Zilog - Z84C0010VEC Datasheet

Welcome to E-XFL.COM

Understanding Embedded - Microprocessors

Embedded microprocessors are specialized computing chips designed to perform specific tasks within an embedded system. Unlike general-purpose microprocessors found in personal computers, embedded microprocessors are tailored for dedicated functions within larger systems, offering optimized performance, efficiency, and reliability. These microprocessors are integral to the operation of countless electronic devices, providing the computational power necessary for controlling processes, handling data, and managing communications.

Applications of Embedded - Microprocessors

Embedded microprocessors are utilized across a broad spectrum of applications, making them indispensable in

Details	
Product Status	Obsolete
Core Processor	Z80
Number of Cores/Bus Width	1 Core, 8-Bit
Speed	10MHz
Co-Processors/DSP	-
RAM Controllers	-
Graphics Acceleration	No
Display & Interface Controllers	-
Ethernet	-
SATA	-
USB	-
Voltage - I/O	5.0V
Operating Temperature	-40°C ~ 100°C (TA)
Security Features	-
Package / Case	44-LCC (J-Lead)
Supplier Device Package	44-PLCC
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z84c0010vec

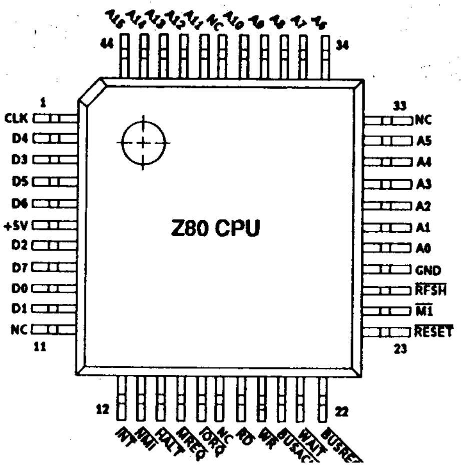

A15 A14 A13 A12 A11 NC A10 A9 A8 A7 A6
44
34
1
CLK
D4
D3
D5
D6
+5V
D2
D7
D0
D1
NC
11
33
NC
A5
A4
A3
A2
A1
A0
GND
RFSH
M1
RESET
23
Z80 CPU
12
22
INT NMI HALT MREQ IORQ NC RD WR BUSACK WAIT BUSREQ

Figure 2a. 44-Pin LQFP, Pin Assignments (Onlv available for 84C00)

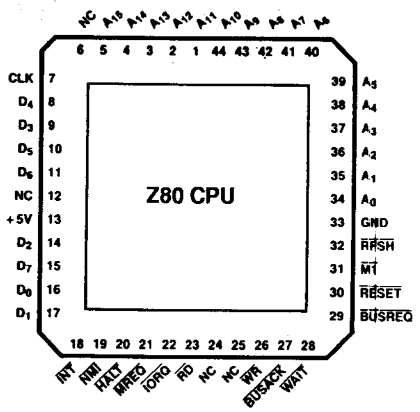


Figure 2b. 44-Pin Chip Carrier Pin Assignments

GENERAL DESCRIPTION

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

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

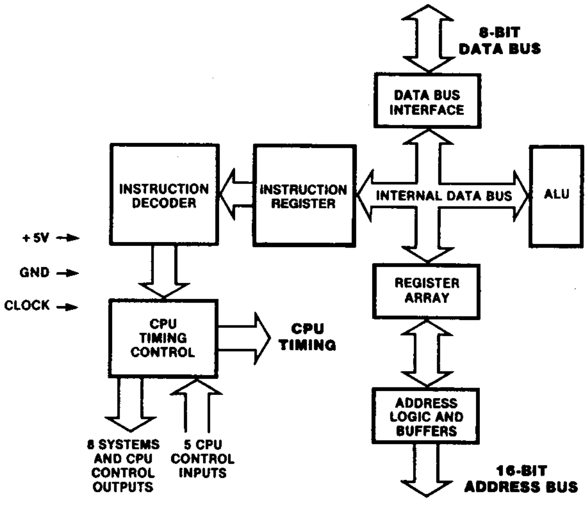


Figure 3. Z80C CPU Block Diagram

Table 1. Z80C CPU Registers

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

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

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

Mode 0 Interrupt Operation. This mode is similar to the 8080 microprocessor interrupt service procedures. The interrupting device places an instruction on the data bus. This is normally a Restart instruction, which will initiate a call to the selected one of eight restart locations in page zero of memory. Unlike the 8080, the Z80 CPU responds to the Call instruction with only one interrupt acknowledge cycle followed by two memory read cycles.

Mode 1 Interrupt Operation. Mode 1 operation is very similar to that for the $\overline{\text{NMI}}$. The principal difference is that the Mode 1 interrupt has only one restart location, 0038H.

Mode 2 Interrupt Operation. This interrupt mode has been designed to most effectively utilize the capabilities of the Z80 microprocessor and its associated peripheral family.The interrupting peripheral device selects the starting address of the interrupt service routine. It does this by placing an 8-bit vector on the data bus during the interrupt acknowledge cycle. The CPU forms a pointer using this byte as the lower 8 bits and the contents of the I register as the upper 8 bits. This points to an entry in a table of addresses for interrupt service routines. The CPU then jumps to the routine at that

address. This flexibility in selecting the interrupt service routine address allows the peripheral device to use several different types of service routines. These routines may be located at any available location in memory. Since the interrupting device supplies the low-order byte of the 2-byte vector, bit 0 (A_0) must be a zero.

Interrupt Enable/Disable Operation. Two flip-flops, IFF_1 and IFF_2, referred to in the register description, are used to signal the CPU interrupt status. Operation of the two flip-flops is described in Table 2. For more details, refer to the *Z80 CPU Technical Manual* (03-0029-01) and *Z80 Assembly Language Programming Manual* (03-0002-01).

Table 2. State of Flip-Flops

Action	IFF_1	IFF_2	Comments
CPU Reset	0	0	Maskable interrupt $\overline{INT}$ disabled
DI instruction execution	0	0	Maskable interrupt $\overline{INT}$ disabled
EI instruction execution	1	1	Maskable interrupt $\overline{INT}$ enabled
LD A,I instruction execution	•	•	IFF_2 → Parity flag
LD A,R instruction execution	•	•	IFF_2 → Parity flag
Accept $\overline{NMI}$	0	•	Maskable interrupt $\overline{INT}$ disabled
RETN instruction execution	IFF_2	•	IFF_2 → IFF_1 at completion of an $\overline{NMI}$ service routine.

INSTRUCTION SET

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

The following is a summary of the instruction set which shows the assembly language mnemonic, the operation, the flag status, and gives comments on each instruction. For an explanation of flag notations and symbols for mnemonic tables, see the Symbolic Notations section which follows these tables. The *Z80 CPU Technical Manual* (03-0029-01), the *Programmer's Reference Guide* (03-0012-03), and *Assembly Language Programming Manual* (03-0002-01) contain significantly more details for programming use.

The instructions are divided into the following categories:

□ 8-bit loads

□ 16-bit loads

□ Exchanges, block transfers, and searches

□ 8-bit arithmetic and logic operations

□ General-purpose arithmetic and CPU control

□ 16-bit arithmetic operations

□ Rotates and shifts

□ Bit set, reset, and test operations

□ Jumps

□ Calls, returns, and restarts

□ Input and output operations

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

□ Immediate

□ Immediate extended

□ Modified page zero

□ Relative

□ Extended

□ Indexed

□ Register

□ Register indirect

□ Implied

□ Bit

8-BIT LOAD GROUP (Continued)

	Symbolic				Flags						Opcode			No. of	No. of M	No. of T	
Mnemonic	Operation	S	Z		H		P/V	N	C	76	543	210	Hex	Bytes	Cycles	States	Comments
LD (IY+d), n	(IY+d) ← n	•	•	X	•	X	•	•	•	11	111	101	FD	4	5	19	
										00	110	110	36				
											←d→						
											←n→						
LD A, (BC)	A ← (BC)	•	•	X	•	X	•	•	•	00	001	010	0A	1	2	7	
LD A, (DE)	A ← (DE)	•	•	X	•	X	•	•	•	00	011	010	1A	1	2	7	
LD A, (nn)	A ← (nn)	•	•	X	•	X	•	•	•	00	111	010	3A	3	4	13	
											←n→						
											←n→						
LD (BC), A	(BC) ← A	•	•	X	•	X	•	•	•	00	000	010	02	1	2	7	
LD (DE), A	(DE) ← A	•	•	X	•	X	•	•	•	00	010	010	12	1	2	7	
LD (nn), A	(nn) ← A	•	•	X	•	X	•	•	•	00	110	010	32	3	4	13	
											←n→						
											←n→						
LD A, I	A ← I	↕	↕	X	0	X	IFF	0	•	11	101	101	ED	2	2	9	
										01	010	111	57				
LD A, R	A ← R	↕	↕	X	0	X	IFF	0	•	11	101	101	ED	2	2	9	
										01	011	111	5F				
LD I, A	I ← A	•	•	X	•	X	•	•	•	11	101	101	ED	2	2	9	
										01	000	111	47				
LD R, A	R ← A	•	•	X	•	X	•	•	•	11	101	101	ED	2	2	9	
										01	001	111	4F				

NOTE: IFF, the content of the interrupt enable flip-flop, (IFF_2), is copied into the P/V flag.

16-BIT LOAD GROUP

	Symbolic				Flags						Opcode			No. of	No. of M	No. of T		
Mnemonic	Operation	S	Z		H		P/V	N	C	76	543	210	Hex	Bytes	Cycles	States	Comments	
LD dd, nn	dd ← nn	•	•	X	•	X	•	•	•	00	dd0	001		3	3	10	dd	Pair
											←n→						00	BC
											←n→						01	DE
LD IX, nn	IX ← nn	•	•	X	•	X	•	•	•	11	011	101	DD	4	4	14	10	HL
										00	100	001	21				11	SP
											←n→							
											←n→							
LD IY, nn	IY ← nn	•	•	X	•	X	•	•	•	11	111	101	FD	4	4	14		
										00	100	001	21					
											←n→							
											←n→							
LD HL, (nn)	H ← (nn+1) L ← (nn)	•	•	X	•	X	•	•	•	00	101	010	2A	3	5	16		
											←n→							
											←n→							
LD dd, (nn)	dd_H ← (nn+1) dd_L ← (nn)	•	•	X	•	X	•	•	•	11	101	101	ED	4	6	20		
										01	dd1	011						
											←n→							
											←n→							

NOTE: $(PAIR)_H$, $(PAIR)_L$ refer to high order and low order eight bits of the register pair respectively. e.g., BC_L = C, AF_H = A.

EXCHANGE, BLOCK TRANSFER, BLOCK SEARCH GROUPS

Mnemonic	Symbolic Operation	S	Z	Flags	H		P/V	N	C	Opcode 76	543	210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
EX DE, HL	DE ↔ HL	•	•	X	•	X	•	•	•	11	101	011	EB	1	1	4	
EX AF, AF′	AF ↔ AF′	•	•	X	•	X	•	•	•	00	001	000	08	1	1	4	
EXX	BC ↔ BC′	•	•	X	•	X	•	•	•	11	011	001	D9	1	1	4	Register bank and auxiliary register bank exchange
	DE ↔ DE′																
	HL ↔ HL′																
EX (SP), HL	H ↔ (SP + 1)	•	•	X	•	X	•	•	•	11	100	011	E3	1	5	19	
	L ↔ (SP)																
EX (SP), IX	IX_H ↔ (SP + 1)	•	•	X	•	X	•	•	•	11	011	101	DD	2	6	23	
	IX_L ↔ (SP)									11	100	011	E3				
EX (SP), IY	IY_H ↔ (SP + 1)	•	•	X	•	X	•	•	•	11	111	101	FD	2	6	23	
	IYL ↔ (SP)									11	100	011	E3				
LDI	(DE) ← (HL)	•	•	X	0	X	[1] ↕	0	•	11	101	101	ED	2	4	16	Load (HL) into (DE), increment the pointers and decrement the byte counter (BC)
	DE ← DE + 1									10	100	000	A0				
	HL ← HL + 1																
	BC ← BC − 1																
LDIR	(DE) ← (HL)	•	•	X	0	X	[2] 0	0	•	11	101	101	ED	2	5	21	If BC ≠ 0
	DE ← DE + 1									10	110	000	B0	2	4	16	If BC = 0
	HL ← HL + 1																
	BC ← BC − 1																
	Repeat until BC = 0																
LDD	(DE) ← (HL)	•	•	X	0	X	[1] ↕	0	•	11	101	101	ED	2	4	16	
	DE ← DE − 1									10	101	000	A8				
	HL ← HL − 1																
	BC ← BC − 1																
LDDR	(DE) ← (HL)	•	•	X	0	X	[2] 0	0	•	11	101	101	ED	2	5	21	If BC ≠ 0
	DE ← DE − 1									10	111	000	B8	2	4	16	If BC = 0
	HL ← HL − 1																
	BC ← BC − 1																
	Repeat until BC = 0																
CPI	A − (HL)	↕	[3] ↕	X	↕	X	[1] ↕	1	•	11	101	101	ED	2	4	16	
	HL ← HL + 1									10	100	001	A1				
	BC ← BC − 1																

NOTE: [1] P/V flag is 0 if the result of BC − 1 = 0, otherwise P/V = 1.
[2] P/V flag is 0 only at completion of instruction.
[3] Z flag is 1 if A = HL , otherwise Z = 0.

EXCHANGE, BLOCK TRANSFER, BLOCK SEARCH GROUPS (Continued)

	Symbolic			Flags						Opcode				No. of	No. of M	No. of T	
Mnemonic	**Operation**	**S**	**Z**		**H**		**P/V**	**N**	**C**	**76**	**543**	**210**	**Hex**	**Bytes**	**Cycles**	**States**	**Comments**
			③				①										
CPIR	A − (HL)	↕	↕	X	↕	X	↕	1	•	11	101	101	ED	2	5	21	If BC ≠ 0 and A ≠ (HL)
	HL ← HL + 1 BC ← BC − 1 Repeat until A = (HL) or BC = 0									10	110	001	B1	2	4	16	If BC = 0 or A = (HL)
			③				①										
CPD	A − (HL) HL ← HL − 1 BC ← BC − 1	↕	↕	X	↕	X	↕	1	•	11 10	101 101	101 001	ED A9	2	4	16	
			③				①										
CPDR	A − (HL)	↕	↕	X	↕	X	↕	1	•	11	101	101	ED	2	5	21	If BC ≠ 0 and A ≠ (HL)
	HL ← HL − 1 BC ← BC − 1 Repeat until A = (HL) or BC = 0									10	111	001	B9	2	4	16	If BC = 0 or A = (HL)

NOTE: ① P/V flag is 0 if the result of BC − 1 = 0, otherwise P/V = 1.
② P/V flag is 0 only at completion of instruction.
③ Z flag is 1 if A = (HL), otherwise Z = 0.

8-BIT ARITHMETIC AND LOGICAL GROUP

	Symbolic			Flags						Opcode				No. of	No. of M	No. of T	
Mnemonic	**Operation**	**S**	**Z**		**H**		**P/V**	**N**	**C**	**76**	**543**	**210**	**Hex**	**Bytes**	**Cycles**	**States**	**Comments**
ADD A, r	A ← A + r	↕	↕	X	↕	X	V	0	↕	10	$\boxed{000}$	r		1	1	4	r Reg.
ADD A, n	A ← A + n	↕	↕	X	↕	X	V	0	↕	11	$\boxed{000}$	110		2	2	7	000 B
											← n →						001 C
																	010 D
ADD A, (HL)	A ← A + (HL)	↕	↕	X	↕	X	V	0	↕	10	$\boxed{000}$	110		1	2	7	011 E
ADD A, (IX + d)	A ← A + (IX + d)	↕	↕	X	↕	X	V	0	↕	11	011	101	DD	3	5	19	100 H
										10	$\boxed{000}$	110					101 L
											← d →						111 A
ADD A, (IY + d)	A ← A + (IY + d)	↕	↕	X	↕	X	V	0	↕	11	111	101	FD	3	5	19	
										10	$\boxed{000}$	110					
											← d →						
ADC A, s	A ← A + s + CY	↕	↕	X	↕	X	V	0	↕		$\boxed{001}$						s is any of r, n,
SUB s	A ← A − s	↕	↕	X	↕	X	V	1	↕		$\boxed{010}$						(HL), (IX + d),
SBC A, s	A ← A − s − CY	↕	↕	X	↕	X	V	1	↕		$\boxed{011}$						(IY + d) as
AND s	A ← A > s	↕	↕	X	1	X	P	0	0		$\boxed{100}$						shown for ADD
OR s	A ← A > s	↕	↕	X	0	X	P	0	0		$\boxed{110}$						instruction. The
XOR s	A ← A⊕s	↕	↕	X	0	X	P	0	0		$\boxed{101}$						indicated bits
CP s	A − s	↕	↕	X	↕	X	V	1	↕		$\boxed{111}$						replace the
																	$\boxed{000}$ in the
																	ADD set above.

BIT SET, RESET AND TEST GROUP

Mnemonic	Symbolic Operation	S	Z		Flags H		P/V	N	C	Opcode 76	543	210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
BIT b, r	$Z \leftarrow r_b$	X	↕	X	1	X	X	0	•	11	001	011	CB	2	2	8	r Reg.
										01	b	r					000 B
BIT b, (HL)	$Z \leftarrow (HL)_b$	X	↕	X	1	X	X	0	•	11	001	011	CB	2	3	12	001 C
										01	b	110					010 D
BIT b,$(IX+d)_b$	$Z \leftarrow (IX+d)_b$	X	↕	X	1	X	X	0	•	11	011	101	DD	4	5	20	011 E
										11	001	011	CB				100 H
											←d→						101 L
										01	b	110					111 A
																	b Bit Tested
BIT b, $(IY+d)_b$	$Z \leftarrow (IY+d)_b$	X	↕	X	1	X	X	0	•	11	111	101	FD	4	5	20	000 0
										11	001	011	CB				001 1
											←d→						010 2
										01	b	110					011 3
SET b, r	$r_b \leftarrow 1$	•	•	X	•	X	•	•	•	11	001	011	CB	2	2	8	100 4
										$\boxed{11}$	b	r					101 5
SET b, (HL)	$(HL)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11	001	011	CB	2	4	15	110 6
										$\boxed{11}$	b	110					111 7
SET b, (IX+d)	$(IX+d)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11	011	101	DD	4	6	23	
										11	001	011	CB				
											←d→						
										$\boxed{11}$	b	110					
SET b, (IY+d)	$(IY+d)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11	111	101	FD	4	6	23	
										11	001	011	CB				
											←d→						
										$\boxed{11}$	b	110					
RES b, m	$m_b \leftarrow 0$ $m \equiv r$, (HL), (IX+d), (IY+d)	•	•	X	•	X	•	•	•	$\boxed{10}$							To form new opcode replace $\boxed{11}$ of SET b, s with $\boxed{10}$. Flags and time states for SET instruction.

NOTE: The notation m_b indicates location m, bit b (0 to 7).

CALL AND RETURN GROUP

	Symbolic			Flags						Opcode				No. of	No. of M	No. of T	
Mnemonic	Operation	S	Z		H		P/V	N	C	76	543	210	Hex	Bytes	Cycles	States	Comments
CALL nn	$(SP-1) \leftarrow PC_H$	•	•	X	•	X	•	•	•	11	001	101	CD	3	5	17	
	$(SP-2) \leftarrow PC_L$										← n →						
	$PC \leftarrow nn$,										← n →						
CALL cc,nn	If condition	•	•	X	•	X	•	•	•	11	cc	100		3	3	10	If cc is false.
	cc is false										← n →						
	continue,										← n →			3	5	17	If cc is true.
	otherwise																
	same as																
	CALL nn																
RET	$PC_L \leftarrow (SP)$	•	•	X	•	X	•	•	•	11	001	001	C9	1	3	10	
	$PC_H \leftarrow (SP+1)$																
RET cc	If condition	•	•	X	•	X	•	•	•	11	cc	000		1	1	5	If cc is false.
	cc is false																
	continue,													/1	3	11	If cc is true.
	otherwise																
	same as RET																cc Condition
																	000 NZ (non-zero)
																	001 Z (zero)
																	010 NC (non-carry)
RETI	Return from	•	•	X	•	X	•	•	•	11	101	101	ED	2	4	14	011 C (carry)
	interrupt									01	001	101	4D				100 PO (parity odd)
RETN[1]	Return from	•	•	X	•	X	•	•	•	11	101	101	ED	2	4	14	101 PE (parity even)
	non-maskable									01	000	101	45				110 P (sign positive)
	interrupt																111 M (sign negative)
RST p	$(SP-1) \leftarrow PC_H$	•	•	X	•	X	•	•	•	11	t	111		1	3	11	t p
	$(SP-2) \leftarrow PC_L$																000 00H
	$PC_H \leftarrow 0$																001 08H
	$PC_L \leftarrow p$																010 10H
																	011 18H
																	100 20H
																	101 28H
																	110 30H
																	111 38H

NOTE: [1]RETN loads $IFF_2 \rightarrow IFF_1$

INPUT AND OUTPUT GROUP

	Symbolic			Flags							Opcode			No. of	No. of M	No. of T	
Mnemonic	Operation	S	Z		H		P/V	N	C	76	543	210	Hex	Bytes	Cycles	States	Comments
IN A, (n)	A ← (n)	•	•	X	•	X	•	•	•	11	011	01	DB	2	3	11	n to $A_0 \sim A_7$
											← n →						Acc. to $A_8 \sim A_{15}$
IN r, (C)	r ← (C)	↕	↕	X	↕	X	P	0	•	11	101	101	ED	2	3	12	C to $A_0 \sim A_7$
	if r = 110 only									01	r	000					B to $A_8 \sim A_{15}$
	the flags will																
	be affected																
			①														
INI	(HL) ← (C)	X	↕	X	X	X	X	1	X	11	101	101	ED	2	4	16	C to $A_0 \sim A_7$
	B ← B − 1									10	100	010	A2				B to $A_8 \sim A_{15}$
	HL ← HL + 1		②														
INIR	(HL) ← (C)	X	1	X	X	X	X	1	X	11	101	101	ED	2	5	21	C to $A_0 \sim A_7$
	B ← B − 1									10	110	010	B2		(If B≠0)		B to $A_8 \sim A_{15}$
	HL ← HL + 1													2	4	16	
	Repeat until														(If B = 0)		
	B = 0																
			①														
IND	(HL) ← (C)	X	↕	X	X	X	X	1	X	11	101	101	ED	2	4	16	C to $A_0 \sim A_7$
	B ← B − 1									10	101	010	AA				B to $A_8 \sim A_{15}$
	HL ← HL − 1		②														
INDR	(HL) ← (C)	X	1	X	X	X	X	1	X	11	101	101	ED	2	5	21	C to $A_0 \sim A_7$
	B ← B − 1									10	111	010	BA		(If B≠0)		B to $A_8 \sim A_{15}$
	HL ← HL − 1													2	4	16	
	Repeat until														(If B = 0)		
	B = 0																
OUT (n), A	(n) ← A	•	•	X	•	X	•	•	•	11	010	011	D3	2	3	11	n to $A_0 \sim A_7$
											← n →						Acc. to $A_8 \sim A_{15}$
OUT (C), r	(C) ← r	•	•	X	•	X	•	•	•	11	101	101	ED	2	3	12	C to $A_0 \sim A_7$
										01	r	001					B to $A_8 \sim A_{15}$
			①														
OUTI	(C) ← (HL)	X	↕	X	X	X	X	1	X	11	101	101	ED	2	4	16	C to $A_0 \sim A_7$
	B ← B − 1									10	100	011	A3				B to $A_8 \sim A_{15}$
	HL ← HL + 1		②														
OTIR	(C) ← (HL)	X	1	X	X	X	X	1	X	11	101	101	ED	2	5	21	C to $A_0 \sim A_7$
	B ← B − 1									10	110	011	B3		(If B≠0)		B to $A_8 \sim A_{15}$
	HL ← HL + 1													2	4	16	
	Repeat until														(If B = 0)		
	B = 0																
			①														
OUTD	(C) ← (HL)	X	↕	X	X	X	X	1	X	11	101	101	ED	2	4	16	C to $A_0 \sim A_7$
	B ← B − 1									10	101	011	AB				B to $A_8 \sim A_{15}$
	HL ← HL − 1																
			②														
OTDR	(C) ← (HL)	X	1	X	X	X	X	1	X	11	101	101	ED	2	5	21	C to $A_0 \sim A_7$
	B ← B − 1									10	111	011			(If B≠0)		B to $A_8 \sim A_{15}$
	HL ← HL − 1													2	4	16	
	Repeat until														(If B = 0)		
	B = 0																

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

SUMMARY OF FLAG OPERATION

	D_7							D_0	
Instructions	**S**	**Z**		**H**		**P/V**	**N**	**C**	**Comments**
ADD A, s; ADC A, s	↕	↕	X	↕	X	V	0	↕	8-bit add or add with carry.
SUB s; SBC A, s; CP s; NEG	↕	↕	X	↕	X	V	1	↕	8-bit subtract, subtract with carry, compare and negate accumulator.
AND s	↕	↕	X	1	X	P	0	0	Logical operation.
OR s, XOR s	↕	↕	X	0	X	P	0	0	Logical operation.
INC s	↕	↕•	X	↕	X	V	0	•	8-bit increment.
DEC s	↕	↕	X	↕	X	V	1	•	8-bit decrement.
ADD DD, ss	•	•	X	X	X	•	0	↕	16-bit add.
ADC HL, ss	↕	↕	X	X	X	V	0	↕	16-bit add with carry.
SBC HL, ss	↕	↕	X	X	X	V	1	↕	16-bit subtract with carry.
RLA; RLCA; RRA; RRCA	•	•	X	0	X	•	0	↕	Rotate accumulator.
RL m; RLC m; RR m; RRC m; SLA m; SRA m; SRL m	↕	↕	X	0	X	P	0	↕	Rotate and shift locations.
RLD; RRD	↕	↕	X	0	X	P	0	•	Rotate digit left and right.
DAA	↕	↕	X	↕	X	P	•	↕	Decimal adjust accumulator.
CPL	•	•	X	1	X	•	1	•	Complement accumulator.
SCF	•	•	X	0	X	•	0	1	Set carry.
CCF	•	•	X	X	X	•	0	↕	Complement carry.
IN r (C)	↕	↕	X	0	X	P	0	•	Input register indirect.
INI; IND; OUTI; OUTD	X	↕	X	X	X	X	1	•	Block input and output. Z = 1 if B ≠ 0, otherwise Z = 0.
INIR; INDR; OTIR; OTDR	X	1	X	X	X	X	1	•	Block input and output. Z = 1 if B ≠ 0, otherwise Z = 0.
LDI; LDD	X	X	X	0	X	↕	0	•	Block transfer instructions. P/V = 1 if BC ≠ 0, otherwise P/V = 0.
LDIR; LDDR	X	X	X	0	X	0	0	•	Block transfer instructions. P/V = 1 if BC ≠ 0, otherwise P/V = 0.
CPI; CPIR; CPD; CPDR	X	↕	X	X	X	↕	1	•	Block search instructions. Z = 1 if A = (HL), otherwise Z = 0. P/V = 1 if BC ≠ 0, otherwise P/V = 0.
LD A; I, LD A, R	↕	↕	X	0	X	IFF	0	•	IFF, the content of the interrupt enable flip-flop, (IFF_2), is copied into the P/V flag.
BIT b, s	X	↕	X	1	X	X	0	•	The state of bit b of location s is copied into the Z flag.

SYMBOLIC NOTATION

Symbol **Operation**

S Sign flag. S = 1 if the MSB of the result is 1.

Z Zero flag. Z = 1 if the result of the operation is 0.

P/V Parity or overflow flag. Parity (P) and overflow (V) share the same flag. Logical operations affect this flag with the parity of the result while arithmetic operations affect this flag with the overflow of the result. If P/V holds parity: P/V = 1 if the result of the operation is even; P/V = 0 if result is odd. If P/V holds overflow, P/V = 1 if the result of the operation produced an overflow. If P/V does not hold overflow, P/V = 0.

H* Half-carry flag. H = 1 if the add or subtract operation produced a carry into, or borrow from, bit 4 of the accumulator.

N* Add/Subtract flag. N = 1 if the previous operation was a subtract.

C Carry/Link flag. C = 1 if the operation produced a carry from the MSB of the operand or result.

Symbol **Operation**

↕ The flag is affected according to the result of the operation.

• The flag is unchanged by the operation.

0 The flag is reset by the operation.

1 The flag is set by the operation.

X The flag is indeterminate.

V P/V flag affected according to the overflow result of the operation.

P P/V flag affected according to the parity result of the operation.

r Any one o the CPU registers A, B, C, D, E, H, L.

s Any 8-bit location for all the addressing modes allowed for the particular instruction.

ss Any 16-bit location for all the addressing modes allowed for that instruction.

ii Any one of the two index registers IX or IY.

R Refresh counter.

n 8-bit value in range < 0, 255 >.

nn 16-bit value in range < 0, 65535 >.

*H and N flags are used in conjunction with the decimal adjust instruction (DAA) to properly correct the result into packed BCD format following addition or subtraction using operands with packed BCD format.

PIN DESCRIPTIONS

A$_0$-A$_{15}$. *Address Bus* (output, active High, 3-state). A$_0$-A$_{15}$ form a 16-bit address bus. The Address Bus provides the address for memory data bus exchanges (up to 64K bytes) and for I/O device exchanges.

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

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

D$_0$-D$_7$. *Data Bus* (input/output, active High, 3-state). D$_0$-D$_7$ constitute an 8-bit bidirectional data bus, used for data exchanges with memory and I/O.

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

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

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

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

$\overline{\text{MREQ}}$. *Memory Request* (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.

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

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

$\overline{\text{RESET}}$. *Reset* (input, active Low). $\overline{\text{RESET}}$ initializes the CPU as follows: it resets the interrupt enable flip-flop, clears the PC and Registers I and R, and sets the interrupt status to Mode 0. During reset time, the address and data bus go to a high-impedance state, and all control output signals go to the inactive state. Note that $\overline{\text{RESET}}$ must be active for a minimum of three full clock cycles before the reset operation is complete.

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

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

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

CPU TIMING

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

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

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

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

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

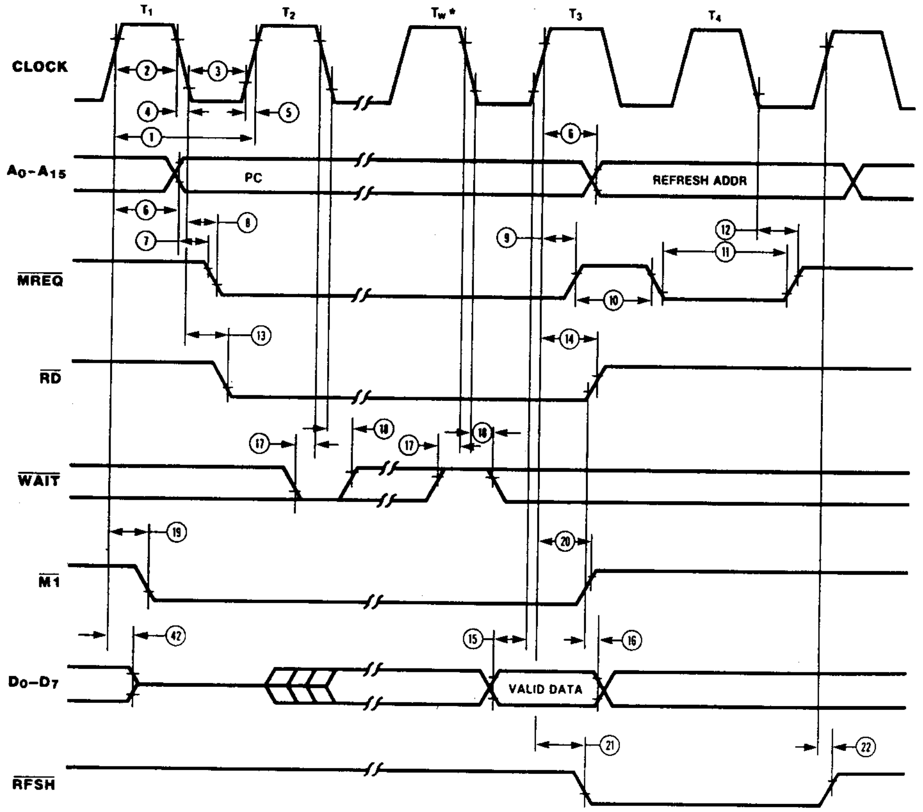
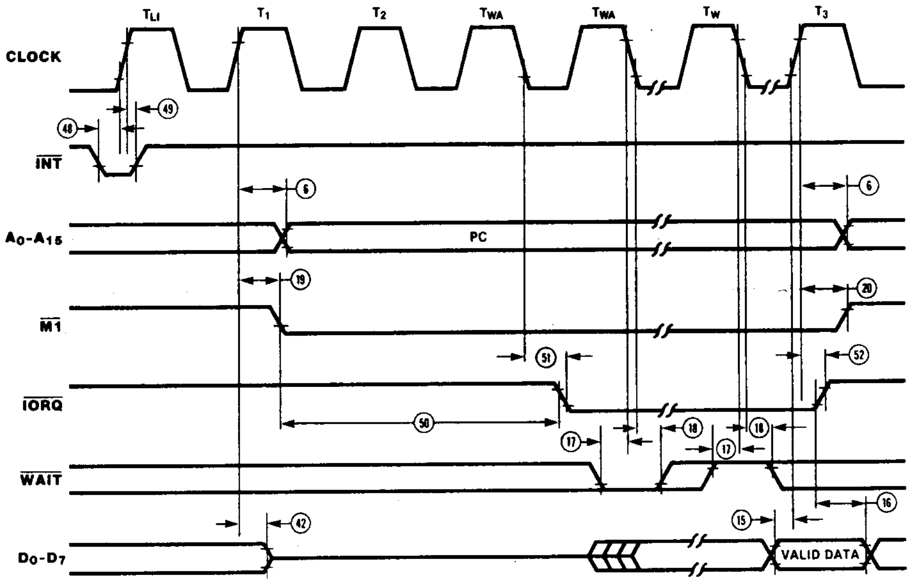
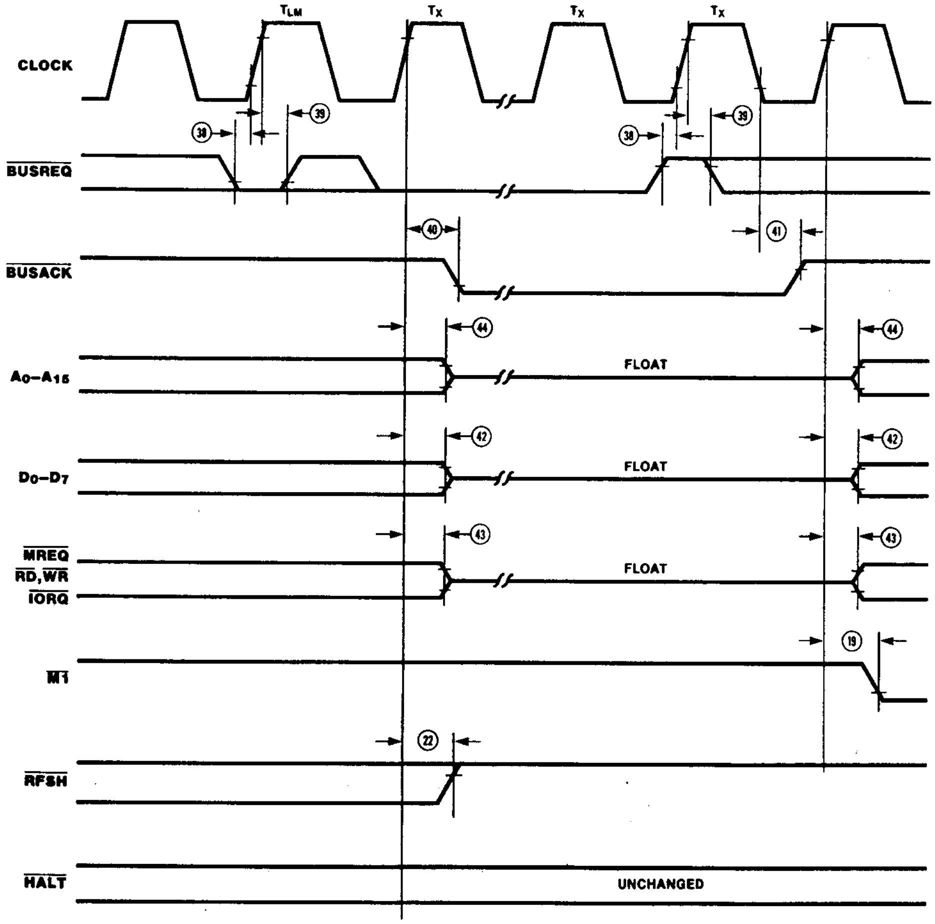


Figure 5. Instruction Opcode Fetch

Interrupt Request/Acknowledge Cycle. The CPU samples the interrupt signal with the rising edge of the last clock cycle at the end of any instruction (Figure 8). When an interrupt is accepted, a special $\overline{M1}$ cycle is generated. During this $\overline{M1}$ cycle, $\overline{IORQ}$ becomes active (instead of $\overline{MREQ}$) to indicate that the interrupting device can place an 8-bit vector on the data bus. The CPU automatically adds two Wait states to this cycle.



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

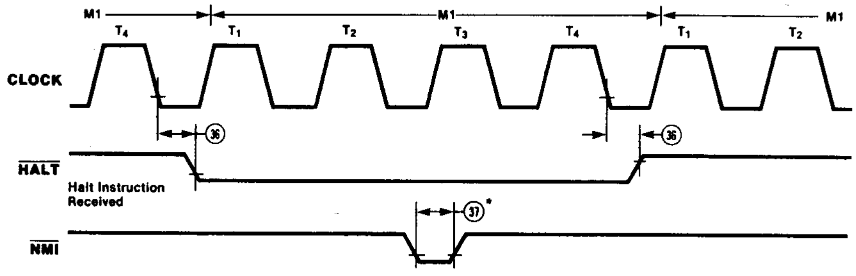


T_{LM}
T_X
T_X
T_X
CLOCK
38
39
38
39
$\overline{\mathrm{BUSREQ}}$
40
41
$\overline{\mathrm{BUSACK}}$
44
44
A_0–A_{15}
FLOAT
42
42
D_0–D_7
FLOAT
43
43
$\overline{\mathrm{MREQ}}$
$\overline{\mathrm{RD}}$,$\overline{\mathrm{WR}}$
$\overline{\mathrm{IORQ}}$
FLOAT
19
$\overline{\mathrm{M1}}$
22
$\overline{\mathrm{RFSH}}$
$\overline{\mathrm{HALT}}$
UNCHANGED

NOTES: 1) T_{LM} = Last state of any M cycle.
2) T_X = An arbitrary clock cycle used by requesting device.

Figure 10. BUS Request/Acknowledge Cycle

Halt Acknowledge Cycle. When the CPU receives a HALT instruction, it executes NOP states until either an $\overline{\text{INT}}$ or $\overline{\text{NMI}}$ input is received. When in the Halt state, the $\overline{\text{HALT}}$ output is active and remains so until an interrupt is received (Figure 11). $\overline{\text{INT}}$ will also force a Halt exit.



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

Figure 11. Halt Acknowledge

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

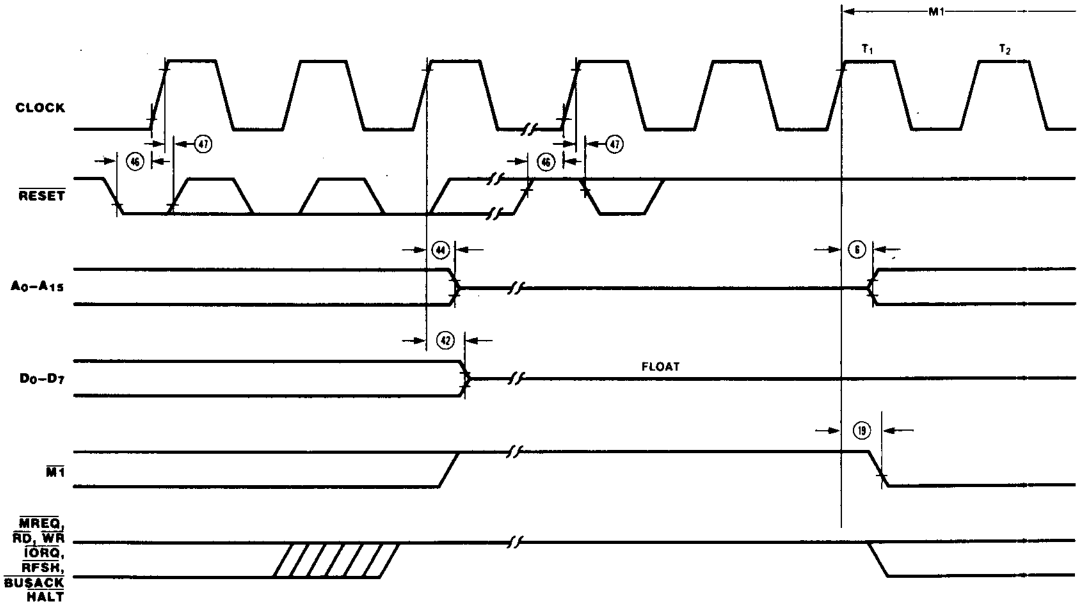

M1
T1
T2
CLOCK
47
46
46
47
$\overline{\text{RESET}}$
44
6
A_0–A_{15}
42
D_0–D_7
FLOAT
19
$\overline{\text{M1}}$
$\overline{\text{MREQ}}$,
$\overline{\text{RD}}$, $\overline{\text{WR}}$
$\overline{\text{IORQ}}$,
$\overline{\text{RFSH}}$,
$\overline{\text{BUSACK}}$
$\overline{\text{HALT}}$

Figure 12. Reset Cycle

Power-Down mode of operation (Only applies to CMOS Z80 CPU).

CMOS Z80 CPU supports Power-Down mode of operation. This mode is also referred to as the "standby mode", and supply current for the CPU goes down as low as 10 uA (Where specified as Icc_2).

Power-Down Acknowledge Cycle. When the clock input to the CPU is stopped at either a High or Low level, the CPU stops its operation and maintains all registers and control signals. However. I_{CC2} (standby supply current) is guaranteed only when the system clock is stopped at a Low level during T_4 of the machine cycle following the execution of the HALT instruction. The timing diagram for the power-down function, when implemented with the HALT **instruction, is shown in Figure 13.**

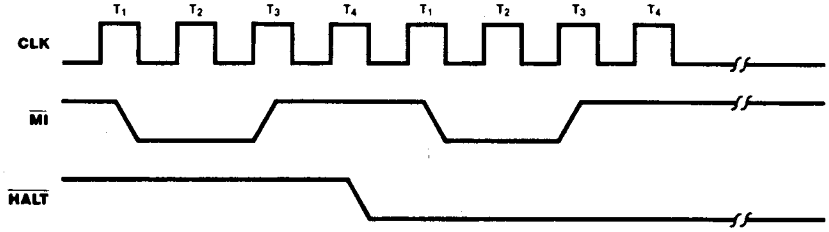


Figure 13. Power-Down Acknowledge

Power-Down Release Cycle. The system clock must be supplied to the CPU to release the power-down state. When the system clock is supplied to the CLK input, the CPU restarts operations from the point at which the power-down state was implemented.

The timing diagrams for the release from power-down mode **are shown in Figure 14.**

NOTES:.

1) When the external oscillator has been stopped to enter the power-down state, some warm-up time may be required to obtain a stable clock for the release.
2) When the HALT instruction is executed to enter the power-down state, the CPU will also enter the Halt state. An interrupt signal (either $\overline{\text{NMI}}$ or $\overline{\text{INT}}$) or a $\overline{\text{RESET}}$ signal must be applied to the CPU after the system clock is supplied in order to release the power-down state.

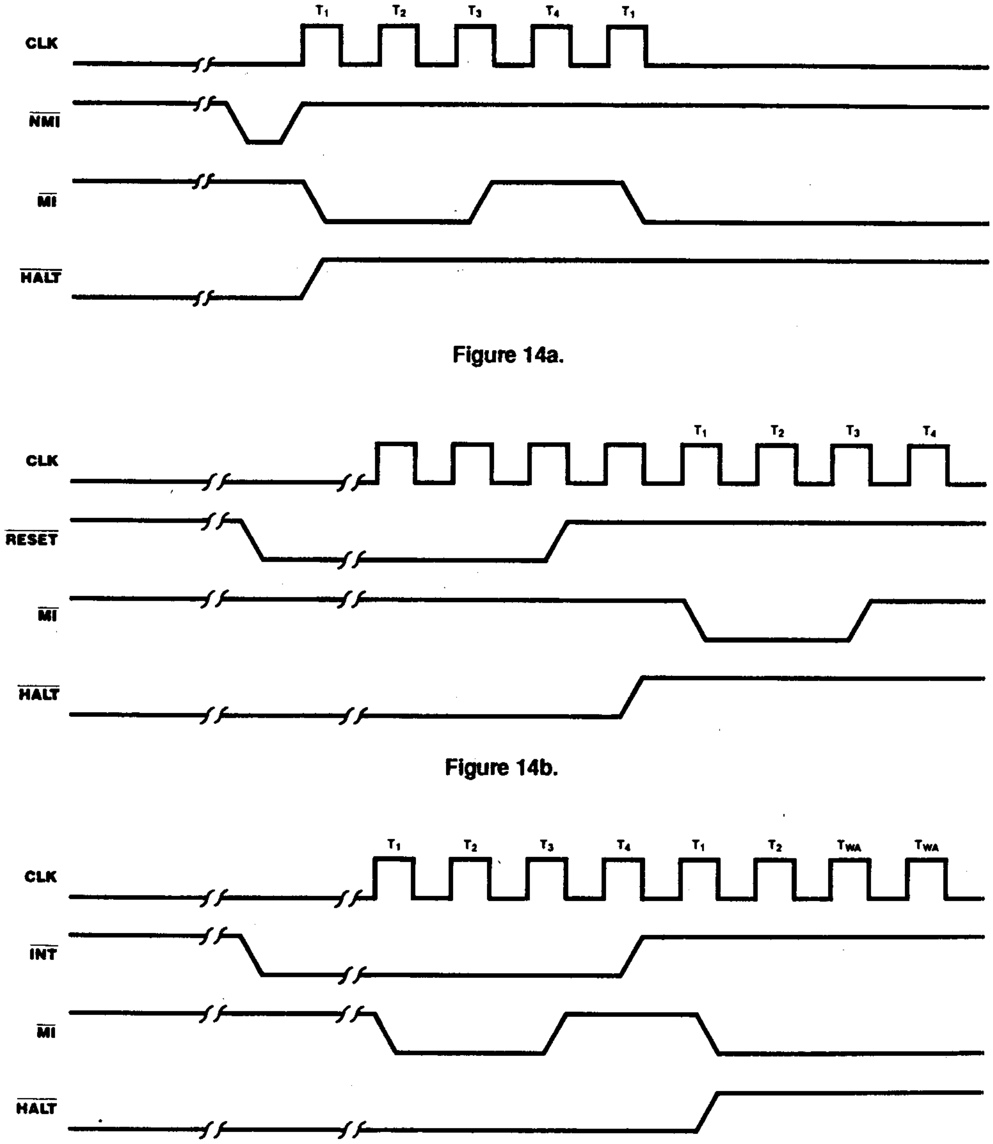


Figure 14a.

Figure 14b.

Figure 14c.

Figure 13. Power-Down Release

DC CHARACTERISTICS (Z84C00/CMOS Z80 CPU)

Symbol	Parameter	Min	Max	Unit	Condition
V_{ILC}	Clock Input Low Voltage	-0.3	0.45	V	
V_{IHC}	Clock Input High Voltage	$V_{CC}-.6$	$V_{CC}+.3$	V	
V_{IL}	Input Low Voltage	-0.3	0.8	V	
V_{IH}	Input High Voltage	2.2	V_{CC}	V	
V_{OL}	Output Low Voltage		0.4	V	$I_{OL} = 2.0$ mA
V_{OH_1}	Output High Voltage	2.4		V	$I_{OH} = -1.6$ mA
V_{OH_2}	Output High Voltage	$V_{CC}-0.8$		V	$I_{OH} = -250\,\mu A$
I_{CC_1}	Power Supply Current **4 MHz**		**20**	**mA**	$V_{CC} = 5V$
	6 MHz		**30**	**mA**	$V_{IH} = V_{CC} - 0.2V$
	8 MHz		**40**	**mA**	$V_{IL} = 0.2V$
	10 MHz		**50**	**mA**	
	20 MHz		100	mA	$V_{cc} = 5V$
I_{CC_2}	Standby Supply Current		10	μA	$V_{CC} = 5V$ $\overline{CLK} = (0)$ $V_{IH} = V_{CC} - 0.2V$ $V_{IL} = 0.2V$
I_{LI}	Input Leakage Current	-10	10	μA	$V_{IN} = 0.4$ to V_{CC}
I_{LO}	3-State Output Leakage Current in Float	-10	10^2	μA	$V_{OUT} = 0.4$ to V_{CC}

1. Measurements made with outputs floating.
2. A_{15}-A_0, D_7-D_0, $\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$, and $\overline{WR}$.
3. I_{CC_2} standby supply current is guaranteed only when the supplied clock is stopped at a low level during T_4 of the machine cycle immediately following the execution of a HALT instruction.

CAPACITANCE

Symbol	Parameter	Min	Max	Unit
C_{CLOCK}	Clock Capacitance		**10**	pf
C_{IN}	Input Capacitance		5	pf
C_{OUT}	Output Capacitance		15	pf

$T_A = 25°C$, f = 1 MHz.
Unmeasured pins returned to ground.

AC CHARACTERISTICS† (Z84C00/CMOS Z80 CPU)

V_{CC}=5.0V ± 10%, unless otherwise specified

No	Symbol	Parameter	Z84C0004**		Z84C0006		Z84C0008		Z84C0010		Z84C0020[1]		Unit	Note
			Min	Max	Min	Max	Min	Max	Min	Max	Min	Max		
1	TcC	Clock Cycle time	250*	DC	162*	DC	125*	DC	100*	DC	50*	DC	nS	
2	TwCh	Clock Pulse width (high)	110	DC	65	DC	55	DC	40	DC	20	DC	nS	
3	TwCl	Clock Pulse width (low)	110	DC	65	DC	55	DC	40	DC	20	DC	nS	
4	TfC	Clock Fall time		30		20		10		10		10	nS	
5	TrC	Clock Rise time		30		20		10		10		10	nS	
6	TdCr(A)	Address vaild from Clock Rise		110		90		80		65		57	nS	[2]
7	TdA(MREQf)	Address valid to /MREQ Fall	65*		35*		20*		5*		-15*		nS	
8	TdCf(MREQf)	Clock Fall to /MREQ Fall delay		85		70		60		55		40	nS	
9	TdCr(MREQr)	Clock Rise to /MREQ Rise delay		85		70		60		55		40	nS	
10	TwMREQh	/MREQ pulse width (High)	110*		65*		45**		30*		10*		nS	[3]
11	TwMREQl	/MREQ pulse width (low)	220*		132*		100*		75*		25*		nS	[3]
12	TdCf(MERQr)	Clock Fall to /MREQ Rise delay		85		70		60		55		40	nS	
13	TdCf(RDf)	Clock Fall to /RD Fall delay		95		80		70		65		40	nS	
14	TdCr(RDr)	Clock Rise to /RD Rise delay		85		70		60		55		40	nS	
15	TsD(Cr)	Data setup time to Clock Rise	35		30		30		25		12		nS	
16	ThD(RDr)	Data hold time after /RD Rise	0		0		0		0		0		nS	
17	TsWAIT(Cf)	/WAIT setup time to Clock Fall	70		60		50		20		7.5		nS	
18	ThWAIT(Cf)	/WAIT hold time after Clock Fall	10		10		10		10		10		nS	
19	TdCr(M1f)	Clock Rise to /M1 Fall delay		100		80		70		65		45	nS	
20	TdCr(M1r)	Clock Rise to /M1 Rise delay		100		80		70		65		45	nS	
21	TdCr(RFSHf)	Clock Rise to /RFSH Fall delay		130		110		95		80		60	nS	
22	TdCr(RFSHr)	Clock Rise to /RFSH Rise delay		120		100		85		80		60	nS	
23	TdCf(RDr)	Clock Fall to /RD Rise delay		85		70		60		55		40	nS	
24	TdCr(RDf)	Clock Rise to /RD Fall delay		85		70		60		55		40	nS	
25	TsD(Cf)	Data setup to Clock Fall during M2, M3, M4 or M5 cycles	50		40		30		25		12		nS	
26	TdA(IORQf)	Address stable prior to /IORQ Fall	180*		107*		75*		50*		0*		nS	
27	TdCr(IORQf)	Clock Rise to /IORQ Fall delay		75		65		55		50		40	nS	
28	TdCf(IORQr)	Clock Fall to /IORQ Rise delay		85		70		60		55		40	nS	
29	TdD(WRf)Mw	Data stable prior to /WR Fall	80*		22*		5*		40*		-10*		nS	
30	TdCf(WRf)	Clock Fall to /WR Fall delay		80		70		60		55		40	nS	
31	TwWR	/WR pulse width	220*		132*		100*		75*		25*		nS	
32	TdCf(WRr)	Clock Fall to /WR Rise delay		80		70		60		55		40	nS	
33	TdD(WRf)IO	Data stable prior to /WR Fall	-10*		-55*		-55*		-10*		-30*		nS	
34	TdCr(WRf)	Clock Rise to /WR Fall delay		65		60		60		50		40	nS	
35	TdWRr(D)	Data stable from /WR Rise	60*		30*		15*		10*		0*		nS	
36	TdCf(HALT)	Clock Fall to /HALT 'L' or 'H'		300		260		225		90		70	nS	
37	TwNMI	/NMI pulse width	80		60		60		60		60		nS	
38	TsBUSREQ (Cr)	/BUSREQ setup time to Clock Rise	50		50		40		30		15		nS	

*For clock periods other than the minimums shown, calculate parameters using the table on the following page. Calculated values above assumed TrC = TfC = 20 ns.

†Units in nanoseconds (ns).

†† **For loading ≥ 50 pf. Decrease width by 10 ns for each additional 50 pf.**

**4 MHz CMOS Z80 is obsoleted and replaced by 6 MHz

Customer Support

For answers to technical questions about the product, documentation, or any other issues with Zilog's offerings, please visit Zilog's Knowledge Base at http://www.zilog.com/kb.

For any comments, detail technical questions, or reporting problems, please visit Zilog's Technical Support at http://support.zilog.com.