NEC

NEC Electronics Inc.

μPD780
HIGH-PERFORMANCE CP/M®-COMPATIBLE NMOS 8-BIT MICROPROCESSOR

Description

The μPD780 is a microprocessor that utilizes a highly consistent architectural organization, a comprehensive instruction set that is a superset of the industry-standard 8080A instruction set, and third-generation technology, to provide a flexible, high-performance, efficient CPU easily adaptable to a very broad range of industrial and commercial applications.

All software developed on 8080A-based systems may be run on 780-based systems as a subset of the full 780 instruction set. In addition, the NEC μPD780 is fully pin-compatible and software-compatible with the Z80® microprocessor and is therefore perfectly suited for CP/M® designs. The NEC μPD780 provides system designers with powerful, wide-range logic capability that requires minimal additional circuitry to complete a microcomputer system.

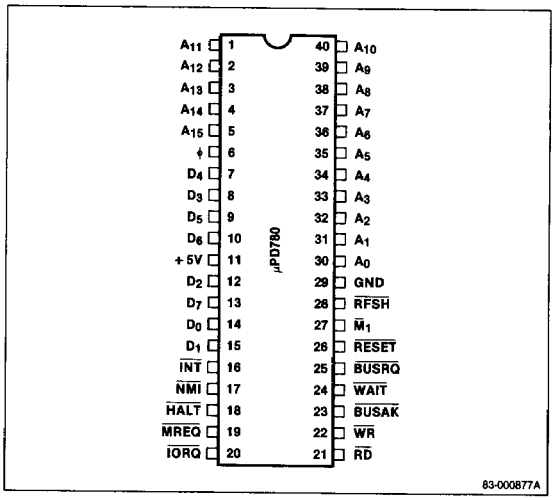
The output signals of the μPD780 are fully decoded and signal timing is fully compatible with industry-standard memory and peripheral devices. Two faster versions of the basic μPD780 (2.5 MHz master clock rate) are offered by the μPD780-1 (4 MHz master clock rate) and the μPD780-2 (6 MHz master clock rate). Other than clock rates, all three versions are identical.

Features

- ☐ Powerful, wide-range logic capability requiring minimal support circuitry
- ☐ Fully Z80® -compatible
- ☐ Industry-standard 8080A software compatibility
- ☐ CP/M® -compatible
- ☐ Comprehensive, powerful instruction set featuring 158 instruction types
- ☐ Vectored, multilevel interrupt structure
- ☐ Highly consistent architectural structure featuring dual register set
- ☐ Foreground/background programming
- ☐ Automatic refreshing of external dynamic memory
- ☐ Signal timing compatible with industry-standard memory and peripheral devices
- ☐ TTL-compatible signals
- ☐ Single-phase +5 V clock and +5 V DC power supply

® Z80 is a registered trademark of Zilog, Inc.
® CP/M is a registered trademark of Digital Research Corporation.

Pin Configuration



Ordering Information

Part Number	Package Type	Max Frequency of Operation
μPD780C	40-pin plastic DIP	2.5 MHz
μPD780C-1	40-pin plastic DIP	4 MHz
μPD780C-2	40-pin plastic DIP	6 MHz

4

Pin Identification

No.	Symbol	Function
1-5, 30-40	A_0-A_{15}	Three-state address bus (output)
6	ϕ	Clock input
7-10, 12-15	D_0-D_7	Three-state, I/O data bus
11	+5 V	Power supply
16	$\overline{INT}$	Interrupt request input
17	$\overline{NMI}$	Non-maskable interrupt input
18	$\overline{HALT}$	Halt state input
19	$\overline{MREQ}$	Memory request output
20	$\overline{IORQ}$	I/O request output
21	$\overline{RD}$	Read output
22	$\overline{WR}$	Write output
23	$\overline{BUSAK}$	Bus acknowledge output
24	$\overline{WAIT}$	Wait state input
25	$\overline{BUSRQ}$	Bus request input
26	$\overline{RESET}$	Reset input
27	$\overline{M_1}$	Machine cycle 1
28	$\overline{RFSH}$	Refresh output
29	GND	Ground

Pin Functions

A_0-A_{15} (Address Bus)

16-bit, three-state output address bus. During refresh operations, lines A_0-A_6 output the external memory address.

D_0-D_7 (Data Bus)

8-bit, three-state I/O data bus.

$\overline{NMI}$ (Non-Maskable Interrupt)

This active low input line is used for non-maskable interrupts. A non-maskable interrupt is always acknowledged at the end of the current instruction, regardless of whether the interrupt enable flip flop has been turned on, except when the $\overline{BUSRQ}$ signal is asserted. Because of the higher priority of the $\overline{BUSRQ}$ signal, it is acknowledged before the $\overline{NMI}$ signal. When $\overline{NMI}$ is acknowledged, program execution automatically restarts from location 0066H.

$\overline{INT}$ (Interrupt Request)

This active low input line is used for interrupt requests by external I/O devices. Interrupts are serviced upon completion of the current instruction if the interrupt enable flip flop has been turned on by the software. There are three interrupt response modes: the mode 0 response is equivalent to an 8080 interrupt response; mode 1 uses location 0038H as a restart address; and mode 2 is a simple vectoring to an interrupt service routine that can be located anywhere in memory.

$\overline{BUSRQ}$ (Bus Request)

This active low input signal is used to place the data bus, address bus, and all three-state bus control signals ($\overline{WR}$, $\overline{RD}$, $\overline{IORQ}$, and $\overline{MREQ}$) in a high-impedance state to allow a requesting device to assume bus control. The $\overline{BUSRQ}$ signal has a higher priority than the NMI signal and is always honored at the end of the current machine cycle.

Excessive DMA operations resulting in long periods in which $\overline{BUSRQ}$ is asserted can impair the CPU's ability to adequately refresh the dynamic RAMs. Also, $\overline{BUSRQ}$ does not have an internal pull-up resistor. For input signals to this pin in a wire-OR'ed configuration, an external pull-up resistor should be used.

$\overline{BUSAK}$ (Bus Acknowledge)

This active low output line is used to inform the device requesting bus control that the data bus, address bus, and all three-state bus controls ($\overline{WR}$, $\overline{RD}$, $\overline{IORQ}$, and $\overline{MREQ}$) are in a high-impedance state and the requesting device can now assume control.

$\overline{WR}$ (Write)

This three-state active low output is used to strobe data from the data bus to external memory or I/O devices. $\overline{WR}$ is asserted to indicate the data bus holds valid data. This line is three-stated during halt or reset conditions.

$\overline{IORQ}$ (I/O Request)

This three-state active low output is used to indicate the lower half of the address bus holds a valid address for an I/O read or write. During interrupt acknowledge cycles, $\overline{IORQ}$ and $\overline{M_1}$ are asserted together to indicate that a vector address can be sent to the data bus.

$\overline{RD}$ (Read)

This three-state active low output is used to strobe data from external memory or I/O devices onto the data bus. $\overline{RD}$ is asserted to indicate the CPU is requesting data from external memory or I/O devices. This line is three-stated during halt or reset conditions.

$\overline{MREQ}$ (Memory Request)

This three-state active low output is used to indicate that the address specified for the memory read or write is valid.

$\overline{M_1}$ (Machine Cycle 1)

This active low output is used to indicate that the current machine cycle is the opcode fetch phase of an instruction execution.

$\overline{HALT}$ (Halt State)

This active low input is used with the $\overline{HALT}$ instruction to initiate a halt state. When $\overline{HALT}$ is asserted, program execution stops and does not resume until an interrupt is generated. During the halt state, NOPs are executed in order to continue memory refresh operations.

$\overline{WAIT}$ (Wait State)

This active low input is used to indicate that the external memory or I/O devices addressed by the CPU are not ready to transfer data. When $\overline{WAIT}$ is asserted, the CPU is placed in a wait condition.

$\overline{RESET}$

This active low input signal is used to initialize the CPU. When $\overline{RESET}$ is asserted, the interrupt enable flip flop is reset, the program counter and the I and R registers are cleared, and interrupt response mode 0 is enabled. In a reset condition, the address and data busses are three-stated and all output control signals are inactive, after which program execution begins from address 0000.

The pulse width of $\overline{RESET}$ must be a minimum of 3 clock cycles in length to reinitialize the CPU and stabilize operation.

$\overline{RFSH}$ (Refresh)

This active low output is used in conjunction with the $\overline{MREQ}$ signal to initiate a refresh read of all external dynamic memory. $\overline{RFSH}$ and $\overline{MREQ}$ are both asserted when the least significant 7 bits of the address on the address bus hold a valid external dynamic memory address.

ϕ (Clock)

This line is an input for external clock sources.

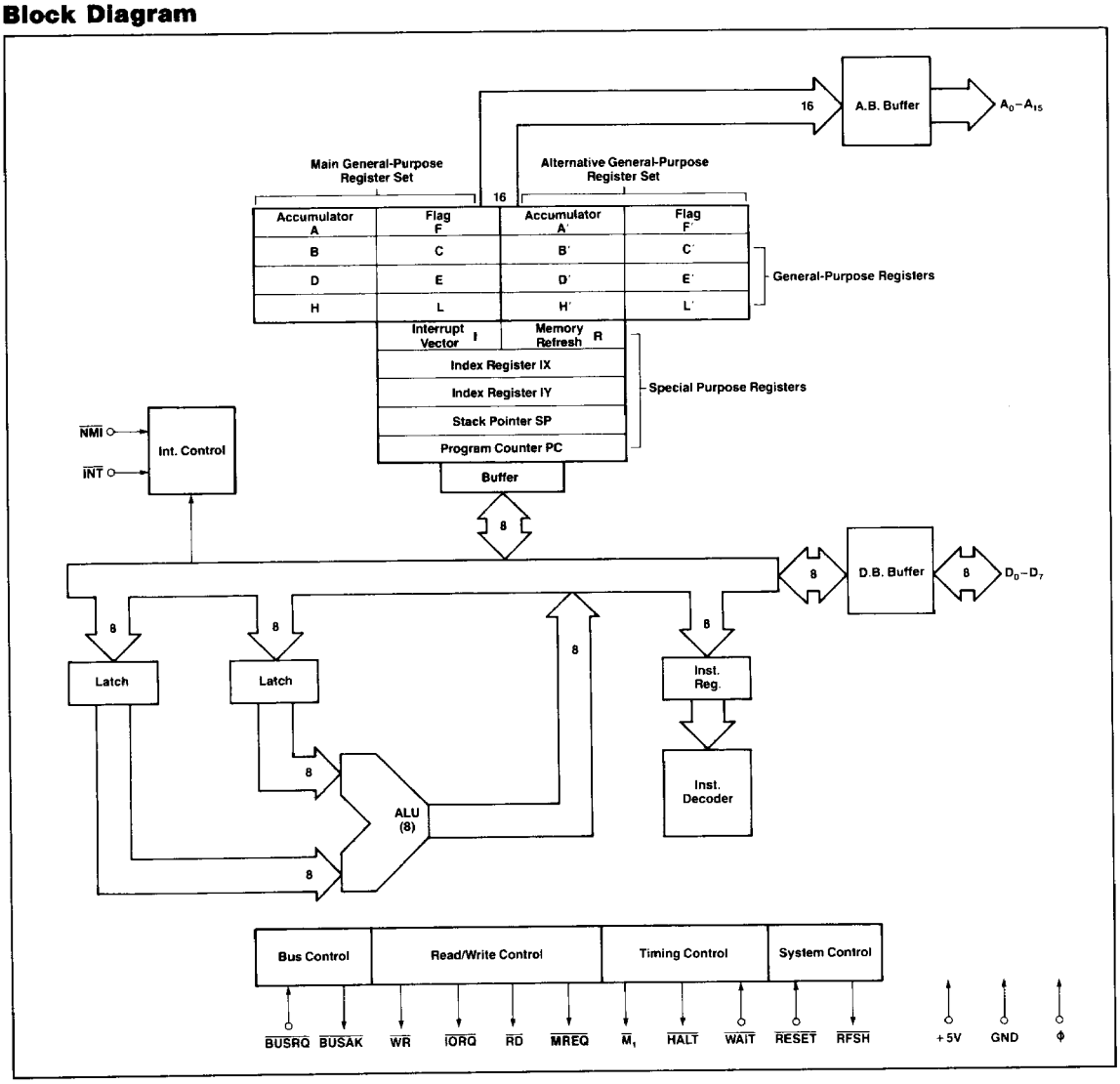
+5 V

Single +5 V power supply.

GND

Ground.

Block Diagram



Architecture

The architecture includes a dual set of six 8-bit general-purpose registers and two 8-bit accumulators and flag registers. A flexible vectored interrupt structure is supported by an 8-bit interrupt vector register that provides the most-significant 8 bits of a pointer to a table of vector addresses, while the requesting device generates the least-significant 8 bits of the pointer. Two 16-bit index registers enable the manipulation of tabular data as well as facilitating code relocation.

Multilevel interrupts as well as virtually unlimited subroutine nesting are supported by a 16-bit stack pointer and complimentary 16-bit program counter, enhancing the speed and efficiency of a wide variety of data-handling operations. Processing efficiency is additionally supported by a special memory refresh register that enables automatic refreshing of all external dynamic memory with minimal processor overhead.

The dual set of general-purpose registers may be used as individual 8-bit registers or paired as 16-bit registers. The dual register set (including a dual accumulator and flag register) not only allows more powerful addressing and data transfer operations, but also permits programming in foreground/background mode for vastly improved throughput.

Standard Test Conditions

The standard test conditions reference all voltages to ground (0 V) and follow the convention that positive current flows into the referenced pin. The listing of AC parameters is based on a load capacitance of 50 pF unless explicitly stated otherwise. For every 50 pF increase in load capacitance there is a 10 ns delay, up to a maximum increase of 200 pF for the data bus and 100 pF for the address bus and the bus control lines.

The operating temperature range is: 0°C to +70°C; $+4.75\ V \leq V_{CC} \leq +5.25\ V$.

Absolute Maximum Ratings

T_A = 25°C

Operating temperature	0°C to +70°C
Storage temperature	−65°C to +150°C
Voltage on any pin	−0.3 to +7 V (1)
Power dissipation	1.5 W

Note:

(1) With respect to ground.

Comment: Exposing the device to stresses above those listed in Absolute Maximum Ratings could cause permanent damage. The device is not meant to be operated under conditions outside the limits described in the operational sections of this specification. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Capacitance

T_A = 25°C

Parameter	Symbol	Limits Min	Limits Max	Unit	Test Conditions
Clock capacitance	C_ϕ		35	pF	f_C=1 MHz
Input capacitance	C_{IN}		5	pF	Unmeasured pins
Output capacitance	C_{OUT}		10	pF	returned to ground.

DC Characteristics

T_A = 0°C to +70°C; V_{CC} = +5 V ± 5% unless otherwise specified

Parameter	Symbol	Limits Min	Limits Typ	Limits Max	Unit	Test Conditions
Clock input low voltage	V_{ILC}	−0.3		0.45	V	
Clock input high voltage	V_{IHC}	V_{CC} −0.6		V_{CC} +0.3	V	
Input low voltage	V_{IL}	−0.3		0.8	V	
Input high voltage	V_{IH}	2.0		V_{CC}	V	
Output low voltage	V_{OL}			0.4	V	I_{OL}=1.8 mA
Output high voltage	V_{OH}	2.4			V	I_{OH}=−250 μA
Power supply Current μPD780	I_{CC}			150	mA	t_C=400 ns
μPD780-1	I_{CC}		90	200	mA	t_C=250 ns
Input leakage current	I_{LI}			10	μA	V_{IN}=0 to V_{CC}
Three-state output leakage current in float	I_{LOH}			10	μA	V_{OUT}=2.4 to V_{CC}
Three-state output leakage current in float	I_{LOL}			−10	μA	V_{OUT}=0.4 V
Data bus leakage current in input mode	I_{LD}			±10	μA	$0 \leq V_{IN} \leq V_{CC}$

4

AC Characteristics

T_A = 0°C to +70°C; V_{CC} = +5 V ± 5%; unless otherwise specified

Parameter	Symbol	Limits μPD780 (2.5 MHz) Min	Max	μPD780-1(4 MHz) Min	Max	μPD780-2 (6 MHz) Min	Max	Unit	Test Conditions
Clock period	t_C	0.4	(1)	0.25	(1)	0.165	(1)	μs	
Clock pulse width, clock high	$t_W(\phi H)$	180	(2)	110	(2)	65	(2)	ns	
Clock pulse width, clock low	$t_W(\phi L)$	180	2000	110	2000	72	2000	ns	
Clock rise and fall time	t_{Rf}		30		30		20	ns	
Address output delay	$t_{D(AD)}$		145		110		90	ns	
Delay to float	$t_{F(AD)}$		110		90		80	ns	
Address stable prior to $\overline{MREQ}$ (Memory cycle)	t_{ACM}	(3)		(3)		(3)		ns	C_L = 50 pF
Address stable prior to $\overline{IORQ}$, $\overline{RD}$ or $\overline{WR}$ (I/O cycle)	t_{ACI}	(4)		(4)		(4)		ns	
Address stable from $\overline{RD}$ or $\overline{WR}$	t_{CA}	(5)		(5)		(5)		ns	
Address stable from $\overline{RD}$ or $\overline{WR}$ during Float	t_{CAF}	(6)		(6)		(6)		ns	
Data output delay	$t_{D(D)}$		230		150		130	ns	
Delay to float during write cycle	$t_{F(D)}$		90		90		80	ns	
Data setup time to rising edge of clock during M_1 cycle	$t_{S\phi(D)}$	50		35		30		ns	
Data setup time to falling edge of clock during M_2 to M_5 cycles	$t_{S\phi(D)}$	60		50		40		ns	C_L = 200 pF
Data stable prior to $\overline{WR}$ (Memory cycle)	t_{DCM}	(7)		(7)		(7)		ns	
Data stable prior to $\overline{WR}$ (I/O cycle)	t_{DCI}	(8)		(8)		(8)		ns	
Data stable from $\overline{WR}$	t_{CDF}	(9)		(9)		(9)		ns	
$\overline{BUSRQ}$ setup time to rising edge of clock	$t_{S(BQ)}$	80		50		50		ns	
$\overline{BUSAK}$ delay from rising edge of clock to $\overline{BUSAK}$ low	$t_{DL(BA)}$		120		100		90	ns	
$\overline{BUSAK}$ delay from falling edge of clock to $\overline{BUSAK}$ high	$t_{DH(BA)}$		110		100		90	ns	C_L = 50 pF
Delay to float ($\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$ and $\overline{WR}$)	$t_{F(C)}$		100		80		70	ns	
$\overline{M_1}$ stable prior to $\overline{IORQ}$ (Interrupt ack.)	t_{MR}	(10)		(10)		(10)		ns	
Any hold time for setup time	t_H	0		0		0		ns	
$\overline{HALT}$ delay time from falling edge of clock	$t_{D(HT)}$		300		300		260	ns	C_L = 50 pF
$\overline{INT}$ setup time to rising edge of clock	$t_{S(IT)}$	80		80		70		ns	
$\overline{IORQ}$ delay from rising edge of clock to $\overline{IORQ}$ low	$t_{DL\phi(IR)}$		90		75		65	ns	
$\overline{IORQ}$ delay from falling edge of clock to $\overline{IORQ}$ low	$t_{DL\bar{\phi}(IR)}$		110		85		70	ns	
$\overline{IORQ}$ delay from rising edge of clock to $\overline{IORQ}$ high	$t_{DH\phi(IR)}$		100		85		70	ns	
$\overline{IORQ}$ delay from falling edge of clock to $\overline{IORQ}$ high	$t_{DH\bar{\phi}(IR)}$		110		85		70	ns	C_L = 50 pF
$\overline{M_1}$ delay from rising edge of clock to $\overline{M_1}$ low	$t_{DL(M_1)}$		130		100		80	ns	
$\overline{M_1}$ delay from rising edge of clock to $\overline{M_1}$ high	$t_{DH(M_1)}$		130		100		80	ns	
$\overline{MREQ}$ delay from falling edge of clock to $\overline{MREQ}$ low	$t_{DL\bar{\phi}(MR)}$		100		85		70	ns	
$\overline{MREQ}$ delay from rising edge of clock to $\overline{MREQ}$ high	$t_{DH\phi(MR)}$		100		85		70	ns	
$\overline{MREQ}$ delay from falling edge of clock to $\overline{MREQ}$ high	$t_{DH\bar{\phi}(MR)}$		100		85		70	ns	
Pulse width, $\overline{MREQ}$ low	$t_{W(MRL)}$	(11)		(11)		(11)		ns	
Pulse width, $\overline{MREQ}$ high	$t_{W(MRH)}$	(12)		(12)		(12)		ns	
Pulse width, $\overline{NMI}$ low	$t_{W(NML)}$	80		80		70		ns	
$\overline{RESET}$ setup time to rising edge of clock	$t_{S(RS)}$	90		60		60		ns	
$\overline{RD}$ delay from rising edge of clock to $\overline{RD}$ low	$t_{DL\phi(RD)}$		100		85		70	ns	
$\overline{RD}$ delay from falling edge of clock to $\overline{RD}$ low	$t_{DL\bar{\phi}(RD)}$		130		95		80	ns	
$\overline{RD}$ delay from rising edge of clock to $\overline{RD}$ high	$t_{DH\phi(RD)}$		100		85		70	ns	
$\overline{RD}$ delay from falling edge of clock to $\overline{RD}$ high	$t_{DH\bar{\phi}(RD)}$		110		85		70	ns	C_L = 30 pF

AC Characteristics (cont)

T_A = 0°C to +70°C; V_{CC} = +5 V ± 5%; unless otherwise specified

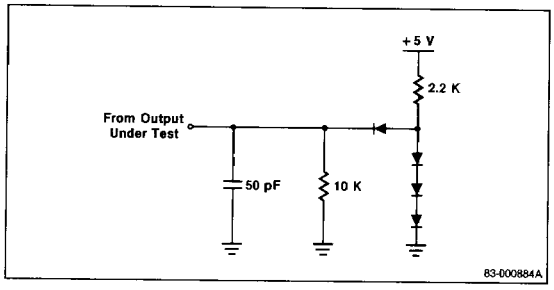
Parameter	Symbol	Limits μPD780 (2.5 MHz) Min	Max	μPD780-1(4 MHz) Min	Max	μPD780-2 (6 MHz) Min	Max	Unit	Test Conditions
$\overline{RFSH}$ delay from rising edge of clock to $\overline{RFSH}$ low	$t_{DL(RF)}$		180		130		110	ns	C_L = 30 pF
$\overline{RFSH}$ delay from rising edge of clock to $\overline{RFSH}$ high	$t_{DH(RF)}$		150		120		100	ns	
$\overline{WAIT}$ setup time to falling edge of clock	$t_{S(WT)}$	70		70		60		ns	
$\overline{WR}$ delay from rising edge of clock to $\overline{WR}$ low	$t_{DL\phi(WR)}$		80		65		60	ns	
$\overline{WR}$ delay from falling edge of clock $\overline{WR}$ low	$t_{DL\bar{\phi}(WR)}$		90		80		70	ns	
$\overline{WR}$ delay from falling edge of clock to $\overline{WR}$ high	$t_{DH\bar{\phi}(WR)}$		100		80		70	ns	
Pulse width to $\overline{WR}$ low	$t_{W(WRL)}$	(13)		(13)		(13)		ns	

Notes:

(1) $t_C = t_{W(\phi H)} + t_{W(\phi L)} + t_R + t_F$
(2) Though the structure of the 780 is static, 200μs is guaranteed maximum.
(3) $t_{ACM} = t_{w(\phi H)} + t_F - 65$ (75)* (50)**
(4) $t_{ACI} = t_c - 70$ (80)* (55)**
(5) $t_{CA} = t_{W(\phi L)} + t_R - 50$ (40)* (50)**
(6) $t_{CAF} = t_{W(\phi L)} + t_R - 45$ (60)* (40)**
(7) $t_{DCM} = t_C - 170$ (210)* (140)**
(8) $t_{DCI} = t_{W(\phi L)} + t_R - 170$ (210)* (140)**
(9) $t_{CDF} = t_{W(\phi L)} + t_R - 70$ (80)* (55)**
(10) $t_{MR} = 2t_C + t_{W(\phi H)} + t_F - 65$ (80)* (50)**
(11) $t_{W(MRL)} = t_C - 30$ (40)* (30)**
(12) $t_{W(MRH)} = t_{W(\phi H)} + t_F - 20$ (30)* (20)**
(13) $t_{W(WR)} = t_C - 30$ (40)* (30)**

* These values apply to the μPD780.
** These values apply to the μPD780-2.

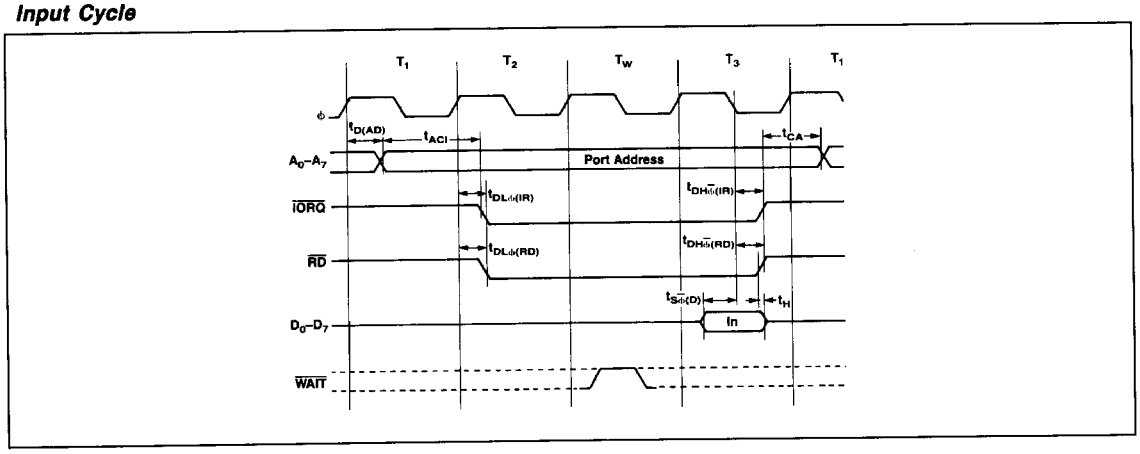
Load Circuit for Output



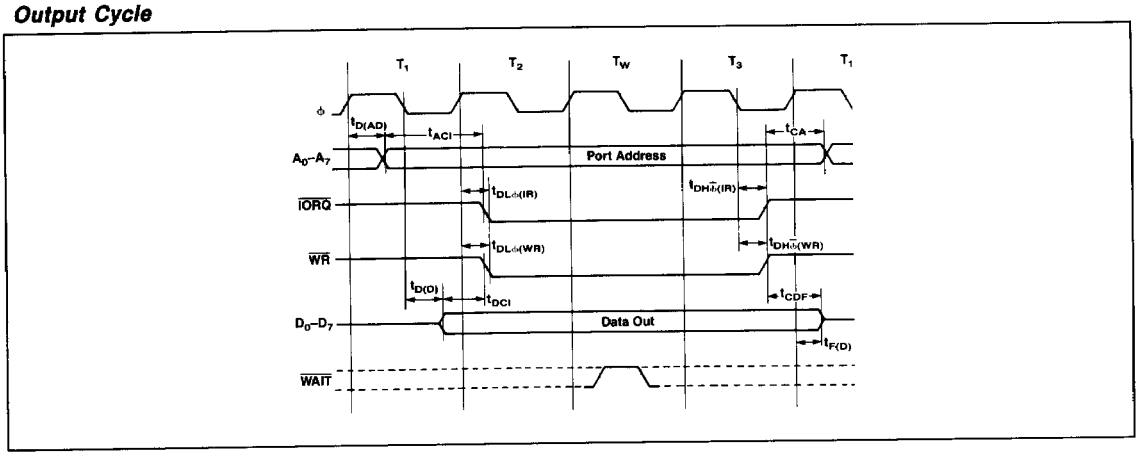
4

Timing Waveforms

Input Cycle

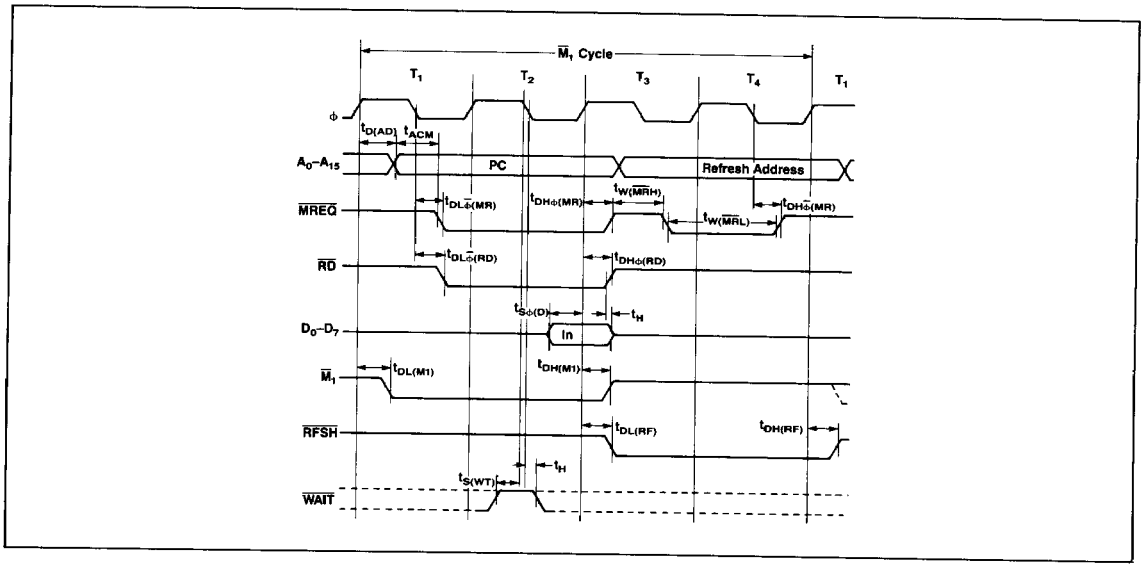


Output Cycle

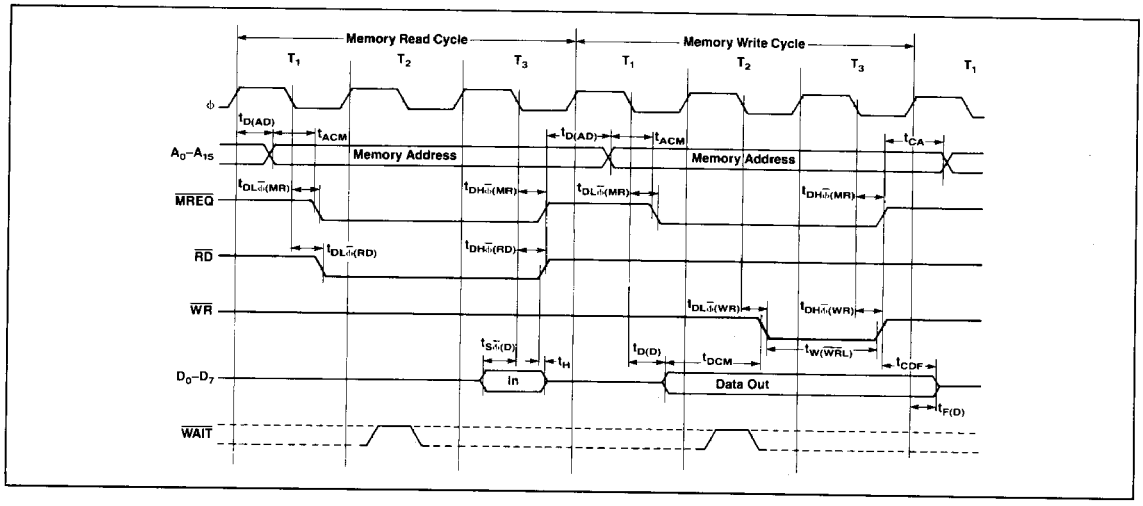


Timing Waveforms (cont)

M_1 Cycle

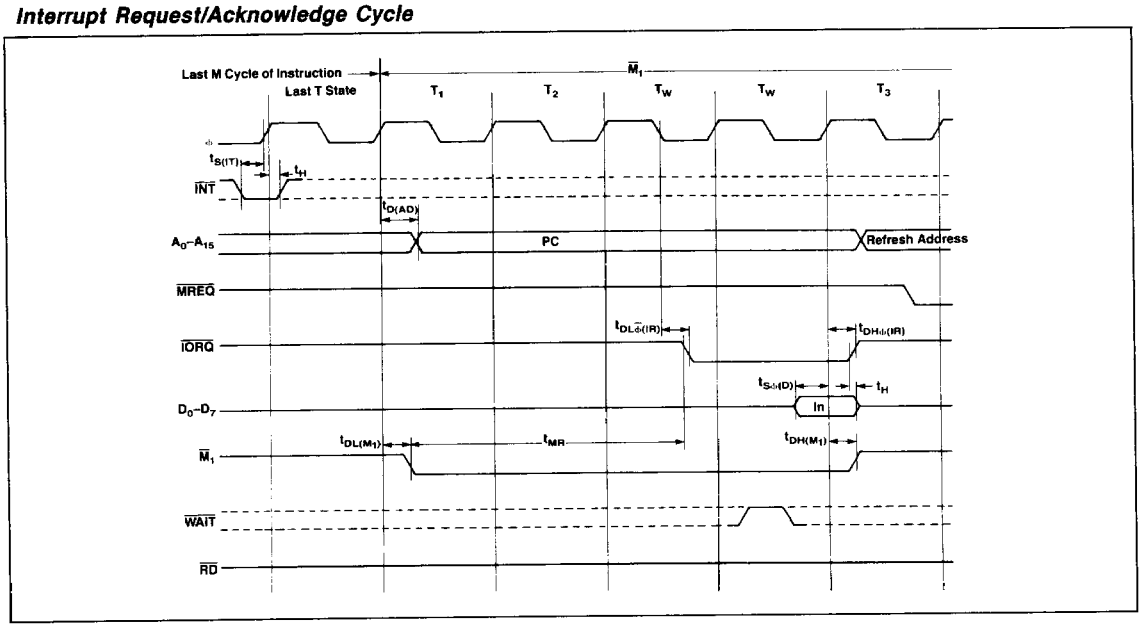


Memory Read/Write Cycles



Timing Waveforms (cont)

Interrupt Request/Acknowledge Cycle



Input and Output Cycles

In I/O operations, a single wait state (T_W) is automatically included to provide adequate time for an I/O port to decode the address from the port address lines and initiate a wait condition if needed.

Opcode Fetch Instruction Cycle

At the beginning of the cycle, the contents of the program counter are placed on the address bus. After approximately one-half cycle, $\overline{MREQ}$ is asserted and its falling edge can be used directly by the external memory as a chip enable signal. The data from the external memory can be gated onto the data bus when $\overline{RD}$ is asserted. The CPU reads the data at the rising edge of T_3. During T_3 and T_4, external dynamic memory is refreshed while the instruction is decoded and executed. The assertion of $\overline{RFSH}$ indicates that the external dynamic memory requires a refresh read.

Memory Read or Write Cycles

In read and write operations, the $\overline{MREQ}$ and $\overline{RD}$ signals function the same as they do in opcode fetch operations. In a write operation $\overline{MREQ}$ is asserted and can be used directly by external memory as a chip enable signal when information on the address bus is stable. The $\overline{WR}$ signal is used as a write strobe to almost any type of semiconductor memory, and is asserted when data on the data bus is stable.

Interrupt Request/Acknowledge Cycle

The interrupt signal is sampled at the rising edge of the final clock pulse at the end of an instruction. When an interrupt is accepted, an M_1 cycle is begun. Instead of $\overline{MREQ}$, $\overline{IORQ}$ is asserted during this cycle to indicate that an 8-bit vector address can be placed on the data bus by the interrupting device. This cycle includes the automatic addition of two wait states to facilitate the implementation of a daisy-chain priority interrupt protocol.

Instruction Set

The instruction set of the μPD780 consists of 158 types of instructions divided into 16 categories as follows:

- 8-bit load operations
- register exchanges
- memory block searches
- 16-bit arithmetic operations
- rotate and shift operations
- jump operations
- restart operations
- miscellaneous operations
- 16-bit load operations
- memory block transfers
- 8-bit arithmetic and logic operations
- bit set, reset, and test operations
- I/O operations
- call operations
- return operations
- general-purpose accumulator and flag operations

This comprehensive instruction set is made more powerful by the array of addressing modes implemented by the architecture, as follows:

- bit addressing
- register-indirect addressing
- immediate addressing
- extended addressing
- implied addressing
- register addressing
- relative addressing
- immediate-extended addressing
- indexed addressing
- modified page zero addressing

Instruction Set Symbol Definitions

Symbol	Description
•	Flag not affected
0	Flag set
X	Flag
↕	Flag affected according to result of operation
V	Overflow set
P	Parity set
IFF	Interrupt flip-flop set
C	Carry/Link
Z	Zero
P/V	Parity/Overflow
S	Sign
N	Add/Subtract
H	Half Carry

4

Instruction Set

Mnemonic	Operation	Description	Operation Code 7 6 5 4 3 2 1 0	No. of Clocks	No. of Bytes	Flags C	Z	P/V	S	N	H
ADC HL, ss	HL ← HL + ss + CY	Add with carry reg. pair ss to HL	1 1 1 0 1 1 0 1 (A) 0 1 s s 1 0 1 0	15	1	‡	‡	V	‡	0	X
ADC A, r	A ← A + r + CY	Add with carry Reg. r to ACC	1 0 0 0 1 r r r (B)	4	1	‡	‡	V	‡	0	‡
ADC A, n	A ← A + n + CY	Add with carrry value n to ACC	1 1 0 0 1 1 1 0 n n n n n n n n	7	2	‡	‡	V	‡	0	‡
ADC A, (HL)	A ← A + (HL) + CY	Add with carry loc. (HL) to ACC	1 0 0 0 1 1 1 0	7	1	‡	‡	V	‡	0	‡
ADC A, (IX + d)	A ← A + (IX + d) + CY	Add with carry loc. (IX + d) to ACC	1 1 0 1 1 1 0 1 1 0 0 0 1 1 1 0 d d d d d d d d	19	3	‡	‡	V	‡	0	‡
ADC A, (IY + d)	A ← A + (IY + d) + CY	Add with carry loc. (IY + d) to ACC	1 1 1 1 1 1 0 1 1 0 0 0 1 1 1 0 d d d d d d d d	19	3	‡	‡	V	‡	0	‡
ADD A, n	A ← A + n	Add value n to ACC	1 1 0 0 0 1 1 0 n n n n n n n n	7	2	‡	‡	V	‡	0	‡
ADD A, r	A ← A + r	Add Reg. r to ACC	1 0 0 0 0 r r r (B)	4	1	‡	‡	V	‡	0	‡
ADD A, (HL)	A ← A + (HL)	Add location (HL) to ACC	1 0 0 0 0 1 1 0	7	1	‡	‡	V	‡	0	‡
ADD A, (IX + d)	A ← A + (IX + d)	Add location (IX + d) to ACC	1 1 0 1 1 1 0 1 1 0 0 0 0 1 1 0 d d d d d d d d	19	3	‡	‡	V	‡	0	‡
ADD A, (IY + d)	A ← A + (IY + d)	Add location (IY + d) to ACC	1 1 1 1 1 1 0 1 1 0 0 0 0 1 1 0 d d d d d d d d	19	3	‡	‡	V	‡	0	‡
ADD HL, ss	HL ← HL + ss	Add Reg. pair ss to HL	0 0 s s 1 0 0 1 (A)	11	1	‡	•	•	•	0	X
ADD IX, pp	IX ← IX + pp	Add Reg. pair pp to IX	1 1 0 1 1 1 0 1 (C) 0 0 p p 1 0 0 1	15	2	‡	•	•	•	0	X
ADD IY, rr	IY ← IY + rr	Add Reg. pair rr to IY	1 1 1 1 1 1 0 1 (D) 0 0 r r 1 0 0 1	15	2	‡	•	•	•	0	X
AND r	A ← A ∧ r	Logical 'AND' of Reg. r ∧ ACC	1 0 1 0 0 r r r (B)	4	1	0	‡	P	‡	0	‡
AND n	A ← A ∧ n	Logical 'AND' of value n ∧ ACC	1 1 1 0 0 1 1 0 n n n n n n n n	7	2	0	‡	P	‡	0	‡
AND (HL)	A ← A ∧ (HL)	Logical 'AND' of loc. (HL) ∧ ACC	1 0 1 0 0 1 1 0	7	1	0	‡	P	‡	0	‡
AND (IX + d)	A ← A ∧ (IX + d)	Logical 'AND' of loc. (IX + d) ∧ ACC	1 1 0 1 1 1 0 1 1 0 1 0 0 1 1 0 d d d d d d d d	19	3	0	‡	P	‡	0	‡
AND (IY + d)	A ← A ∧ (IY + d)	Logical 'AND' of loc. (IY + d) ∧ ACC	1 1 1 1 1 1 0 1 1 0 1 0 0 1 1 0 d d d d d d d d	19	3	0	‡	P	‡	0	‡

Instruction Set (cont)

Mnemonic	Operation	Description	Operation Code 7 6 5 4 3 2 1 0	No. of Clocks	No. of Bytes	Flags C	Z	P/V	S	N	H
CPIR	A – (HL) HL ← HL + 1 BC ← BC – 1 until A = (HL) or BC = 0	Compare location (HL) and ACC, increment HL, decrement BC Repeat until BC = C	1 1 1 0 1 1 0 1 1 0 1 1 0 0 0 1	21 if BC = 0 and A≠(HL) 16 if BC = 0 or A = (HL)	2	•	‡(2)	‡(1)	‡	1	‡
CPL	A← A	Complement ACC (1's comp.)	0 0 1 0 1 1 1 1	4	1	•	•	•	•	1	1
DAA		Decimal adjust ACC	0 0 1 0 0 1 1 1	4	1	‡	‡	P	‡	•	‡
DEC r	r ← r – 1	Decrement Reg. r	0 0 r r r 1 0 1 (B)	4	1	•	‡	V	‡	1	‡
DEC (HL)	(HL) ← (HL) – 1	Decrement loc. (HL)	0 0 1 1 0 1 0 1	11	1	•	‡	V	‡	1	‡
DEC (IX + d)	(IX + d) ← (IX + d) – 1	Decrement loc. (IX + d)	1 1 0 1 1 1 0 1 0 0 1 1 0 1 0 1 d d d d d d d d	23	3	•	‡	V	‡	1	‡
DEC (IY + d)	(IY + d) ← (IY + d) –1	Decrement loc. (IY + d)	1 1 1 1 1 1 0 1 0 0 1 1 0 1 0 1 d d d d d d d d	23	3	•	‡	V	‡	1	‡
DEC IX	IX ← IX – 1	Decrement IX	1 1 0 1 1 1 0 1 0 0 1 0 1 0 1 1	10	2	•	•	•	•	•	•
DEC IY	IY ← IY – 1	Decrement IY	1 1 1 1 1 1 0 1 0 0 1 0 1 0 1 1	10	2	•	•	•	•	•	•
DEC ss	ss ← ss – 1	Decrement Reg. pair ss	0 0 s s 1 0 1 1 (A)	6	1	•	•	•	•	•	•
DI	IFF ← 0	Disable interrupts	1 1 1 1 0 0 1 1	4	1	•	•	•	•	•	•
DJNZ, e	B ← B – 1 if B = 0 continue if B ≠ 0, PC ← PC + e	Decrement B and jump relative if B = 0	0 0 0 1 0 0 0 0 ← e-2 →	8	2	•	•	•	•	•	•
EI	IFF ← 1	Enable interrupts	1 1 1 1 1 0 1 1	4	1	•	•	•	•	•	•
EX (SP), HL	H ↔ (SP + 1), L ↔ (SP)	Exchange the location (SP) and HL	1 1 1 0 0 0 1 1	19	1	•	•	•	•	•	•
EX (SP), IX	IX_H ↔ (SP + 1) IX_L ↔ (SP)	Exchange the location (SP) and IX	1 1 0 1 1 1 0 1 1 1 1 0 0 0 1 1	23	2	•	•	•	•	•	•
EX (SP), IY	IY_H ↔ (SP + 1) IY_L ↔ (SP)	Exchange the location (SP) and IY	1 1 1 1 1 1 0 1 1 1 1 0 0 0 1 1	23	2	•	•	•	•	•	•
EX AF, AF′	AF ↔ AF′	Exchange the contents of AF, AF′	0 0 0 0 1 0 0 0	4	1	•	•	•	•	•	•
EX DE, HL	DE ↔ HL	Exchange the contents of DE and HL	1 1 1 0 1 0 1 1	4	1	•	•	•	•	•	•
EXX	BC ↔ BC′ DE ↔ DE′, HL ↔ HL′	Exchange the contents of BC, DE, HL with contents of BC′, DE′, HL′, respectively	1 1 0 1 1 0 0 1	4	1	•	•	•	•	•	•
HALT	Processor Halted	HALT (wait for interrupt or reset)	0 1 1 1 0 1 1 0	4	1	•	•	•	•	•	•

Instruction Set (cont)

Mnemonic	Operation	Description	Operation Code 7 6 5 4 3 2 1 0	No. of Clocks	No. of Bytes	Flags C	Z	P/V	S	N	H
BIT b, (HL)	$Z \leftarrow (\overline{HL})_b$	Test BIT b of location (HL)	1 1 0 0 1 0 1 1 (E) 0 1 b b b 1 1 0	12	2	•	‡	X	X	0	1
BIT b, (IX + d)	$Z \leftarrow (\overline{IX + d})_b$	Test BIT b at location (IX + d)	1 1 0 1 1 1 0 1 (E) 1 1 0 0 1 0 1 1 d d d d d d d d 0 1 b b b 1 1 0	20	4	•	‡	X	X	0	1
BIT b, (IY + d)	$Z \leftarrow (\overline{IY + d})_b$	Test BIT b at location (IY + d)	1 1 1 1 1 1 0 1 (E) 1 1 0 0 1 0 1 1 d d d d d d d d 0 1 b b b 1 1 0	20	4	•	‡	X	X	0	1
BIT b, r	$Z \leftarrow \bar{r}_b$	Test BIT of Reg. r	1 1 0 0 1 0 1 1 0 1 b b b r r r (B) (E)	8	2	•	‡	X	X	0	1
CALL cc, nn	If condition cc false continues, else same as CALL nn	Call subroutine at location nn if condition cc is true	1 1 ← cc → 1 0 0 (H) n n n n n n n n n n n n n n n n	10	3	•	•	•	•	•	•
CALL nn	$(SP - 1) \leftarrow PC_H$ $(SP - 2) \leftarrow PC_L$ $PC \leftarrow nn$	Unconditional call subroutine at location nn	1 1 0 0 1 1 0 1 n n n n n n n n n n n n n n n n	17	3	•	•	•	•	•	•
CCF	$CY \leftarrow \overline{CY}$	Complement carry flag	0 0 1 1 1 1 1 1	4	1	‡	•	•	•	0	X
CP r	A − r	Compare Reg. r with ACC	1 0 1 1 1 r r r (B)	4	1	‡	‡	V	‡	1	‡
CP n	A − n	Compare value n with ACC	1 1 1 1 1 1 1 0 n n n n n n n n	7	2	‡	‡	V	‡	1	‡
CP (HL)	A − (HL)	Compare loc. (HL) with ACC	1 0 1 1 1 1 1 0	7	1	‡	‡	V	‡	1	‡
CP (IX + d)	A − (IX + d)	Compare loc. (IX + d) with ACC	1 1 0 1 1 1 0 1 1 0 1 1 1 1 1 0 d d d d d d d d 1 1 1 1 1 1 0 1	19	4	‡	‡	V	‡	1	‡
CP (IY + d)		Compare loc. (IY + d) with ACC	1 0 1 1 1 1 1 0 d d d d d d d d	19	2	‡	‡	V	‡	1	‡
CPD	A − (HL) HL ← HL − 1 BC ← BC − 1	Compare location (HL) and ACC, decrement HL and BC	1 1 1 0 1 1 0 1 1 0 1 0 1 0 0 1	16	2	•	‡(2)	‡(1)	‡	1	‡
CPDR	A − (HL) HL ← HL − 1 BC ← BC − 1 until A = (HL) or BC = 0	Compare location (HL) and ACC, decrement HL and BC, repeat until BC = 0	1 1 1 0 1 1 0 1 1 0 1 1 1 0 0 1	21 if BC = 0 and A≠(HL) 16 if BC = 0 or A = (HL)	2	•	‡(2)	‡(1)	‡	1	‡
CPI	A − (HL) HL ← HL + 1, BC ← BC − 1	Compare location (HL) and ACC, increment HL and decrement BC	1 1 1 0 1 1 0 1 1 0 1 0 0 0 0 1	16	2	•	‡(2)	‡(1)	‡	1	‡

Instruction Set (cont)

Mnemonic	Operation	Description	Operation Code 7	6	5	4	3	2	1	0	No. of Clocks	No. of Bytes	Flags C	Z	P/V	S	N	H
IM 0		Set interrupt mode 0	1	1	1	0	1	1	0	1	8	2	•	•	•	•	•	•
			0	1	0	0	0	1	1	0								
IM 1		Set interrupt mode 1	1	1	1	0	1	1	0	1	8	2	•	•	•	•	•	•
			0	1	0	1	0	1	1	0								
IM 2		Set interrupt mode 2	1	1	1	0	1	1	0	1	8	2	•	•	•	•	•	•
			0	1	0	1	1	1	1	0								
IN A, (n)	A ← (n)	Load ACC with input from device n	1	1	0	1	1	0	1	1	11	2	•	•	•	•	•	•
			n	n	n	n	n	n	n	n								
IN r, (C)	r ← (C)	Load Reg. r with input from device (C)	1	1	1	0	1	1	0	1 (I)	12	2	•	‡	P	‡	0	‡
			0	1	r	r	r	0	0	0								
INC (HL)	(HL) ← (HL) + 1	Increment location (HL)	0	0	1	1	0	1	0	0	11	1	•	‡	V	‡	0	‡
INC IX	IX ← IX + 1	Increment IX	1	1	0	1	1	1	0	1	10	2	•	•	•	•	•	•
			0	0	1	0	0	0	1	1								
INC (IX + d)	(IX + d) ← (IX + d) + 1	Increment location (IX + d)	1	1	0	1	1	1	0	1	23	3	•	‡	V	‡	0	‡
			0	0	1	1	0	1	0	0								
			d	d	d	d	d	d	d	d								
INC IY	IY ← IY + 1	Increment IY	1	1	1	1	1	1	0	1	10	2	•	•	•	•	•	•
			0	0	1	0	0	0	1	1								
INC (IY + d)	(IY + d) ← (IY + d) + 1	Increment location (IY + d)	1	1	1	1	1	1	0	1	23	3	•	‡	V	‡	0	‡
			0	0	1	1	0	1	0	0								
			d	d	d	d	d	d	d	d								
INC r	r ← r + 1	Increment Reg. r	0	0	r	r	r	1	0	0 (B)	4	1	•	‡	V	‡	0	‡
INC ss	ss ← ss + 1	Increment Reg. pair ss	0	0	s	s	0	0	1	1 (A)	6	1	•	•	•	•	•	•
IND	(HL) ← (C) B ← B′ − 1, HL ← HL − 1	Load location (HL) with input from port (C), decrement HL and B	1	1	1	0	1	1	0	1	16	2	•	‡(3)	X	X	1	X
			1	0	1	0	1	0	1	0								
INDR	(HL) ← (C) B ← B − 1 HL ← HL − 1 until B = 0	Load location (HL) with input from port (C), decrement HL and decrement B, repeat until B = 0	1	1	1	0	1	1	0	1	21	2	•	1	X	X	1	X
			1	0	1	1	1	0	1	0								
INI	(HL) ← (C) B ← B − 1, HL ← HL + 1	Load location (HL) with input from port (C), and increment HL and decrement B	1	1	1	0	1	1	0	1	16	2	•	‡(3)	X	X	1	X
			1	0	1	0	0	0	1	0								
INIR	(HL) ← (C) B ← B − 1 HL ← HL + 1 until B = 0	Load location (HL) with input from port (C), increment HL and decrement B, repeat until B = 0	1	1	1	0	1	1	0	1	21	2	•	1	X	X	1	X
			1	0	1	1	0	0	1	0								
JP (HL)	PC ← HL	Unconditional jump to (HL)	1	1	1	0	1	0	0	1	4	1	•	•	•	•	•	•
JP (IX)	PC ← IX	Unconditional jump to (IX)	1	1	0	1	1	1	0	1	8	2	•	•	•	•	•	•
			1	1	1	0	1	0	0	1								
JP (IY)	PC ← IY	Unconditional jump to (IY)	1	1	1	1	1	1	0	1	8	2	•	•	•	•	•	•
			1	1	1	0	1	0	0	1								

Instruction Set (cont)

Mnemonic	Operation	Description	Operation Code 7 6 5 4 3 2 1 0	No. of Clocks	No. of Bytes	Flags C	Z	P/V	S	N	H
JP cc, nn	If cc true PC ← nn else continue	Jump to location nn if continue cc	1 1 ← cc → 0 1 0 (H) n n n n n n n n n n n n n n n n	10	3	•	•	•	•	•	•
JP nn	PC ← nn	Unconditional jump to location nn	1 1 0 0 0 0 1 1 n n n n n n n n n n n n n n n n	10	3	•	•	•	•	•	•
JR C, e	If C = 0 continue If C = 1 PC ← PC + e	Jump relative to PC + e, if carry = 1	0 0 1 1 1 0 0 0 ← e-2 →	7 if condition met, 12 if not	2	•	•	•	•	•	•
JR e	PC ← PC + e	Unconditional jump relative to PC + e	0 0 0 1 1 0 0 0 ← e-2 →	12	2	•	•	•	•	•	•
JR NC, e	If C = 1 continue If C = 0 PC ← PC + e	Jump relative to PC + e if carry = 0	0 0 1 1 0 0 0 0 ← e-2 →	7	2	•	•	•	•	•	•
JR NZ, e	If Z = 1 continue	Jump relative to PC + e if non-zero (Z = 0)	0 0 1 0 0 0 0 0 ← e-2 →	7	2	•	•	•	•	•	•
JR Z, e	if Z = 0 continue	Jump relative to PC + e if zero (Z = 1)	0 0 1 0 1 0 0 0 ← e-2 →	7	2	•	•	•	•	•	•
LD A, (BC)	A ← (BC)	Load ACC with location (BC)	0 0 0 0 1 0 1 0	7	1	•	•	•	•	•	•
LD A, (DE)	A ← (DE)	Load ACC with location (DE)	0 0 0 1 1 0 1 0	7	1	•	•	•	•	•	•
LD A, I	A ← I	Load ACC with I	1 1 1 0 1 1 0 1 0 1 0 1 0 1 1 1	9	2	•	‡	IFF	‡	0	0
LD A, (nn)	A ← (nn)	Load ACC with location nn	0 0 1 1 1 0 1 0 n n n n n n n n n n n n n n n n	13	3	•	•	•	•	•	•
LD A, R	A ← R	Load ACC with Reg. R	1 1 1 0 1 1 0 1 0 1 0 1 1 1 1 1	9	2	•	‡	IFF	‡	0	0
LD (BC), A	(BC) ← A	Load location (BC) with ACC	0 0 0 0 0 0 1 0	7	1	•	•	•	•	•	•
LD (DE), A	(DE) ← A	Load location (DE) with ACC	0 0 0 1 0 0 1 0	7	1	•	•	•	•	•	•
LD (HL), n	(HL) ← n	Load location (HL) with value n	0 0 1 1 0 1 1 0 n n n n n n n n	10	2	•	•	•	•	•	•
LD ss, nn	ss ← nn	Load Reg. pair ss with value nn	0 0 s s 0 0 0 1 (A) n n n n n n n n n n n n n n n n	20	4	•	•	•	•	•	•
LD HL, (nn)	H ← (nn + 1)	Load HL with location (nn)	0 0 1 0 1 0 1 0 n n n n n n n n n n n n n n n n	16	3	•	•	•	•	•	•
LD (HL), r	(HL) ← r	Load location (HL) with Reg. r	0 1 1 1 0 r r r (B)	7	1	•	•	•	•	•	•
LD I, A	I ← A	Load I with ACC	1 1 1 0 1 1 0 1 0 1 0 0 0 1 1 1	9	2	•	•	•	•	•	•

Instruction Set (cont)

Mnemonic	Operation	Description	Operation Code 7 6 5 4 3 2 1 0	No. of Clocks	No. of Bytes	Flags C	Z	P/V	S	N	H
LD IX, nn	IX ← nn	Load IX with value nn	1 1 0 1 1 1 0 1 0 0 1 0 0 0 0 1 n n n n n n n n n n n n n n n n	19	4	•	•	•	•	•	•
LD IX, (nn)	IX_H ← (nn + 1) IX_L ← (nn)	Load IX with location (nn)	1 1 0 1 1 1 0 1 0 0 1 0 1 0 1 0 n n n n n n n n n n n n n n n n	20	4	•	•	•	•	•	•
LD (IX + d), n	(IX + d) ← n	Load location (IX + d) with value n	1 1 0 1 1 1 0 1 0 0 1 1 0 1 1 0 d d d d d d d d n n n n n n n n	19	4	•	•	•	•	•	•
LD (IX + d), r	(IX + d) ← r	Load location (IX + d) with Reg. r	1 1 0 1 1 1 0 1 (B) 0 1 1 1 0 r r r d d d d d d d d	19	3	•	•	•	•	•	•
LD IY, nn	IY ← nn	Load IY with value nn	1 1 1 1 1 1 0 1 0 0 1 0 0 0 0 1 n n n n n n n n n n n n n n n n	14	4	•	•	•	•	•	•
LD IY, (nn)	IY_H ← (nn + 1) IY_L ← (nn)	Load IY with location (nn)	1 1 1 1 1 1 0 1 0 0 1 0 1 0 1 0 n n n n n n n n n n n n n n n n	20	4	•	•	•	•	•	•
LD ss, (nn)	ss_H ← (nn + 1) ss_L ← (nn)	Load Reg. pair dd with location (nn)	1 1 1 0 1 1 0 1 (A) 0 1 s s 1 0 1 1 n n n n n n n n n n n n n n n n	20	4	•	•	•	•	•	•
LD (IY + d), n	(IY + d) ← n	Load (IY + d) with value n	1 1 1 1 1 1 0 1 0 0 1 1 0 1 1 0 d d d d d d d d n n n n n n n n	19	4	•	•	•	•	•	•
LD (IY + d), r	(IY + d) ← r	Load location (IY + d) with Reg. r	1 1 1 1 1 1 0 1 (B) 0 1 1 1 0 r r r d d d d d d d d	19	3	•	•	•	•	•	•
LD (nn), A	(nn) ← A	Load location (nn) with ACC	0 0 1 1 0 0 1 0 n n n n n n n n n n n n n n n n	13	3	•	•	•	•	•	•
LD (nn), ss	(nn + 1) ← ss_H (nn) ← ss_L	Load location (nn) with Reg. pair dd	1 1 1 0 1 1 0 1 (A) 0 1 s s 0 0 1 1 n n n n n n n n n n n n n n n n	20	4	•	•	•	•	•	•

Instruction Set (cont)

Mnemonic	Operation	Description	Operation Code 7 6 5 4 3 2 1 0	No. of Clocks	No. of Bytes	Flags C	Z	P/V	S	N	H
LD (nn), HL	(nn + 1) ← H (nn) ← L	Load location (nn) with HL	0 0 1 0 0 0 1 0 n n n n n n n n n n n n n n n n	16	3	•	•	•	•	•	•
LD (nn), IX	(nn +1) ← IX_H (nn) ← IX_L	Load location (nn) with IX	1 1 0 1 1 1 0 1 0 0 1 0 0 0 1 0 n n n n n n n n n n n n n n n n	20	4	•	•	•	•	•	•
LD(nn), IY	(nn +1) ← IY_H (nn) ← IY_L	Load location (nn) with IY	1 1 1 1 1 1 0 1 0 0 1 0 0 0 1 0 n n n n n n n n n n n n n n n n	20	4	•	•	•	•	•	•
LD R, A	R ← A	Load R with ACC	1 1 1 0 1 1 0 1 0 1 0 0 1 1 1 1	9	2	•	•	•	•	•	•
LD r, (HL)	r ← (HL)	Load Reg. r with location (HL)	0 1 r r r 1 1 0 (B)	7	1	•	•	•	•	•	•
LD r, (IX + d)	r ← (IX + d)	Load Reg. r with location (IX + d)	1 1 0 1 1 1 0 1 (B) 0 1 r r r 1 1 0 d d d d d d d d	19	3	•	•	•	•	•	•
LD r, (IY +d)	r ← (IY +d)	Load Reg. r with location (IY + d)	1 1 1 1 1 1 0 1 (B) 0 1 r r r 1 1 0 d d d d d d d d	19	3	•	•	•	•	•	•
LD r, n	r ←n	Load Reg. r with value n	0 0 r r r 1 1 0 (B) n n n n n n n n	7	2	•	•	•	•	•	•
LD, r, r'	r ← r'	Load Reg. r with Reg. r'	0 1 r r r r' r' r' (F)	4	1	•	•	•	•	•	•
LD SP, HL	SP ← HL	Load SP with HL	1 1 1 1 1 0 0 1	6	1	•	•	•	•	•	•
LD SP, IX	SP ← IX	Load SP with IX	1 1 0 1 1 1 0 1 1 1 1 1 1 0 0 1	10	2	•	•	•	•	•	•
LD SP, IY	SP ← IY	Load SP with IY	1 1 1 1 1 1 0 1 1 1 1 1 1 0 0 1	10	2	•	•	•	•	•	•
LDD	(DE) ← (HL) DE ← DE − 1 HL ← HL − 1, BC ← BC − 1	Load location (DE) with location (HL), decrement DE, HL and BC	1 1 1 0 1 1 0 1 1 0 1 0 1 0 0 0	16	2	•	•	‡	•	0	0
LDDR	(DE) ← (HL) DE ← DE − 1 HL ← HL − 1, BC ← BC − 1 until BC = 0	Load location (DE) with location (HL)	1 1 1 0 1 1 0 1 1 0 1 1 1 0 0 0	21	2	•	•	0	•	0	0

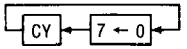
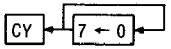
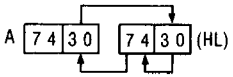
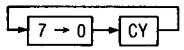
Instruction Set (cont)

Mnemonic	Operation	Description	Operation Code 7	6	5	4	3	2	1	0	No. of Clocks	No. of Bytes	Flags C	Z	P/V	S	N	H
LDI	(DE) ← (HL) DE ← DE + 1 HL ← HL + 1 BC ← BC − 1	Load location (DE) with location (HL), increment DE, HL; decrement BC	1	1	1	0	1	1	0	1	16	2	•	•	‡(T)	•	0	0
			1	0	1	0	0	0	0	0								
LDIR	(DE) ← (HL) DE ← DE + 1 HL ← HL + 1 BC ← BC − 1 until BC = 0	Load location (DE) with location (HL); increment DE, HL, decrement BC and repeat until BC = 0	1	1	1	0	1	1	0	1	21 if BC ≠ 0 16 if BC = 0	2	•	•	0	•	0	0
			1	0	1	1	0	0	0	0								
NEG	A ← 0 − A	Negate ACC (2's complement)	1	1	1	0	1	1	0	1	8	2	‡	‡	V	‡	1	‡
			0	1	0	0	0	1	0	0								
NOP		No operation	0	0	0	0	0	0	0	0	4	1	•	•	•	•	•	•
OR r	A ← AV r	Logical 'OR' of Reg. r and ACC	1	0	1	1	0	r	r	r (B)	4	1	0	‡	P	‡	0	‡
OR n	A ← AV n	Logical 'OR' of value n and ACC	1	1	1	1	0	1	1	0	7	2	•	‡	P	‡	0	‡
			n	n	n	n	n	n	n	n								
OR (HL)	A ← AV (HL)	Logical 'OR' of loc. (HL) and ACC	1	0	1	1	0	1	1	0	7	1	•	‡	P	‡	0	‡
OR (IX + d)	A ← (IX + d)	Logical 'OR' of loc. (IX + d) Λ ACC	1	1	0	1	1	1	0	1	19	3	•	‡	P	‡	0	‡
			1	0	1	1	0	1	1	0								
			d	d	d	d	d	d	d	d								
OR (IY +d)	A ← AV (IY + d)	Logical 'OR' of loc. (IY + d) Λ ACC	1	1	1	1	1	1	0	1	19	3	•	‡	P	‡	0	‡
			1	0	1	1	0	1	1	0								
			d	d	d	d	d	d	d	d								
OTDR	(C) ← (HL) B ← B − 1 HL ← HL − 1 until B = 0	Load output port (C) with contents of location (HL), decrement HL and B, repeat until B = 0	1	1	1	0	1	1	0	1	21 if B ≠ 0 16 if B = C	2	•	1	X	X	1	X
			1	0	1	1	1	0	1	1								
OTIR	(C) ← (HL) B ← B − 1 HI ← HL + 1 until B = 0	Load output port (C) with location (HL), increment HL, decrement B, repeat until B = 0	1	1	1	0	1	1	0	1	21 if B ≠ 0 16 if B = C	2	•	1	X	X	1	X
OUT (C), r	(C) ← r	Load output port (C) with Reg. r	1	1	1	0	1	1	0	1 (B)	12	2	•	•	•	•	•	•
			0	1	r	r	r	0	0	1								
OUT (n), A	(n) ← A	Load output port (n) with ACC	1	1	0	1	0	0	1	1	11	2	•	•	•	•	•	•
			n	n	n	n	n	n	n	n								
OUTD	(C) ← (HL) B ← B − 1, HL ← HL − 1	Load output port (C) with location (HL), increment HL and decrement B	1	1	1	0	1	1	0	1	16	2	•	‡(3)	X	X	1	X
			1	0	1	0	1	0	1	1								
OUTI	(C) ← (HL) B ← B − 1, HL ← HL + 1	Load output port (C) with location (HL), increment HL and decrement B	1	1	1	0	1	1	0	1	16	2	•	‡(3)	X	X	1	X
			1	0	1	0	0	0	1	1								

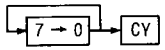
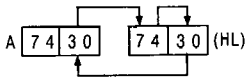
Instruction Set (cont)

Mnemonic	Operation	Description	Operation Code 7 6 5 4 3 2 1 0	No. of Clocks	No. of Bytes	Flags C	Z	P/V	S	N	H
POP IX	$IX_H \leftarrow (SP + 1)$ $IX_L \leftarrow (SP)$	Load IX with top of stack	1 1 0 1 1 1 0 1 1 1 1 0 0 0 0 1	14	2	•	•	•	•	•	•
POP IY	$IY_H \leftarrow (SP + 1)$ $IY_L \leftarrow (SP)$	Load IY with top of stack	1 1 1 1 1 1 0 1 1 1 1 0 0 0 0 1	14	2	•	•	•	•	•	•
POP qq	$qq_H \leftarrow (SP + 1)$ $qq_L \leftarrow (SP)$	Load Reg. pair qq with top of stack	1 1 q q 0 0 0 1 (G)	10	1	•	•	•	•	•	•
PUSH IX	$(SP - 2) \leftarrow IX_L$ $(SP - 1) \leftarrow IX_H$	Load IX onto stack	1 1 0 1 1 1 0 1 1 1 1 0 0 1 0 1	15	2	•	•	•	•	•	•
PUSH IY	$(SP - 2) \leftarrow IY_L$ $(SP - 1) \leftarrow IY_H$	Load IY onto stack	1 1 1 1 1 1 0 1 1 1 1 0 0 1 0 1	15	2	•	•	•	•	•	•
PUSH qq	$(SP - 2) \leftarrow qq_L$ $(SP - 1) \leftarrow qq_H$	Load Reg. pair qq onto stack	1 1 q q 0 1 0 1 (G)	11	1	•	•	•	•	•	•
RES b,r	$S_b \leftarrow 0$	Reset Bit b of Reg. r	1 1 0 0 1 0 1 1 (B) 1 0 b b b r r r (E)	8	2	•	•	•	•	•	•
RES b, (HL)	$S_b \leftarrow 0$, (HL)	Reset Bit b of loc. (HL)	1 1 0 0 1 0 1 1 1 0 b b b 1 1 0	15	2	•	•	•	•	•	•
RES b, (IX + d)	$S_b \leftarrow 0$ (IX + d)	Reset Bit b of loc. (IX + d)	1 1 0 1 1 1 0 1 1 1 0 0 1 0 1 1 d d d d d d d d 1 0 b b b 1 10	23	4	•	•	•	•	•	•
RES b, (IY + d)	$S_b \leftarrow 0$, (IY + d)	Reset Bit b of loc. (IY + d)	1 1 1 1 1 1 0 1 1 1 0 0 1 0 1 1 d d d d d d d d 1 0 b b b 1 1 0	23	4	•	•	•	•	•	•
RET	$PC_L \leftarrow (SP)$ $PC_H \leftarrow (SP + 1)$	Return from subroutine	1 1 0 0 1 0 0 1	10	1	•	•	•	•	•	•
RET cc	If condition cc is false cont. else $(PC_L \leftarrow (SP)$ $PC_H \leftarrow (SP + 1)$	Return from subroutine if condition cc is true	1 1 ← cc → 0 0 0 (H)	5 if CC false 11 if CC true	1	•	•	•	•	•	•
RETI		Return from interrupt	1 1 1 0 1 1 0 1 0 1 0 0 1 1 0 1	14	2	•	•	•	•	•	•
RETN		Return from non-maskable interrupt	1 1 1 0 1 1 0 1 0 1 0 0 0 1 0 1	14	2	•	•	•	•	•	•
RL r		Rotate left through carry Reg. r	1 1 0 0 1 0 1 1 (B) 0 0 0 1 0 r r r	2	2	‡	‡	P	‡	0	0
RL (HL)		Rotate left through carry loc. (HL)	1 1 0 0 1 0 1 1 0 0 0 1 0 1 1 0	4	2	‡	‡	P	‡	0	0

Instruction Set (cont)

Mnemonic	Operation	Description	Operation Code 7	6	5	4	3	2	1	0	No. of Clocks	No. of Bytes	Flags C	Z	P/V	S	N	H
RL (IX + d)	 m r, (HL)	Rotate left through carry loc. (IX + d)	1	1	0	1	1	1	0	1	6	4	‡	‡	P	‡	0	0
			1	1	0	0	1	0	1	1								
			d	d	d	d	d	d	d	d								
			0	0	0	1	0	1	1	0								
RL (IY + d)	(IX + d), (IY + d), A	Rotate left through carry loc. (IY + d)	1	1	1	1	1	1	0	1	6	4	‡	‡	P	‡	0	0
			1	1	0	0	1	0	1	1								
			d	d	d	d	d	d	d	d								
			0	0	0	1	0	1	1	0								
RLA		Rotate left ACC through carry	0	0	0	1	0	1	1	1	4	1	‡	•	•	•	0	0
RLC (HL)		Rotate location (HL) left circular	1	1	0	0	1	0	1	1	15	2	‡	‡	P	‡	0	0
			0	0	0	0	0	1	1	0								
RLC (IX + d)		Rotate location (IX + d) left circular	1	1	0	1	1	1	0	1	23	4	‡	‡	P	‡	0	0
			1	1	0	0	1	0	1	1								
			d	d	d	d	d	d	d	d								
			0	0	0	0	0	1	1	0								
RLC (IY + d)	 m = r, (HL), (IX + d), (IY + d), A	Rotate location (IY + d) left circular	1	1	1	1	1	1	0	1	23	4	‡	‡	P	‡	0	0
			1	1	0	0	1	0	1	1								
			d	d	d	d	d	d	d	d								
			0	0	0	0	0	1	1	0								
RLC r		Rotate Reg. r left circular	1	1	0	0	1	0	1	1 (B)	8	2	‡	‡	P	‡	0	0
			0	0	0	0	0	r	r	r								
RLCA		Rotate left circular ACC	0	0	0	0	0	1	1	1	4	1	‡	•	•	•	0	0
RLD		Rotate digit left and right between ACC and location (HL)	1	1	1	0	1	1	0	1	18	2	•	‡	P	‡	0	0
			0	1	1	0	1	1	1	1								
RR r		Rotate right through carry Reg. r	1	1	0	0	1	0	1	1 (B)	2	2	‡	‡	P	‡	0	0
			0	0	0	1	1	r	r	r								
RR (HL)		Rotate right through carry loc. (HL)	1	1	0	0	1	0	1	1	4	2	‡	‡	P	‡	0	0
			0	0	0	1	1	1	1	0								
RR (IX + d)		Rotate right through carry loc. (IX + d)	1	1	0	1	1	1	0	1	6	4	‡	‡	P	‡	0	0
			1	1	0	0	1	0	1	1								
			d	d	d	d	d	d	d	d								
			0	0	0	1	1	1	1	0								
RR (IY + d)	 m = r, (HL), (IX + d), (IY + d), A	Rotate right through carry loc. (IY + d)	1	1	1	1	1	1	0	1	6	4	‡	‡	P	‡	0	0
			1	1	0	0	1	0	1	1								
			d	d	d	d	d	d	d	d								
			0	0	0	1	1	1	1	0								
RRA		Rotate right ACC through carry	0	0	0	1	1	1	1	1	4	1	‡	•	•	•	0	0
RRC r		Rotate Reg. r right circular	1	1	0	0	1	0	1	1 (B)	2	2	‡	‡	P	‡	0	0
			0	0	0	0	1	r	r	r								

Instruction Set (cont)

Mnemonic	Operation	Description	Operation Code 7 6 5 4 3 2 1 0	No. of Clocks	No. of Bytes	Flags C	Z	P/V	S	N	H
RRC (HL)		Rotate loc. (HL) right circular	1 1 0 0 1 0 1 1 0 0 0 0 1 1 1 0	4	2	‡	‡	P	‡	0	0
RRC (IX + d)		Rotate loc (IX + d) right circular	1 1 0 1 1 1 0 1 1 1 0 0 1 0 1 1 d d d d d d d d 0 0 0 0 1 1 1 0	6	4	‡	‡	P	‡	0	0
RRC (IY + d)	m = r, (HL), (IX + d), (IY + d), A	Rotate loc. (IY + d) right circular	1 1 1 1 1 1 0 1 1 1 0 0 1 0 1 1 d d d d d d d d 0 0 0 0 1 1 1 0	6	4	‡	‡	P	‡	0	0
RRCA		Rotate right circular ACC	0 0 0 0 1 1 1 1	4	1	‡	•	•	•	0	0
RRD		Rotate digit right and then left between ACC and location (HL)	1 1 1 0 1 1 0 1 0 1 1 0 0 1 1 1	18	2	•	‡	P	‡	0	0
RST_t	(SP − 1) ← PC_H (SP − 2) ← PC_L PC_H ← 0, PC_L ← T	Restart to location T	1 1 t t t 1 1 1	11	1	•	•	•	•	•	•
SBC A, r	A ← A − r CY	Subtract Reg. r from ACC w/carry	1 0 0 1 1 r r r (B)	4	1	‡	‡	V	‡	1	‡
SBC A, n	A ← A − n − CY	Subtract value n from ACC with carry	1 1 0 1 1 1 1 0 n n n n n n n n	7	2	‡	‡	V	‡	1	‡
SBC A, (HL)	A ← A − (HL) − CY	Sub. loc. (HL) from ACC w/carry	1 0 0 1 1 1 1 0	7	1	‡	‡	V	‡	1	‡
SBC A, (IX + d)	A ← A − (IX + d) − CY	Subtract loc. (IX + d) from ACC with carry	1 1 0 1 1 1 0 1 1 0 0 1 1 1 1 0 d d d d d d d d	19	3	‡	‡	V	‡	1	‡
SBC A, (IY + d)	A ← A − (IY + d) − CY	Subtract loc. (IY + d) from ACC with carry	1 1 1 1 1 1 0 1 1 0 0 1 1 1 1 0 d d d d d d d d	19	3	‡	‡	V	‡	1	‡
SBC HL, ss	HL ← HL − ss − CY	Subtract Reg. pair ss from HL with carry	1 1 1 0 1 1 0 1 (A) 0 1 s s 0 0 1 0	15	2	‡	‡	V	‡	1	X
SCF	CY ← 1	Set carry flag (C = 1)	0 0 1 1 0 1 1 1	4	1	1	•	•	•	0	0
SET b, (HL)	$(HL)_b$ ← 1	Set Bit b of location (HL)	1 1 0 0 1 0 1 1 (E) 1 1 b b b 1 1 0	15	2	•	•	•	•	•	•
SET b, (IX + d)	$(IX + d)_b$ ← 1	Set Bit b of location (IX + d)	1 1 0 1 1 1 0 1 (E) 1 1 0 0 1 0 1 1 d d d d d d d d 1 1 b b b 1 1 0	23	4	•	•	•	•	•	•

Instruction Set (cont)

Mnemonic	Operation	Description	Operation Code 7 6 5 4 3 2 1 0	No. of Clocks	No. of Bytes	Flags C	Z	P/V	S	N	H
SET b, (IY + d)	$(IY + d)_b \leftarrow 1$	Set Bit b of location (IY + d)	1 1 1 1 1 1 0 1 (E) 1 1 0 0 1 0 1 1 d d d d d d d d 1 1 b b b 1 1 0	23	4	•	•	•	•	•	•
SET b, r	$r_b \leftarrow 1$	Set Bit b of Reg. r	1 1 0 0 1 0 1 1 (B) 1 1 b b b r r r	8	2	•	•	•	•	•	•
SLA r		Shift Reg. r left arithmetic	1 1 0 0 1 0 1 1 (B) 0 0 1 0 0 r r r	8	2	‡	‡	P	‡	0	0
SLA (HL)	CY ← 7 ← 0 ← 0	Shift loc. (HL) left arithmetic	1 1 0 0 1 0 1 1 0 0 1 0 0 1 1 0	15	2	‡	‡	P	‡	0	0
SLA (IX + d)	m = r, (HL), (IX + d), (IY + d)	Shift loc. (IX + d) left arithmetic	1 1 0 1 1 1 0 1 1 1 0 0 1 0 1 1 d d d d d d d d 0 0 1 0 0 1 1 0	23	4	‡	‡	P	‡	0	0
SLA (IY + d)		Shift loc. (IY + d) left arithmetic	1 1 1 1 1 1 0 1 1 1 0 0 1 0 1 1 d d d d d d d d 0 0 1 0 0 1 1 0	23	4	‡	‡	P	‡	0	0
SRA r		Shift Reg. r right arithmetic	1 1 0 0 1 0 1 1 (B) 0 0 1 0 1 r r r	8	2	‡	‡	P	‡	0	0
SRA (HL)	7 → 0 → CY	Shift loc. (HL) right arithmetic	1 1 0 0 1 0 1 1 0 0 1 0 1 1 1 0	15	2	‡	‡	P	‡	0	0
SRA (IX + d)	m = r, (HL), (IX + d), (IY + d)	Shift loc. (IX + d) right arithmetic	1 1 0 1 1 1 0 1 1 1 0 0 1 0 1 1 d d d d d d d d 0 0 1 0 1 1 1 0	23	4	‡	‡	P	‡	0	0
SRA (IY + d)		Shift loc. (IY + d) right arithmetic	1 1 1 1 1 1 0 1 1 1 0 0 1 0 1 1 d d d d d d d d 0 0 1 0 1 1 1 0	23	4	‡	‡	P	‡	0	0
SRL r		Shift Reg. r right logical	1 1 0 0 1 0 1 1 (B) 0 0 1 1 1 r r r	8	2	‡	‡	P	‡	0	0
SRL (HL)	0 → 7 → 0 → CY	Shift loc. (HL) right logical	1 1 0 0 1 0 1 1 0 0 1 1 1 1 1 0	15	2	‡	‡	P	‡	0	0
SRL (IX + d)	m = r, (HL), (IX + d), (IY + d)	Shift loc. (IX + d) right logical	1 1 0 1 1 1 0 1 1 1 0 0 1 0 1 1 d d d d d d d d 0 0 1 1 1 1 1 0	23	4	‡	‡	P	‡	0	0

Instruction Set (cont)

Mnemonic	Operation	Description	Operation Code 7 6 5 4 3 2 1 0	No. of Clocks	No. of Bytes	Flags C	Z	P/V	S	N	H
SRL (IY + d)		Shift loc. (IY + d) right logical	1 1 1 1 1 1 0 1 1 1 0 0 1 0 1 1 d d d d d d d d 0 0 1 1 1 1 1 0	23	4	‡	‡	P	‡	0	0
SUB r	A ← A − r	Subtract Reg. r from ACC	1 0 0 1 0 r r r (B)	4	1	‡	‡	V	‡	1	‡
SUB n	A ← A − n	Subtract value n from ACC	1 1 0 1 0 1 1 0 n n n n n n n n	7	2	‡	‡	V	‡	1	‡
SUB (HL)	A ← A − (HL)	Subtract loc. (HL) from ACC	1 0 0 1 0 1 1 0	7	1	‡	‡	V	‡	1	‡
SUB (IX + d)	A ← A − (IX + d)	Subtract loc. (IX + d) from ACC	1 1 0 1 1 1 0 1 1 0 0 1 0 1 1 0 d d d d d d d d	19	3	‡	‡	V	‡	1	‡
SUB (IY + d)	A ← A − (IY + d)	Subtract loc. (IY + d) from ACC	1 1 1 1 1 1 0 1 1 0 0 1 0 1 1 0 d d d d d d d d	19	3	‡	‡	V	‡	1	‡
XOR r	A ← A ∀ r	Exclusive 'OR' Reg. r and ACC	1 0 1 0 1 r r r (B)	4	1	‡	‡	P	‡	1	‡
XOR n	A ← A ∀ n	Exclusive 'OR' value n and ACC	1 1 1 0 1 1 1 0 n n n n n n n n	7	2	‡	‡	P	‡	1	‡
XOR (HL)	A ← A ∀ (HL)	Exclusive 'OR' loc. (HL) and ACC	1 0 1 0 1 1 1 0	7	1	‡	‡	P	‡	1	‡
XOR (IX + d)	A ← A ∀ (IX + d)	Exclusive 'OR' loc. (IX + d) and ACC	1 1 0 1 1 1 0 1 1 0 1 0 1 1 1 0 d d d d d d d d	19	3	‡	‡	P	‡	1	‡
XOR (IY + d)	A ← A ∀ (IY + d)	Exclusive 'OR' loc. (IY + d) and ACC	1 1 1 1 1 1 0 1 1 0 1 0 1 1 1 0 d d d d d d d d	19	3	‡	‡	P	‡	1	‡

Note:

(1) P/V flag is 0 if B = 0, else P/V = 1
(2) Z = 1 if A = (HL), else Z = 0
(3) If B = 0, Z flag set, else reset

A		B		C		D		E		F		G		H			I	
Reg ss		Reg r		Reg pp		Reg rr		Bit b		Reg r, r'		Reg qq		CC	Condition	Relevant Flag	Reg r	
BC	00	A	111	BC	00	BC	00	0	000	A	111	BC	00	000	NZ	Non zero	Z	B 000
DE	01	B	000	DE	01	DE	01	1	001	B	000	DE	01	001	Z	Zero	Z	C 001
HL	10	C	001	IX	10	IY	10	2	010	C	001	HL	10	010	NC	Non carry	C	D 010
SP	11	D	010	SP	11	SP	11	3	011	D	010	AF	11	011	C	Carry	C	E 011
		E	011					4	100	E	011			100	PO	Parity odd	P/V	H 100
		H	100					5	101	H	100			101	PE	Parity even	P/V	L 101
		L	101					6	110	L	101			110	P	Sign positive	S	F 110
								7	111					111	M	Sign negative	S	A 111