NEC

MOS INTEGRATED CIRCUIT

μPD70330, 70330(A)

V35™
16-BIT SINGLE-CHIP MICROCONTROLLERS

A stricter quality assurance program is applied to the μPD70330(A) than to the μPD70330 (standard) (NEC calls this program "Special" in respect to the quality grade category).

The μPD70330 (or V35) is a single-chip microcontroller consisting of 16-bit CPU, RAM, serial interface, timer, DMA controller, interrupt controller, etc. integrated to one chip. The μPD70330 is software-compatible with the 8/16 bit microprocessor μPD70108/70116 (or V20™/V30™).

Its functions are described in detail in the manual indicated below. Please read this manual before starting design.

- V25™, V35 User's Manual-Hardware : IEM-1220
- V25, V35 family User's Manual-Instructions : IEU-847

b

FEATURES

- Internal 16-bit architecture and external 16-bit data bus
- Software compatible with the μPD70108 and 70116 (in the native mode). Additional instructions are included.
- Minimum instruction cycle: 250 ns/8 MHz
- On-chip RAM: 256 words x 8-bit
- Memory space: 1 Mbytes
- Register bank (memory mapped): eight banks
- Input port with comparator (port T): eight bits
- I/O lines (Input port: four bits, Input/output ports: 20 bits)
- Serial interface dedicated baud rate generator on-chipped: two channels
  - Asynchronous mode
  - I/O interface mode
- Interrupt controller
  - Programmable priority (eight levels)
  - Vectored interrupt
  - Register bank switching
  - Macro service
- DRAM, pseudo SRAM refresh function
- DMA controller: two channels
  - Four DMA transfer modes
- 16-bit timer: two channels
- Time base counter (20 bits): one channel
- Clock generator on-chipped
- Programmable wait function
- Standby function (STOP or HALT)

The information in this document is subject to change without notice.

Document No. U10170EJ8V0DS00 (8th edition)
Date Published October 1995 P
Printed in Japan

The mark ★ shows revised points.

6427525 0062927 9T2

★ ORDERING INFORMATION

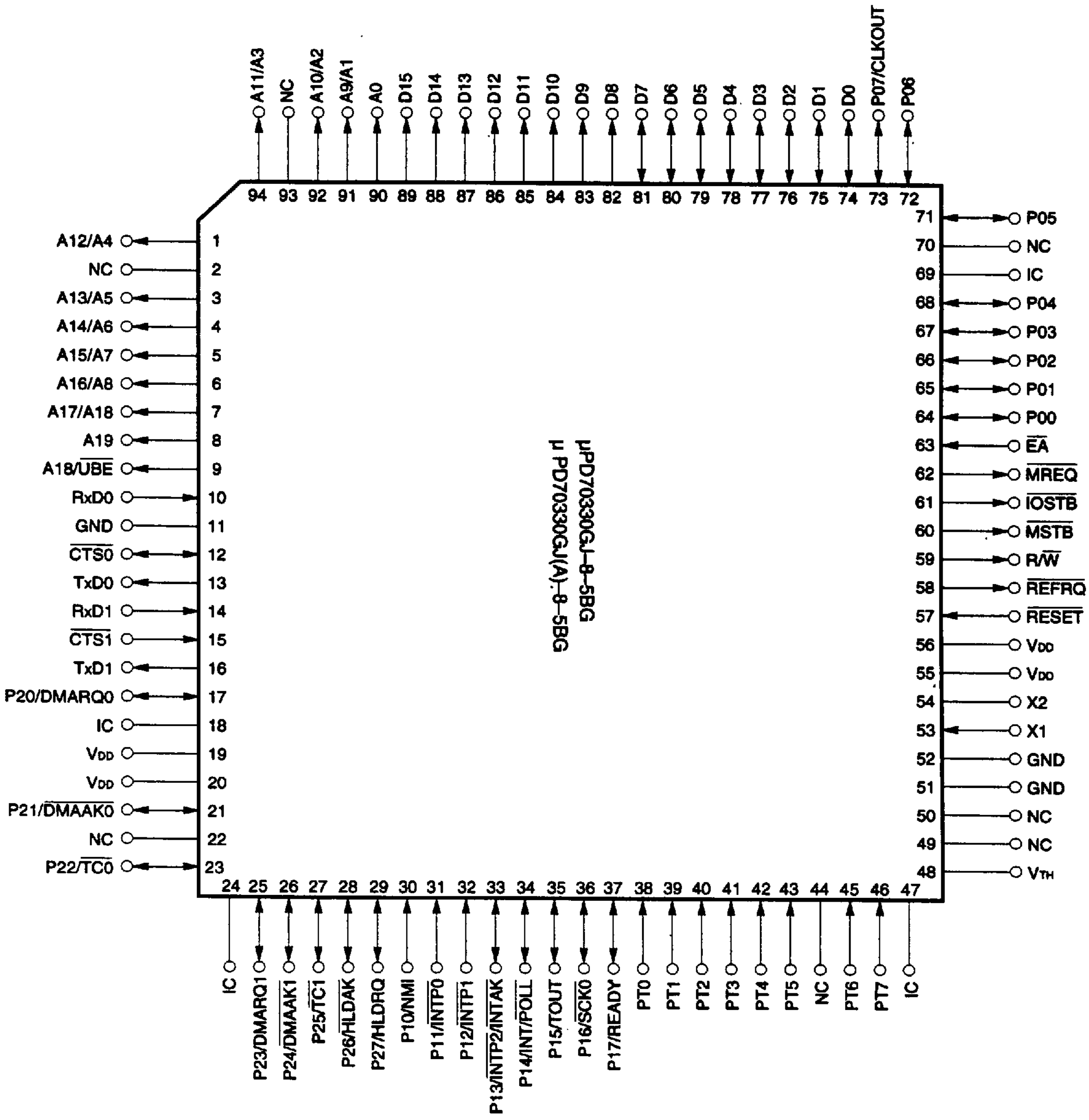
Part Number	Package	Frequency (MHz)	Quality grade
μPD70330L-8	84-pin plastic QFJ (1150 x 1150 mil)	8	Standard
μPD70330GJ-8-5BG	94-pin plastic QFP (20 x 20 mm)	8	Standard
μPD70330GJ(A)-8-5BG	94-pin plastic QFP (20 x 20 mm)	8	Special

Remark The plastic QFJ is a new name of PLCC.

Please refer to "Quality grade on NEC Semiconductor Devices" (Document number IEI-1209) published by NEC Corporation to know the specification of quality grade on the devices and its recommended applications.

PIN CONFIGURATION (Top view) ★

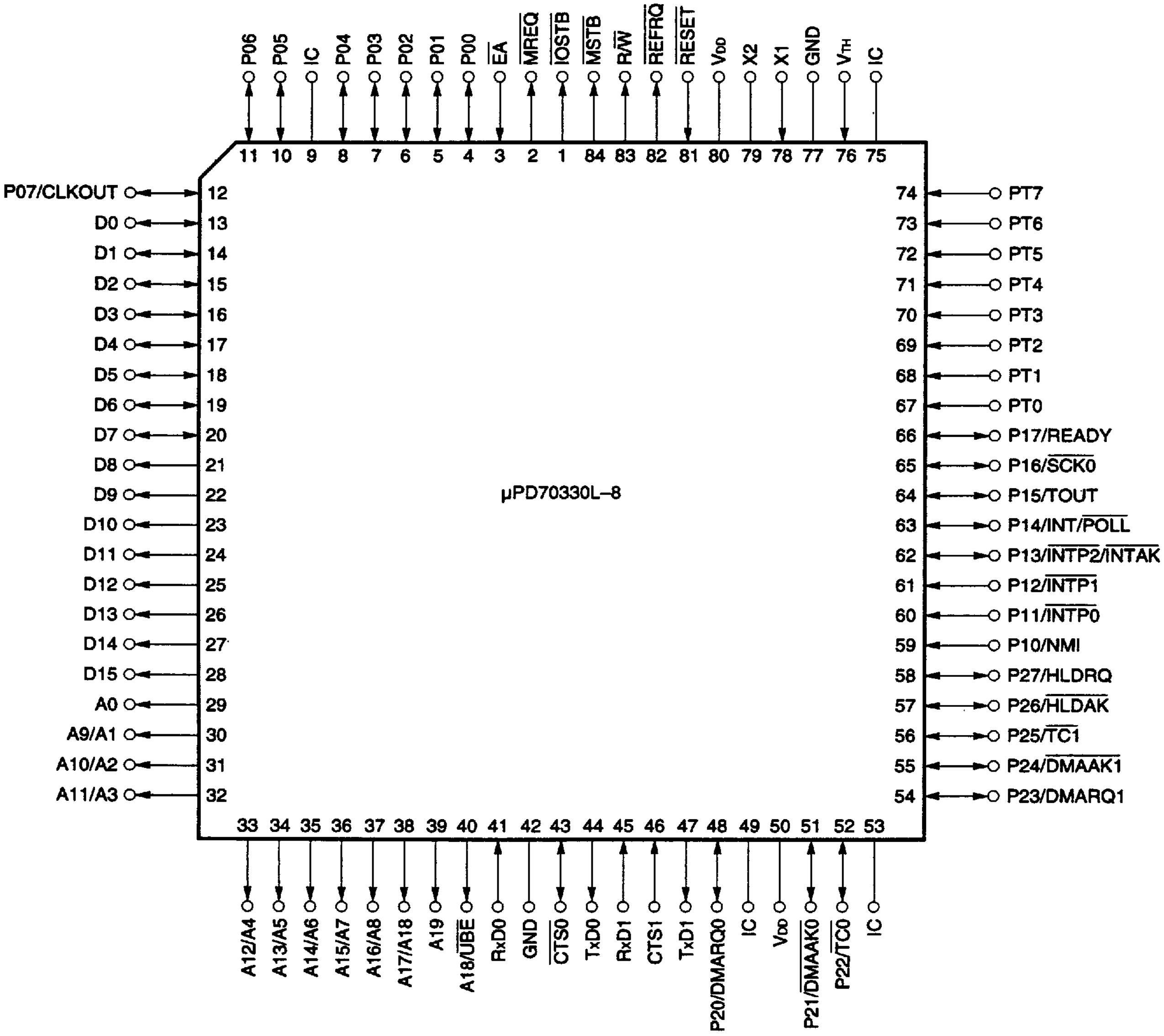
94-pin plastic QFP (20 x 20 mm)



NC: Non-Connection
IC : Internally Connected

Cautions 1. IC pins should be independently connected to V_{DD} through resistors (3 to 10 kΩ).
2. $\overline{EA}$ pins should be connected to GND through resistors (3 to 10 kΩ).

84-pin plastic QFJ (1150 x 1150 mil)

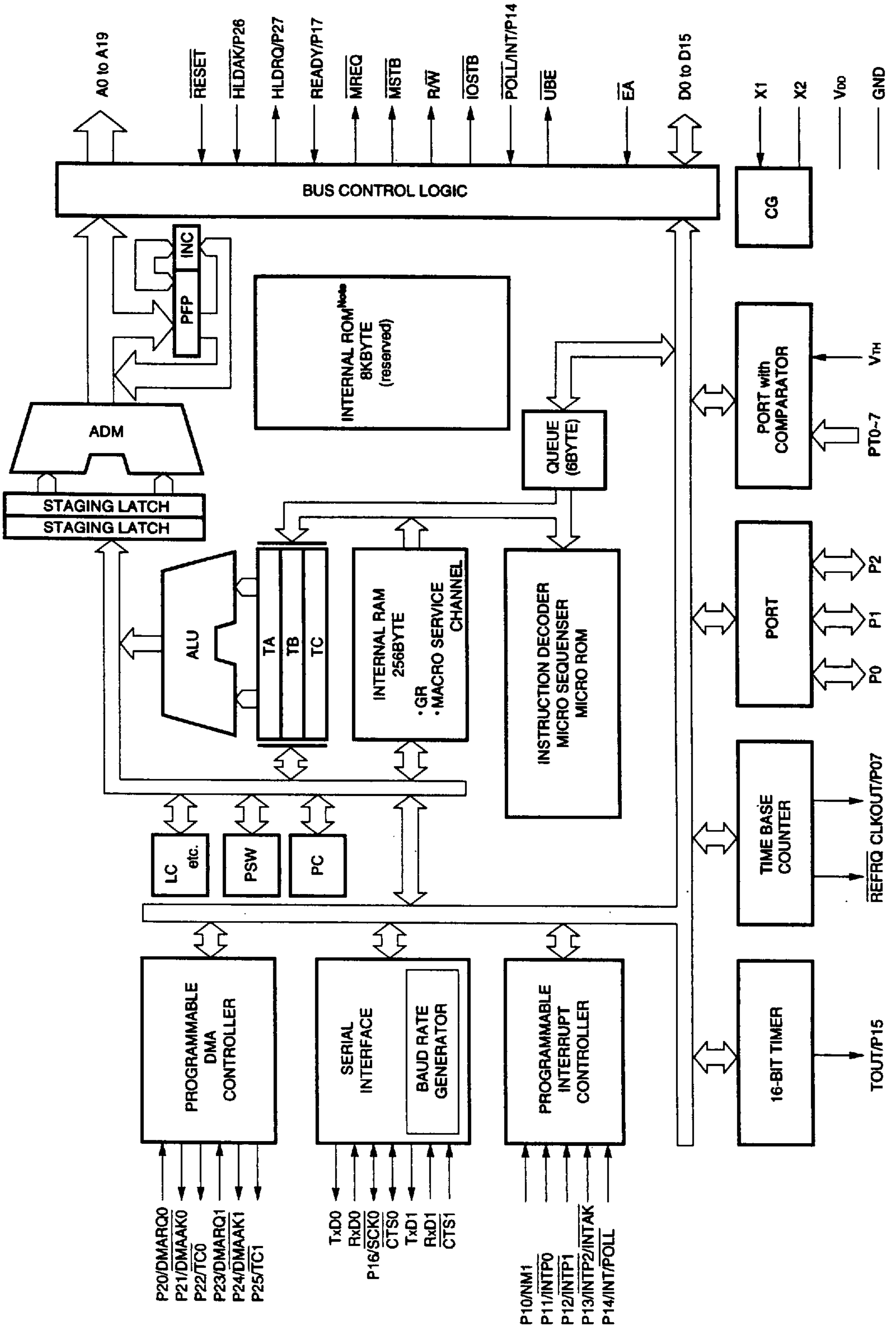


IC : Internally Connected

Cautions 1. IC pins should be independently connected to V_{DD} through resistors (3 to 10 kΩ).
2. $\overline{EA}$ pins should be connected to GND through resistors (3 to 10 kΩ).

INTERNAL BLOCK DIAGRAM

★



Note Not to be used by users

6427525 0062931 323

CONTENTS

1. PIN FUNCTIONS

1.1 Port Pins

Pin Name	I/O	Port Function	Control Function
P00 to P06	I/O	Eight-bit bidirectional input/output port for which input or output mode can be specified bitwise.	—
P07/CLKOUT	I/O/O		System clock output
P10/NMI	I	Nonmaskable interrupt request input (input port)	—
P11/$\overline{\text{INTP0}}$		External interrupt request input and input port	
P12/$\overline{\text{INTP1}}$			
P13/$\overline{\text{INTP2}}$/$\overline{\text{INTAK}}$	I/I/O		INT acknowledge signal output
P14/$\overline{\text{POLL}}$/INT	I/O/I/I	Bidirectional input/output port and $\overline{\text{POLL}}$ input	External interrupt request input
P15/TOUT	I/O/O	Bidirectional input/output port; input or output mode can be specified bitwise	Timer output
P16/$\overline{\text{SCK0}}$			Serial clock output
P17/READY	I/O/I		READY input
P20/DMARQ0	I/O/I	Eight-bit bidirectional input/output port; input or output mode can be specified bitwise	DMA request input (CH0)
P21/$\overline{\text{DMAAK0}}$	I/O/O		DMA acknowledge output (CH0)
P22/$\overline{\text{TC0}}$			DMA termination output (CH0)
P23/DMARQ1	I/O/I		DMA request input (CH1)
P24/$\overline{\text{DMAAK1}}$	I/O/O		DMA acknowledge output (CH1)
P25/$\overline{\text{TC1}}$			DMA termination output (CH1)
P16/HLDAK	I/O/O		HOLD acknowledge output
P27/HLDRQ	I/O/I		HOLD input
PT0 to PT7	I	Input port with 8-bit comparator	—

Remark All port pins become input ports after $\overline{\text{RESET}}$ returns high.
When using P13/$\overline{\text{INTP2}}$/$\overline{\text{INTAK}}$ as an $\overline{\text{INTAK}}$ pin, be sure to pull up the pin to avoid a malfunction of external-interrupt controller after $\overline{\text{RESET}}$ returns high.

1.2 Other Pins

Pin Name	I/O	Function
TxD0	O	Serial data output
TxD1		
RxD0	I	Serial data input
RxD1		
$\overline{\text{CTS0}}$	I/O	CTS input in asynchronous mode. Receive clock input/output in I/O interface mode.
$\overline{\text{CTS1}}$	I	CTS input
$\overline{\text{REFRQ}}$	O	DRAM refresh pulse output
V_{TH}	I	Comparator reference voltage input
$\overline{\text{RESET}}$		Reset signal input
$\overline{\text{EA}}$		External memory access (connected to GND through resistors (3 to 10 kΩ))
X1	I	Crystal resonator/ceramic resonator connection pins for system clock oscillator. External clock is input with opposing phases to the X1 and X2 pins.
X2		
D0 to D15	I/O	8-bit data bus
A0	O	Select low-order memory bank.
$\overline{\text{UBE}}$		Select high-order memory bank.
A1 to A19		Output 19-bit address
$\overline{\text{MREQ}}$		Output indicating that memory bus cycle is started.
$\overline{\text{MSTB}}$		Memory read or memory write strobe output.
R/$\overline{\text{W}}$		Read or write cycle identification signal output.
$\overline{\text{IOSTB}}$		I/O read or I/O write strobe output.
V_{DD}		Positive power supply pin. (Connect all V_{DD} pins).
GND		GND pin (Connect all GND pins)
IC		Internal connect independently connected to V_{DD} through resistors (3 to 10 kΩ).

2. INSTRUCTION SET

The instruction set for the μPD70330 is upward compatible with the μPD70108/μPD70116 instruction set in the native mode.

2.1 Additional Instructions to the μPD70108/μPD70116

New instructions added to the μPD70108/μPD70116 insruction set from the μPD70332 and μPD70330 are as follows:

(1) Conditional branch instruction

- BTCLR: Special function register bit test instruction
  When BTCLR is executed, the target special function register bit is cleared if the bit is set to 1, and a branch is made to the short-label described in the operand. If the bit is set to 0, the next instruction is executed. PSW does not change.

(Description format)

Mnemonic	Operand		
	Special function register address	Special function register bit	Branch destination
BTCLR	sfr	imm3	short-label

(2) Interrupt instructions

- RETRBI: Register bank return instruction
  The RETRBI instruction is used to return from the interrupt service routine using the register bank switching function. It cannot be used to return from a vectored interrupt.

(Description format)

Mnemonic	Operand
RETRBI	None

- FINT: Instruction to inform the interrupt controller that interrupt service terminates
  If an interrupt other than NMI, INT, or software interrupt is used, execute the FINT instruction before a return instruction from the interrupt. Do not use the FINT instruction in NMI, INT, or software interrupts.

(Description format)

Mnemonic	Operand
FINT	None

(3) CPU instruction

- STOP: Transition instruction to the STOP state

(Description format)

Mnemonic	Operand
STOP	None

(4) Register bank change instruction

- BRKCS: Register bank change instruction
  The current register bank is changed to the register bank indicated by the contents of the low-order three bits of the 16-bit register described in the operand. A branch is taken to the address obtained from PS prestored in the new register bank and vector PC. To return from the new register bank, use the RETRBI instruction.

(Description format)

Mnemonic	Operand
BRKCS	reg16

- TSKSW: Register bank change instruction
  The current register bank is changed to the register bank indicated by the contents of the three bits of the 16-bit register described in the operand. A branch is taken to the address obtained from PS prestored in the new register bank and the PC save area.

(Description format)

Mnemonic	Operand
TSKSW	reg16

(5) Data transfer instructions

- MOVSPA: SS and SP transfer instruction
  The SS and SP values before register banks are changed are transferred to SS and SP in the new register bank indicated.

(Description format)

Mnemonic	Operand
MOVSPA	None

- MOVSPB: SS and SP transfer instruction
  The SS and SP values in the current register bank are transferred to SS and SP in the new register bank indicated by the contents of the low-order three bits of the 16-bit register described in the operand.

(Description format)

Mnemonic	Operand
MOVSPB	reg16

When the following instructions of the μPD70108/μPD70116 instruction set are used on the μPD70330, note the following:

- Input/output instruction: Primitive input/output instruction — If the PSW $\overline{\text{IBRK}}$ flag is reset to 0, the instruction is not executed and an interrupt occurs. To use the input/output instruction, set the $\overline{\text{IBRK}}$ flag to 1.
- FPO instruction: The instruction is not executed and an interrupt occurs.

2.2 Instruction Set Operation

Table 2-1. Operand Identifier

Identifier	Description
reg	8/16-bit general-purpose register
reg8	8-bit general-purpose register
reg16	16-bit general-purpose register
dmem	8/16-bit memory location
mem	8/16-bit memory location
mem8	8-bit memory location
mem16	16-bit memory location
mem32	32-bit memory location
sfr	8-bit special function register location
imm	Constant in the range of 0-FFFFH
imm3	Constant in the range of 0-7
imm4	Constant in the range of 0-FH
imm8	Constant in the range of 0-FFH
imm16	Constant in the range of 0-FFFFH
acc	Register AW or AL
sreg	Segment register
src–table	256-byte conversion table name
src–block	Name of block addressed in register IX
dst–block	Name of block addressed in register IY
near–proc	Procedure in the current program segment
far–proc	Procedure in another program segment
near–label	Label in the current program segment
short–label	Label in the range of –128 to +127 bytes from the instruction end
far–label	Label in another program segment
memptr16	Word containing location offset in the current program segment to which control is to be transferred
memptr32	Double-word containing the segment base address and location offset in another program segment to which control is to be transferred
regptr16	16-bit general-purpose register containing location offset in another program segment to which control is to be transferred
pop–value	Number of bytes discarded from stack (0 to 64 K, normally even-number)
fp–op	Immediate value to determine the instruction code of an external floating-point operation chip
R	Register set

Table 2-2. Operand Code Identifier

Identifier	Description
W	Byte/word specification bit (0: byte, 1:word). However, when s = 1, sign extended byte data is specified 16-bit operand even if W = 1.
reg	Register field (000-111)
mem	Memory field (000-111)
mod	Mode field (00-10)
s	Sign extension specification bit (0: sign is not extended, 1: sign is extended)
X,XXX,YYY, ZZZ	Data to determine the instruction code of an external floating-point operation chip

Table 2-3. Operation Identifier

Identifier	Description
AW	Accumulator (16 bits)
AH	Accumulator (upper byte)
AL	Accumulator (lower byte)
BW	Register BW (16 bits)
CW	Register CW (16 bits)
CL	Register CW (lower byte)
DW	Register DW (16 bits)
SP	Stack pointer (16 bits)
PC	Program counter (16 bits)
PSW	Program status word (16 bits)
IX	Index register (source) (16 bits)
IY	Index register (destination) (16 bits)
PS	Program segment register (16 bits)
DS1	Data segment 1 register (16 bits)
DS0	Data segment 0 register (16 bits)
SS	Stack segment register (16 bits)
AC	Auxiliary carry flag
CY	Carry flag
P	Parity flag
S	Sign flag
Z	Zero flag
DIR	Direction flag
IE	Interrupt enable flag

Identifier	Description
V	Overflow flag
BRK	Break flag
MD	Mode flag
(···)	Contents of memory indicated in ( )
disp	Displacement (8 or 16 bits)
ext-disp8	16 bits of 8-bit displacement with extended sign
temp	Temporary register (8, 16, or 32 bits)
tmpcy	Temporary carry flag (one bit)
seg	Immediate segment data (16 bits)
offset	Immediate offset data (16 bits)
<-	Transfer direction
+	Addition
−	Subtraction
x	Multiplication
÷	Division
%	Modulo
∧	AND
∨	OR
⊻	Exclusive OR
xxH	2-digit hexadecimal number
xxxxH	4-digit hexadecimal number

Table 2-4. Flag Operation Identifier

Identifier	Description
(blank)	No change
0	Cleared to 0
1	Set to 1
x	Set to 1 or cleared according to the result
U	Undefined
R	Previously saved value is restored

Table 2-5. Selection of 8- or 16-Bit General Purpose Register

reg	W = 0	W = 1
000	AL	AW
001	CL	CW
010	DL	DW
011	BL	BW
100	AH	SP
101	CH	BP
110	DH	IX
111	BH	IY

Table 2-6. Segment Register Selection

sreg	
00	DS1
01	PS
10	SS
11	DS0

Table 2-7. Number of Clocks for Each Memory Addressing

mem \ mod	00		01		10	
		Clocks		Clocks		Clocks
000	BW + IX	3	BW + IX + disp8	3	BW + IX + disp16	4
001	BW + IY	3	BW + IY + disp8	3	BW + IY + disp16	4
010	BP + IX	3	BP + IX + disp8	3	BP + IX + disp16	4
011	BP + IY	3	BP + IY + disp8	3	BP + IY + disp16	4
100	IX	3	IX + disp8	3	IX + disp16	4
101	IY	3	IY + disp8	3	IY + disp16	4
110	Direct address	3	BP + disp8	3	BP + disp16	4
111	BW	3	BW + disp8	3	BW + disp16	4

In memory operand, the number of clocks varies depending on the addressing mode. Apply the numeric values listed in Table 2-8 to those written as "EA" in the list of the number of clocks.

"T" denotes the number of wait states. Apply any number of wait states from 0 (no wait).

The instruction fetch cycle is not counted as the number of clocks.

There are some branch instructions for which such description as the example below is provided.

The description indicates as follows:

Example 15/8 .... 15: the number of clock cycles when branched
8: the number of clock cycles when not branched

6427525 0062941 272

2.3 Intruction Set Table

Group	Mnemonic	Operand	Operation Code		Number of bytes	Operation	Flags					
			7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0			AC	CY	V	P	S	Z
Data transfer	MOV	reg, reg	1 0 0 0 1 0 1 W	1 1 reg reg	2	reg <- reg						
		mem, reg	1 0 0 0 1 0 0 W	mod reg mem	2 – 4	(mem) <- reg						
		reg, mem	1 0 0 0 1 0 1 W	mod reg mem	2 – 4	reg <- (mem)						
		mem, imm	1 1 0 0 0 1 1 W	mod 0 0 0 mem	3 – 6	(mem) <- imm						
		reg, imm	1 0 1 1 W reg		2 – 3	reg <- imm						
		acc, dmem	1 0 1 0 0 0 0 W		3	When W = 0, AL <- (dmem) When W = 1, AH <- (dmem+1), AL <- (dmem)						
		dmem, acc	1 0 1 0 0 0 1 W		3	When W = 0, (dmem) <- AL When W = 1, (dmem+1) <- AH, (dem) <- AL						
		sreg, reg16	1 0 0 0 1 1 1 0	1 1 0 sreg reg	2	sreg <- reg16 sreg: SS, DS0, DS1						
		sreg, mem16	1 0 0 0 1 1 1 0	mod 0 sreg mem	2 – 4	sreg <- (mem16) sreg: SS, DS0, DS1						
		reg16, sreg	1 0 0 0 1 1 0 0	1 1 0 sreg reg	2	reg16 <- sreg						
		mem16, sreg	1 0 0 0 1 1 0 0	mod 0 sreg mem	2 – 4	(mem16) <- sreg						
		DS0, reg16, mem32	1 1 0 0 0 1 0 1	mod reg mem	2 – 4	reg16 <- (mem32) DS0 <- (mem32+2)						
		DS1, reg16, mem32	1 1 0 0 0 1 0 0	mod reg mem	2 – 4	reg16 <- (mem32) DS1 <- (mem32+2)						
		AH, PSW	1 0 0 1 1 1 1 1		1	AH <- S, Z, F1, AC, F0, P, $\overline{\text{IBRK}}$, CY						
		PSW, AH	1 0 0 1 1 1 1 0		1	S, Z, F1, AC, F0, P, $\overline{\text{IBRK}}$, CY <- AH	x	x		x	x	x
	LDEA	reg16, mem16	1 0 0 0 1 1 0 1	mod reg mem	2 – 4	reg16 <- mem16						
	TRANS	src-table	1 1 0 1 0 1 1 1		1	AL <- (BW+AL)						
	XCH	reg, reg	1 0 0 0 0 1 1 W	1 1 reg reg	2	reg <-> reg						
		mem, reg reg, mem	1 0 0 0 0 1 1 W	mod reg mem	2 – 4	(mem) <-> reg						
		AW, reg16 reg16, AW	1 0 0 1 0 reg		1	AW <-> reg16						
	MOVSPA Note		0 0 0 0 1 1 1 1	0 0 1 0 0 1 0 1	2	SS and SP of the new register bank <- SS and SP of the old register bank						
	MOVSPB Note	reg16	0 0 0 0 1 1 1 1 1 1 1 1 1 reg	1 0 0 1 0 1 0 1	3	SS and SP of the new register bank indicated by reg16 <- SS and SP of the old register						

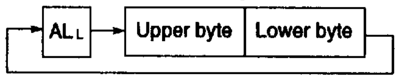
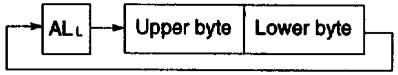
Note New added instruction from the μPD70108/70116

Group	Mnemonic	Operand	Operation Code 7 6 5 4 3 2 1 0	Operation Code 7 6 5 4 3 2 1 0	Number of bytes	Operation	AC	CY	V	P	S	Z
Repeat prefix	REPC		0 1 1 0 0 1 0 1		1	While CW ≠ 0, primitive block transfer instruction of the subsequent byte is executed and CW is decremented by one. If an interrupt is held, it is processed. When CY ≠ 1, loop is terminated.						
	REPNC		0 1 1 0 0 1 0 0		1	Same as above When CY ≠ 0, loop is terminated.						
	REP REPE REPZ		1 1 1 1 0 0 1 1		1	While CW ≠ 0, primitive block transfer instruction of the subsequent byte is executed and CW is decremented by one. If an interrupt is held, it is processed. When the primitive block transfer instruction is CMPBK or CMPM and Z ≠ 1, loop is terminated.						
	REPNE REPNZ		1 1 1 1 0 0 1 0		1	Same as above When Z ≠ 0, loop is terminated.						
Primitive block transfer	MOVBK	dst-block, src-block	1 0 1 0 0 1 0 W		1	When W = 0, (IY)<- (IX) DIR = 0: IX <- IX+1, IY <- IY+1 DIR = 1: IX <- IX−1, IY <- IY−1						
						When W = 1, (IY+1, IY) <- (IX+1, IX) DIR = 0: IX <- IX+2, IY <- IY+2 DIR = 1: IX <- IX−2, IY <- IY−2						
	CMPBK	src-block, dst-block	1 0 1 0 0 1 1 W		1	When W = 0, (IX) − (IY) DIR = 0: IX <- IX+1, IY <- IY+2 DIR = 1: IX <- IX−1, IY <- IY−2	x	x	x	x	x	x
						When W = 1, (IX+1, IX) − (IY+1, IY) DIR = 0: IX <- IX+2, IY <- IY+2 DIR = 1: IX <- IX−2, IY <- IY−2						
	CMPM	dst-block	1 0 1 0 1 1 1 W		1	When W = 0, AL − (IY) DIR = 0: IY<- IY+1; DIR = 1: IY <- IY−1	x	x	x	x	x	x
						When W = 1, AW − (IY+1, IY) DIR = 0: IY<- IY+2; DIR = 1: IY <- IY−2						
	LDM	src-block	1 0 1 0 1 1 0 W		1	WHen W = 0, AL<- (IX) DIR = 0: IX<- IX+1; DIR = 1: IX <- IX−1						
						When W = 1, AW<- (IX+1, IX) DIR = 0: IX+2; DIR = 1: IX <- IX −2						

Group	Mnemonic	Operand	Operation Code 7 6 5 4 3 2 1 0	Operation Code 7 6 5 4 3 2 1 0	Number of bytes	Operation	AC	CY	V	P	S	Z
Primitive block transfer	STM	dst-block	1 0 1 0 1 0 1 W		1	When W = 0, (IY) <- AL DIR = 0: IY <- IY+1; DIR = 1: IY <- IY−1						
						When W = 1 (IY+1, IY) <- AW DIR = 0: IY <- IY+2; DIR = 1: IY <- IY−2						
Bit field operation	INS	reg8, reg8	0 0 0 0 1 1 1 1	0 0 1 1 0 0 0 1	3	16-bit field <- AW						
			1 1 reg reg									
		reg8, imm4	0 0 0 0 1 1 1 1	0 0 1 1 1 0 0 1	4	16-bit field <-AW						
			1 1 0 0 0 reg									
	EXT	reg8, reg8	0 0 0 0 1 1 1 1	0 0 1 1 0 0 1 1	3	AW <- 16-bit field						
			1 1 reg reg									
		reg8, imm4	0 0 0 0 1 1 1 1	0 0 1 1 1 0 1 1	4	AW <- 16-bit field						
			1 1 0 0 0 reg									
Input/ output	IN Note	acc, imm8	1 1 1 0 0 1 0 W		2	When W = 0, AL <- (imm8) When W = 1, AH <- (imm8+1), AL <- (imm8)						
		acc, DW	1 1 1 0 1 1 0 W		1	When W = 0, AL <- (DW) When W = 1, AH <- (DW+1), AL <- (DW)						
	OUT Note	imm8, acc	1 1 1 0 0 1 1 W		2	When W = 0, (imm8) <- AL When W = 1, (imm8+1) <- AH, (imm8) <- AL						
		DW, acc	1 1 1 0 1 1 1 W		1	When W = 0, (DW) <- AL When W = 1, (DW+1) <- AH, (DW) <- AL						
Primitive input/ output	INM Note	dst-block, DW	0 1 1 0 1 1 0 W		1	When W = 0, (IY) <- (DW) DIR = 0: IY <- IY+1; DIR = 1: IY <- IY−1						
						When W = 1, (IY+1, IY) <- (DW+1, DW) DIR = 0: IY <- IY+2; DIR = 1: IY <- IY−2						
	OUTM Note	DW, src-block	0 1 1 0 1 1 1 W		1	When W = 0, (DW) <- (IX) DIR = 0: IX <- IX+1; DIR = 1: IX <- IX−1						
						When W = 1, (DW+1, DW) <- (IX+1, IX) DIR = 0: IX <- IX+2; DIR = 1: IX <- IX−2						

Note When $\overline{\text{IBRK}}$=0, a software interrupt automatically occurs, and the instruction is not executed.

Group	Mnemonic	Operand	Operation Code 7 6 5 4 3 2 1 0	Operation Code 7 6 5 4 3 2 1 0	Number of bytes	Operation	Flags AC	CY	V	P	S	Z
Addition and subtraction	ADD	reg, reg	0 0 0 0 0 0 1 W	1 1 reg reg	2	reg <- reg+reg	x	x	x	x	x	x
		mem, reg	0 0 0 0 0 0 0 W	mod reg mem	2 – 4	(mem) <- (mem)+reg	x	x	x	x	x	x
		reg, mem	0 0 0 0 0 0 1 W	mod reg mem	2 – 4	reg <- reg+(mem)	x	x	x	x	x	x
		reg, imm	1 0 0 0 0 0 s W	1 1 0 0 0 reg	3 – 4	reg <- reg+imm	x	x	x	x	x	x
		mem, imm	1 0 0 0 0 0 s W	mod 0 0 0 mem	3 – 6	(mem) <- (mem)+imm	x	x	x	x	x	x
		acc, imm	0 0 0 0 0 1 0 W		2 – 3	When W = 0, AL <- AL+imm When W = 1, AW <- AW+imm	x	x	x	x	x	x
	ADDC	reg, reg	0 0 0 1 0 0 1 W	1 1 reg reg	2	reg <- reg+reg+CY	x	x	x	x	x	x
		mem, reg	0 0 0 1 0 0 0 W	mod reg mem	2 – 4	(mem) <- (mem)+reg+CY	x	x	x	x	x	x
		reg, mem	0 0 0 1 0 0 1 W	mod reg mem	2 – 4	reg <- reg+(mem)+CY	x	x	x	x	x	x
		reg, imm	1 0 0 0 0 0 s W	1 1 0 1 0 reg	3 – 4	reg <- reg+imm+CY	x	x	x	x	x	x
		mem, imm	1 0 0 0 0 0 s W	mod 0 1 0 mem	3 – 6	(mem) <- (mem)+imm+CY	x	x	x	x	x	x
		acc, imm	0 0 0 1 0 1 0 W		2 – 3	When W = 0, AL <- AL+imm+CY When W = 1, AW <- AW+imm+CY	x	x	x	x	x	x
	SUB	reg, reg	0 0 1 0 1 0 1 W	1 1 reg reg	2	reg <- reg–reg	x	x	x	x	x	x
		mem, reg	0 0 1 0 1 0 0 W	mod reg mem	2 – 4	(mem) <- (mem)–reg	x	x	x	x	x	x
		reg, mem	0 0 1 0 1 0 1 W	mod reg mem	2 – 4	reg <- reg–(mem)	x	x	x	x	x	x
		reg, imm	1 0 0 0 0 0 s W	1 1 1 0 1 reg	3 – 4	reg <- reg–imm	x	x	x	x	x	x
		mem, imm	1 0 0 0 0 0 s W	mod 1 0 1 mem	3 – 6	(mem) <- (mem)–imm	x	x	x	x	x	x
		acc, imm	0 0 1 0 1 1 0 W		2 – 3	When W = 0, AL <- AL–imm When W = 1, AW <- AW–imm	x	x	x	x	x	x
	SUBC	reg, reg	0 0 0 1 1 0 1 W	1 1 reg reg	2	reg <- reg–reg–CY	x	x	x	x	x	x
		mem, reg	0 0 0 1 1 0 0 W	mod reg mem	2 – 4	(mem) <- (mem)–reg–CY	x	x	x	x	x	x
		reg, mem	0 0 0 1 1 0 1 W	mod reg mem	2 – 4	reg <- reg–(mem)–CY	x	x	x	x	x	x
		reg, imm	1 0 0 0 0 0 s W	1 1 0 1 1 reg	3 – 4	reg <- reg–imm–CY	x	x	x	x	x	x
		mem, imm	1 0 0 0 0 0 s W	mod 0 1 1 mem	3 – 6	(mem) <- (mem)–imm–CY	x	x	x	x	x	x
		acc, imm	0 0 0 1 1 1 0 W		2 – 3	When W = 0, AL <- AL–imm–CY When W = 1, AW <- AW–imm–CY	x	x	x	x	x	x

Group	Mnemonic	Operand	Operation Code		Number of bytes	Operation	Flags					
			7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0			AC	CY	V	P	S	Z
BCD operation	ADD4S		0 0 0 0 1 1 1 1	0 0 1 0 0 0 0 0	2	dst BCD string <- dst BCD string + src BCD string Note	U	x	U	U	U	x
	SUB4S		0 0 0 0 1 1 1 1	0 0 1 0 0 0 1 0	2	dst BCD string <- dst BCD string – src BCD string Note	U	x	U	U	U	x
	CMP4S		0 0 0 0 1 1 1 1	0 0 1 0 0 1 1 0	2	dst BCD string – src BCD string Note	U	x	U	U	U	x
	ROL4	reg8	0 0 0 0 1 1 1 1 1 1 0 0 0 reg	0 0 1 0 1 0 0 0	3	reg 						
		mem8	0 0 0 0 1 1 1 1 mod 0 0 0 mem	0 0 1 0 1 0 0 0	3 – 5	mem 						
	ROR4	reg8	0 0 0 0 1 1 1 1 1 1 0 0 0 reg	0 0 1 0 1 0 1 0	3	reg 						
		mem8	0 0 0 0 1 1 1 1 mod 0 0 0 mem	0 0 1 0 1 0 1 0	3 – 5	mem 						
Increment and decrement	INC	reg8	1 1 1 1 1 1 1 0	1 1 0 0 0 reg	2	reg8 <- reg8+1	x		x	x	x	x
		mem	1 1 1 1 1 1 1 W	mod 0 0 0 mem	2 – 4	(mem) <- (mem)+1	x		x	x	x	x
		reg16	0 1 0 0 0 reg		1	reg16 <- reg16+1	x		x	x	x	x
	DEC	reg8	1 1 1 1 1 1 1 0	1 1 0 0 1 reg	2	reg8 <- reg8+1	x		x	x	x	x
		mem	1 1 1 1 1 1 1 W	mod 0 0 1 mem	2 – 4	(mem) <- (mem)–1	x		x	x	x	x
		reg16	0 1 0 0 1 reg		1	reg16 <- reg16–1	x		x	x	x	x

n: Half of the number of BCD digits

Note The number of BCD digits is specified in the CL register. CL can be set in the range of 1 to 254.

Group	Mnemonic	Operand	Operation Code 7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	Number of bytes	Operation	Flags AC	CY	V	P	S	Z
Multipli-cation	MULU	reg8	1 1 1 1 0 1 1 0	1 1 1 0 0 reg	2	AW <- AL x reg8 AH = 0: CY<- 0, V <- 0 AH ≠ 0: CY <- 1, V <- 1	U	x	x	U	U	U
		mem8	1 1 1 1 0 1 1 0	mod 1 0 0 mem	2 – 4	AW <- AL x (mem8) AH = 0: CY<- 0, V <- 0 AH ≠ 0: CY <- 1, V <- 1	U	x	x	U	U	U
		reg16	1 1 1 1 0 1 1 1	1 1 1 0 0 reg	2	DW, AW <- AW x reg16 DW = 0: CY<- 0, V <- 0 DW = 1: CY <- 1, V <- 1	U	x	x	U	U	U
		mem16	1 1 1 1 0 1 1 1	mod 1 0 0 mem	2 – 4	DW, AW <- AW x (mem16) DW = 0: CY<- 0, V <- 0 DW = 1: CY <- 1, V <- 1	U	x	x	U	U	U
	MUL	reg8	1 1 1 1 0 1 1 0	1 1 1 0 1 reg	2	AW <- AL x reg8 AH = AL sign extension: CY<- 0, V <- 0 AH ≠ AL sign extension: CY<- 1, V <- 1	U	x	x	U	U	U
		mem8	1 1 1 1 0 1 1 0	mod 1 0 1 mem	2 – 4	AW <- AL x (mem8) AH = AL sign extension: CY<- 0, V <- 0 AH ≠ AL sign extension: CY<- 1, V <- 1	U	x	x	U	U	U
		reg16	1 1 1 1 0 1 1 1	1 1 1 0 1 reg	2	DW, AW <- AW x reg16 DW = AW sign extension: CY<- 0, V <- 0 DW ≠ AW sign extension: CY<- 1, V <- 1	U	x	x	U	U	U
		mem16	1 1 1 1 0 1 1 1	mod 1 0 1 mem	2 – 4	DW, AW <- AW x (mem16) DW = AW sign extension: CY<- 0, V <- 0 DW ≠ AW sign extension: CY<- 1, V <- 1	U	x	x	U	U	U
		reg16, (reg16) Note, imm8	0 1 1 0 1 0 1 1	1 1 reg reg	3	reg16 <- reg16 x imm8 Product ≤ 16 bits: CY <- 0, V <- 0 Product > 16 bits: CY <- 1, V <- 1	U	x	x	U	U	U
		reg16, mem16 imm8	0 1 1 0 1 0 1 1	mod reg mem	3 – 5	reg16 <- (mem16) x imm8 Product ≤ 16 bits: CY <- 0, V <- 0 Product > 16 bits: CY <- 1, V <- 1	U	x	x	U	U	U
		reg16, (reg16) Note, imm8	0 1 1 0 1 0 0 1	1 1 reg reg	4	reg16 <- reg16 x imm16 Product ≤ 16 bits: CY <- 0, V <- 0 Product > 16 bits: CY <- 1, V <- 1	U	x	x	U	U	U
		reg16, mem16, imm8	0 1 1 0 1 0 0 1	mod reg mem	4 – 6	reg16 <- (mem16) x imm16 Product ≤ 16 bits: CY <- 0, V <- 0 Product > 16 bits: CY <- 1, V <- 1	U	x	x	U	U	U

Note The second operand can be omitted. If it is omitted, the same register as th e first operand is assumed to be specified.

Group	Mnemonic	Operand	Operation Code 7 6 5 4 3 2 1 0	Operation Code 7 6 5 4 3 2 1 0	Number of bytes	Operation	AC	CY	V	P	S	Z
Unsigned division	DIVU	reg8	1 1 1 1 0 1 1 0	1 1 1 1 0 reg	2	temp <- AW When temp + reg8 ≤ FFH, AH <- temp%reg8, AL <- temp + reg8 When temp + reg8 > FFH, (SP−1, SP−2) <- PSW, (SP−3, SP−4) <- PS (SP−5, SP−6) <- PC, SP <- SP−6 IE <- 0, BRK <- 0, PS <- (3, 2), PC <- (1, 0)	U	U	U	U	U	U
		mem8	1 1 1 1 0 1 1 0	mod 1 1 0 mem	2 – 4	temp <- AW When temp + (mem8) ≤ FFH, AH <- temp%(mem8), AL <- temp + (mem8) When temp + (mem8) > FFH, (SP−1, SP−2) <- PSW, (SP−3, SP−4) <- PS (SP−5, SP−6) <- PC, SP <- SP−6 IE <- 0, BRK <- 0, PS <- (3, 2), PC <- (1, 0)	U	U	U	U	U	U
		reg16	1 1 1 1 0 1 1 1	1 1 1 1 0 reg	2	temp <- DW, AW When temp + reg16 ≤ FFFFH, DW <- temp%reg16, AW <- temp + reg16 When temp + reg16 > FFFFH, (SP−1, SP−2) <- PSW, (SP−3, SP−4) <- PS (SP−5, SP−6) <- PC, SP <- SP−6 IE <- 0, BRK <- 0, PS <- (3, 2), PC <- (1, 0)	U	U	U	U	U	U
		mem16	1 1 1 1 0 1 1 1	mod 1 1 0 mem	2 – 4	temp <- DW, AW When temp + (mem16) ≤ FFFFH, DW <- temp%(mem16), AW <- temp + (mem16) When temp + (mem16) > FFFFH, (SP−1, SP−2) <- PSW, (SP−3,SP−4) <- PS (SP−5, SP−6) <- PC, SP <- SP−6 IE <- 0, BRK <- 0, PS <- (3, 2), PC <- (1, 0)	U	U	U	U	U	U

Group	Mnemonic	Operand	Operation Code 7 6 5 4 3 2 1 0	Operation Code 7 6 5 4 3 2 1 0	Number of bytes	Operation	Flags AC	CY	V	P	S	Z
Signed division	DIV	reg8	1 1 1 1 0 1 1 0	1 1 1 1 1 reg	2	temp <- AW When temp ÷ reg8 > 0 and temp ÷ reg8 ≤ 7FH or temp ÷ reg8 < 0 and temp ÷ reg8 > 0–7FH–1, AH <- temp%reg8, AL <- temp ÷ reg8 When temp ÷ reg8 > 0 and temp ÷ reg8 > 7FH or temp ÷ reg8 > 0 and temp ÷ reg8 < 0–7FH–1 (SP–1, SP–2) <- PSW, (SP–3, SP–4) <- PS (SP–5, SP–6) <- PC, SP <- SP–6 IE <- 0, BRK <- 0, PS <- (3, 2), PC <- (1, 0)	U	U	U	U	U	U
		mem8	1 1 1 1 0 1 1 0	mod 1 1 1 mem	2 – 4	temp <- AW When temp ÷ (mem8) > 0 and temp ÷ (mem8) ≤ 7FH or temp ÷ (mem8) < 0 and temp ÷ (mem8) > 0–7FH–1, AH <- temp%(mem8), AL <- temp ÷ (mem8) When temp ÷ (mem8) > 0 and temp ÷ (mem8) > 7FH or temp ÷ (mem8) > 0 and temp ÷ (mem8) < 0–7FH–1, (SP–1, SP–2) <- PSW, (SP–3,SP–4) <-PS (SP–5, SP–6) <- PC, SP <- SP–6 IE <- 0, BRK <- 0, PS <- (3, 2), PC <- (1, 0)	U	U	U	U	U	U
		reg16	1 1 1 1 0 1 1 1	1 1 1 1 1 reg	2	temp <- DW, AW When temp ÷ reg16 > 0 and temp ÷ reg16 ≤ 7FFFH or temp ÷ reg16 < 0 and temp ÷ reg16 > 0–7FFFH–1, DW <- temp%reg16, AW <- temp ÷ reg16 When temp ÷ reg16 > 0 and temp ÷ reg16 > 7FFFH or temp ÷ reg16 > 0 and temp ÷ reg16 < 0–7FFFH–1, (SP–1,SP–2) <- PSW, (SP-3, SP–4) <-PS (SP–5, SP–6) <- PC, SP <- SP – 6 IE <- 0, BRK <- 0, PS <- (3, 2), PC <- (1, 0)	U	U	U	U	U	U
		mem16	1 1 1 1 0 1 1 1	mod 1 1 1 mem	2 – 4	temp <- DW, AW When temp ÷ (mem16) > 0 and temp ÷ (mem16) ≤ 7FFFH or temp ÷ (mem16) < 0 and temp ÷ (mem16) > 0–7FFFH–1, DW <- temp%(mem16), AW <- temp ÷ (mem16) When temp ÷ (mem16) > 0 and temp ÷ (mem16) > 7FFFH or temp ÷ (mem16) > 0 and temp ÷ (mem16) < 0–7FFFH–1, (SP–1, SP–2) <- PSW, (SP–3, SP–4) <-PS (SP–5, SP–6) <- PC, SP <- SP–6 IE <- 0, BRK <- 0, PS <- (3, 2), PC <- (1, 0)	U	U	U	U	U	U

Group	Mnemonic	Operand	Operation Code		Number of bytes	Operation	Flags					
			7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0			AC	CY	V	P	S	Z
BCD adjustment	ADJBA		0 0 1 1 0 1 1 1		1	When AL ∧ 0FH > 9 or AC = 1, AL <- AL+6 AH <- AH+1, AC <- 1, CY <- AC, AL <- AL ∧ 0FH	x	x	U	U	U	U
	ADJ4A		0 0 1 0 0 1 1 1		1	When AL ∧ 0FH > 9 or AC = 1, AL <- AL+6, AC <- 1 When AL > 9FH or CY = 1 AL <- AL+60H, CY <- 1	x	x	U	x	x	x
	ADJBS		0 0 1 1 1 1 1 1		1	When AL ∧ 0FH > 9 or AC = 1, AL <- AL–6, AH <- AH–1, AC <- 1 CY <- AC, AL <- AL ∧ 0FH	x	x	U	U	U	U
	ADJ4S		0 0 1 0 1 1 1 1		1	When AL ∧ 0FH > 9 or AC = 1, AL <- AL–6, AC <- 1 When AL > 9FH or CY = 1 AL <- AL–60H, CY <- 1	x	x	U	x	x	x
Data conversion	CVTBD		1 1 0 1 0 1 0 0	0 0 0 0 1 0 1 0	2	AH <- AL + 0AH, AL <- AL%0AH	U	U	U	x	x	x
	CVTDB		1 1 0 1 0 1 0 1	0 0 0 0 1 0 1 0	2	AL <- AH x 0AH + AL, AH <- 0	U	U	U	x	x	x
	CVTBW		1 0 0 1 1 0 0 0		1	When AL < 80H, AH <- 0; otherwise, AH <- FFH						
	CVTWL		1 0 0 1 1 0 0 1		1	When AW < 8000H, DW <- 0; otherwise, DW <- FFFFH						
Comparison	CMP	reg, reg	0 0 1 1 1 0 1 W	1 1 reg reg	2	reg – reg	x	x	x	x	x	x
		mem, reg	0 0 1 1 1 0 0 W	mod reg mem	2 – 4	(mem) – reg	x	x	x	x	x	x
		reg, mem	0 0 1 1 1 0 1 W	mod reg mem	2 – 4	reg – (mem)	x	x	x	x	x	x
		reg, imm	1 0 0 0 0 0 s W	1 1 1 1 1 reg	3 – 4	reg – imm	x	x	x	x	x	x
		mem, imm	1 0 0 0 0 0 s W	mod 1 1 1 mem	3 – 6	(mem) – imm	x	x	x	x	x	x
		acc, imm	0 0 1 1 1 1 0 W		2 – 3	When W = 0, AL – imm When W = 1, AW – imm	x	x	x	x	x	x
Comple-ment operation	NOT	reg	1 1 1 1 0 1 1 W	1 1 0 1 0 reg	2	reg <- $\overline{reg}$						
		mem	1 1 1 1 0 1 1 W	mod 0 1 0 mem	2 – 4	(mem <- $\overline{(mem)}$)						
	NEG	reg	1 1 1 1 0 1 1 W	1 1 0 1 0 reg	2	reg <- $\overline{reg}$ + 1	x	x	x	x	x	x
		mem	1 1 1 1 0 1 1 W	mod 0 1 0 mem	2 – 4	(mem) <- $\overline{(mem)}$ + 1	x	x	x	x	x	x

Group	Mnemonic	Operand	Operation Code 7 6 5 4 3 2 1 0	Operation Code 7 6 5 4 3 2 1 0	Number of bytes	Operation	Flags AC	CY	V	P	S	Z
Logical operation	TEST	reg, reg	1 0 0 0 0 1 0 W	1 1 reg reg	2	reg ∧ reg	U	0	0	x	x	x
		mem, reg reg, mem	1 0 0 0 0 1 0 W	mod reg mem	2 – 4	(mem) ∧ reg	U	0	0	x	x	x
		reg, imm	1 1 1 1 0 1 1 W	1 1 0 0 0 reg	3 – 4	reg ∧ imm	U	0	0	x	x	x
		mem, imm	1 1 1 1 0 1 1 W	mod 0 0 0 mem	3 – 6	(mem) ∧ imm	U	0	0	x	x	x
		acc, imm	1 0 1 0 1 0 0 W		2 – 3	When W = 0, AL ∧ imm8 When W = 1, AW ∧ imm16	U	0	0	x	x	x
	AND	reg, reg	0 0 1 0 0 0 1 W	1 1 reg reg	2	reg <- reg ∧ reg	U	0	0	x	x	x
		mem, reg	0 0 1 0 0 0 0 W	mod reg mem	2 – 4	(mem) <- (mem) ∧ reg	U	0	0	x	x	x
		reg, mem	0 0 1 0 0 0 1 W	mod reg mem	2 – 4	reg <- reg ∧ (mem)	U	0	0	x	x	x
		reg, imm	1 0 0 0 0 0 0 W	1 1 1 0 0 reg	3 – 4	reg <- reg ∧ imm	U	0	0	x	x	x
		mem, imm	1 0 0 0 0 0 0 W	mod 1 0 0 mem	3 – 6	(mem) <- (mem) ∧ imm	U	0	0	x	x	x
		acc, imm	0 0 1 0 0 1 0 W		2 – 3	When W = 0, AL <- AL ∧ imm8 When W = 1, AW <- AW ∧ imm16	U	0	0	x	x	x
	OR	reg, reg	0 0 0 0 1 0 1 W	1 1 reg reg	2	reg <- reg ∨ reg	U	0	0	x	x	x
		mem, reg	0 0 0 0 1 0 0 W	mod reg mem	2 – 4	(mem) <- (mem) ∨ reg	U	0	0	x	x	x
		reg, mem	0 0 0 0 1 0 1 W	mod reg mem	2 – 4	reg <- rev ∨ (mem)	U	0	0	x	x	x
		reg, imm	1 0 0 0 0 0 0 W	1 1 0 0 1 reg	3 – 4	reg <- reg ∨ imm	U	0	0	x	x	x
		mem, imm	1 0 0 0 0 0 0 W	mod 0 0 1 mem	3 – 6	(mem) <- (mem) ∨ imm	U	0	0	x	x	x
		acc, imm	0 0 0 0 1 1 0 W		2 – 3	When W = 0, AL <- AL ∨ imm8 When W = 1, AW <- AW ∨ imm16	U	0	0	x	x	x
	XOR	reg, reg	0 0 1 1 0 0 1 W	1 1 reg reg	2	reg <- reg ⊻ reg	U	0	0	x	x	x
		mem, reg	0 0 1 1 0 0 0 W	mod reg mem	2 – 4	(mem) <- (mem) ⊻ reg	U	0	0	x	x	x
		reg, mem	0 0 1 1 0 0 1 W	mod reg mem	2 – 4	reg <- reg ⊻ (mem)	U	0	0	x	x	x
		reg, imm	1 0 0 0 0 0 0 W	1 1 1 1 0 reg	3 – 4	reg <- reg ⊻ (imm)	U	0	0	x	x	x
		mem, imm	1 0 0 0 0 0 0 W	mod 1 1 0 mem	3 – 6	(mem) <- (mem) ⊻ imm	U	0	0	x	x	x
		acc, imm	0 0 1 1 0 1 0 W		2 – 3	When W = 0, AL <- AL ⊻ imm8 When W = 1, AW <- AW ⊻ imm16	U	0	0	x	x	x

Group	Mnemonic	Operand	Operation Code 7 6 5 4 3 2 1 0	Operation Code 7 6 5 4 3 2 1 0	Number of bytes	Operation	AC	CY	V	P	S	Z
Bit operation	TEST1	reg8, CL	0 0 0 1 0 0 0 0	1 1 0 0 0 reg	3	reg8 bit NO. CL = 0: Z<-1 reg8 bit NO. CL = 1: Z<-0	U	0	0	U	U	x
		mem8, CL	0 0 0 0	mod 0 0 0 mem	3 – 5	(mem8) bit NO. CL = 0: Z<-1 (mem8) bit NO. CL = 1: Z<-0	U	0	0	U	U	x
		reg16, CL	0 0 0 1	1 1 0 0 0 reg	3	reg16 bit NO. CL = 0: Z<-1 reg16 bit NO. CL = 1: Z<-0	U	0	0	U	U	x
		mem16, CL	0 0 0 1	mod 0 0 0 mem	3 – 5	(mem16) bit NO. CL = 0: Z<-1 (mem16) bit NO. CL = 1: Z<-0	U	0	0	U	U	x
		reg8, imm3	1 0 0 0	1 1 0 0 0 reg	4	reg8 bit NO. imm3 = 0: Z<-1 reg8 bit NO. imm3 = 1: Z<-0	U	0	0	U	U	x
		mem8, imm3	1 0 0 0	mod 0 0 0 mem	4 – 6	(mem8) bit NO. imm3 = 0: Z<-1 (mem8) bit NO. imm3 = 1: Z<-0	U	0	0	U	U	x
		reg16, imm4	1 0 0 1	1 1 0 0 0 reg	4	reg16 bit NO. imm4 = 0: Z<-1 reg16 bit NO. imm4 = 1: Z<-0	U	0	0	U	U	x
		mem16, imm4	1 0 0 1	mod 0 0 0 mem	4 – 6	(mem16) bit NO. imm4 = 0: Z<-1 (mem16) bit NO. imm4 = 1: Z<-0	U	0	0	U	U	x
	NOT1	reg8, CL	0 1 1 0	1 1 0 0 0 reg	3	reg8 bit NO. CL <- $\overline{\text{reg8 bit NO. CL}}$						
		mem8, CL	0 1 1 0	mod 0 0 0 mem	3 – 5	(mem8) bit NO. CL <- $\overline{\text{(mem8) bit NO. CL}}$						
		reg16, CL	0 1 1 1	1 1 0 0 0 reg	3	reg16 bit NO. CL <- $\overline{\text{reg16 bit NO. CL}}$						
		mem16, CL	0 1 1 1	mod 0 0 0 mem	3 – 5	(mem16) bit NO. CL <- $\overline{\text{(mem16) bit NO. CL}}$						
		reg8, imm3	1 1 1 0	1 1 0 0 0 reg	4	reg8 bit NO. imm3 <- $\overline{\text{reg8 bit NO. imm3}}$						
		mem8, imm3	1 1 1 0	mod 0 0 0 mem	4 – 6	(mem8) bit NO. imm3 <- $\overline{\text{(mem8) bit NO. imm3}}$						
		reg16, imm4	1 1 1 1	1 1 0 0 0 reg	4	reg16 bit NO. imm3 <- $\overline{\text{reg16 bit NO. imm4}}$						
		mem16, imm4	1 1 1 1	mod 0 0 0 mem	4 – 6	(mem16) bit NO. imm3 <- $\overline{\text{(mem16) bit NO. imm4}}$						

Second byte [Note] Third byte [Note]

Note First byte = 0FH

	NOT1	CY	1 1 1 1 0 1 0 1		1	CY <- $\overline{\text{CY}}$		x				

Group	Mnemonic	Operand	Operation Code (Second byte[Note]) 7 6 5 4 3 2 1 0	Operation Code (Third byte[Note]) 7 6 5 4 3 2 1 0	Number of bytes	Operation	AC	CY	V	P	S	Z
Bit operation	CLR1	reg8, CL	0 0 0 1 0 0 1 0	1 1 0 0 0 reg	3	reg8 bit NO. CL <- 0						
		mem8, CL	0 0 1 0	mod 0 0 0 mem	3 – 5	(mem8) bit NO. CL <- 0						
		reg16, CL	0 0 1 1	1 1 0 0 0 reg	3	reg16 bit NO. CL <- 0						
		mem16, CL	0 0 1 1	mod 0 0 0 mem	3 – 5	(mem16) bit NO. CL <- 0						
		reg8, imm3	1 0 1 0	1 1 0 0 0 reg	4	reg8 bit NO. imm3 <- 0						
		mem8, imm3	1 0 1 0	mod 0 0 0 mem	4 – 6	(mem8) bit NO. imm3 <- 0						
		reg16, imm4	1 0 1 1	1 1 0 0 0 reg	4	reg16 bit NO. imm4 <- 0						
		mem16, imm4	1 0 1 1	mod 0 0 0 mem	4 – 6	(mem16) bit NO. imm4 <- 0						
	SET1	reg8, CL	0 1 0 0	1 1 0 0 0 reg	3	reg8 bit NO. CL <- 1						
		mem8, CL	0 1 0 0	mod 0 0 0 mem	3 – 5	(mem8) bit NO. CL <- 1						
		reg16, CL	0 1 0 1	1 1 0 0 0 reg	3	reg16 bit NO. CL <- 1						
		mem16, CL	0 1 0 1	mod 0 0 0 mem	3 – 5	(mem16) bit NO. CL <- 1						
		reg8, imm3	1 1 0 0	1 1 0 0 0 reg	4	reg8 bit NO. imm3 <- 1						
		mem8, imm3	1 1 0 0	mod 0 0 0 mem	4 – 6	(mem8) bit NO. imm3 <- 1						
		reg16, imm4	1 1 0 1	1 1 0 0 0 reg	4	reg16 bit NO. imm4 <- 1						
		mem16, imm4	1 1 0 1	mod 0 0 0 mem	4 – 6	(mem16) bit NO. imm4 <- 1						

Note First byte = 0FH

	CLR1	CY	1 1 1 1 1 0 0 0		1	CY <- 0		0				
		DIR	1 1 1 1 1 1 0 0		1	DIR <- 0						
	SET1	CY	1 1 1 1 1 0 0 1		1	CY <- 1		1				
		DIR	1 1 1 1 1 1 0 1		1	DIR <- 1						

Group	Mnemonic	Operand	Operation Code		Number of bytes	Operation	Flags					
			7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0			AC	CY	V	P	S	Z
Shift	SHL	reg, 1	1 1 0 1 0 0 0 W	1 1 1 0 0 reg	2	CY <- reg MSB, reg <- reg x reg2 When reg MSB ≠ CY, V<- 1 When reg MSB = CY, V<- 0	U	x	x	x	x	x
		mem, 1	1 1 0 1 0 0 0 W	mod 1 0 0 mem	2 – 4	CY <- (mem) MSB, (mem) <- (mem) x 2 When (mem) MSB ≠ CY, V <- 1 When (mem) MSB = CY, V <- 0	U	x	x	x	x	x
		reg, CL	1 1 0 1 0 0 1 W	1 1 1 0 0 reg	2	While temp <- CL and temp ≠ 0, the following operation is repeated: CY <- reg MSB, reg <- reg x 2 temp <- temp –1	U	x	U	x	x	x
		mem, CL	1 1 0 1 0 0 1 W	mod 1 0 0 mem	2 – 4	While temp <- CL and temp ≠ 0, the following operation is repeated: CY <- (mem) MSB, (mem) <- (mem) x 2 temp <- temp–1	U	x	U	x	x	x
		reg, imm8	1 1 0 0 0 0 0 W	1 1 1 0 0 reg	3	While temp <- imm8, temp ≠ 0, the following operation is repeated: CY <- reg MSB and reg <- reg x 2 temp <- temp–1	U	x	U	x	x	x
		mem, imm8	1 1 0 0 0 0 0 W	mod 1 0 0 mem	3 – 5	While temp <- imm8 and temp ≠ 0, the following operation is repeated: CY <- (mem) MSB, (mem) <- (mem) x 2 temp <- temp–1	U	x	U	x	x	x

Group	Mnemonic	Operand	Operation Code		Number of bytes	Operation	Flags					
			7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0			AC	CY	V	P	S	Z
Shift	SHR	reg, 1	1 1 0 1 0 0 0 W	1 1 1 0 1 reg	2	CY <- reg LSB, reg <- reg ÷ 2 reg MSB ≠ second MSB of reg: V <- 1 reg MSB = second MSB of reg: V <- 0	U	x	x	x	x	x
		mem, 1	1 1 0 1 0 0 0 W	mod 1 0 1 mem	2 – 4	CY <- (mem) MSB, (mem) <- (mem) ÷ 2 (mem) MSB ≠ second MSB of reg: V <- 1 (mem) MSB = second MSB of reg: V <- 0	U	x	x	x	x	x
		reg, CL	1 1 0 1 0 0 1 W	1 1 1 0 1 reg	2	While temp <- CL and temp ≠ 0, the following operation is repeated: CY <- reg LSB, reg <- reg ÷ 2 temp <- temp–1	U	x	U	x	x	x
		mem, CL	1 1 0 1 0 0 1 W	mod 1 0 1 mem	2 – 4	While temp <- CL and temp ≠ 0, the following operation is repeated: CY <- (mem) LSB, (mem) <- (mem) ÷ 2 temp <- temp–1	U	x	U	x	x	x
		reg, imm8	1 1 0 0 0 0 0 W	1 1 1 0 1 reg	3	While temp <- imm8 and temp ≠ 0, the following operation is repeated: CY <- reg LSB, reg <- reg ÷ 2 temp <- temp–1	U	x	U	x	x	x
		mem, imm8	1 1 0 0 0 0 0 W	mod 1 0 1 mem	3 – 5	While temp <- imm8 and temp ≠ 0, the following operation is repeated: CY <- (mem) LSB, (mem) <- (mem) ÷ 2 temp <- temp–1	U	x	U	x	x	x
	SHRA	reg, 1	1 1 0 1 0 0 0 W	1 1 1 1 1 reg	2	CY <- reg LSB, reg <- reg ÷ 2, V <- 0 Operand MSB does not change.	U	x	0	x	x	x
		mem, 1	1 1 0 1 0 0 0 W	mod 1 1 1 mem	2 – 4	CY <- (mem) LSB, (mem) <- (mem) ÷ 2, V <- 0 Operand MSB does not change.	U	x	0	x	x	x
		reg, CL	1 1 0 1 0 0 1 W	1 1 1 1 1 reg	2	While temp <- CL and temp ≠ 0, the following operation is repeated: CY <- reg LSB, reg <- reg ÷ 2 temp <-temp–1, operand MSB does not change.	U	x	U	x	x	x
		mem, CL	1 1 0 1 0 0 1 W	mod 1 1 1 mem	2 – 4	While temp <- CL and temp ≠ 0, the following operation is repeated: CY <- (mem) LSB, (mem) <- (mem) ÷ 2 temp <- temp–1, operand MSB does not change.	U	x	U	x	x	x
		reg, imm8	1 1 0 0 0 0 0 W	1 1 1 1 1 reg	3	While temp <- imm8, temp ≠ 0, the following operation is repeated: CY <- reg LSB, reg <- reg ÷ 2 temp <- temp–1, operand MSB does not change.	U	x	U	x	x	x
		mem, imm8	1 1 0 0 0 0 0 W	mod 1 1 1 mem	3 – 5	While temp <- imm8 and temp ≠ 0, the following operation is repeated: CY <- (mem) LSB, (mem) <- (mem) ÷ 2 temp <- temp–1, operand MSB does not change.	U	x	U	x	x	x

Group	Mnemonic	Operand	Operation Code 7 6 5 4 3 2 1 0	Operation Code 7 6 5 4 3 2 1 0	Number of bytes	Operation	AC	CY	V	P	S	Z
Rotate	ROL	reg, 1	1 1 0 1 0 0 0 W	1 1 0 0 0 reg	2	CY <- reg MSB, reg <- reg x reg2+CY reg MSB ≠ CY: V<-1 reg MSB = CY: V<-0		x	x			
		mem, 1	1 1 0 1 0 0 0 W	mod 0 0 0 mem	2 – 4	CY <- (mem) MSB, (mem) <- (mem) x 2+CY (mem) MSB ≠ CY: V<-1 (mem) MSB = CY: V<-0		x	x			
		reg, CL	1 1 0 1 0 0 1 W	1 1 0 0 0 reg	2	While temp <- CL and temp ≠ 0, the following operation is repeated: CY <- reg MSB, reg <- reg x 2+CY temp <- temp –1		x	U			
		mem, CL	1 1 0 1 0 0 1 W	mod 0 0 0 mem	2 – 4	While temp <- CL and temp ≠ 0, the following operation is repeated: CY <- (mem) MSB, (mem) <- (mem) x 2+CY temp <- temp –1		x	U			
		reg, imm8	1 1 0 0 0 0 0 W	1 1 0 0 0 reg	3	While temp <- imm8, temp ≠ 0, the following operation is repeated: CY <- reg MSB, reg <- reg x 2 +CY temp <- temp –1		x	U			
		mem, imm8	1 1 0 0 0 0 0 W	mod 0 0 0 mem	3 – 5	While temp <- imm8, temp ≠ 0, the following operation is repeated: CY <- (mem) MSB, (mem) <- (mem) x 2+CY temp <- temp –1		x	U			
	ROR	reg, 1	1 1 0 1 0 0 0 W	1 1 0 0 1 reg	2	CY <- reg LSB, reg <- reg ÷ 2 reg MSB <- CY reg MSB ≠ second MSB of reg: V<-1 reg MSB = second MSB of reg: V<-0		x	x			
		mem, 1	1 1 0 1 0 0 0 W	mod 0 0 1 mem	2 – 4	CY <- (mem) LSB, (mem) <- (mem) ÷ 2 (mem) MSB <- CY (mem) MSB ≠ second MSB of (mem): V<-1 (mem) MSB = second MSB of (mem): V<-0		x	x			
		reg, CL	1 1 0 1 0 0 1 W	1 1 0 0 1 reg	2	While temp <- CL, temp ≠ 0, the following operation is repeated: CY <- reg LSB, reg <- reg ÷ 2 reg MSB <- CY temp <- temp –1		x	U			
		mem, CL	1 1 0 1 0 0 1 W	mod 0 0 1 mem	2 – 4	While temp <- CL, temp ≠ 0, the following operation is repeated: CY <- (mem) LSB, (mem) <- (mem) ÷ 2 (mem) MSB <- CY temp <- temp–1		x	U			

Group	Mnemonic	Operand	Operation Code		Number of bytes	Operation	Flags						
			7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0			AC	CY	V	P	S	Z	
Rotate	ROR	reg, imm8	1 1 0 0 0 0 0 W	1 1 0 0 1 reg	3	While temp <- imm8, temp ≠ 0, the following operation is repeated: CY <- reg LSB, reg <- reg ÷ 2 reg MSB <- CY temp <- temp −1		x	U				★
		mem, imm8	1 1 0 0 0 0 0 W	mod 0 0 1 mem	3 – 5	While temp <- imm8, temp ≠ 0, the following operation is repeated: CY <- (mem) LSB, (mem) <- (mem) ÷ 2 (mem) MSB <- CY temp <- temp −1		x	U				★
	ROLC	reg, 1	1 1 0 1 0 0 0 W	1 1 0 1 0 reg	2	tmpcy <- CY, CY <- reg MSB reg <- reg x 2+tmpcy reg MSB ≠ CY: V<-1 reg MSB = CY: V<-0		x	x				
		mem, 1	1 1 0 1 0 0 0 W	mod 0 1 0 mem	2 – 4	tmpcy <- CY, CY <- (mem) MSB (mem) <- (mem) x 2+tmpcy (mem) MSB ≠ CY: V<-1 (mem) MSB = CY: V<-0		x	x				
		reg, CL	1 1 0 1 0 0 1 W	1 1 0 1 0 reg	2	While temp <- CL, temp ≠ 0, the following operation is repeated: tmpcy <- CY, CY <- reg MSB reg <- reg x 2+tmpcy temp <- temp −1		x	U				★
		mem, CL	1 1 0 1 0 0 1 W	mod 0 1 0 mem	2 – 4	While temp <- CL, temp ≠ 0, the following operation is repeated: tmpcy <- CY, CY <- (mem) MSB (mem) <- (mem) x 2+tmpcy temp <- temp -1		x	U				★
		reg, imm8	1 1 0 0 0 0 0 W	1 1 0 1 0 reg	3	While temp <- imm8, temp ≠ 0, the following operation is repeated: tmpcy <- CY, CY <- reg MSB reg <- reg x 2+tmpcy temp <- temp-1		x	U				★
		mem, imm8	1 1 0 0 0 0 0 W	mod 0 1 0 mem	3 – 5	While temp <- imm8, temp ≠ 0, the following operation is repeated: tmpcy <- CY, CY <- (mem) MSB (mem) <- (mem) x 2+tmpcy temp <- temp -1		x	U				★

Group	Mnemonic	Operand	Operation Code		Number of bytes	Operation	Flags					
			7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0			AC	CY	V	P	S	Z
Rotate	RORC	reg, 1	1 1 0 1 0 0 0 W	1 1 0 1 1 reg	2	tmpcy <- CY, CY <- reg LSB reg <- reg + 2 reg MSB <- tmpcy reg MSB ≠ second MSB of reg: V <- 1 reg MSB = second MSB of reg: V <- 0		x	x			
		mem, 1	1 1 0 1 0 0 0 W	mod 0 1 1 mem	2 – 4	tmpcy <- CY, CY <- (mem) LSB (mem) <- (mem) + 2 (mem) MSB <- tmpcy (mem) MSB ≠ second MSB of (mem): V <- 1 (mem) MSB = second MSB of (mem): V <- 0		x	x			
		reg, CL	1 1 0 1 0 0 1 W	1 1 0 1 1 reg	2	While temp <- CL, temp ≠ 0, the following operation is repeated: tmpcy <- CY, CY <- reg LSB reg <- reg + 2 reg MSB <- tmpcy temp <- temp –1		x	U			
		mem, CL	1 1 0 1 0 0 1 W	mod 0 1 1 mem	2 – 4	While temp <- CL, temp ≠ 0, the following operation is repeated: tmpcy <- CY, CY <- (mem) LSB (mem) <- (mem) + 2 (mem) MSB <- tmpcy temp <- temp –1		x	U			
		reg, imm8	1 1 0 0 0 0 0 W	1 1 0 1 1 reg	3	While temp <- imm8, temp ≠ 0, the following operation is repeated: tmpcy <- CY, CY <- reg LSB reg <- reg + 2 reg MSB <- tmpcy temp <- temp –1		x	U			
		mem, imm8	1 1 0 0 0 0 0 W	mod 0 1 1 mem	3 – 5	While temp <- imm8, temp ≠ 0, the following operation is repeated: tmpcy <- CY, CY <- (mem) LSB (mem) <- (mem) + 2 (mem) MSB <- tmpcy temp <- temp –1		x	U			

Group	Mnemonic	Operand	Operation Code 7 6 5 4 3 2 1 0	Operation Code 7 6 5 4 3 2 1 0	Number of bytes	Operation	Flags AC	CY	V	P	S	Z
Subroutine control	CALL	near-proc	1 1 1 0 1 0 0 0		3	(SP–1, SP–2) <- PC, SP <- SP–2 PC <- PC+disp						
		regptr16	1 1 1 1 1 1 1 1	1 1 0 1 0 reg	2	(SP–1, SP–2) <- PC, PC <- regptr16 SP <- SP–2						
		memptr16	1 1 1 1 1 1 1 1	mod 0 1 0 mem	2 – 4	(SP–1, SP–2) <- PC, SP <- SP–2 PC <- (memptr16)						
		far-proc	1 0 0 1 1 0 1 0		5	(SP–1, SP–2) <- PS, (SP–3, SP–4) <- PC SP <- SP–4 PS <- seg, PC <- offset						
		memptr32	1 1 1 1 1 1 1 1	mod 0 1 1 mem	2 – 4	(SP–1, SP–2) <- PS, (SP–3, SP–4) <- PC SP <- SP–4 PS <- (memptr32+2), PC <- (memptr32)						
	RET		1 1 0 0 0 0 1 1		1	PC <- (SP+1, SP) SP <- SP+2						
		pop-value	1 1 0 0 0 0 1 0		3	PC <- (SP+1, SP) SP <- SP+2, SP <- SP+pop-value						
			1 1 0 0 1 0 1 1		1	PC <- (SP+1, SP) PS <- (SP+3, SP+2) SP <- SP+4						
		pop-value	1 1 0 0 1 0 1 0		3	PC <- (SP+1, SP) PS <- (SP+3, SP+2) SP <- SP+4, SP <- SP+pop-value						
Stack operation	PUSH	mem16	1 1 1 1 1 1 1 1	mod 1 1 0 mem	2 – 4	(SP–1, SP–2) <- (mem16) SP <- SP–2						
		reg16	0 1 0 1 0 reg		1	(SP–1, SP–2) <- reg16 SP <- SP–2						
		sreg	0 0 0 sreg 1 1 0		1	(SP–1, SP–2) <- sreg SP <- SP–2						
		PSW	1 0 0 1 1 1 0 0		1	(SP–1, SP–2) <- PSW SP <- SP–2						
		R	0 1 1 0 0 0 0 0		1	Push registers on the stack						
		imm 8	0 1 1 0 1 0 1 0		2	(SP–1, SP–2) <- imm8 sign extension SP <- SP–2						
		imm 16	0 1 1 0 1 0 0 0		3	(SP–1, SP–2) <- imm16 SP <- SP–2						

Group	Mnemonic	Operand	Operation Code 7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	Number of bytes	Operation	Flags AC	CY	V	P	S	Z	
Stack operation	POP	mem16	1 0 0 0 1 1 1 1	mod 0 0 0 mem	2 – 4	SP <- SP+2 (mem16) <- (SP–1, SP–2)							★
		reg16	0 1 0 1 1 reg		1	SP <- SP+2 reg16 <- (SP–1, SP–2)							★
		sreg	0 0 0 sreg 1 1 1		1	SP <- SP+2 sreg <- (SP–1, SP–2) sreg: SS, DS0, DS1							★
		PSW	1 0 0 1 1 1 0 1		1	SP <- SP+2 PSW <- (SP–1, SP–2)	R	R	R	R	R	R	★
		R	0 1 1 0 0 0 0 1		1	Pop registers from the stack							
	PREPARE	imm16, imm8	1 1 0 0 1 0 0 0		4	Prepare New Stack Frame							
	DISPOSE		1 1 0 0 1 0 0 1		1	Dispose of Stack Frame							
Branch	BR	near-label	1 1 1 0 1 0 0 1		3	PC <- PC+disp							
		short-label	1 1 1 0 1 0 1 1		2	PC <- PC+ext–disp8							
		regptr16	1 1 1 1 1 1 1 1	1 1 1 0 0 reg	2	PC <- regptr16							
		memptr16	1 1 1 1 1 1 1 1	mod 1 0 0 mem	2 – 4	PC <- (memptr16)							
		far-label	1 1 1 0 1 0 1 0		5	PS <- set PC <- offset							
		memptr32	1 1 1 1 1 1 1 1	mod 1 0 1 mem	2 – 4	PS <- (memptr32+2) PC <- (memptr32)							

Group	Mnemonic	Operand	Operation Code 7 6 5 4 3 2 1 0	Operation Code 7 6 5 4 3 2 1 0	Number of bytes	Operation		Flags AC	CY	V	P	S	Z
Conditional branch	BV	short-label	0 1 1 1 0 0 0 0		2	if V = 1	PC <- PC+ext–disp8						
	BNV	short-label	0 0 0 1		2	if V = 0	PC <- PC+ext–disp8						
	BC BL	short-label	0 0 1 0		2	if CY = 1	PC <- PC+ext–disp8						
	BNC BNL	short-label	0 0 1 1		2	if CY = 0	PC <- PC+ext–disp8						
	BE BZ	short-label	0 1 0 0		2	if Z = 1	PC <- PC+ext–disp8						
	BNE BNZ	short-label	0 1 0 1		2	if Z = 0	PC <- PC+ext–disp8						
	BNH	short-label	0 1 1 0		2	if CY ∨ Z = 1	PC <- PC+ext–disp8						
	BH	short-label	0 1 1 1		2	if CY ∨ Z = 0	PC <- PC+ext–disp8						
	BN	short-label	1 0 0 0		2	if S = 1	PC <- PC+ext–disp8						
	BP	short-label	1 0 0 1		2	if S = 0	PC <- PC+ext–disp8						
	BPE	short-label	1 0 1 0		2	if P = 1	PC <- PC+ext–disp8						
	BPO	short-label	1 0 1 1		2	if P = 0	PC <- PC+ext–disp8						
	BLT	short-label	1 1 0 0		2	if S ⊻ V = 1	PC <- PC+ext–disp8						
	BGE	short-label	1 1 0 1		2	if S ⊻ V = 0	PC <- PC+ext–disp8						
	BLE	short-label	1 1 1 0		2	if (S ⊻ V) V Z = 1	PC <- PC+ext–disp8						
	BGT	short-label	1 1 1 1		2	if (S ⊻ V) V Z = 0	PC <- PC+ext–disp8						
	DBNZNE	short-label	1 1 1 0 0 0 0 0		2	CW = CW–1 if Z = 0 and CW ≠ 0	PC <- PC+ext–disp8						
	DBNZE	short-label	0 0 0 1		2	CW = CW–1 if Z = 1 and CW ≠ 0	PC <- PC+ext–disp8						
	DBNZ	short-label	0 0 1 0		2	CW = CW–1 if CW ≠ 0	PC <- PC+ext–disp8						
	BCWZ	short-label	0 0 1 1		2	if CW = 0	PC <- PC+ext–disp8						
	BTCLR[Note]	sfr, imm3, short-label	0 0 0 0 1 1 1 1	1 0 0 1 1 1 0 0	5	When (sfr) bit No. imm3 = 1, PC <- PC+ext-disp8, (sfr) bit No. imm3 <- 0							

Note New added instruction from the μPD70108/μPD70116

Group	Mnemonic	Operand	Operation Code		Number of bytes	Operation	Flags					
			7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0			AC	CY	V	P	S	Z
Interrupt	BRK	3	1 1 0 0 1 1 0 0		1	(SP–1, SP–2) <- PSW, (SP–3, SP–4) <- PS, (SP–5, SP–6) <- PC, SP <- SP–6 IE <- 0, BRK <- 0 PS <- (15, 14), PC <- (13, 12)						
		imm8 (≠3)	1 1 0 0 1 1 0 1		2	(SP–1, SP–2) <- PSW, (SP–3, SP–4) <- PS, (SP–5, SP–6) <- PC, SP <- SP–6 IE <- 0, BRK <- 0 PS <- (n x 4+3, n x 4+2), PC <- (n x 4+1, n x 4) n = imm8						
	BRKV		1 1 0 0 1 1 1 0		1	When V = 1, (SP–1, SP–2) <- PSW (SP–3, SP–4) <- PS, (SP–5, SP–6) <- PC, SP <- SP–6 IE <- 0, BRK <- 0 PS <- (19, 18), PC <- (17, 16)						
	RETI		1 1 0 0 1 1 1 1		1	PC <- (SP+1, SP), PS <- (SP+3, SP+2), PSW <- (SP+5, SP+4), SP <- SP+6	R	R	R	R	R	R
	RETRBI[Note]		0 0 0 0 1 1 1 1	1 0 0 1 0 0 0 1	2	PC <- Save PC, PSW <- Save PSW	R	R	R	R	R	R
	FINT[Note]		0 0 0 0 1 1 1 1	1 0 0 1 0 0 1 0	2	Inform the internal interrupt controller in CPU that the interrupt service routine terminates.						
	CHIKIND	reg16, mem32	0 1 1 0 0 0 1 0	mod reg mem	2 – 4	When (mem32) > reg16 or (mem32+2) < reg16, (SP–1, SP–2) <- PSW, (SP–3, SP–4) <- PS, (SP–5, SP–6) <- PC, SP <- SP–6 IE <- 0, BRK <- 0 PS <- (23, 22), PC <- (21, 20)						
Register bank switch	BRKCS[Note]	reg16	0 0 0 0 1 1 1 1	0 0 1 0 1 1 0 1	3	RB2-0 <- reg16 lower 3 bits, IE <- 0, BRK <- 0, Save PSW <- PSW, Save PC <- PC, PC <- Vector PC						
			1 1 0 0 0 reg									
	TSKSW[Note]	reg16	0 0 0 0 1 1 1 1	1 0 0 1 0 1 0 0	3	RB2-0 <- reg16 lower 3 bits, Old register bank Save PSW, Save PC <- PSW, PC, PSW, PC <- New register bank Save PSW, Save PC	x	x	x	x	x	x
			1 1 1 1 1 reg									

Note New added instruction from the μPD70108/μPD70116

Group	Mnemonic	Operand	Operation Code 7 6 5 4 3 2 1 0	Operation Code 7 6 5 4 3 2 1 0	Number of bytes	Operation	Flags AC	CY	V	P	S	Z
CPU control	HALT		1 1 1 1 0 1 0 0		1	CPU Halt						
	STOP[Note 2]		0 0 0 0 1 1 1 1	1 0 0 1 1 1 1 0	2	CPU Stop						
	POLL		1 0 0 1 1 0 1 1		1	Poll and wait						
	DI		1 1 1 1 1 0 1 0		1	IE <- 0						
	EI		1 1 1 1 1 0 1 1		1	IE <- 1						
	BUSLOCK		1 1 1 1 0 0 0 0		1	Bus Lock Prefix						
	FPO1 [Note 3]	fp-op	1 1 0 1 1 X X X	1 1 Y Y Y Z Z Z	2	No Operation						
		fp-op, mem	1 1 0 1 1 X X X	mod Y Y Y mem	2 – 4	data bus <- (mem)						
	FPO2 [Note 3]	fp-op	0 1 1 0 0 1 1 X	1 1 Y Y Y Z Z Z	2	No Operation						
		fp-op, mem	0 1 1 0 0 1 1 X	mod Y Y Y mem	2 – 4	data bus <- (mem)						
	NOP		1 0 0 1 0 0 0 0		1	No Operation						
	Note 1		0 0 1 sreg 1 1 0		1	Segment override prefix						

Notes 1. DS0:, DS1:, PS:, SS:

2. New added instruction from the μPD70108/μPD70116

3. In the μPD70330, the instruction is not executed and an interrupt occurs.

Table 2-8. The Number of Clocks

Group	Mnemonic	Operand	Byte processing: On-chip RAM access enable	Byte processing: On-chip RAM access disable	Word processing: On-chip RAM access enable	Word processing: On-chip RAM access disable
Data transfer	MOV	reg, reg	2	2	2	2
		mem, reg	EA+5+T	EA+2	EA+5+T	EA+2
		reg, mem	EA+7+T	EA+7+T	EA+7+T	EA+7+T
		mem, imm	EA+6+T	EA+6+T	EA+6+T	EA+6+T
		reg, imm	5	5	6	6
		acc, dmem	10+T	10+T	10+T	10+T
		dmem, acc	8+T	5	8+T	5
		sreg, reg16	–	–	4	4
		sreg, mem16	–	–	EA+9+T	EA+9+T
		reg16, sreg	–	–	3	3
		mem16, sreg	–	–	EA+6+T	EA+3
		DS0, reg16, mem32	–	–	EA+17+2·T	EA+17+2·T
		DS1, reg16, mem32	–	–	EA+17+2·T	EA+17+2·T
		AH, PSW	2	2	–	–
		PSW, AH	3	3	–	–
	LDEA	re16, mem16	–	–	EA+2	EA+2
	TRANS	src-table	11+T	11+T	–	–
	XCH	reg, reg	3	3	3	3
		mem, reg/reg, mem	EA+12+2·T	EA+9+T	EA+12+2·T	EA+9+T
		AW, reg16/reg16, AW	–	–	4	4
	MOVSPA		–	–	16	16
	MOVSPB	reg16	–	–	11	11

Group	Mnemonic	Operand	Byte processing		Word processing	
			On-chip RAM access enable	On-chip RAM access disable	On-chip RAM access enable	On-chip RAM access disable
Repeat prefix	REPC		2	2	2	2
	REPNC		2	2	2	2
	REP/REPE/ REPZ		2	2	2	2
	REPNE/ REPNZ		2	2	2	2
Primitive block transfer	MOVBK Note 1	dst-block, src-block	22+2·T	17+T	22+2·T	19+T
			16+(18+2·T)·n	16+(13+T)·n	16+(18+2·T)·n	16+(10+T)·n
	CMPBK Note1	dst-block, src-block	25+2·T	21+2·T	25+2·T	19+2·T
			16+(23+2·T)n	16+(23+2·T)·n	16+(23+2·T)·n	16+(23+2·T)·n
	CMPM Note 1	dst-block, src-block	18+T	18+T	19+2·T	19+2·T
			16+(16+T)·n	16+(16+T)·n	16+(16+T)·n	16+(16+T)·n
	LDM Note 1	dst-block, src-block	13+T	13+T	13+T	13+T
			16+(11+T)·n	16+(11+T)·n	16+(11+T)·n	16+(11+T)·n
	STM Note 1	dst-block, src-block	13+T	10	13+T	10
			16+(9+T)·n	16+(7+T)·n	16+(9+T)·n	16+(5+T)·n
Bit field operation	INS	reg8, reg8	–	–	63 to 155	63 to 155
		reg8, imm4	–	–	64 to 156	64 to 156
	EXT	reg8, reg8	–	–	41 to 121	41 to 121
		reg8, imm8	–	–	42 to 122	42 to 122
Input/ output	IN Note 2	acc, imm8	15+T	15+T	15+T	15+T
		acc, DW	14+T	14+T	14+T	14+T
	OUT Note 2	imm8, acc	11+T	11+T	9+T	9+T
		DW, acc	10+T	10+T	8+T	8+T

Notes 1. $n \geq 1$

2. IBRK = 1

6427525 0062965 706

Group	Mnemonic	Operand	Byte processing: On-chip RAM access enable	Byte processing: On-chip RAM access disable	Word processing: On-chip RAM access enable	Word processing: On-chip RAM access disable
Primitive I/O	INM [Note]	dst-block, DW	21+2·T	19+2·T	19+2·T	15+2·T
			18+(15+2·T)·n	18+(13+2·T)·n	18+(13+2·T)·n	18+(9+2·T)·n
	OUTM [Note]	DW, dst-block	21+2·T	19+2·T	19+2·T	15+2·T
			18+(15+2·T)·n	18+(13+2·T)·n	18+(13+2·T)·n	18+(9+2·T)·n
Addition and subtraction	ADD	reg, reg	2	2	2	2
		mem, reg	EA+10+2·T	EA+7+T	EA+10+2·T	EA+7+T
		reg, mem	EA+7+T	EA+7+T	EA+7+T	EA+7+T
		reg, imm	5	5	6	6
		mem, imm	EA+11+2·T	EA+9+2·T	EA+12+2·T	EA+8+2·T
		acc, imm	5	5	6	6
	ADDC	reg, reg	2	2	2	2
		mem, reg	EA+10+2·T	EA+7+T	EA+10+2·T	EA+7+T
		reg, mem	EA+7+T	EA+7+T	EA+7+T	EA+7+T
		reg, imm	5	5	6	6
		mem, imm	EA+11+2·T	EA+9+2·T	EA+12+2·T	EA+8+2·T
		acc, imm	5	5	6	6
	SUB	reg, reg	2	2	2	2
		mem, reg	EA+10+2·T	EA+7+T	EA+10+2·T	EA+7+T
		reg, mem	EA+7+T	EA+7+T	EA+7+T	EA+7+T
		reg, imm	5	5	6	6
		mem, imm	EA+11+2·T	EA+9+2·T	EA+12+2·T	EA+8+2·T
		acc, imm	5	5	6	6

Note n ≥ 1, IBRK = 1

Group	Mnemonic	Operand	Byte processing		Word processing	
			On-chip RAM access enable	On-chip RAM access disable	On-chip RAM access enable	On-chip RAM access disable
Addition and subtraction	SUBC	reg, reg	2	2	2	2
		mem, reg	EA+10+2·T	EA+7+T	EA+10+2·T	EA+7+T
		reg, mem	EA+7+T	EA+7+T	EA+7+T	EA+7+T
		reg, imm	5	5	6	6
		mem, imm	EA+11+2·T	EA+9+2·T	EA+12+2·T	EA+8+2·T
		acc, imm	5	5	6	6
BCD operation	ADD4S Note		22+(30+3·T)·n	22+(28+3·T)·n	–	–
	SUB4S Note		22+(30+3·T)·n	22+(28+3·T)·n	–	–
	CMP4S Note		22+(25+2·T)·n	22+(25+2·T)·n	–	–
	ROL4	reg8	17	17	–	–
		mem8	EA+20+2·T	EA+18+2·T	–	–
	ROR4	reg8	21	21	–	–
		mem8	EA+26+2·T	EA+24+2·T	–	–
Increment and decrement	INC	reg8	5	5	–	–
		mem8	EA+13+2·T	EA+11+2·T	EA+13+2·T	EA+9+2·T
		reg16	–	–	2	2
	DEC	reg8	5	5	–	–
		mem8	EA+13+2·T	EA+11+2·T	EA+13+2·T	EA+9+2·T
		reg16	–	–	2	2
Multi-plication	MULU	reg8	24	24	–	–
		mem8	EA+27+T	EA+27+T	–	–
		reg16	–	–	32	32
		mem16	–	–	EA+33+T	EA+33+T

Note n: The number of BCD digit x 1/2

Group	Mnemonic	Operand	Byte processing: On-chip RAM access enable	Byte processing: On-chip RAM access disable	Word processing: On-chip RAM access enable	Word processing: On-chip RAM access disable
Multi-plication	MUL	reg8	31 to 40	31 to 40	–	–
		mem8	EA+34+T to EA+43+T	EA+34+T to EA+43+T	–	–
		reg16	–	–	39 to 48	39 to 48
		mem16	EA+42+T to EA+51+T	EA+42+T to EA+51+T	–	–
		reg16,(reg16,) imm8	–	–	39 to 49	39 to 49
		reg16, mem16, imm8	EA+42+T to EA+52+T	EA+42+T to EA+52+T	–	–
		reg16,(reg16,) imm16	–	–	40 to 50	40 to 50
		reg16, mem16, imm16	EA+43+T to EA+53+T	EA+43+T to EA+53+T	–	–
Unsigned division	DIVU Note	reg8	31	31	–	–
		mem8	EA+34+T	EA+34+T	–	–
		reg16	–	–	39	39
		mem16	–	–	EA+43+2·T	EA+43+2·T
Signed division	DIV Note	reg8	46 to 56	46 to 56	–	–
		mem8	EA+49+T to EA+59+T	EA+49+T to EA+59+T	–	–
		reg16	–	–	54 to 64	54 to 64
		mem16	EA+57+T to EA+67+T	EA+57+T to EA+67+T	–	–
BCD adjustment	ADJBA		17	17	–	–
	ADJ4A		9	9	–	–
	ADJBS		17	17	–	–
	ADJ4S		9	9	–	–
Encode/ decode	CVTBD		19	19	–	–
	CVTDB		20	20	–	–
	CVTBW		3	3	–	–
	CVTWL		–	–	8	8

★ **Note** When a divide error occurs, add 58 + 5 W to 62 + 5 W (W: Wait number) to the normal clock number.

Group	Mnemonic	Operand	Byte processing: On-chip RAM access enable	Byte processing: On-chip RAM access disable	Word processing: On-chip RAM access enable	Word processing: On-chip RAM access disable
Comparison	CMP	reg, reg	2	2	2	2
		mem, reg	EA+7+T	EA+7+T	EA+7+T	EA+7+T
		reg, mem	EA+7+T	EA+7+T	EA+7+T	EA+7+T
		reg, imm	5	5	6	6
		mem, imm	EA+8+T	EA+8+T	EA+9+T	EA+9+T
		acc, imm	5	5	6	6
Complement	NOT	reg	5	5	5	5
		mem	EA+13+2·T	EA+10+T	EA+13+2·T	EA+10+T
	NEG	reg	5	5	5	5
		mem	EA+13+2·T	EA+10+T	EA+13+2·T	EA+10+T
Logical	TEST	reg, reg	4	4	4	4
		mem, reg/reg, mem	EA+12+T	EA+12+T	EA+11+2·T	EA+11+2·T
		reg, imm	7	7	8	8
		mem, imm	EA+9+T	EA+9+T	EA+10+T	EA+10+T
		acc, imm	5	5	6	6
	AND	reg, reg	2	2	2	2
		mem, reg	EA+10+2·T	EA+7+T	EA+10+2·T	EA+7+T
		reg, mem	EA+7+T	EA+7+T	EA+7+T	EA+7+T
		reg, imm	5	5	6	6
		mem, imm	EA+11+2·T	EA+9+2·T	EA+12+2·T	EA+8+2·T
		acc, imm	5	5	6	6
	OR	reg, reg	2	2	2	2
		mem, reg	EA+10+2·T	EA+7+T	EA+10+2·T	EA+7+T
		reg, mem	EA+7+T	EA+7+T	EA+7+T	EA+7+T

Group	Mnemonic	Operand	Byte processing: On-chip RAM access enable	Byte processing: On-chip RAM access disable	Word processing: On-chip RAM access enable	Word processing: On-chip RAM access disable
Logical	OR	reg, imm	5	5	6	6
		mem, imm	EA+11+2·T	EA+9+2·T	EA+12+2·T	EA+8+2·T
		acc, imm	5	5	6	6
	XOR	reg, reg	2	2	2	2
		mem, reg	EA+10+2·T	EA+7+T	EA+10+2·T	EA+7+T
		reg, mem	EA+7+T	EA+7+T	EA+7+T	EA+7+T
		reg, imm	5	5	6	6
		mem, imm	EA+11+2·T	EA+9+2·T	EA+12+2·T	EA+8+2·T
		acc, imm	5	5	6	6
Bit manipulation	TEST1	reg8, CL	7	7	–	–
		mem8, CL	EA+12+T	EA+12+T	–	–
		reg16, CL	–	–	7	7
		mem16, CL	–	–	EA+12+T	EA+12+T
		reg8, imm3	6	6	–	–
		mem8, imm3	EA+9+T	EA+9+T	–	–
		reg16, imm4	–	–	6	6
		mem16, imm4	–	–	EA+9+T	EA+9+T
	NOT1	reg8, CL	7	7	–	–
		mem8, CL	EA+15+2·T	EA+12+T	–	–
		reg16, CL	–	–	7	7
		mem16, CL	–	–	EA+15+2·T	EA+12+T
		reg8, imm3	6	6	–	–
		mem8, imm3	EA+12+2·T	EA+9+T	–	–

Group	Mnemonic	Operand	Byte processing		Word processing	
			On-chip RAM access enable	On-chip RAM access disable	On-chip RAM access enable	On-chip RAM access disable
Bit manipulation	NOT1	reg16, imm4	–	–	6	6
		mem16, imm4	–	–	EA+12+2·T	EA+9+T
	NOT1	CY	2	2	2	2
	CLR1	reg8, CL	8	8	–	–
		mem8, CL	EA+16+2·T	EA+13+T	–	–
		reg16, CL	–	–	8	8
		mem16, CL	–	–	EA+16+2·T	EA+13+T
		reg8, imm3	7	7	–	–
		mem8, imm3	EA+13+2·T	EA+10+T	–	–
		reg16, imm4	–	–	7	7
		mem16, imm4	–	–	EA+13+2·T	EA+9+T
	SET1	reg8, CL	7	7	–	–
		mem8, CL	EA+15+2·T	EA+12+T	–	–
		reg16, CL	–	–	7	7
		mem16, CL	–	–	EA+15+2·T	EA+12+T
		reg8, imm3	6	6	–	–
		mem8, imm3	EA+12+2·T	EA+9+T	–	–
		reg16, imm4	–	–	6	6
		mem16, imm4	–	–	EA+12+2·T	EA+9+T
	CLR1	CY	2	2	2	2
		DIR	2	2	2	2
	SET1	CY	2	2	2	2
		DIR	2	2	2	2

Group	Mnemonic	Operand	Byte processing		Word processing	
			On-chip RAM access enable	On-chip RAM access disable	On-chip RAM access enable	On-chip RAM access disable
Shift	SHL Note	reg, 1	8	8	8	8
		mem, 1	EA+16+2·T	EA+13+T	EA+16+2·T	EA+13+T
		reg, CL	11+2·n	11+2·n	11+2·n	11+2·n
		mem, CL	EA+19+2·T+2·n	EA+16+T+2·n	EA+19+2·T+2·n	EA+16+T+2·n
		reg, imm8	9+2·n	9+2·n	9+2·n	9+2·n
		mem, imm8	EA+15+2·T+2·n	EA+12+T+2·n	EA+15+2·T+2·n	EA+12+T+2·n
	SHR Note	reg, 1	8	8	8	8
		mem, 1	EA+16+2·T	EA+13+T	EA+16+2·T	EA+13+T
		reg, CL	11+2·n	11+2·n	11+2·n	11+2·n
		mem, CL	EA+19+2·T+2·n	EA+16+T+2·n	EA+19+2·T+2·n	EA+16+T+2·n
		reg, imm8	9+2·n	9+2·n	9+2·n	9+2·n
		mem, imm8	EA+15+2·T+2·n	EA+12+T+2·n	EA+15+2·T+2·n	EA+12+T+2·n
	SHRA Note	reg, 1	8	8	8	8
		mem, 1	EA+16+2·T	EA+13+T	EA+16+2·T	EA+13+T
		reg, CL	11+2·n	11+2·n	11+2·n	11+2·n
		mem, CL	EA+19+2·T+2·n	EA+16+T+2·n	EA+19+2·T+2·n	EA+16+T+2·n
		reg, imm8	9+2·n	9+2·n	9+2·n	9+2·n
		mem, imm8	EA+15+2·T+2·n	EA+12+T+2·n	EA+15+2·T+2·n	EA+12+T+2·n
Rotate	ROL Note	reg, 1	8	8	8	8
		mem, 1	EA+16+2·T	EA+13+T	EA+16+2·T	EA+13+T
		reg, CL	11+2·n	11+2·n	11+2·n	11+2·n
		mem, CL	EA+19+2·T+2·n	EA+16+T+2·n	EA+19+2·T+2·n	EA+16+T+2·n
		reg, imm8	9+2·n	9+2·n	9+2·n	9+2·n
		mem, imm8	EA+15+2·T+2·n	EA+12+T+2·n	EA+15+2·T+2·n	EA+12+T+2·n

Note n: The number of shift

Group	Mnemonic	Operand	Byte processing: On-chip RAM access enable	Byte processing: On-chip RAM access disable	Word processing: On-chip RAM access enable	Word processing: On-chip RAM access disable
Rotate	ROR Note	reg, 1	8	8	8	8
		mem, 1	EA+16+2·T	EA+13+T	EA+16+2·T	EA+13+T
		reg, CL	11+2·n	11+2·n	11+2·n	11+2·n
		mem, CL	EA+19+2·T+2·n	EA+16+T+2·n	EA+19+2·T+2·n	EA+16+T+2·n
		reg, imm8	9+2·n	9+2·n	9+2·n	9+2·n
		mem, imm8	EA+15+2·T+2·n	EA+12+T+2·n	EA+15+2·T+2·n	EA+12+T+2·n
	ROLC Note	reg, 1	8	8	8	8
		mem, 1	EA+16+2·T	EA+13+T	EA+16+2·T	EA+13+T
		reg, CL	11+2·n	11+2·n	11+2·n	11+2·n
		mem, CL	EA+19+2·T+2·n	EA+16+T+2·n	EA+19+2·T+2·n	EA+16+T+2·n
		reg, imm8	9+2·n	9+2·n	9+2·n	9+2·n
		mem, imm8	EA+15+2·T+2·n	EA+12+T+2·n	EA+15+2·T+2·n	EA+12+T+2·n
	RORC Note	reg, 1	8	8	8	8
		mem, 1	EA+16+2·T	EA+13+T	EA+16+2·T	EA+13+T
		reg, CL	11+2·n	11+2·n	11+2·n	11+2·n
		mem, CL	EA+19+2·T+2·n	EA+16+T+2·n	EA+19+2·T+2·n	EA+16+T+2·n
		reg, imm8	9+2·n	9+2·n	9+2·n	9+2·n
		mem, imm8	EA+15+2·T+2·n	EA+12+T+2·n	EA+15+2·T+2·n	EA+12+T+2·n
Subroutine control	CALL	near-proc	–	–	21+T	17+T
		regptr16	–	–	21+T	17+T
		memptr16	–	–	EA+24+2·T	EA+22+2·T
		far-proc	–	–	36+2·T	32+2·T
		memptr32	–	–	EA+32+4·T	EA+20+4·T

Note n: The number of shift

Group	Mnemonic	Operand	Byte processing		Word processing	
			On-chip RAM access enable	On-chip RAM access disable	On-chip RAM access enable	On-chip RAM access disable
Subroutine control	RET		–	–	19+T	19+T
		pop-value	–	–	19+T	19+T
			–	–	27+2·T	27+2·T
		pop-value	–	–	28+2·T	28+2·T
Stack manipulation	PUSH	mem16	–	–	EA+16+2·T	EA+12+2·T
		reg16	–	–	13+T	9+T
		sreg	–	–	10+T	7
		PSW	–	–	9+T	6
		R	–	–	74+8·T	50
		imm8	–	–	12+T	9
		imm16	–	–	13+T	10
	POP	mem16	–	–	EA+14+2·T	EA+11+T
		reg16	–	–	11+T	11+T
		sreg	–	–	12+T	12+T
		PSW	–	–	13+T	13+T
		R	–	–	74+8·T	58
	PREPARE	imm16, imm8	when imm8 = 0, 26+T when imm8 = 1, 37+2·T when imm8 = n, n > 1, 44+19(n–1)2·n·T			
	DISPOSE		–	–	11+T	11+T
Branch	BR	near-label	–	–	12	12
		short-label	–	–	12	12
		regptr16	–	–	13	13
		memptr16	–	–	EA+16+T	EA+16+T

Group	Mnemonic	Operand	Byte processing		Word processing	
			On-chip RAM access enable	On-chip RAM access disable	On-chip RAM access enable	On-chip RAM access disable
Branch	BR	far-label	–	–	15	15
		memptr32	–	–	EA+23+2·T	EA+23+2·T
Conditional branch	BV	short-label	–	–	15/8	15/8
	BVN	short-label	–	–	15/8	15/8
	BC/BL	short-label	–	–	15/8	15/8
	BNC/BNL	short-label	–	–	15/8	15/8
	BE/BZ	short-label	–	–	15/8	15/8
	BNE/BNZ	short-label	–	–	15/8	15/8
	BNH	short-label	–	–	15/8	15/8
	BH	short-label	–	–	15/8	15/8
	BN	short-label	–	–	15/8	15/8
	BP	short-label	–	–	15/8	15/8
	BPE	short-label	–	–	15/8	15/8
	BPO	short-label	–	–	15/8	15/8
	BLT	short-label	–	–	15/8	15/8
	BGE	short-label	–	–	15/8	15/8
	BLE	short-label	–	–	15/8	15/8
	BGT	short-label	–	–	15/8	15/8
	DBNZNE	short-label	–	–	17/8	17/8
	DBNZE	short-label	–	–	17/8	17/8
	BCWZ	short-label	–	–	15/8	15/8
	BTCLR	sfr, imm3 short-label	29	29	–	–
Interrupt	BRK	3	–	–	50+5·T	38+5·T
		imm8 (≠3)	–	–	51+5·T	39+5·T

Group	Mnemonic	Operand	Byte processing: On-chip RAM access enable	Byte processing: On-chip RAM access disable	Word processing: On-chip RAM access enable	Word processing: On-chip RAM access disable
Interrupt	BRKV		–	–	50+5·T	38+5·T
	RETI		–	–	40+3·T	34+T
	RETRBI		–	–	12	12
	FINT		2	2	2	2
	CHKIND	reg16, mem32	–	–	EA+24+2·T	EA+24+2·T
	BRKCS	reg16	–	–	15	15
	TSKSW	reg16	–	–	20	20
CPU control	HALT		–	–	–	–
	STOP		–	–	–	–
	POLL		–	–	–	–
	DI		4	4	4	4
	EI		12	12	12	12
	BUSLOCK		2	2	2	2
	FPO1	fp-op	–	–	55+5·T	43+5·T
		fp-op, mem	–	–	55+5·T	43+5·T
	FPO2	fp-op	–	–	55+5·T	43+5·T
		fp-op, mem	–	–	55+5·T	43+5·T
	NOP		4	4	4	4
Note			2	2	2	2

Note Segment override prefix (DS0:, DS1:, PS:, SS:)

3. ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS (T_A = 25 °C)

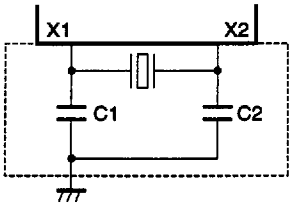
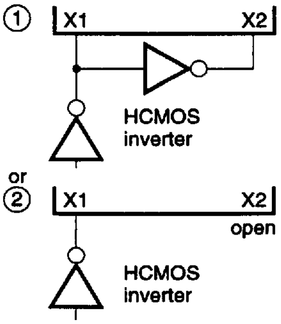
PARAMETER	SYMBOL	TEST CONDITION	RATINGS	UNIT
Supply Voltage	V_{DD}		–0.5 to +7.0	V
Input Voltage	V_{TH}		–0.5 to V_{DD} + 0.5	V
	V_I		–0.5 to V_{DD} + 0.5	V
Output Voltage	V_O		–0.5 to V_{DD} + 0.5	V
Output Low Current	I_{OL}	Each output pin	4.0	mA
		Total	50	mA
Output High Current	I_{OH}	Each output pin	–2.0	mA
		Total	–20	mA
Operating Ambient Temperature	T_A		–40 to +85	°C
Storage Temperature	T_{stg}		–65 to +150	°C

Cautions 1. Do not connect output (and bidirectional) pins each other. Do not connect output (or bidirectional) pins directly to the V_{DD}, V_{CC}, or GND line. However, open drain pin and open collector pins can be directly connected to V_{DD}, V_{CC}, or GND line. If timing design is made so that no signal conflict occurs, three-state pins can also be connected directly to three-state pins of external circuit.

2. Exposure to Absolute Maximum Ratings for extended periods may affect device reliability; exceeding the ratings could cause permanent damage. The parameters apply independently. The device should be operated within the limits specified under DC and AC Characteristics.

OSCILLATOR CHARACTERISTICS

(T_A = –40 °C to +85 °C, V_{DD} = +5.0 V ± 10%, V_{SS} = 0 V, 0 V ≤ V_{TH} ≤ V_{DD} + 0.1 V)

CONFIGURATION	RECOMMENDED CIRCUIT	PARAMETER	MIN.	MAX.	UNIT
Internal Oscillator (Ceramic or Crystal Resonator)		Oscillation frequency (f_{XX})	4	16	MHz
External Clock		X1 input frequency (f_X)	4	16	MHz
		X1 rise, fall time (t_{XR}, t_{XF})	0	20	ns
		X1 input high, low level width (t_{WXH}, t_{WXL})	20	250	ns

Cautions 1. Mount the capacitors and crystal or ceramic resonator as close to pins X1 and X2 as possible.

2. Do not route other signal lines through the dotted line area.

Recommended Oscillator Constants

Ceramic resonator

Manufacturer	Product number	Recommended constants	
		C1 (pF)	C2 (pF)
Kyocera	KBR-10.0M Note 1	33	33
Murata	CSA7.37MT040 Note 2	100	100
	CSA10.0MT Note 1	47	47
	CSA11.0MT Note 2		
	CSA16.0MX040 Note 1	30	30
TDK	FCR10.0M2S Note 2	30	30
	FCR16.0M2S Note 2	15	6
	FCR16.0M2G Note 2	22	10

Notes 1. T_A = –10 to +70 °C if these resonators are used.

2. T_A = –20 to +80 °C if these resonators are used.

Crystal resonator

Manufacturer	Product number	Recommended constants	
		C1 (pF)	C2 (pF)
Kinseki	HC-49/U (KR-100)	22	22
	HC-49/U (KR-160)	22	22

Remark For the characteristics of each resonator, consult each manufacturer.

CAPACITANCE (T_A = 25 °C, V_{DD} = 0 V)

PARAMETER	SYMBOL	TEST CONDITION	MIN.	TYP.	MAX.	UNIT
Input Capacitance	C_I	f_c = 1 MHz Unmeasured pins returned to 0 V			10	pF
Output Capacitance	C_O				20	pF
Input/output Capacitance	C_{IO}				20	pF

DC CHARACTERISTICS (T_A = –40 °C to +85 °C, V_{DD} = +5.0 V ± 10 %)

PARAMETER	SYMBOL	TEST CONDITION	MIN.	TYP.	MAX.	UNIT
Input Low Voltage	V_{IL}		0		0.8	V
Input High Voltage	V_{IH1}	Except $\overline{RESET}$, P10/NMI, X1, or X2	2.2		V_{DD}	V
	V_{IH2}	$\overline{RESET}$, P10/NMI, X1, X2	0.8V_{DD}		V_{DD}	V
Output Low Voltage	V_{OL}	I_{OL} = 1.6 mA			0.45	V
Output High Voltage	V_{OH}	I_{OH} = –0.4 mA	V_{DD} – 1.0			V
Input Current	I_I	$\overline{EA}$, P10/NMI; $0 \le V_I \le V_{DD}$			±20	μA
Input Leakage Current	I_{LI}	Except $\overline{EA}$, P10/NM; $0 \le V_I \le V_{DD}$			±10	μA
Output Leakage Current	I_{LO}	$0 \le V_O \le V_{DD}$			±10	μA
V_{TH} Current	I_{TH}	$0\ V \le V_{TH} \le V_{DD}$		0.5	1.0	mA
V_{DD} Supply Current	I_{DD1}	Operation mode		65	120	mA
	I_{DD2}	HALT mode		25	50	mA
	I_{DD3}	STOP mode		10	30	μA

AC CHARACTERISTICS (T_A = −40 °C to +85 °C, V_{DD} = +5.0 V ± 10 %)

PARAMETER	SYMBOL	TEST CONDITION	MIN.	MAX.	UNIT
X1 Input Cycle Time	t_{CYX}		62	250	ns
X1 Input High, Low Level Width	t_{WXH}, t_{WXL}		20		ns
X1 Input Rise, Fall Time	t_{XR}, t_{XF}			20	ns
CLKOUT Output Cycle Time	t_{CYK}	fx/2, T = t_{CYK}	125	2000	ns
CLKOUT Output High, Low Level Width	t_{WKH}, t_{WKL}		0.5T−15		ns
CLKOUT Output Rise, Fall Time	t_{KR}, t_{KF}			15	ns
Input Rise, Fall Time	t_{IR}, t_{IF}	Except $\overline{RESET}$, MNI, X1, X2		20	ns
	t_{IRS}, t_{IFS}	$\overline{RESET}$, NMI		30	ns
Output Rise, Fall Time	t_{OR}, t_{OF}	Except CLKOUT		20	ns
CLKOUT -> Address Delay Time	t_{DKA}			90	ns
Address -> Data Input Delay Time	t_{DADR}			(n+1.5)T−90	ns
$\overline{MREQ}$ ↓ -> Data Delay Time	t_{DMRD}			(n+2)T−75	ns
$\overline{MSTB}$ ↓ -> Data Delay Time	t_{DMSD}			(n+1)T−75	ns
$\overline{MREQ}$ ↓ -> $\overline{MSTB}$ ↓ Delay Time	t_{DMRMS1}	Read operation	T−35	T+35	ns
	t_{DMRMS2}	Write operation	(n+1)T−35	(n+1)T+35	ns
$\overline{MREQ}$ Low Level Width	t_{WMRL}		(n+2)T−30	(n+2)T+30	ns
$\overline{MREQ}$ ↓ -> Address Hold Time	t_{HMRA}		0.5T−30		ns
Address Hold Time from $\overline{MREQ}$ ↑	t_{HMA}		0.5T−30		ns
Data Input Hold Time from $\overline{MREQ}$ ↑	t_{HMDR}		0		ns
Control Signal Recovery Time	t_{RVC}		T−25		ns
Address -> Data Output Delay Time	t_{DADW}			0.5T+50	ns

PARAMETER	SYMBOL	TEST CONDITION	MIN.	MAX.	UNIT
Address Setup Time to $\overline{MREQ}$ ↓	t_{DAMR}		0.5T–30		ns
R/$\overline{W}$ -> $\overline{MSTB}$ ↓ Delay Time	t_{DRMS}		0.5T–30		ns
	t_{DWMS}		(n+0.5)T–30		ns
$\overline{MSTB}$ Low Level Width	t_{WMSL1}	Read operation	(n+1)T–30		ns
	t_{WMSL2}	Write operation	1T–30	1T+30	ns
Data Output Setup Time to $\overline{MSTB}$ ↑	t_{SDM}		(n+2)T–50		ns
Address Setup Time to $\overline{IOSTB}$ ↓	t_{DAIS}		0.5T–30		ns
$\overline{IOSTB}$ ↓ -> Data Delay Time	t_{DISD}			(n+1) T–90	ns
$\overline{MREQ}$ ↓ -> $\overline{IOSTB}$ ↓ Delay Time	t_{DMRIS}		1T–35		ns
$\overline{IOSTB}$ Low Level Width	t_{WISL}		(n+1)T–30		ns
Address Hold Time from $\overline{IOSTB}$ ↑	t_{HISA}		0.5T–30		ns
Data Input Hold Time from $\overline{IOSTB}$ ↑	t_{HISDR}		0		ns
Data Output Setup Time to $\overline{IOSTB}$ ↑	t_{SDIS}		(n+2)T–50		ns
$\overline{DMARQ}$ Setup Time to $\overline{MREQ}$ ↓	t_{SDADQ}	Demand release mode		1T	ns
$\overline{DMARQ}$ Hold Time from $\overline{DMAAK}$ ↓	t_{HDADQ}	Demand release mode	0		ns
$\overline{DMAAK}$ Output Low Level Width	t_{WDMRL}	Read operation	(n+2.5)T–30		ns
$\overline{DMAAK}$ ↓ -> $\overline{TC}$ ↓ Delay Time	t_{DDATC}			0.5T+50	ns
$\overline{TC}$ Low Level Width	t_{WTCL}		2T–30		ns
$\overline{DMAAK}$ Output Low Level Width	t_{WDMWL}	Write operation	(n+2)T–30		ns
Address Setup Time to $\overline{REFRQ}$ ↓	t_{DARF}		0.5T–30		ns
$\overline{REFRQ}$ Low Level Width	t_{WRFL}		(n+2)T–30		ns
Address Hold Time from $\overline{REFRQ}$ ↑	t_{HRFA}		0.5T–30		ns

PARAMETER	SYMBOL	TEST CONDITION	MAX.	MAX.	UNIT
$\overline{RESET}$ Low Level Width	t_{WRSL1}	STOP mode release/ power-on reset	30		ms
	t_{WRSL2}	System reset	5		μs
READY Setup Time to $\overline{MREQ}$ ↓	t_{SCRY0}	n ≥ 2		2T–100	ns
	t_{SCRY}	n ≥ 3		nT–100	ns
READY Hold Time from $\overline{MREQ}$ ↓	t_{HCRY0}	n = 2	2T		ns
	t_{HCRY}	n ≥ 3	nT		ns
	t_{HCRY1}	n ≥ 3	(n–1)T		ns
READY Setup Time to $\overline{IOSTB}$ ↓	t_{SSRY0}	n ≥ 2		T–100	ns
	t_{SSRY}	n ≥ 3		(n–1)T–100	ns
READY Hold Time from $\overline{IOSTB}$ ↓	t_{HSRY0}	n = 2	1T		ns
	t_{HSRY}	n ≥ 3	(n–1)T		ns
	t_{HSRY1}	n ≥ 3	(n–2)T		ns
HLDRQ Setup Time to CLKOUT ↑	t_{SHQK}		30		ns
CLKOUT ↑ -> $\overline{HLDAK}$ Delay Time	t_{DKHA}			80	ns
Bus Float -> $\overline{HLDAK}$ ↓ Delay Time	t_{CFHA}		1T–50		ns
$\overline{HLDAK}$ ↑ -> Bus Output Delay Time	t_{DHAC}		1T–50		ns
HLDRQ ↓ -> $\overline{HLDAK}$ ↑ Delay Time	t_{DHQHA}			3T+160	ns
HLDRQ ↓ -> Bus Output Delay Time	t_{DHQC}		3T+30		ns
HLDRQ Low Level Width	t_{WHQL}		1.5T		ns
$\overline{HLDAK}$ Low Level Width	t_{WHAL}		1T		ns
INTDMARQ Setup Time to CLKOUT ↑	t_{SIQK}		30		ns
INTDMARQ High, Low Level Width	t_{WIQH}, t_{WIQL}		8T		ns

Remark n denotes the number of wait states. n = 0 denotes no wait.

PARAMETER	SYMBOL	TEST CONDITION	MIN.	MAX.	UNIT
$\overline{\text{POLL}}$ Setup Time to CLKOUT ↑	t_{SPLK}		30		ns
NMI High, Low Level Width	t_{WNIH}, t_{WNIL}		5		μs
$\overline{\text{CTS}}$ Low Level Width	t_{WCTL}		2T		ns
INT Setup Time to CLKOUT ↑	t_{SIRK}		30		ns
CLKOUT ↓ -> $\overline{\text{INTAK}}$ ↓ Delay Time	t_{DKIA}			80	ns
INT Hold Time from $\overline{\text{INTAK}}$ ↓	t_{HIAIQ}		0		ns
$\overline{\text{INTAK}}$ Low Level Width	t_{WIAL}		2T–30		ns
$\overline{\text{INTAK}}$ High Level Width	t_{WIAH}		1T–30		ns
$\overline{\text{INTAK}}$ ↓ -> Data Delay Time	t_{DIAD}			2T–130	ns
Data Hold Time from $\overline{\text{INTAK}}$ ↑	t_{HIAD}		0	0.5T	ns
$\overline{\text{SCK0}}$ Cycle Time	t_{CYTK}		1000		ns
$\overline{\text{SCK0}}$ High, Low Level Width	t_{WSTH}, t_{WSTL}		450		ns
$\overline{\text{SCK0}}$ ↓ -> TxD Delay Time	t_{DTKD}			210	ns
$\overline{\text{SCK0}}$ ↓ -> TxD Hold Time	t_{HTKD}		20		ns
$\overline{\text{CTS0}}$ Cycle Time	t_{CYRK}		1000		ns
$\overline{\text{CTS0}}$ High, Low Level Width	t_{WSRH}, t_{WSRL}		420		ns
RxD Setup, Hold Time to/from $\overline{\text{CTS0}}$ ↑	t_{SRDK}, t_{HKRD}		80		ns

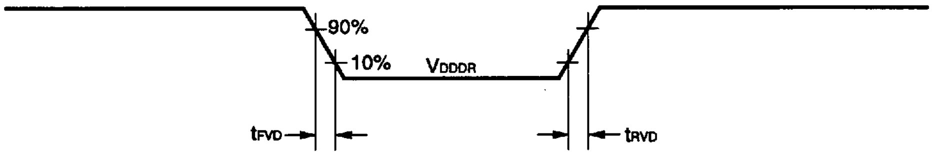
COMPARATOR CHARACTERISTICS (T_A = −40 °C to +85 °C, V_{DD} = +5.0 V ± 10 %)

PARAMETER	SYMBOL	TEST CONDITION	MIN.	TYP.	MAX.	UNIT
Comparison Accuracy	V_{ACOMP}				±100	mV
Threshold Voltage	V_{TH}		0		V_{DD}+0.1	V
Comparison time	t_{COMP}		64		65	t_{CYK}
PT Input Voltage	V_{IPT}		0		V_{DD}	V

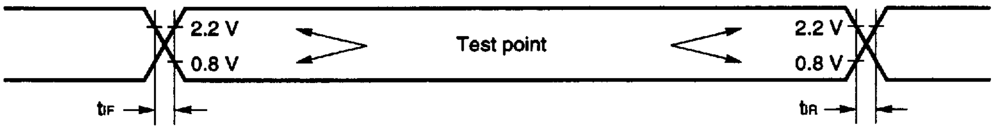
DATA MEMORY STOP MODE LOW SUPPLY VOLTAGE DATA RETENTION
(T_A = −40 °C to +85 °C)

PARAMETER	SYMBOL	TEST CONDITION	MIN.	MAX.	UNIT
Data Retention Supply Voltage	V_{DDDR}		2.5	5.5	V
V_{DD} Rise, Fall Time	t_{RVD}, t_{FVD}		200		μs

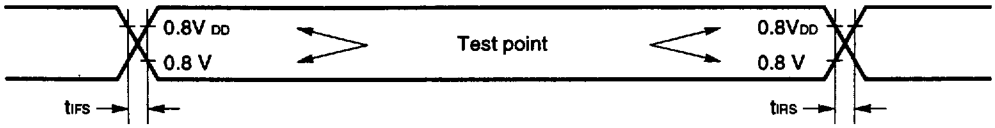
Data Retention Timing



AC Test Input Waveform (Except $\overline{\text{RESET}}$, NMI, X1, or X2)

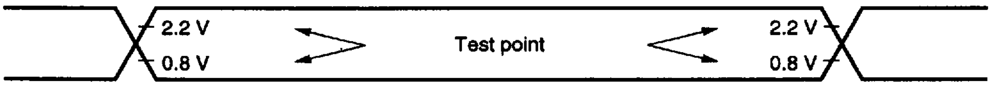


AC Test Output Test Points ($\overline{\text{RESET}}$, NMI, X1, or X2)

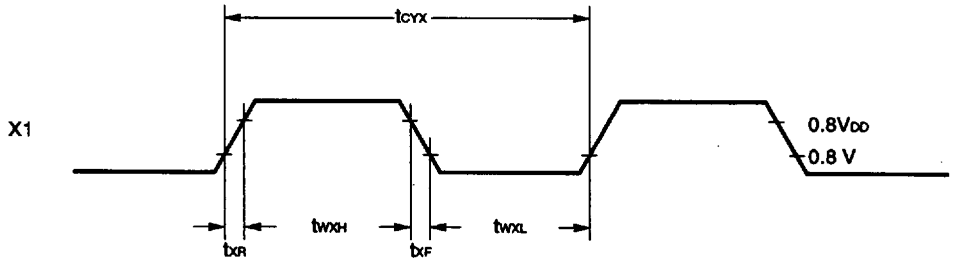


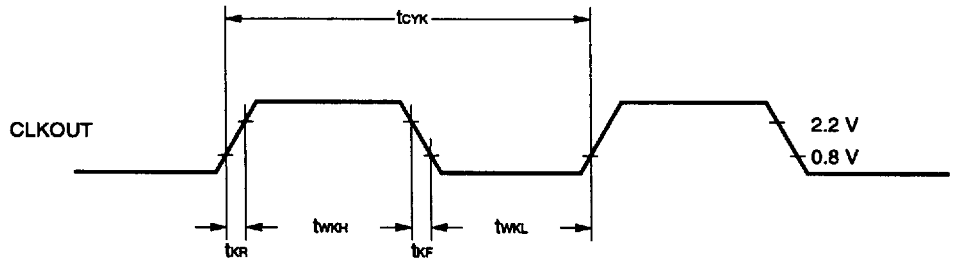
AC Test Output Test Points

Output load condition: 100 pF

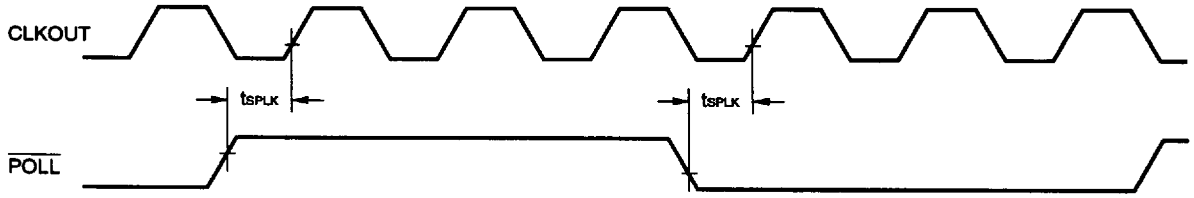


Clock Timing





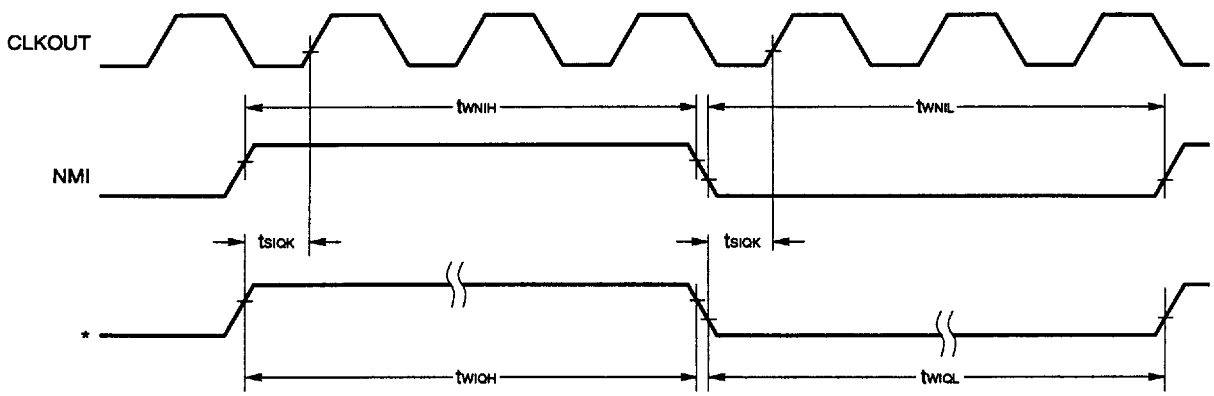
POLL Input Timing



CTS0-CTS1 Input Timing

6427525 0062987 377

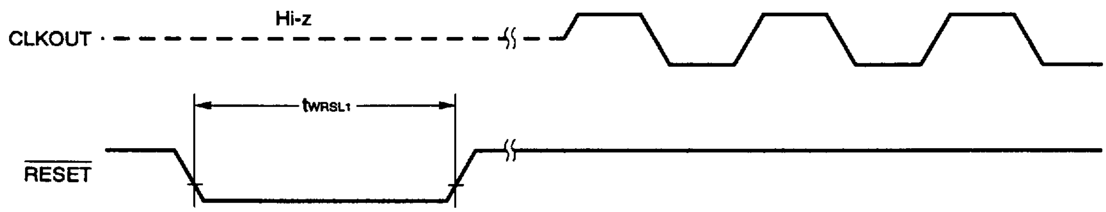
Interrupt Input/DMA Input Timing



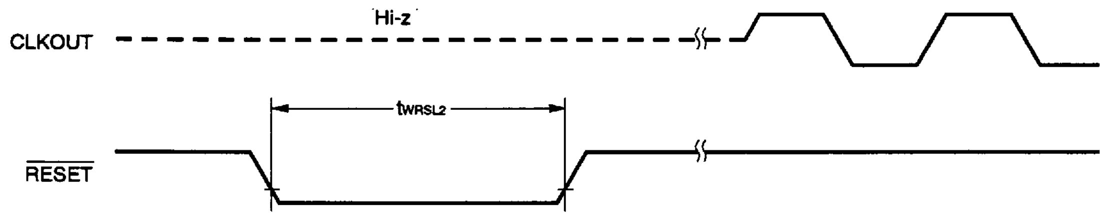
* $\overline{\text{INTP0}}$-$\overline{\text{INTP2}}$, DMARQ0-DMARQ1

$\overline{\text{RESET}}$ Input Timing

At release of STOP mode/Power-on reset:

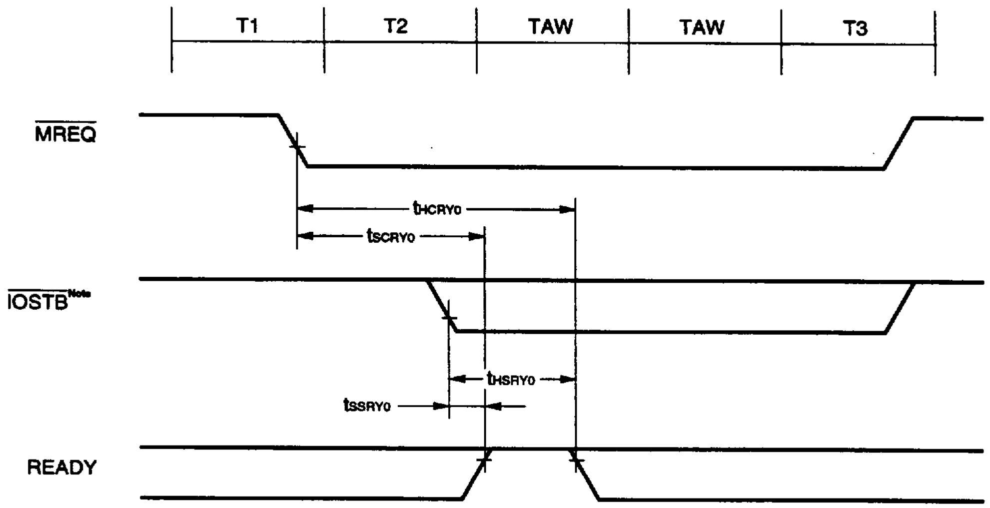


★ **At system reset:**

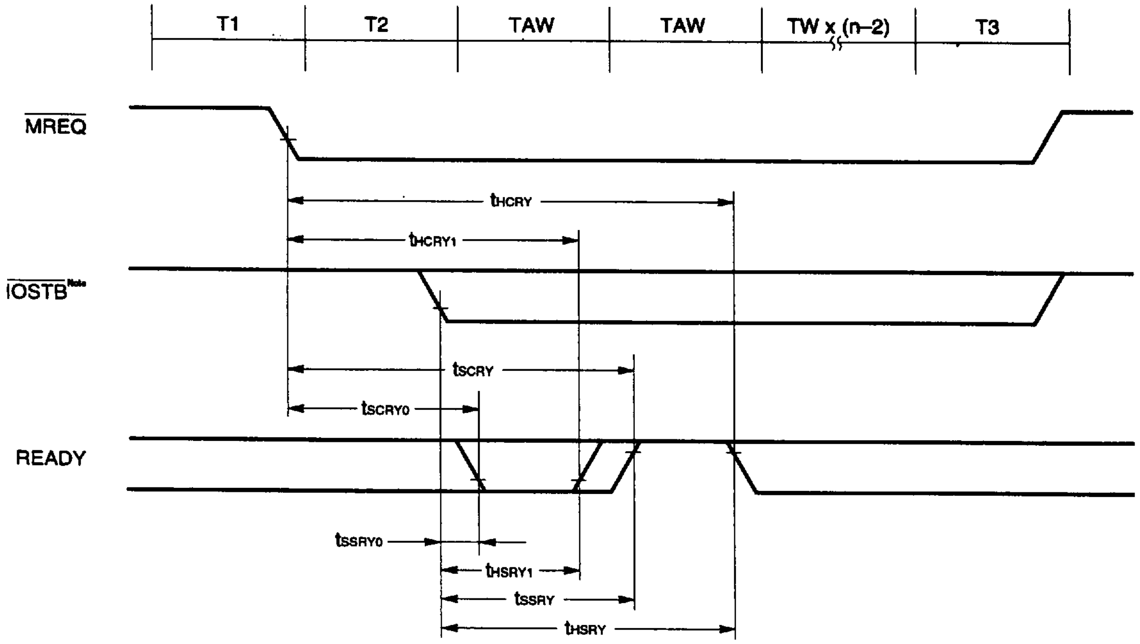


Ready Timing

At insertion of two-wait state:



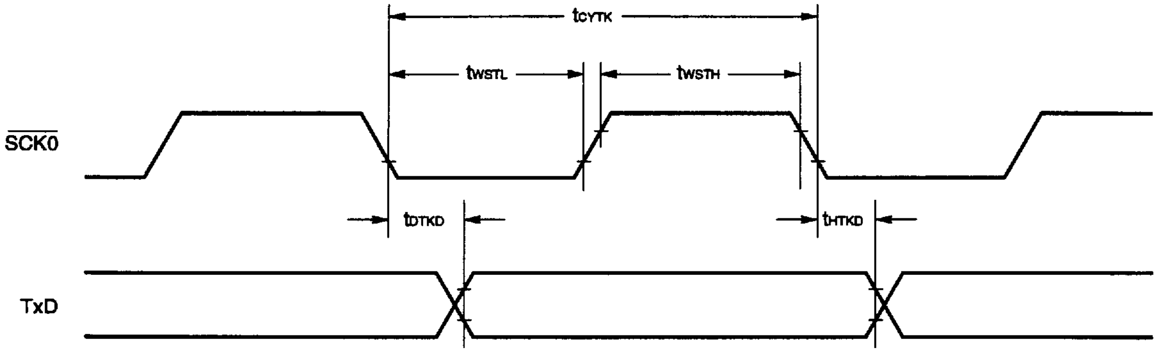
At insertion of additional (n–2) state [n ≥ 3]:



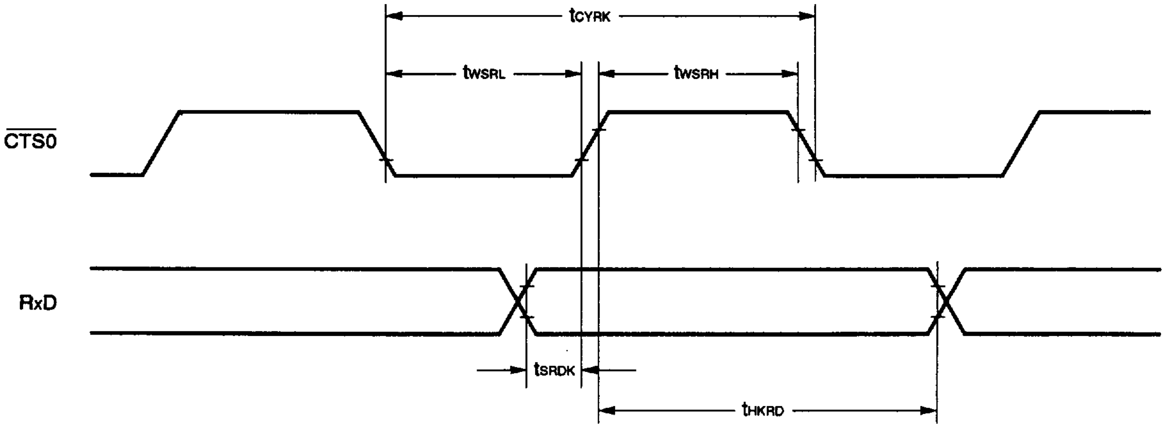
Note Only I/O cycle output

Serial Operation

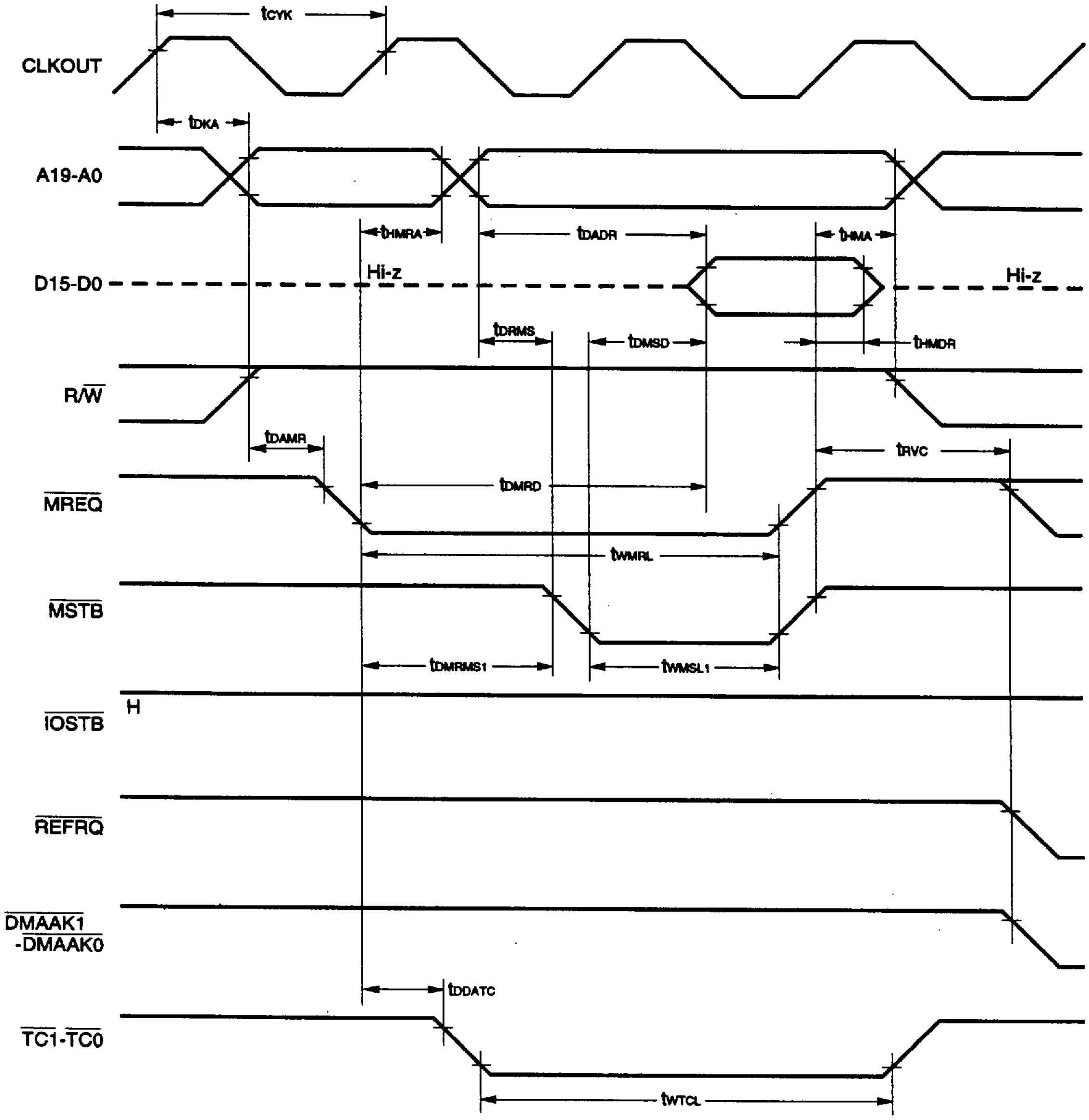
At transmission of I/O interface mode:



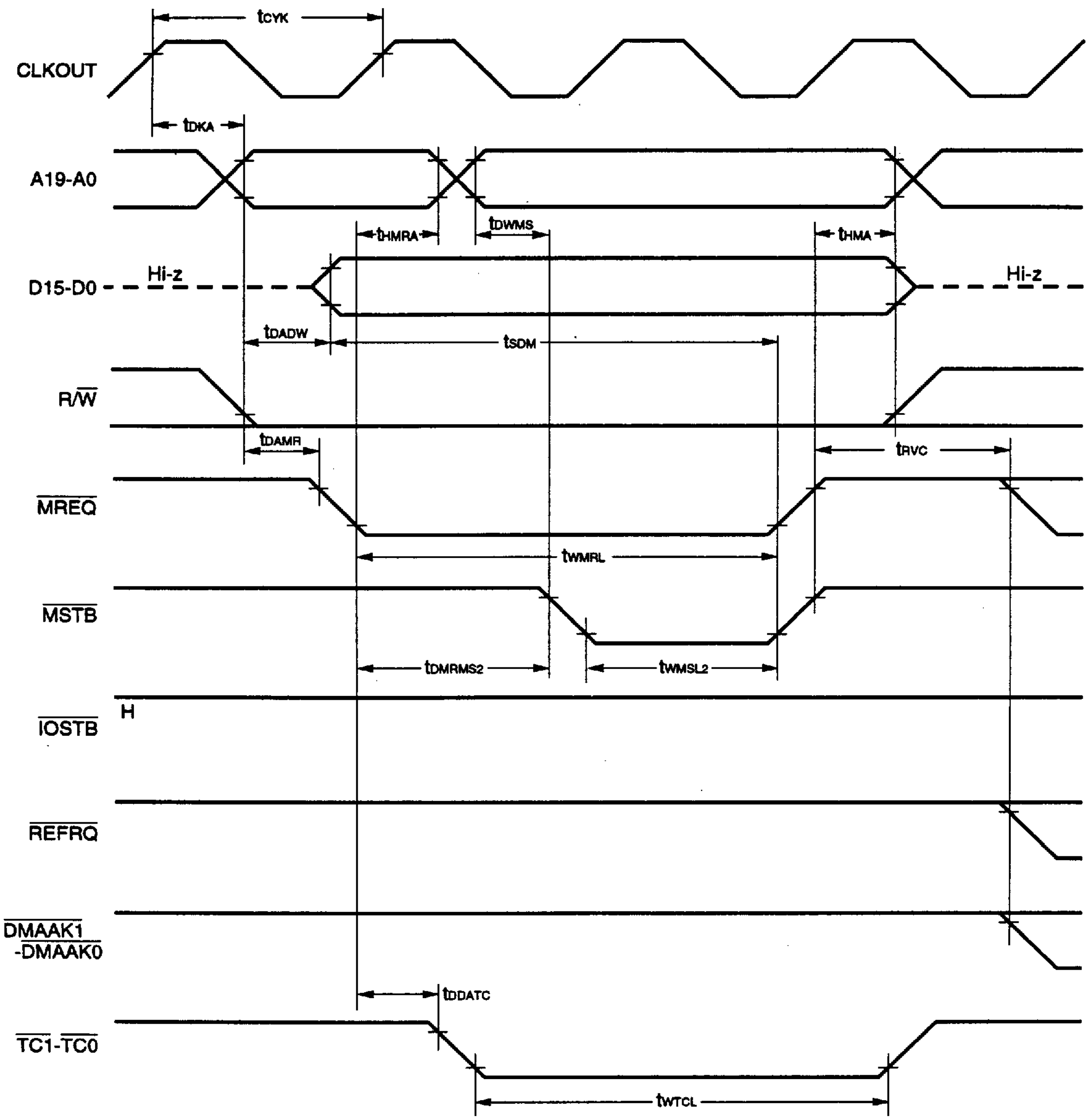
At receiving I/O interface mode:



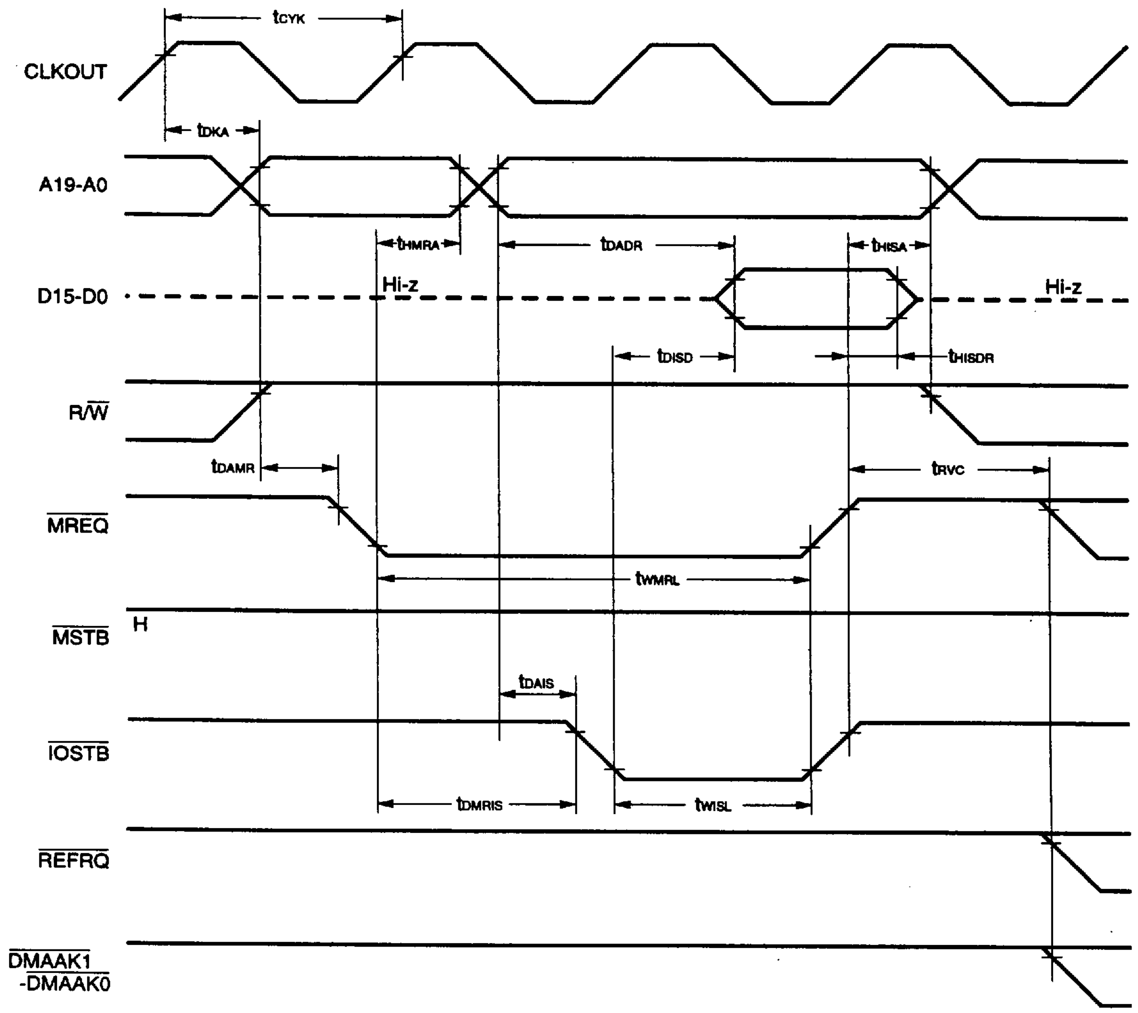
Read Operation



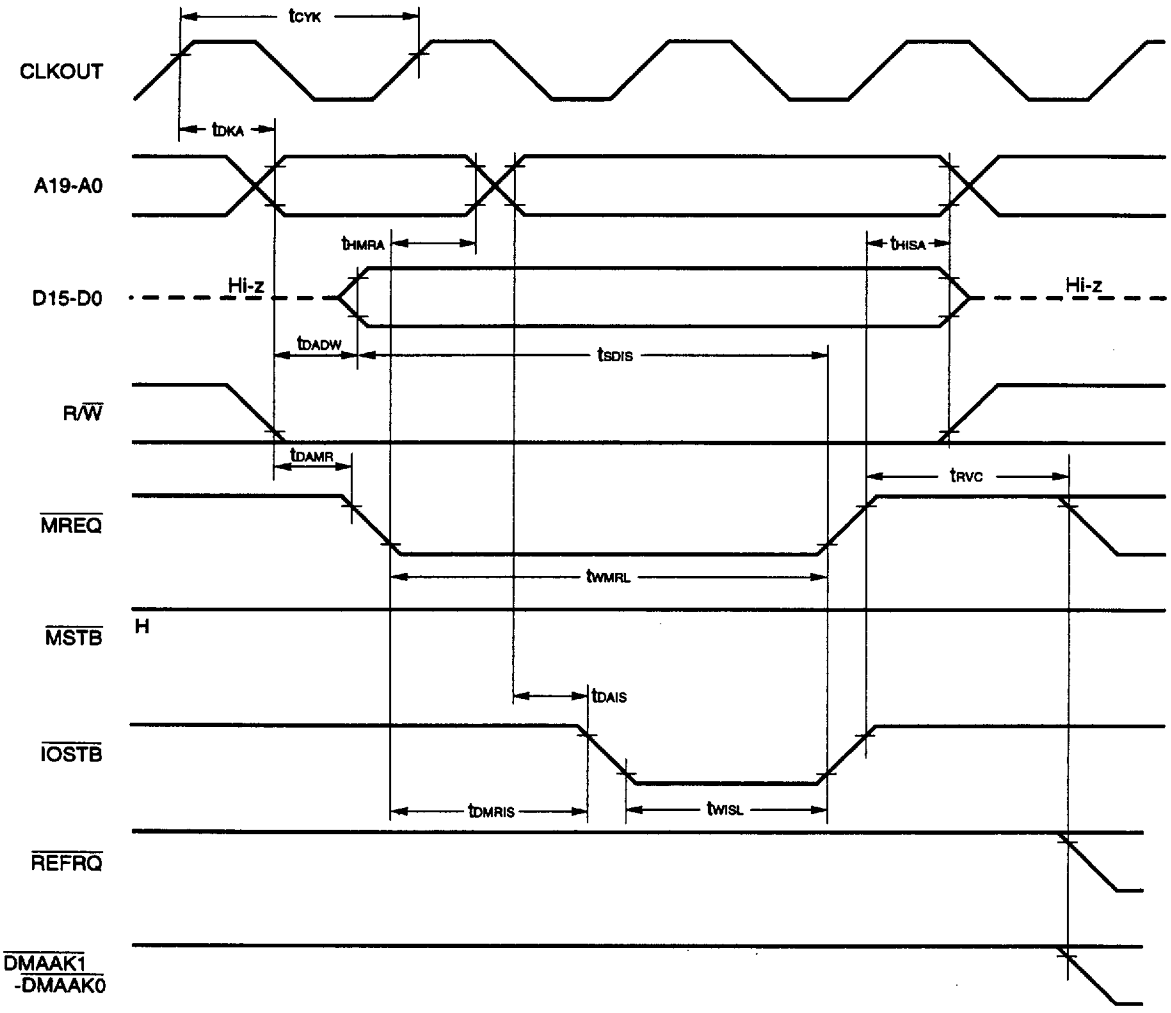
Write Operation



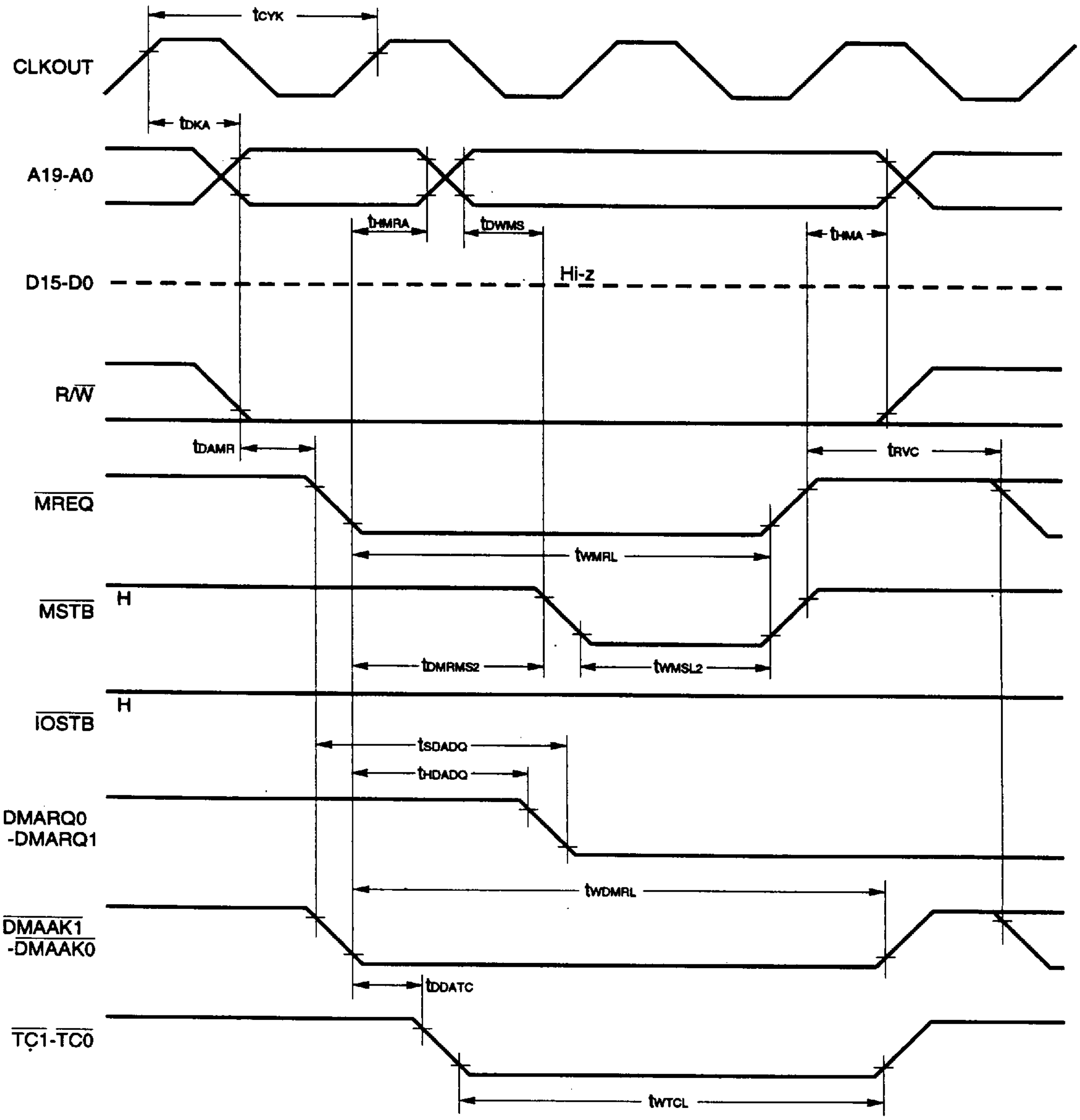
I/O Read Timing



I/O Write Timing

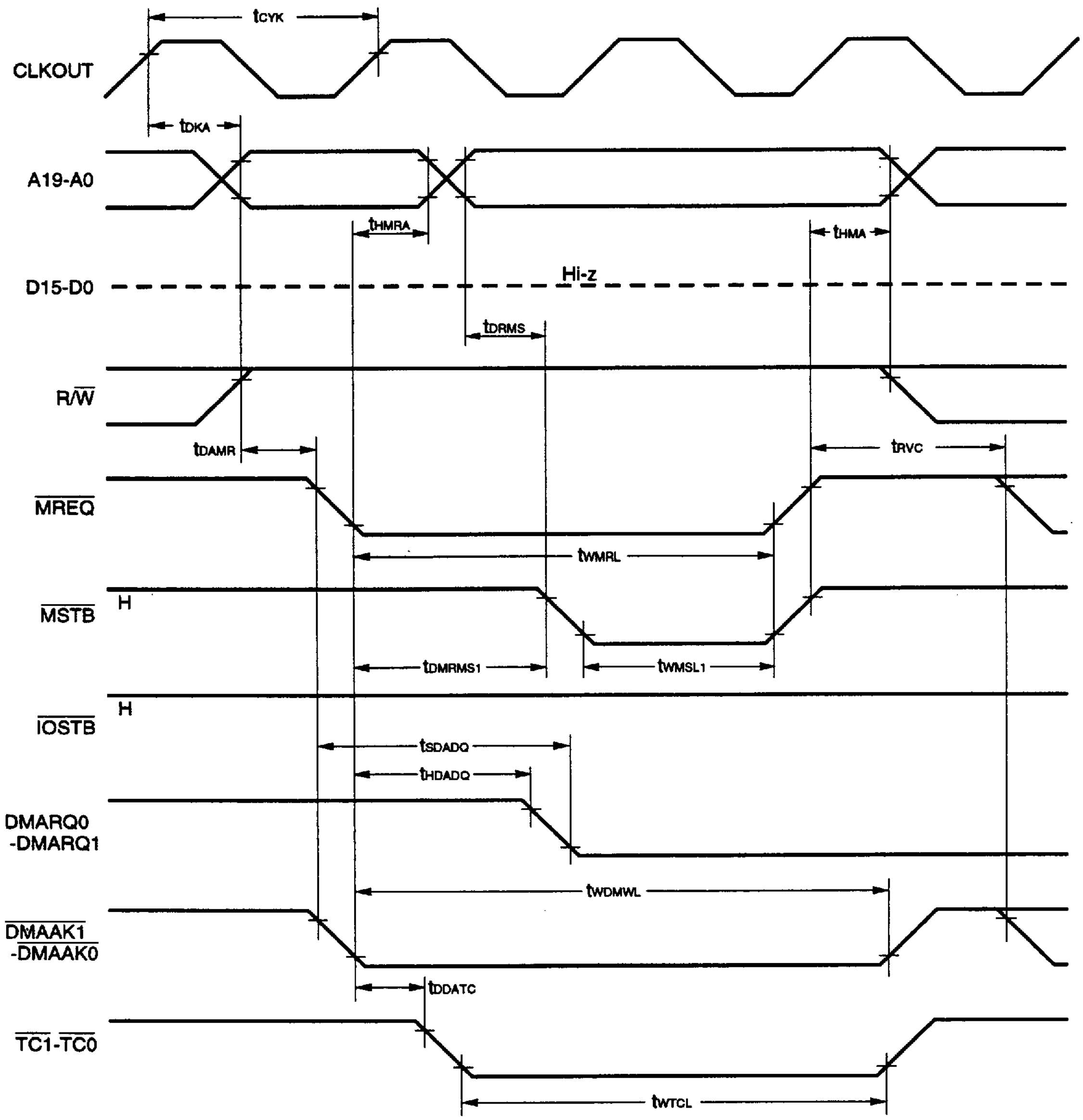


DMA (I/O -> Memory) Timing

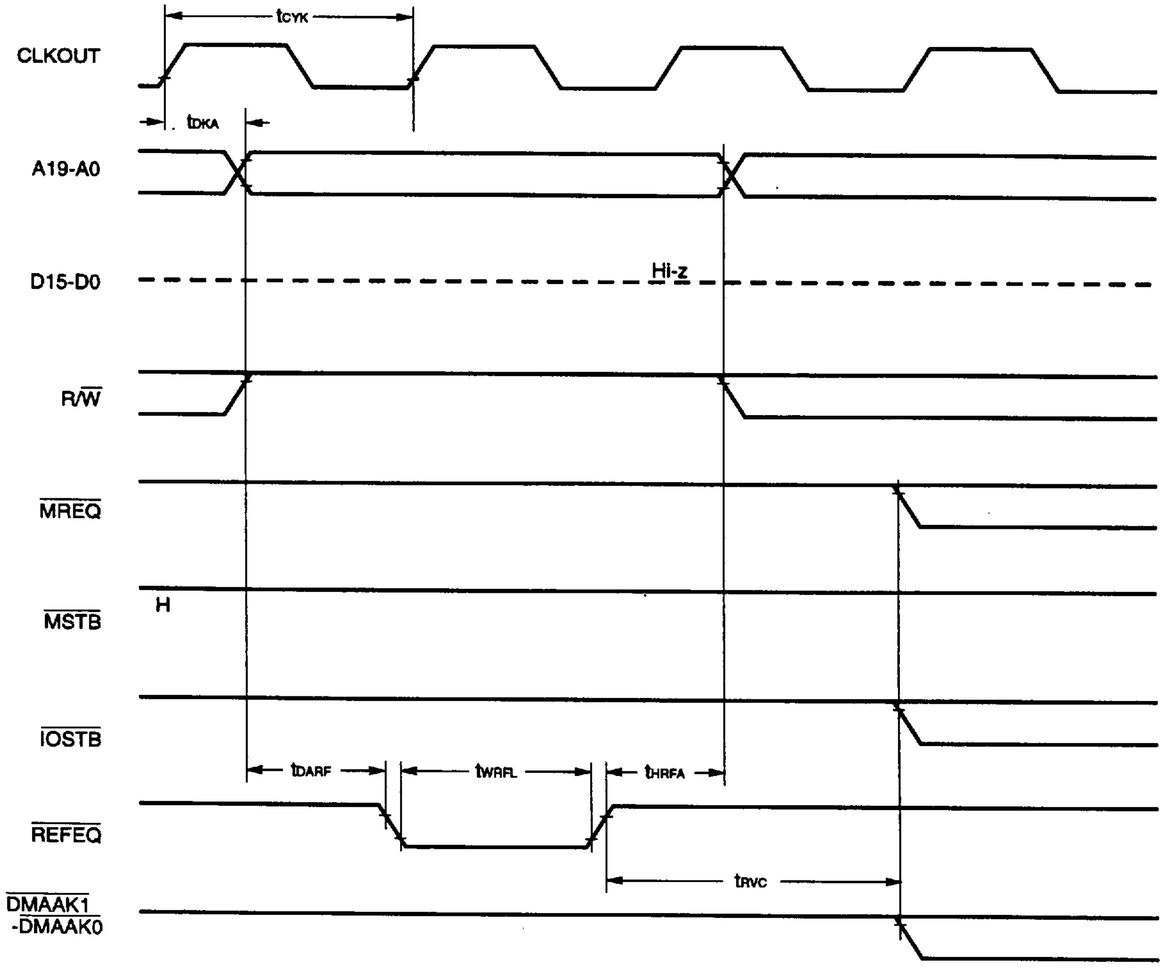


6427525 0062995 443

DMA (Memory -> I/O) Timing

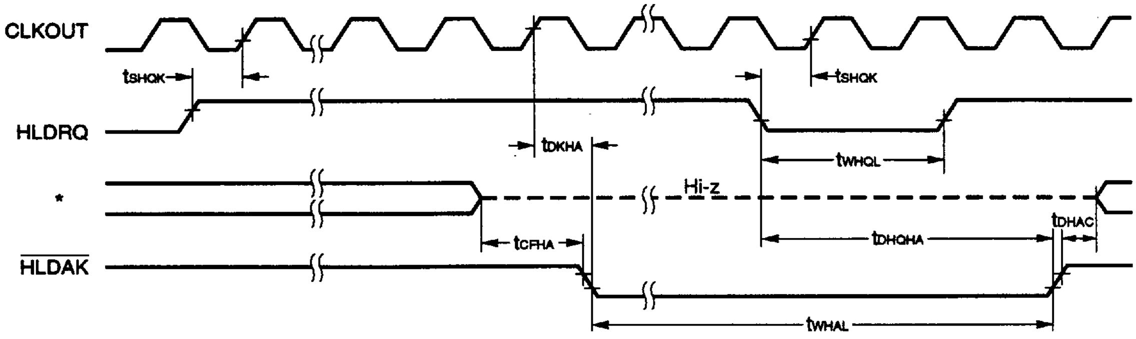


Refresh Timing

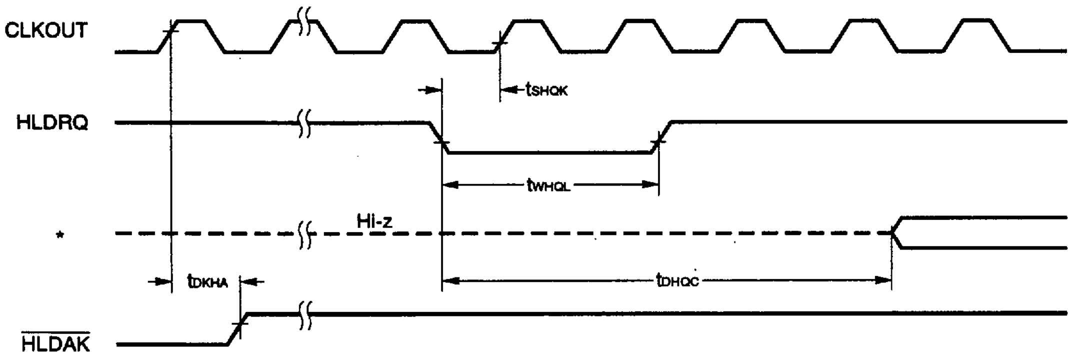


Hold Request/Acknowledge Timing

Normal mode:

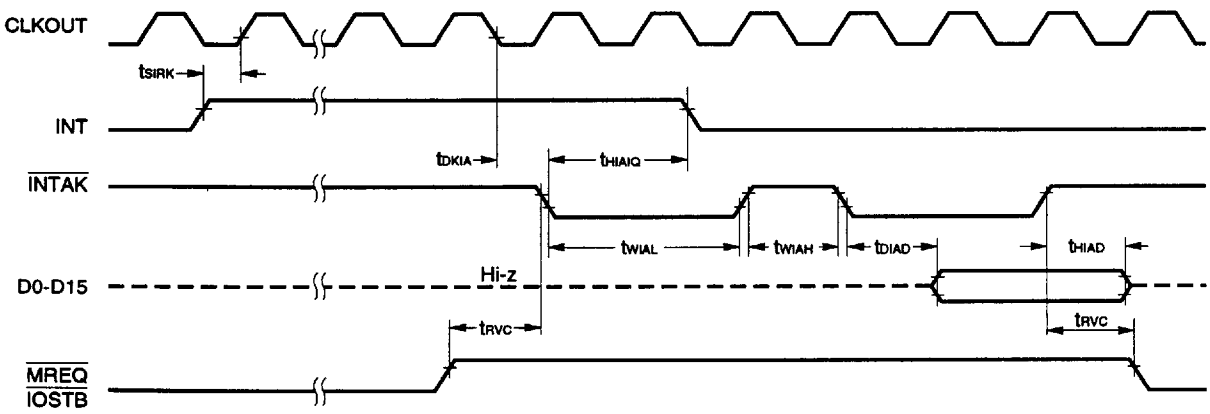


Release of HOLD mode in refresh:

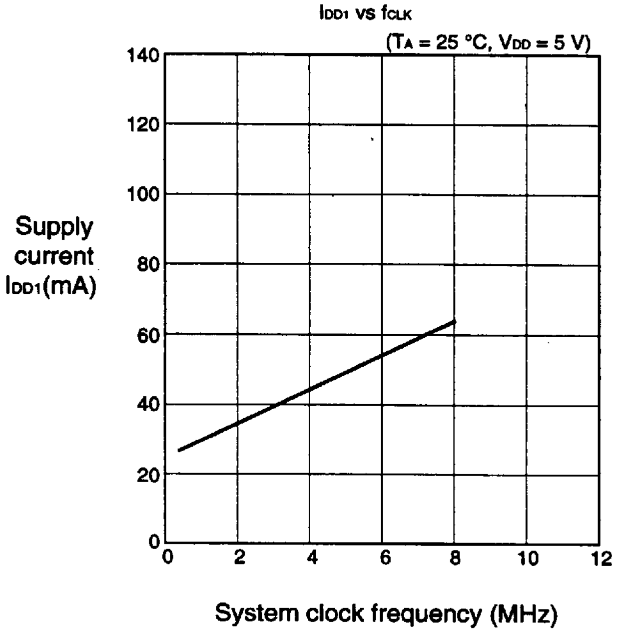


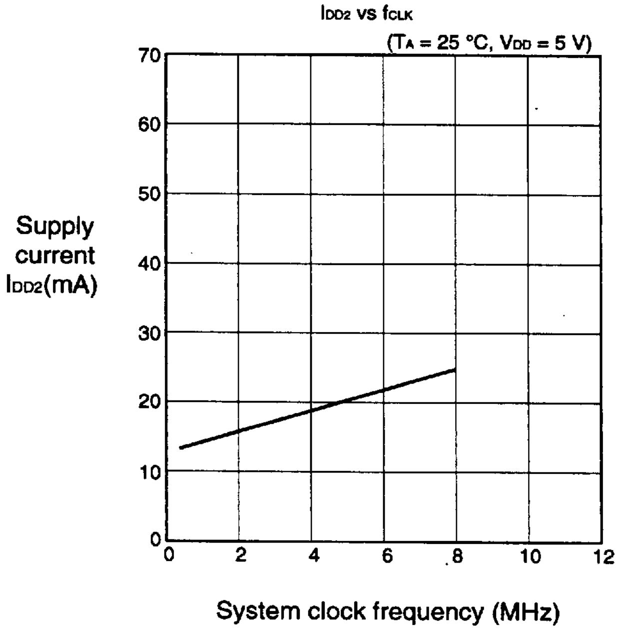
*: A19–A0, D15–D0, $\overline{\text{MREQ}}$, $\overline{\text{MSTB}}$, $\overline{\text{IOSTB}}$, R/$\overline{\text{W}}$

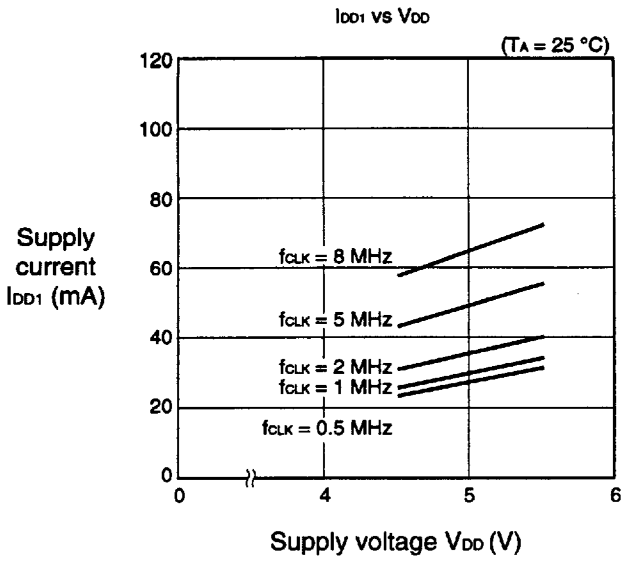
External Interrupt Request/Acknowledge Timing

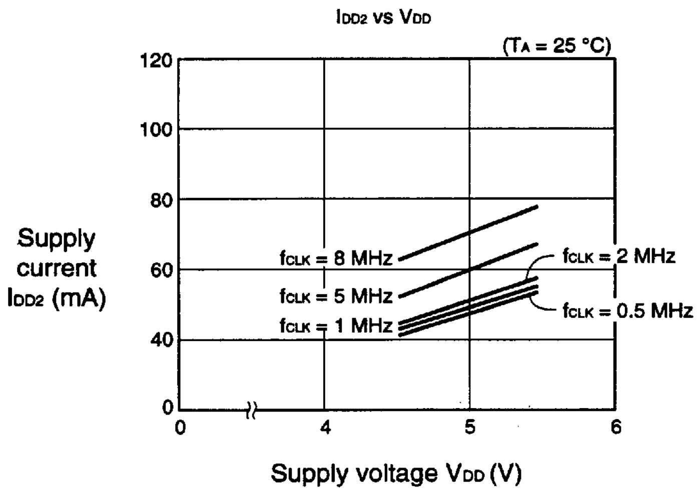


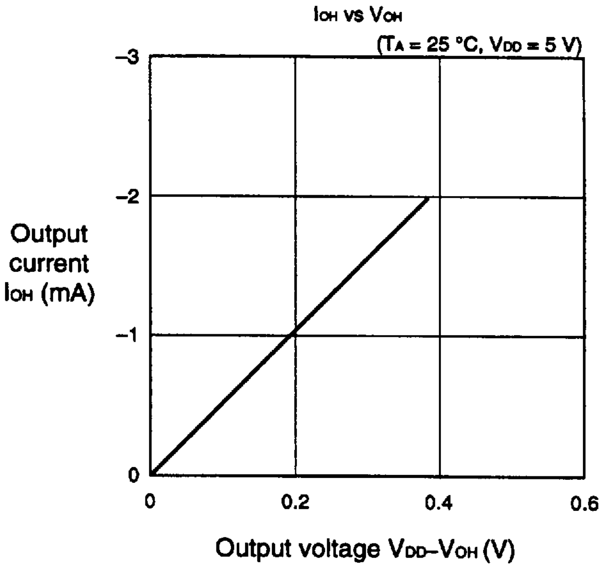
4. CHARACTERISTIC CURVES

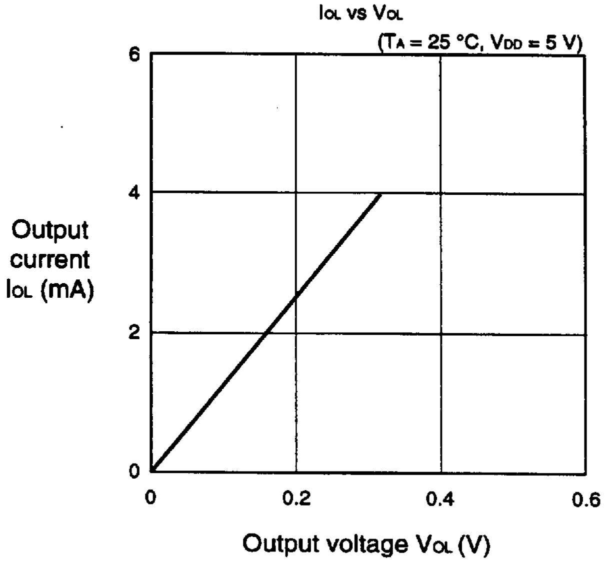





IDD1 vs VDD
(TA = 25 °C)
Supply current IDD1 (mA)
120
100
80
60
40
20
0
fCLK = 8 MHz
fCLK = 5 MHz
fCLK = 2 MHz
fCLK = 1 MHz
fCLK = 0.5 MHz
0
4
5
6
Supply voltage VDD (V)

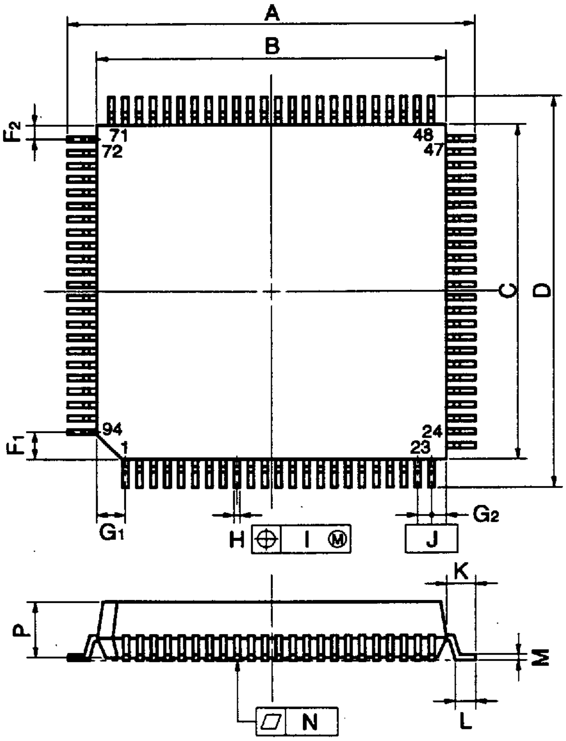

IDD2 vs VDD
(TA = 25 °C)
Supply current IDD2 (mA)
120
100
80
60
40
0
fCLK = 8 MHz
fCLK = 5 MHz
fCLK = 1 MHz
fCLK = 2 MHz
fCLK = 0.5 MHz
0
4
5
6
Supply voltage VDD (V)


I_{OH} vs V_{OH}
(T_A = 25 °C, V_{DD} = 5 V)
–3
–2
–1
0
Output current I_{OH} (mA)
0
0.2
0.4
0.6
Output voltage V_{DD}–V_{OH} (V)

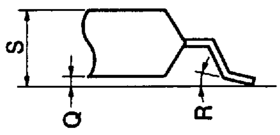

I_{OL} vs V_{OL}
(T_A = 25 °C, V_{DD} = 5 V)
6
4
2
0
Output current I_{OL} (mA)
0
0.2
0.4
0.6
Output voltage V_{OL} (V)

5. PACKAGE DRAWINGS

94 PIN PLASTIC QFP (□20)



detail of lead end



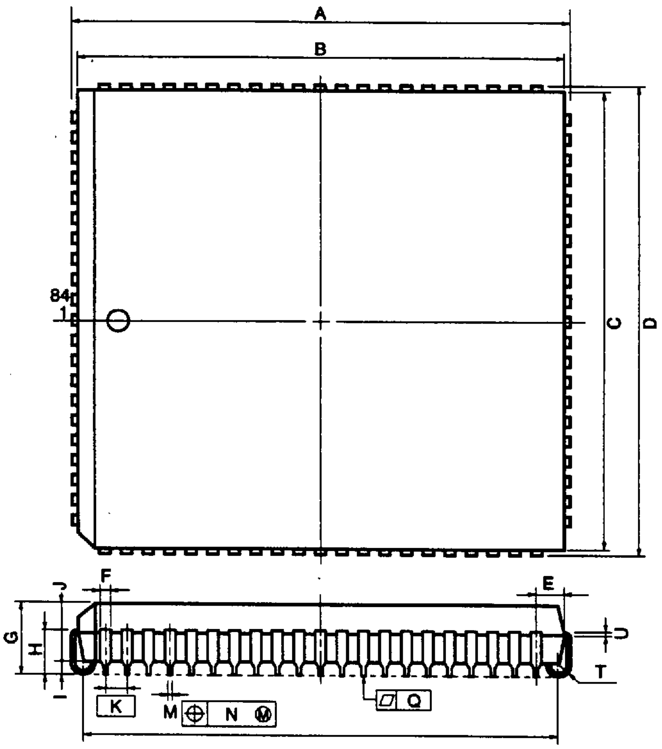
NOTE

Each lead centerline is located within 0.15 mm (0.006 inch) of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS	INCHES
A	23.2±0.4	$0.913^{+0.017}_{-0.016}$
B	20.0±0.2	$0.787^{+0.009}_{-0.008}$
C	20.0±0.2	$0.787^{+0.009}_{-0.008}$
D	23.2±0.4	$0.913^{+0.017}_{-0.016}$
F_1	1.6	0.063
F_2	0.8	0.031
G_1	1.6	0.063
G_2	0.8	0.031
H	0.35±0.10	$0.014^{+0.004}_{-0.005}$
I	0.15	0.006
J	0.8 (T.P.)	0.031 (T.P.)
K	1.6±0.2	0.063±0.008
L	0.8±0.2	$0.031^{+0.009}_{-0.008}$
M	$0.15^{+0.10}_{-0.05}$	$0.006^{+0.004}_{-0.003}$
N	0.10	0.004
P	3.7	0.146
Q	0.1±0.1	0.004±0.004
R	5°±5°	5°±5°
S	4.0 MAX.	0.158 MAX.

S94GJ-80-5BG-3

84 PIN PLASTIC QFJ (□1150 mil)



P84L-50A3-2

NOTE
Each lead centerline is located within 0.12 mm (0.005 inch) of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS	INCHES
A	30.2±0.2	1.189±0.008
B	29.28	1.153
C	29.28	1.153
D	30.2±0.2	1.189±0.008
E	1.94±0.15	$0.076^{+0.007}_{-0.006}$
F	0.6	0.024
G	4.4±0.2	$0.173^{+0.009}_{-0.008}$
H	2.8±0.2	$0.110^{+0.009}_{-0.008}$
I	0.9 MIN.	0.035 MIN.
J	3.4	0.134
K	1.27 (T.P.)	0.050 (T.P.)
M	0.40±0.10	$0.016^{+0.004}_{-0.005}$
N	0.12	0.005
P	28.20±0.20	$1.110^{+0.009}_{-0.008}$
Q	0.15	0.006
T	R 0.8	R 0.031
U	$0.20^{+0.10}_{-0.05}$	$0.008^{+0.004}_{-0.002}$

★ 6. RECOMMENDED SOLDERING CONDITIONS

The following conditions must be met when soldering this product.
For more details, refer to our document **"SEMICONDUCTOR DEVICE MOUNTING TECHNOLOGY MANUAL" (IEI-1207).**
Please consult with an NEC sales representative when using other soldering process or under different soldering conditions.

Table 6-1. Surface Mounting Type Soldering Conditions

(1) μPD70330L-8 : 84-pin plastic QFJ (1150 x 1150 mil)

Soldering process	Soldering conditions	Symbol
VPS	Peak temperature of package surface: 215°C or below, Reflow time: 40 seconds or less, Number of reflow process: 1 Exposure limit: 2 days[Note] (16 hours pre-baking is required at 125°C afterwards)	VP15-162-1
Partial heating	Pin temperature: 300°C or below, Flow time: 3 seconds or less (per device side)	—

Note Exposure limit before soldering after dry-pack package is opened.
Storage condition: 25°C and relative humidity at 65% or less.

(2) μPD70330GJ-8-5BG : 94-pin plastic QFP (20 x 20 mm)
μPD70330GJ(A)-8-5BG : 94-pin plastic QFP (20 x 20 mm)

J specification product

Soldering process	Soldering conditions	Symbol
Infrared ray reflow	Peak temperature of package surface: 235°C or below, Reflow time: 30 seconds or less, Number of reflow process: 2 or less Exposure limit: 7 days[Note] (36 hours pre-baking required at 125°C afterwards) <precautions> (1) The second reflow should be started after the device temperature raised by the first reflow has returned to room temperature. (2) Flux washing must not be performed by the use of water after the first reflow.	IR35-367-2
VPS	Peak temperature of package surface: 215°C or below, Reflow time: 40 seconds or less, Number of reflow process: 2 or less Exposure limit: 7 days[Note] (36 hours pre-baking required at 125°C afterwards) <precautions> (1) The second reflow should be started after the device temperature raised by the first reflow has returned to room temperature. (2) Flux washing must not be performed by the use of water after the first reflow.	VP15-367-2
Wave soldering	Peak temperature of package : 260°C or below, Reflow time: 10 seconds or less, Number of reflow process: 1, Preheating temperature: 120°C MAX. (Package surface) Exposure limit: 7 days[Note] (36 hours pre-baking required at 125°C afterwards)	WS60-367-1
Partial heating	Pin temperature: 300°C or below, Flow time: 3 seconds or less (per device side)	—

Note Exposure limit before soldering after dry-pack package is opened.
Storage condition: 25 °C and relative humidity at 65 % or less.

Caution Do not use several soldering methods in combination (except partial heating).

6172C