL) are loaded to the high-order four bits (7, 6, 5, and 4) of that same memory location. The previous value of those high-order four bits are loaded to the low-order four bits of the Accumulator (A). The previous value of the low-order four bits of the Accumulator are loaded to the low-order four bits of memory location (HL). The value of the high-order bits of the Accumulator are unaffected.

Timing

M Cycles	Z80 T States	Z180x T States
5	18 (4, 4, 3, 4 int, 3)	16 (3, 3, 3, 4 int, 3)

Flags

Flag	Value
S	Set if Accumulator is Negative after operation; reset otherwise
Z	Set if Accumulator is 0 after operation; reset otherwise
H	Reset
P/V	Set if parity of Accumulator is Even after operation; reset otherwise
N	Reset
C	Not affected

Example

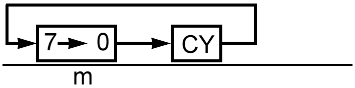
The value of Register pair `HL` is `5000H`. The value of the Accumulator and memory location `5000H` is:

	7	6	5	4	3	2	1	0	
AA	0	1	1	1	1	0	1	0	Accumulator
	0	0	1	1	0	0	0	1	(5000H)

At execution of `RLD,` the value of the Accumulator and memory location `5000H` is:

	7	6	5	4	3	2	1	0	
AA	0	1	1	1	0	0	1	1	Accumulator
	0	0	0	1	1	0	1	0	(5000H)

Operation



Format

RR r	1	1	0	0	1	0	1	1	CB
	0	0	0	1	1	← r →			

RR (HL)	1	1	0	0	1	0	1	1	CB
	0	0	0	1	1	1	1	0	1E

RR (IX+d)	1	1	0	1	1	1	0	1	DD
	1	1	0	0	1	0	1	1	CB
	← d →								
	0	0	0	1	1	1	1	0	1E

RR (IY+d)	1	1	1	1	1	1	0	1	FD
	1	1	0	0	1	0	1	1	CB
	← d →								
	0	0	0	1	1	1	1	0	1E

Description

The value of operand `m` is rotated right 1 bit position through the Carry Flag. The content of Bit `0` is loaded to the Carry Flag and the previous content of the Carry Flag is loaded to Bit 7. Other flags are set as described below. The `m` can be a

Register r, a memory location selected by the value of Register pair HL, or a memory location selected by the sum of the value of an index Register IX or IY and a signed 8-bit displacement d. In the register form, r selects the register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
RR r	2	8 (4, 4)	7 (3, 3, 1 int)
RR (HL)	4	15 (4, 4, 4, 3)	13 (3, 3, 3, 1 int, 3)
RR (IX+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)
RR (IY+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)

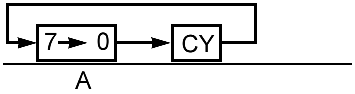
Flags

Flag	Value
S	Set if the result is Negative; reset otherwise
Z	Set if the result is 0; reset otherwise
H	Reset
P/V	Set if resulting parity is Even; reset otherwise
N	Reset
C	Data from Bit 0 of source register

Example

The the value of Register pair HL is 4343H. The memory location 4343H value is DDH. The Carry Flag value is 0. At execution of RR (HL), location 4343H value is 6EH. The Carry Flag value is 1.

Operation



Format

AA	0	0	0	1	1	1	1	1	1F

Description

The value of the Accumulator (A) are rotated right 1 bit position through the Carry Flag. The previous content of the Carry Flag is loaded to Bit 7. The previous content of Bit `0` is loaded to the Carry Flag.

Timing

M Cycles	Z80 T States	Z18x T States
1	4	3

Flags

Flag	Value
S	Not affected
Z	Not affected
H	Reset
P/V	Not affected
N	Reset
C	Data from Bit 0 of Accumulator

Example

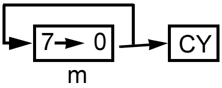
The the value of the Accumulator and the Carry Flag is:

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

At execution of RRA, the value of the Accumulator and the Carry Flag value is:

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

Operation



Format

RRC r

1	1	0	0	1	0	1	1	CB
0	0	0	0	1	← r →			

RRC (HL)

1	1	0	0	1	0	1	1	CB
0	0	0	0	1	1	1	0	0E

RRC (IX+d)

1	1	0	1	1	1	0	1	DD
1	1	0	0	1	0	1	1	CB
← d →								
0	0	0	0	1	1	1	0	0E

RRC (IY+d)

1	1	1	1	1	1	0	1	FD
1	1	0	0	1	0	1	1	CB
← d →								
0	0	0	0	1	1	1	0	0E

Description

The value of the m operand are rotated right one bit position. The content of Bit 0 is loaded to the Carry Flag and also to Bit 7, other flags are set as described below. The m can be a Register r, a memory location selected by the value of Register pair HL, or a memory location selected by the sum of the value of an index

Register IX or IY and a signed 8-bit displacement d. In the register form, r selects the register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
RRC r	2	8 (4, 4)	7 (3, 3, 1 int)
RRC (HL)	4	15 (4, 4, 4, 3)	13 (3, 3, 3, 1 int, 3)
RRC (IX+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)
RRC (IY+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)

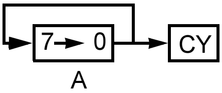
Flags

Flag	Value
S	Set if the result is Negative; reset otherwise
Z	Set if the result is 0; reset otherwise
H	Reset
P/V	Set if resulting parity is Even; reset otherwise
N	Reset
C	Data from Bit 0 of source register

Example

The value of Register A is 31H. At execution of `RRC A,` the A value is 98H and the Carry Flag value is 1.

Operation



Format

AA	0	0	0	0	1	1	1	1	0F

Description

The value of the Accumulator (A) is rotated right 1 bit position. Bit 0 is loaded to the Carry Flag and also to Bit 7.

Timing

M Cycles	Z80 T States	Z18x T States
1	4	3

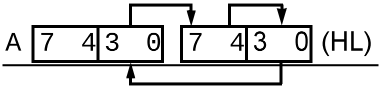
Flags

Flag	Value
S	Not affected
Z	Not affected
H	Reset
P/V	Not affected
N	Reset
C	Data from Bit 0 of Accumulator

Example

The Accumulator value is 11H. At execution of RRCA, the Accumulator value is 88H and the Carry Flag value is 1.

Operation



Format

1	1	1	0	1	1	0	1	ED
0	1	1	0	0	1	1	1	67

Description

The value of the low-order four bits (bits 3, 2, 1, and 0) of memory location (HL) are loaded to the low-order four bits of the Accumulator (A). The previous value of the low-order four bits of the Accumulator are loaded to the high-order four bits (7, 6, 5, and 4) of location (HL). The previous value of the high-order four bits of (HL) are loaded to the low-order four bits of (HL). The value of the high order bits of the Accumulator are unaffected.

Timing

M Cycles	Z80 T States	Z18x T States
4	18 (4, 4, 3, 4 int, 3)	16 (3, 3, 3, 4 int, 3)

Flags

Flag	Value
S	Set if Accumulator is Negative after operation; reset otherwise
Z	Set if Accumulator is 0 after operation; reset otherwise
H	Reset
P/V	Set if parity of Accumulator is Even after operation; reset otherwise
N	Reset
C	Not affected

Example

The value of Register pair HL is 5000H. The value of the Accumulator and memory location 5000H is:

	7	6	5	4	3	2	1	0	
AA	1	0	0	0	0	1	0	0	Accumulator
	0	0	1	0	0	0	0	0	(5000H)

At execution of RRD, the Accumulator and memory location 5000H is:

	7	6	5	4	3	2	1	0	
AA	1	0	0	0	0	0	0	0	Accumulator
	0	1	0	0	0	0	1	0	(5000H)

Operation

```
(SP–1) ← PCH, (SP–2) ←PCL, SP ← SP–2, PCH ← 0, PCL ← p
```

Format

AA	1	1	←t→			1	1	1

Description

The PC value is pushed to the external memory stack, and the page-0 memory location, indicated by operand p, is loaded to the PC. Program execution then begins with the opcode in the address now pointed to by PC. The push is accomplished using the following sequence:

1. Decrementing the value of the SP
2. Loading the high-order byte of PC to the memory address now pointed to by SP
3. decrementing SP again
4. Loading the low-order byte of PC to the address now pointed to by SP

The RST instruction allows for a jump to one of eight addresses in the table below.

Hex Value (p)	Hex Value (t)
00H	000
08H	001
10H	010
18H	011
20H	100
28H	101
30H	110
38H	111

Timing

M Cycles	Z80 T States	Z18x T States
3	11 (5, 3, 3)	11 (3, 2 int, 3, 3)

Example

The value of the PC is 15B3H. The SP value is 1000H. At the execution of RST 18H (Object code 1101111), the value of memory location 0FFFH is 15H. The value of memory location 0FFEH is B3H. The value of the SP is 0FFEH. The value of the PC is 0018H, the address of the next opcode to be fetched.

Operation

A ← A − s − CY

Format

SBC A, r	1	0	0	1	1	←	r	→	
SBC A, n	1	1	0	1	1	1	1	0	DE
	←			n				→	
SBC A, (HL)	1	0	0	1	1	1	1	0	9E
SBC A, (IX+d)	1	1	0	1	1	1	0	1	DD
	1	0	0	1	1	1	1	0	9E
	←			d				→	
SBC A, (IY+d)	1	1	1	1	1	1	0	1	FD
	1	0	0	1	1	1	1	0	9E
	←			d				→	

Description

The s operand and the Carry Flag are subtracted from the value of the Accumulator, the result is stored in A, and the flags are set as described below. The s can be a Register r, an immediate value n in the instruction, a memory location selected by the value of Register pair HL, or a memory location selected by the sum of the value of an index Register `IX` or `IY` and a signed 8-bit displacement `d`. In the register form, `r` selects a source register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
SBC A, r	1	4	4 (3 + 1 int)
SBC A, n	2	7 (4, 3)	6 (3, 3)
SBC A, (HL)	2	7 (4, 3)	6 (3, 3)
SBC A, (IX+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)
SBC A, (IY+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)

Flags

Flag	Value
S	Set if the result is Negative; reset otherwise
Z	Set if the result is 0; reset otherwise
H	Set if Borrow from Bit 4; reset otherwise
P/V	Set if an Overflow occurs; reset otherwise
N	Set
C	Set if Borrow; reset otherwise

Example

The the Accumulator value is `16H`. The Carry Flag is set. The Register pair `HL` value is `3433H`, and address `3433H` value is `05H`. At execution of `SBC A, (HL)`, the Accumulator value is `10H`.

Operation

HL ← Hl – ss – CY

Format

1	1	1	0	1	1	0	1	ED
0	1	s	s	0	0	1	0	

Description

The value of the Register pair ss (any of BC, DE, HL, or SP) and the Carry Flag are subtracted from the value of Register pair HL, the result is stored in HL, and the Carry Flag indicates if a Borrow is needed. Operand ss is specified as follows in the assembled object code.

Register Pair	Hex Value (ss)
BC	00
DE	01
HL	10
SP	11

Timing

M Cycles	Z80 T States	Z18x T States
2	15 (4, 4, 7 int)	10 (3, 3, 4 int)

Flags

Flag	Value
S	Set if the result is Negative; reset otherwise
Z	Set if the result is 0; reset otherwise
H	Set if a Borrow from Bit 12; reset otherwise

Flag	Value
S	Set if the result is Negative; reset otherwise
P/V	Set if an Overflow occurs; reset otherwise
N	Set
C	Set if Borrow; reset otherwise

Example

The the value of Register pair `HL` is `9999H`. The value of `DE` is `1111H`, and the Carry Flag is set. At execution of `SBC HL, DE,` Register pair `HL` value is `8887H`.

Operation

`CY ←1`

Format

0	0	1	1	0	1	1	1	37

Description

The Carry Flag is set.

Timing

M Cycles	Z80 T States	Z18x T States
1	4	3

Flags

Flag	Value
S	Not affected
Z	Not affected
H	Reset
P/V	Not affected
N	Reset
C	Set

NOTE: There is no specific Clear Carry Flag instruction. Use `OR A,` A to clear the Carry Flag.

Operation

m ← m OR (2 ^ b)

Format

SET b, r

1	1	0	0	1	0	1	1	CB
1	1	← b →			← r →			

SET b, (HL)

1	1	0	0	1	0	1	1	CB
1	1	← b →			1	1	0	

SET b, (IX+d)

1	1	0	1	1	1	0	1	DD
1	1	0	0	1	0	1	1	CB
← d →								
1	1	← b →			1	1	0	

SET b, (IY+d)

1	1	1	1	1	1	0	1	FD
1	1	0	0	1	0	1	1	CB
← d →								
1	1	← b →			1	1	0	

Description

Bit b of operand m is set to 1. No flags are affected. The b can be 0, for the least significant bit, through Bit 7 for the most significant bit. The m can be a Register r, a memory location selected by the value of Register pair HL, or a memory location selected by the sum of the value of an index Register IX or IY and a signed 8-bit displacement d. In the register form, r selects the register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

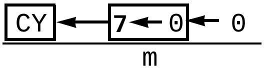
Timing

Instruction	M Cycles	Z80 T States	Z18x T States
SET b, r	2	8 (4, 4)	7 (3, 3, 1 int)
SET b, (HL)	4	15 (4, 4, 4, 3)	13 (3, 3, 3, 1 int, 3)
SET b, (IX+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)
SET b, (IY+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)

Example

The Register pair HL value is 4567H. The memory location 4567H value is 33H. At execution of SET 7, (HL), memory location 4567H value is B3H.

Operation



Format

SLA r	1	1	0	0	1	0	1	1	CB
	0	0	1	0	0	←	r	→	
SLA (HL)	1	1	0	0	1	0	1	1	CB
	0	0	1	0	0	1	1	0	26
SLA (IX+d)	1	1	0	1	1	1	0	1	DD
	1	1	0	0	1	0	1	1	CB
	←			d				→	
	0	0	1	0	0	1	1	0	26
SLA (IY+d)	1	1	1	1	1	1	0	1	FD
	1	1	0	0	1	0	1	1	CB
	←			d				→	
	0	0	1	0	0	1	1	0	26

Description

The operand m is arithmetically shifted left one bit position. The content of Bit 7 is copied to the Carry Flag, other flags are set as described below. The m can be a Register r, a memory location selected by the value of Register pair HL, or a memory location selected by the sum of the value of an index Register IX and IY and a signed 8-bit displacement d. In the register form, r selects the register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
SLA r	2	8 (4, 4)	7 (3, 3, 1 int)
SLA (HL)	4	15 (4, 4, 4, 3)	13 (3, 3, 3, 1 int, 3)
SLA (IX+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)
SLA (IY+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)

Flags

Flag	Value
S	Set if the result is Negative; reset otherwise
Z	Set if the result is 0; reset otherwise
H	Reset
P/V	Set if parity is Even; reset otherwise
N	Reset
C	Data from Bit 7

Example

The Register L value is B1H. At execution of SLA L, the value of Register L is 62H and the Carry Flag sets.

Operation

Enter Sleep or System Stop mode

Format

1	1	1	0	1	1	0	1	ED
0	1	1	1	0	1	1	0	76

Description

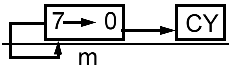
If the IOSTOP bit in the 8018x processor I/O Control Register is 0, this instruction places the device in Sleep mode. Sleep mode is deeper than Halt mode (see HALT instruction) because the internal CPU clock stops, the DMA channels do not operate, DRAM refresh (if any) stops, the Address bus is 3-stated, and control signals are driven high except when they are 3-stated while BUSACK is Low. Similar to Halt mode, Sleep mode can terminated by a Reset or by an interrupt request from an external or internal source, including the ASCIs, PRTs, and CSI/O. More details on Sleep mode and exit from Sleep mode are given in the Sleep mode section.

If the IOSTOP bit in the IOCR is 1 when this instruction is executed, the device enters System Stop mode. This mode differs from Sleep mode because the ASCIs, PRTs, and CSI/O are stopped, and they cannot generate an interrupt to terminate the mode.

Timing

M Cycles	18x T States
2	Indefinite: 8 (3, 3, 2 int) minimum

Operation



Format

SRA r

1	1	0	0	1	0	1	1	CB
0	0	1	0	1	←	r	→	

SRA (HL)

1	1	0	0	1	0	1	1	CB
0	0	1	0	1	1	1	0	2E

SRA (IX+d)

1	1	0	1	1	1	0	1	DD
1	1	0	0	1	0	1	1	CB
←				d			→	
0	0	1	0	1	1	1	0	2E

SRA (IY+d)

1	1	1	1	1	1	0	1	FD
1	1	0	0	1	0	1	1	CB
←				d			→	
0	0	1	0	1	1	1	0	2E

Description

The operand `m` is arithmetically shifted right one bit position. The content of Bit `0` is loaded to the Carry Flag and Bit 7 is unchanged. Other flags are set as described below. The `m` can be a Register `r`, a memory location selected by the value of Register pair `HL`, or a memory location selected by the sum of the value of an index Register IX or IY and a signed 8-bit displacement d. In the register form, r selects the register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
SRA r	2	8 (4, 4)	7 (3, 3, 1 int)
SRA (HL)	4	15 (4, 4, 4, 3)	13 (3, 3, 3, 1 int, 3)
SRA (IX+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)
SRA (IY+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)

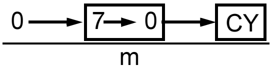
Flags

Flag	Value
S	Set if the result is Negative; reset otherwise
Z	Set if the result is 0; reset otherwise
H	Reset
P/V	Set if parity is Even; reset otherwise
N	Reset
C	Data from Bit 0 of source register

Example

The the value of the Index Register IX is 1000H. The value of memory location 1003H is B8H. At execution of SRA (IX+3), memory location 1003H value is DCH, and the Carry Flag is 0.

Operation



Format

SRL r	1	1	0	0	1	0	1	1	CB
	0	0	1	1	1	←	r	→	

SRL (HL)	1	1	0	0	1	0	1	1	CB
	0	0	1	1	1	1	1	0	3E

SRL (IX+d)	1	1	0	1	1	1	0	1	DD
	1	1	0	0	1	0	1	1	CB
	←			d				→	
	0	0	1	1	1	1	1	0	3E

SRL (IY+d)	1	1	1	1	1	1	0	1	FD
	1	1	0	0	1	0	1	1	CB
	←			d				→	
	0	0	1	1	1	1	1	0	3E

Description

The value of operand m are shifted right one bit position. The content of Bit 0 is loaded to the Carry Flag, Bit 7 is reset, and other flags are set as described below. The m can be a Register r, a memory location selected by the value of Register pair HL, or a memory location selected by the sum of the value of an index

Register IX or IY and a signed 8-bit displacement d. In the register form, r selects the register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
SRL r	2	8 (4, 4)	7 (3, 3, 1 int)
SRL (HL)	4	15 (4, 4, 4, 3)	13 (3, 3, 3, 1 int, 3)
SRL (IX+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)
SRL (IY+d)	6	23 (4, 4, 3, 5, 4, 3)	19 (3, 3, 3, 3, 3, 1 int, 3)

Flags

Flag	Value
S	Reset
Z	Set if the result is 0; reset otherwise
H	Reset
P/V	Set if parity is Even; reset otherwise
N	Reset
C	Data from Bit 0 of source register

Example

The Register B value is 8FH. At execution of SRL B, Register B value is 47H and the Carry Flag sets.

Operation

A ← A – s

Format

SUB A, r	1	0	0	1	0	← r →			
SUB A, n	1	1	0	1	0	1	1	0	D6
	← n →								
SUB A, (HL)	1	0	0	1	0	1	1	0	96
SUB A, (IX+d)	1	1	0	1	1	1	0	1	DD
	1	0	0	1	0	1	1	0	96
	← d →								
SUB A, (IY+d)	1	1	1	1	1	1	0	1	FD
	1	0	0	1	0	1	1	0	96
	← d →								

Description

The s operand is subtracted from the value of Accumulator A, the result is stored in A, and the flags are set as described below. The s can be any of a Register r, an immediate value n in the instruction, a memory location selected by the value of Register pair HL, or a memory location selected by the sum of the value of an index Register IX or IY and a signed 8-bit displacement d. In the register form, r selects a source register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycle	Z80 T States	Z18x T States
SUB A, r	1	4	4 (3 + 1 int)
SUB A, n	2	7 (4, 3)	6 (3, 3)
SUB A, (HL)	2	7 (4, 3)	6 (3, 3)
SUB A, (IX+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)
SUB A, (IY+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)

Flags

Flag	Value
S	Set if the result is Negative; reset otherwise
Z	Set if the result is 0; reset otherwise
H	Set if Borrow from Bit 4; reset otherwise
P/V	Set if an Overflow occurs; reset otherwise
N	Set
C	Set if Borrow; reset otherwise

Example

The Accumulator value is 29H. The Register D value is 11H. At execution of SUB D, the Accumulator value is 18H.

Operation

Test Accumulator (A AND s)

Format

TST A, r	1	1	1	0	1	1	0	1	ED
	0	0	←	r	→	1	0	0	
TST A, (HL)	1	1	1	0	1	1	0	1	ED
	0	0	1	1	0	1	0	0	34
TST A, n	1	1	1	0	1	1	0	1	ED
	0	1	1	0	0	1	0	0	64
	←				n			→	

Description

The value of the A register are logically ANDed with the s operand, which can be a register, the memory location pointed to by Register pair HL, or an immediate value. The result is discarded, but the flags are set to indicate the result as shown below. For the register case the r field of the instruction is encoded as follows:

Register	**r**
B	000
C	001
D	010
E	011
H	100

Register	r
L	101
A	111

Timing

Instruction	M Cycles	18x T States
TST A, r	2	7 (3, 3, 1 int)
TST A, (HL)	3	10 (3, 3, 3, 1 int)
TST A, n	3	9 (3, 3, 3)

Flags

Flag	Value
S	Set if there are 1s in Bit 7 of A and the operand, reset otherwise
Z	Set if all 8 bits of the result are 0, reset otherwise
H	Set
P/V	set if the resulting parity is Even; reset otherwise
N	Reset
C	Reset

Example

The A value is 83H. The B value is 7FH. At execution of instruction TST A, B, a (discarded) AND results in a value of 03H, and the flags are:

Flag	Value
S	0
Z	0
P/V	1

Operation

```
Set flags per ((C) AND n)
```

Format

1	1	1	0	1	1	0	1	ED
0	1	1	1	0	1	0	0	34
←				n			→	

Description

The value of the C Register are placed on `A7-A0` as an I/O port address, with zeroes on `A15-A8`. The value of the input port selected by this address (if any) are read and logically ANDed with the immediate data operand `n`. The result is discarded, but the flags are set to indicate the result as shown below.

Timing

M Cycles	T States (180 Register)	T States (18x Other Reg)
3	12 (3, 3, 3, 3 int)	13 (3, 3, 4, 3 int)

Flags

Flag	Value
S	Set if there are 1s in Bit 7 of both the input byte and the operand, reset otherwise
Z	Set if all 8 bits of the result are 0, reset otherwise
H	Set
P/V	Set if the resulting parity is Even; reset otherwise
N	Reset
C	Reset

Example

The value of C is 45H. The input port 0045H value is 3AH. At execution of instruction TSTIO 88H, a (discarded) AND results in a value of 08H, and the flags are:

Flag	Value
S	0
Z	0
P/V	0

Operation

A ← A XOR s

Format

XOR A, r	1	0	1	0	1	← r →			
XOR A, n	1	1	1	0	1	1	1	0	EE
	← n →								
XOR A, (HL)	1	0	1	0	1	1	1	0	AE
XOR A, (IX+d)	1	1	0	1	1	1	0	1	DD
	1	0	1	0	1	1	1	0	AE
	← d →								
XOR A, (IY+d)	1	1	1	1	1	1	0	1	FD
	1	0	1	0	1	1	1	0	AE
	← d →								

Description

The s operand is logically exclusive ORed with the value of the Accumulator A, the result is stored in A, and the flags are set as described below. The s can be any of a Register r, an immediate value n in the instruction, a memory location selected by the value of Register pair HL, or a memory location selected by the sum of the value of an index Register IX or IY and a signed 8-bit displacement d. In the register form, r selects a source register as follows:

Register	Hex Value (r)
B	000
C	001
D	010
E	011
H	100
L	101
A	111

Timing

Instruction	M Cycles	Z80 T States	Z18x T States
XOR A, r	1	4	4 (3 + 1 int)
XOR A, n	2	7 (4, 3)	6 (3, 3)
XOR A, (HL)	2	7 (4, 3)	6 (3, 3)
XOR A, (IX+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)
XOR A, (IY+d)	4	19 (4, 4, 3, 5 int, 3)	14 (3, 3, 3, 2 int, 3)

Flags

Flag	Value
S	Set if the result is Negative; reset otherwise
Z	Set if the result is 0; reset otherwise
H	Reset
P/V	Set if resulting parity is Even; reset otherwise
N	Reset
C	Reset

Example

The Accumulator value is 96H. At execution of XOR A, 5DH, the Accumulator value is CBH

Appendix A
Op Code Maps

INTRODUCTION

The following pages describe how instructions are encoded in the Z8018X processors.

The X (horizontal) axis of each table is the less significant four bits (nibble) or hex digit of an Op Code byte. The Y (vertical) axis represents the more significant four bits (nibble) or hex digit.

Table 26 describes how the 18X processors decode the first byte of an instruction. If the first byte is any of the hex values `CBH`, `DDH`, `EDH`, or `FDH`, the corresponding box in Table 26 through Table 30 directs the user to one of the more tables, which describe how the 18X processors decode the second byte of an instruction.

For instructions whose first two bytes are `DDCBH` or `FDCBH`, the corresponding box in Table 28 or Table 30 directs the user to Table 31 or Table 32, which describes how the 18X processors decode the fourth byte of an instruction.

Table 26. Op Code Map (1st Byte)

Lower Nibble (Hex)

Upper Nibble (Hex)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
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 D,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 H,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)	SUB 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 2 7)	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 2 8)	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 2 9)	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 3 0)	CP A,n	RST 38H
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Table 26. Op Code Map (1st Byte) (continued)

Lower Nibble (Hex)															
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

NOTES:

n = 8-Bit data
nn = 16-Bit address or data
d = Signed 8-bit displacement

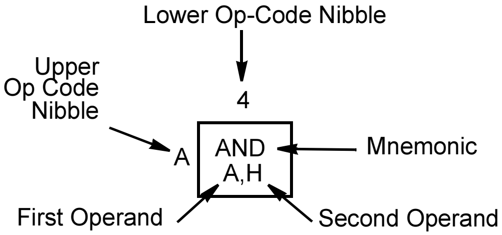


Table 27. Op Code Map (2nd Op Code after 0CBH)

Lower Nibble (Hex)

Upper Nibble (Hex)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
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 27. Op Code Map (2nd Op Code after 0CBH) (continued)

Lower Nibble (Hex)

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

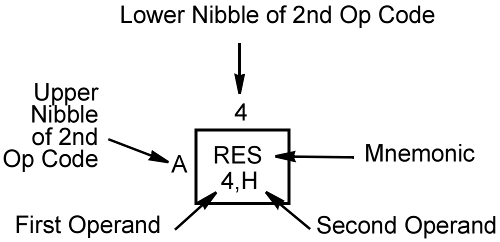


Table 28. Op Code Map (2nd Op Code After 0DDH)

Lower Nibble (Hex)

Upper Nibble (Hex)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
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 3 1)				
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

Table 28. Op Code Map (2nd Op Code After 0DDH) (continued)

Lower Nibble (Hex)															
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

NOTES:

n = 8-Bit data
nn = 16-Bit address or data
d = Signed 8-bit displacement

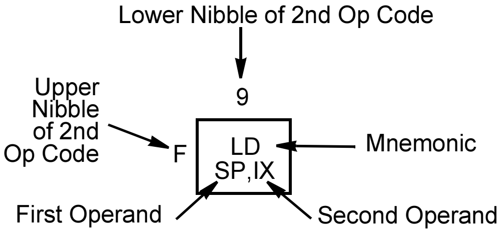


Table 29. Op Code Map (2nd Op Code After 0EDH)

Lower Nibble (Hex)

Upper Nibble (Hex)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
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	TSTIO n		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

Table 29. Op Code Map (2nd Op Code After 0EDH) (continued)

Lower Nibble (Hex)

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

NOTES:

n = 8-Bit data
nn = 16-Bit address or data
d = Signed 8-bit displacement

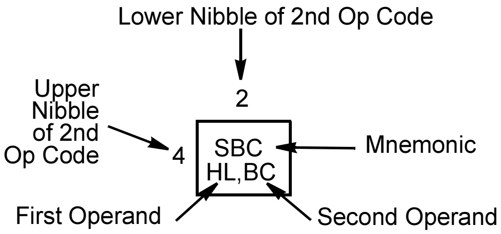


Table 30. Op Code Map (2nd Op Code After 0FDH)

Lower Nibble (Hex)

Upper Nibble (Hex)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
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 3 2)				
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

Table 30. Op Code Map (2nd Op Code After 0FDH) (continued)

Lower Nibble (Hex)

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

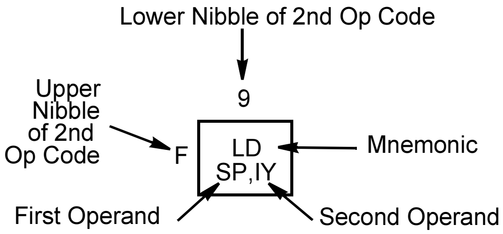


Table 31. Op Code Map (4th Byte, after 0DDH, 0CBH, and d)

Lower Nibble (Hex)

Upper Nibble (Hex)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
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

Table 31. 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

NOTE:

d = Signed 8-bit displacement

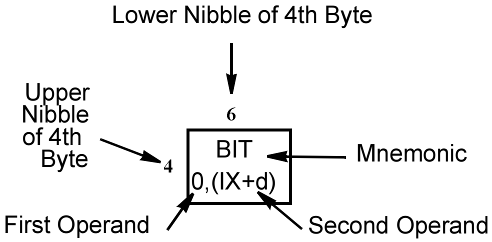


Table 32. Op Code Map (4th Byte, after 0FDH, 0CBH, and D)

Lower Nibble (Hex)

Upper Nibble (Hex)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
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

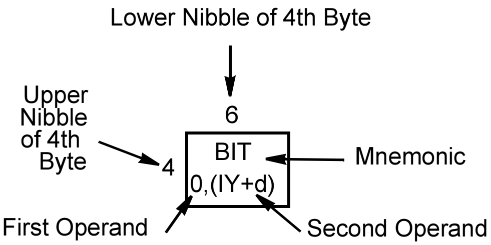
Table 32. Op Code Map (4th Byte, after 0FDH, 0CBH, and D) (continued)

Lower Nibble (Hex)

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

NOTE:

d = Signed 8-bit displacement



Appendix B
Instruction Execution

BUS AND CONTROL SIGNAL CONDITION

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
ADD HL, TT	MC_1	$T_1T_2T_3$	1st op code Address	1st op code	0	1	0	1	0	1	0
		TiTiTiTi	[1]	[2]	1	1	1	1	1	1	1
ADD IX, TT ADD IY, TT	MC_1	$T_1T_2T_3$	1st op code Address	1st op code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op code Address	2nd op code	0	1	0	1	0	1	1
		TiTiTiTi	[1]	[2]	1	1	1	1	1	1	1
ADC HL, SS SBC HL, SS	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
		TiTiTiTi	[1]	[2]	1	1	1	1	1	1	1
ADD A,r ADC A,r SUB A,r SBC A,r AND A,r OR A,r XOR A,r CP A,r	MC_1	$T_1T_2T_3$	1st op code Address	1st op code	0	1	0	1	0	1	0
	MC_2	Ti	[1]	[2]	1	1	1	1	1	1	1
ADD A,n ADC A,n SUB A,n SBC A,n AND A,n OR A,n XOR A,n CP A,n	MC_1	$T_1T_2T_3$	1st op code Address	1st op code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
ADD A, (HL) ADC A, (HL) SUB A, (HL) SBC A, (HL) AND A, (HL) OR A, (HL) XOR A,(HL) CP A, (HL)	MC_1	$T_1T_2T_3$	1st op code Address	1st op code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
ADD A, (IX+d) ADD A, (IY+d) ADC A, (IX+d) ADC A, (IY+d) SUB A, (IX+d) SUB A, (IY+d) SBC A, (IX+d)	MC_1	$T_1T_2T_3$	1st op code Address	1st op code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op code Address	2nd op code	0	1	0	1	0	1	1
SBC A, (IY+d) AND A, (IX+d) AND A, (IY+d) OR A, (IX+d) OR A, (IY+d) XOR A, (IX+d) XOR A, (IY+d) CP A, (IX+d) CP A, (IY+d)	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
		TiTiTi	[1]	[2]	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	IX+d or IY+d	DATA	0	1	0	1	1	1	1
BIT b, r	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
BIT b, (HL)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
BIT b, (IX+d) BIT b, (IY+d)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	3rd op-code Address	3rd op-code	0	1	0	1	0	1	1
	MC_5	$T_1T_2T_3$	IX+d or IY+d	DATA	0	1	0	1	1	1	1
CALL mn	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	SP-1	PCH	1	0	0	1	1	1	1
	MC_5	$T_1T_2T_3$	SP-2	PCL	1	0	0	1	1	1	1
CALL CC,mn (If condition is False)	MC_1	$T_1T_2T_3$	1st op code Address	1st op code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
CALL CC,mn (If condition is True)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	SP-1	PCH	1	0	0	1	1	1	1
	MC_5	$T_1T_2T_3$	SP-2	PCL	1	0	0	1	1	1	1
CCF	MC_1	$T_1T_2T_3$	1st op code Address	1st op code	0	1	0	1	0	1	0
CPI CPD	MC_1	$T_1T_2T_3$	1st op code Address	1st op code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op code Address	2nd op code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
		TiTiTi	[1]	[2]	1	1	1	1	1	1	1
CPIR CPDR (While BC ≠ 0 and A ≠ (HL))	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
		TiTiTiTiTi	[1]	[2]	1	1	1	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{\text{M1}}$ when $\overline{\text{IOC}}$ 1 and $\overline{\text{IOC}}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
CPIR CPDR (When BC = 0 or A = (HL))	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
		TiTiTi	[1]	[2]	1	1	1	1	1	1	1
CPL	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
DAA	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
		Ti	[1]	[2]	1	1	1	1	1	1	1
DI [3]	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
DJNZ d (If B ≠ 0)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
		Ti [4]	[1]	[2]	1	1	1	1	1	1	1
	MC_2	$T_1T_2T_3$	1st operand Address	d-2	0	1	0	1	1	1	1
		TiTi	[1]	[2]	1	1	1	1	1	1	1
DJNZ d (If B = 0)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
		Ti [4]	[1]	[2]	1	1	1	1	1	1	1
	MC_2	$T_1T_2T_3$	1st operand Address	d-2	0	1	0	1	1	1	1
EI [3]	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
EX DE, HL EXX	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
EX AF, AF’	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
		Ti	[1]	[2]	1	1	1	1	1	1	1
EX (SP), HL	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	SP	new L	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	SP+1	new H	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	SP+1	old H	1	0	0	1	1	1	1
	MC_5	$T_1T_2T_3$	SP	old L	1	0	0	1	1	1	1
EX (SP), IX EX (SP), IY	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	SP	new IXL or IYL	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	SP+1	new IXH or IYH	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_5	$T_1T_2T_3$	SP+1	old IXH or IYH	1	0	0	1	1	1	1
	MC_6	$T_1T_2T_3$	SP	old IXL or IYL	1	0	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
HALT	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	–	–	Next op-code Address	Next op-code	0	1	0	1	0	0	0
IM 0 IM 1 IM 2	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
INC r DEC r	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
		Ti	[1]	[2]	1	1	1	1	1	1	1
INC (HL) DEC (HL)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	1	0	0	1	1	1	1
INC (IX+d) INC (IY+d) DEC (IX+d) DEC (IY+d)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
		TiTi	[1]	[2]	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	IX+d or IY+d	DATA	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_5	$T_1T_2T_3$	IX+d or IY+d	DATA	1	0	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
INC ee DEC ee	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
		Ti	[1]	[2]	1	1	1	1	1	1	1
INC IX INC IY DEC IX DEC IY	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
IN A, (n)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	m to A_{7-0} A to A_{15-8}	DATA	0	1	1	0	1	1	1
IN r, (C)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	BC	DATA	0	1	1	0	1	1	1
IN0 r,(n)[6]	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	n to A_{7-0} 00H to A_{15-8}	DATA	0	1	1	0	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{\text{M1}}$ when $\overline{\text{IOC}}$ 1 and $\overline{\text{IOC}}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
INI IND	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	BC	DATA	0	1	1	0	1	1	1
	MC_4	$T_1T_2T_3$	HL	DATA	1	0	0	1	1	1	1
INIR INDR (While B $\neq$ 0)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	BC	DATA	0	1	1	0	1	1	1
	MC_4	$T_1T_2T_3$	HL	DATA	1	0	0	1	1	1	1
		TiTi	[1]	[2]	1	1	1	1	1	1	1
INIR INDR (If B = 0)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	BC	DATA	0	1	1	0	1	1	1
	MC_4	$T_1T_2T_3$	HL	DATA	1	0	0	1	1	1	1
JP mn	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{\text{M1}}$ when $\overline{\text{IOC}}$ 1 and $\overline{\text{IOC}}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
JP CC, mn (If CC is False)	MC$_1$	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
JP CC, mn (If CC is True)	MC$_1$	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC$_3$	$T_1T_2T_3$	2nd operand Adress	m	0	1	0	1	1	1	1
JP (HL)	MC$_1$	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
JP (IX) JP (IY)	MC$_1$	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
JR d	MC$_1$	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	$T_1T_2T_3$	1st operand Address	d-2	0	1	0	1	1	1	1
		TiTi	[1]	[2]	1	1	1	1	1	1	1
JR C,d JR NC,d JR Z,d JR NZ,d (If condition is False)	MC$_1$	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	$T_1T_2T_3$	1st operand Address	d-2	0	1	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{\text{M1}}$ when $\overline{\text{IOC}}$ 1 and $\overline{\text{IOC}}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
JR C,d JR NC,d JR Z,d JR NZ,d (If condition is True)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	d-2	0	1	0	1	1	1	1
		TiTi	[1]	[2]	1	1	1	1	1	1	1
LD r, r’	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
		Ti	[1]	[2]	1	1	1	1	1	1	1
LD r, n	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
LD r, (HL)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
LD r, (IX+d) LD r, (IY+d)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
		TiTiTi	[1]	[2]	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	IX+d or IY+d	DATA	0	1	0	1	1	1	1
LD (HL), r	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
		T_i	[1]	[2]	1	1	1	1	1	1	1
	MC_2	$T_1T_2T_3$	HL	(r)	1	0	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
LD (IX + d), r LD (IY + d), r	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
		$T_iT_iT_i$	[1]	[2]	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	IX+d or IY+d	(r)	1	0	0	1	1	1	1
LD (HL), n	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	HL	n	1	0	0	1	1	1	1
LD (IX+d), n LD (IY+d), n	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	2nd operand Address	n	0	1	0	1	1	1	1
	MC_5	$T_1T_2T_3$	IX+d or IY+d	n	1	0	0	1	1	1	1
LD A, (BC) LD A, (DE)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	BC or DE	DATA	0	1	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{\text{M1}}$ when $\overline{\text{IOC}}$ 1 and $\overline{\text{IOC}}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
LD A, (mn)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	mn	DATA	0	1	0	1	1	1	1
LD (BC), A LD (DE), A	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_2	$T_1T_2T_3$	BC or DE	(A)	1	0	0	1	1	1	1
LD (mn), A	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	mn	(A)	1	0	0	1	1	1	1
LD A, I [3] LD A, R [3] LD I, A	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
LD R, A	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
LD ee, mn	MC$_1$	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC$_3$	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
LD IX, mn LD IY, mn	MC$_1$	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC$_3$	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC$_4$	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
LD HL, (mn)	MC$_1$	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC$_3$	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
	MC$_4$	$T_1T_2T_3$	mn	DATA	0	1	0	1	1	1	1
	MC$_5$	$T_1T_2T_3$	mn+1	DATA	0	1	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{\text{M1}}$ when $\overline{\text{IOC}}$ 1 and $\overline{\text{IOC}}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
LD ee, (mn)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
	MC_5	$T_1T_2T_3$	mn	DATA	0	1	0	1	1	1	1
	MC_6	$T_1T_2T_3$	mn+1	DATA	0	1	0	1	1	1	1
LD IX, (mn) LD IY, (mn)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
	MC_5	$T_1T_2T_3$	mn	DATA	0	1	0	1	1	1	1
	MC_6	$T_1T_2T_3$	mn+1	DATA	0	1	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
LD (mn), HL	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	mn	L	1	0	0	1	1	1	1
	MC_5	$T_1T_2T_3$	mn+1	H	1	0	0	1	1	1	1
LD (mn), ee	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_5	$T_1T_2T_3$	mn	eeL	1	0	0	1	1	1	1
	MC_6	$T_1T_2T_3$	mn+1	eeH	1	0	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{\text{M1}}$ when $\overline{\text{IOC}}$ 1 and $\overline{\text{IOC}}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
LD (mn), IX LD (mn), IY	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_5	$T_1T_2T_3$	mn	IXL or IYL	1	0	0	1	1	1	1
	MC_6	$T_1T_2T_3$	mn+1	IXH or IYH	1	0	0	1	1	1	1
LD SP, HL	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
		Ti	[1]	[2]	1	1	1	1	1	1	1
LD SP, IX LD SP, IY	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
LDI LDD	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	DE	DATA	1	0	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{\text{M1}}$ when $\overline{\text{IOC}}$ 1 and $\overline{\text{IOC}}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
LDIR LDDR (While BC ≠ 0)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	DE	DATA	1	0	0	1	1	1	1
		TiTi	[1]	[2]	1	1	1	1	1	1	1
LDIR LDDR (If BC = 0)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	DE	DATA	1	0	0	1	1	1	1
MLT SS [6]	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
		TiTiTiTi TiTiTiTi TiTiTi	[1]	[2]	1	1	1	1	1	1	1
NEG	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
NOP	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{\text{M1}}$ when $\overline{\text{IOC}}$ 1 and $\overline{\text{IOC}}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
OUT (n), A	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_3	$T_1T_2T_3$	m to A_{7-0} A to A_{15-8}	(A)	1	0	1	0	1	1	1
OUT (C), r	MC_1	$T_1T_2T_3$	1st op-code	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_3	$T_1T_2T_3$	BC	(r)	1	0	1	0	1	1	1
OUT0 (n), r [6]	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	n to A_{7-0} 00H to A_{15-8}	(r)	1	0	1	0	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
OTIM [6] OTDM [6]	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	C to A_{7-0} 00H To A_{15-8}	DATA	1	0	1	0	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
OTIMR [6] OTDMR [6] (While B ≠ 0)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	C to A_{7-0} 00H to A_{15-8}	DATA	1	0	1	0	1	1	1
		TiTiTi	[1]	[2]	1	1	1	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
OTIMR [6] OTDMR [6] (If B = 0)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	C To A_{7-0} 00H to A_{15-8}	DATA	1	0	1	0	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
OUTI OUTD	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	BC	DATA	1	0	1	0	1	1	1
OTIR OTDR (While B ≠ 0)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	BC	DATA	1	0	1	0	1	1	1
		TiTi	[1]	[2]	1	1	1	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
OTIR OTDR (If B = 0)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	BC	DATA	1	0	1	0	1	1	1
POP pp	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	SP	DATA	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	SP+1	DATA	0	1	0	1	1	1	1
POP IX POP IY	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	SP	DATA	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	SP+1	DATA	0	1	0	1	1	1	1
PUSH pp	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
		TiTi	[1]	[2]	1	1	1	1	1	1	1
	MC_2	$T_1T_2T_3$	SP-1	ppH	1	0	0	1	1	1	1
	MC_3	$T_1T_2T_3$	SP-2	ppL	1	0	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
PUSH IX PUSH IY	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
		TiTi	[1]	[2]	1	1	1	1	1	1	1
	MC3	$T_1T_2T_3$	SP-1	IXH or IYH	1	0	0	1	1	1	1
	MC_4	$T_1T_2T_3$	SP-2	IXL or IYL	1	0	0	1	1	1	1
RET	MC_1	$T_1T_2T_3$	1st op-code	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	SP	DATA	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	SP+1	DATA	0	1	0	1	1	1	1
RET CC (If condition is False)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
		TiTi	[1]	[2]	1	1	1	1	1	1	1
RET CC (If condition is True)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_2	$T_1T_2T_3$	SP	DATA	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	SP+1	DATA	0	1	0	1	0	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
RETI (M1E=1) RETN	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	SP	DATA	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	SP+1	DATA	0	1	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{\text{M1}}$ when $\overline{\text{IOC}}$ 1 and $\overline{\text{IOC}}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
RETI (M1E=0)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0 [5] 1	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0 [5] 1	1	1
		TiTiTi	[1]	[2]	1	1	1	1	1 [5] 1	1	1
	MC_3	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0 [5] 0	1	1
		Ti	[1]	[2]	1	1	1	1	1 [5] 1	1	1
	MC_4	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0 [5] 0	1	1
	MC_5	$T_1T_2T_3$	SP	data	0	1	0	1	1 [5] 1	1	1
	MC_6	$T_1T_2T_3$	SP+1	data	0	1	0	1	1 [5] 1	1	1
RLCA RLA RRCA RRA	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
RLC r RL r RRC r RR r SLA r SRA r SRL r	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
RLC (HL) RL (HL) RRC (HL) RR (HL) SLA (HL) SRA (HL) SRL (HL)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	HL	DATA	1	0	0	1	1	1	1
RLC (IX+d) RLC (IY+d) RL (IX+d) RL (IY+d) RRC (IX+d) RRC (IY+d) RR (IX+d) RR (IY+d) SLA (IX+d) SLA (IY+d) SRA (IX+d) SRA (IY+d) SRL (IX+d) SRL (IY+d)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	3rd op-code Address	3rd op-code	0	1	0	1	0	1	1
	MC_5	$T_1T_2T_3$	IX+d or IY+d	DATA	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_6	$T_1T_2T_3$	IX+d or IY+d	DATA	1	0	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
RLD RRD	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
		TiTiTiTi	[1]	[2]	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	HL	DATA	1	0	0	1	1	1	1
RST p	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
		TiTi	[1]	[2]	1	1	1	1	1	1	1
	MC_2	$T_1T_2T_3$	SP-1	PCH	1	0	0	1	1	1	1
	MC_3	$T_1T_2T_3$	SP-2	PCL	1	0	0	1	1	1	1
SCF	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
SET b,r RES b,r	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
SET b, (HL) RES b, (HL)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	HL	DATA	1	0	0	1	1	1	1

SET b, (IX+d)
SET b, (IY+d)
RES b, (IX+d)
RES b, (IY+d)
	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0

	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	3rd op-code Address	3rd op-code	0	1	0	1	0	1	1
	MC_5	$T_1T_2T_3$	IX+d or IY+d	DATA	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_6	$T_1T_2T_3$	IX+d or IY+d	DATA	1	0	0	1	1	1	1
SLP[6]	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	–	–	7FFFFH	[2]	1	1	1	1	1	0	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 26. Instruction

Instruction	Machine Cycle	States	Address	Data	RD	WR	MREQ	IORQ	M1	Halt	St
TSTIO n [6]	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC$_3$	T$_1$T$_2$T$_3$	1st operand	n	0	1	0	1	1	1	1
	MC$_4$	T$_1$T$_2$T$_3$	C to A$_{7-0}$ 00H to A$_{15-8}$	DATA	0	1	1	0	1	1	1
TST r [6]	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
TST n [6]	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC$_3$	T$_1$T$_2$T$_3$	1st operand Address	n	0	1	0	1	1	1	1
TST (HL) [6]	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC$_3$	T$_1$T$_2$T$_3$	HL	DATA	0	1	0	1	1	1	1

[1] (ADDRESS) Invalid
[2](DATA) High Impedance.
[3]Interrupt request is not sampled.
[4]DMA, REFRESH, or BUS RELEASE cannot be executed after this state. (Request is ignored).
[5]The upper and lower data include the state of $\overline{M1}$ when $\overline{IOC}$ 1 and $\overline{IOC}$ 0 respectively.
[6]New added instructions to Z8018X.

Table 27. Interrupt Cycles

Operation	Machine Cycle	States	ADDRESS	DATA	RD	WR	MREQ	IORQ	M1	HALT	ST
$\overline{\text{NMI}}$	MC_1	$T_1T_2T_3$	Next op-code Address (PC)		0	1	0	1	0	1	0
		TiTi	[1]	[2]	1	1	1	1	1	1	1
	MC_2	$T_1T_2T_3$	SP-1	PCH	1	0	0	1	1	1	1
	MC_3	$T_1T_2T_3$	SP-2	PCL	1	0	0	1	1	1	1
$\overline{\text{INT0 MODE 0}}$ (RST INSERTED)	MC_1	$T_1T_2T_W$ T_WT_3	Next op-code Address (PC)	1st op-code	1	1	1	0	0	1	0
		TiTi	[1]	[2]	1	1	1	1	1	1	1
	MC_2	$T_1T_2T_3$	SP-1	PCH	1	0	0	1	1	1	1
	MC_3	$T_1T_2T_3$	SP-2	PCL	1	0	0	1	1	1	1
$\overline{\text{INT0 MODE 0}}$ (CALL INSERTED)	MC_1	$T_1T_2T_W$ T_WT_3	Next op-code Address (PC)	1st op-code	1	1	1	0	0	1	0
	MC_2	$T_1T_2T_3$	PC	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	PC+1	m	0	1	0	1	1	1	1
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	SP-1	PC+2(H)	1	0	0	1	1	1	1
	MC_5	$T_1T_2T_3$	SP-2	PC+2(L)	1	0	0	1	1	1	1
$\overline{\text{INT}}_0$ MODE 1	MC_1	$T_1T_2T_W$ T_WT_3	Next op-code Address (PC)		1	1	1	0	0	1	0
	MC_2	$T_1T_2T_3$	SP-1	PCH	1	0	0	1	1	1	1
	MC_3	$T_1T_2T_3$	SP-2	PCL	1	0	0	1	1	1	1

[1] (ADDRESS) = invalid
[2] (DATA) = high impedance.

Table 27. Interrupt Cycles (continued)

Operation	Machine Cycle	States	ADDRESS	DATA	RD	WR	MREQ	IORQ	M1	HALT	ST
$\overline{\text{INT}}_0$ MODE 2	MC_1	$T_1T_2T_W$ T_WT_3	Next op-code Address (PC)	Vector	1	1	1	0	0	1	0
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_2	$T_1T_2T_3$	SP-1	PCH	1	0	0	1	1	1	1
	MC_3	$T_1T_2T_3$	SP-2	PCL	1	0	0	1	1	1	1
	MC_4	$T_1T_2T_3$	I, Vector	DATA	0	1	0	1	1	1	1
	MC_5	$T_1T_2T_3$	I, Vector+1	DATA	0	1	0	1	1	1	1
$\overline{\text{INT}}_1$ $\overline{\text{INT}}_2$ Internal Interrupts	MC_1	$T_1T_2T_W$ T_WT_3	Next op-code Address (PC)		1	1	1	1	1	1	0
		Ti	[1]	[2]	1	1	1	1	1	1	1
	MC_2	$T_1T_2T_3$	SP-1	PCH							
	MC_3	$T_1T_2T_3$	SP-2	PCL	1	0	0	1	1	1	1
	MC_4	$T_1T_2T_3$	I, Vector	DATA	0	1	0	1	1	1	1
	MC_5	$T_1T_2T_3$	I, Vector+1	DATA	0	1	0	1	1	1	1

[1] (ADDRESS) = invalid

[2] (DATA) = high impedance.