DATA SHEET

MOS INTEGRATED CIRCUIT

μPD70320

V25™
16/8-BIT SINGLE-CHIP MICROCONTROLLER

The μPD70320 (V25) is a single-chip microcontroller on which 16-bit CPU, RAM, serial interface, timer, DMA controller, interrupt controller, etc. are all integrated. The μPD70320 is compatible with the 8/16-bit microprocessor μPD70108/70116 (V20™/V30™) on the software level.

The details of the functions are described in the following User's Manuals. Be sure to read it before starting design.

• V25, V35™ User's Manual — Hardware	**: IEM-1220**
• V25, V35 Family User's Manual — Instructions	**: U12120J (Japanese version)**

FEATURES

- Internal 16-bit architecture and external 8-bit data bus
- Compatible with μPD70108/70116 (in native mode) on software level (some instructions added)
- Minimum instruction cycle : 400 ns/5 MHz (μPD70320)
  250 ns/8 MHz (μPD70320-8)
- On-chip RAM : 256 words × 8 bits
- Input port (port T) with comparator : 8 bits
- I/O lines (input port : 4 bits, input/output port : 20 bits)
- Serial interface (internal dedicated baud rate generator) : 2 channels
  Asynchronous mode and I/O interface mode
- Interrupt controller
  - Programmable priority (8 levels)
  - Vectored interrupt function
  - Register bank switching function
  - Macro service function
- DRAM and pseudo SRAM refreshing functions
- DMA controller : 2 channels
- 16-bit timer : 2 channels
- Time base counter
- On-chip clock generator
- Programmable wait function
- Standby function (STOP/HALT)

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

Document No. U10090EJ8V0DS00 (8th edition)
Date Published November 1997 N
Printed in Japan

The mark ★ shows major revised points.

★ ORDERING INFORMATION

Part Number	Package	Max. Operating Frequency (MHz)
μPD70320L	84-pin plastic QFJ (1150 × 1150 mils)	5
μPD70320L-8	84-pin plastic QFJ (1150 × 1150 mils)	8
μPD70320GJ-5BG	94-pin plastic QFP (20 × 20 mm)	5
μPD70320GJ-8-5BG	94-pin plastic QFP (20 × 20 mm)	8

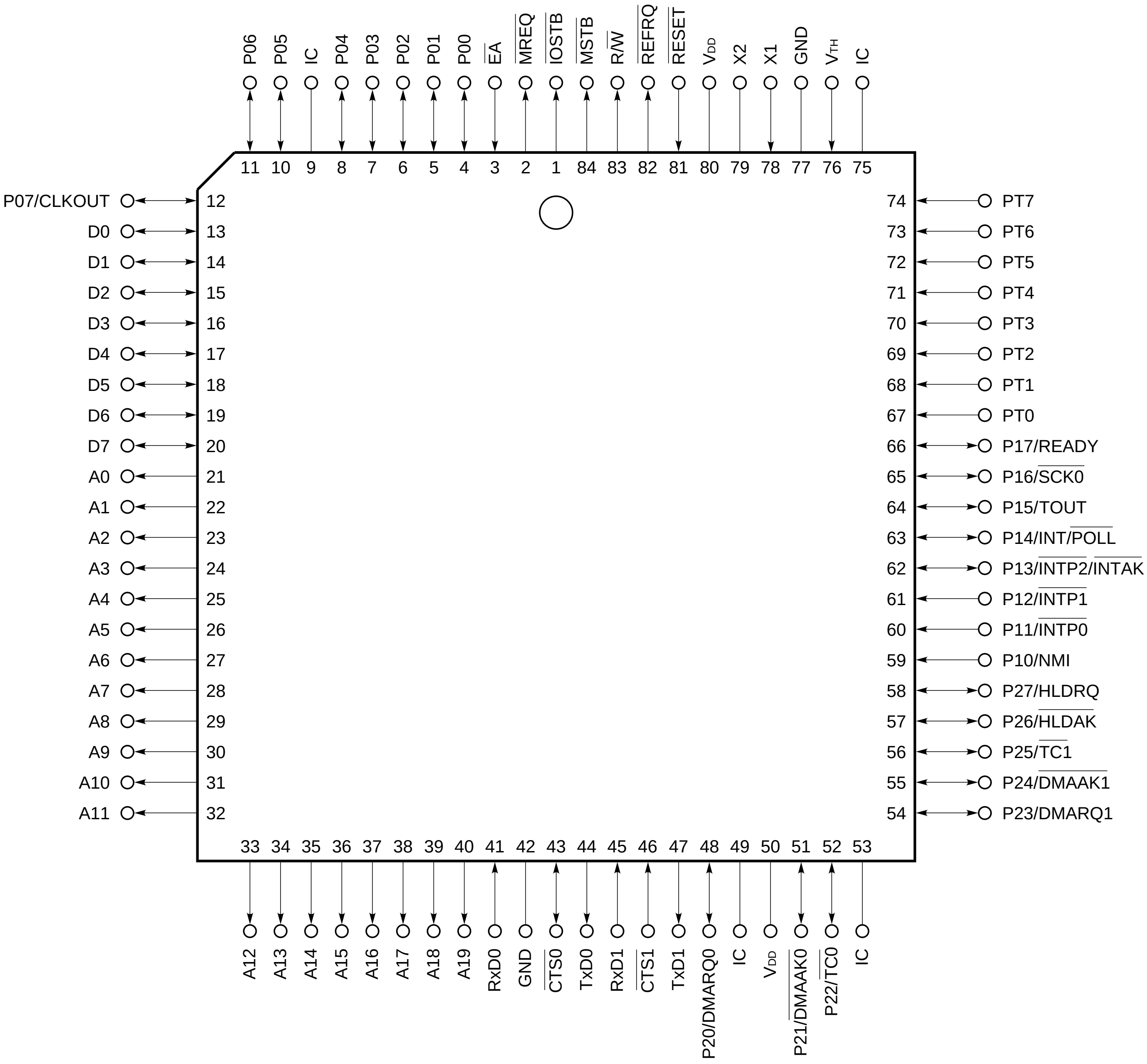
Remark The plastic QFJ is a new name of the PLCC.

PIN CONFIGURATION (Top View)

84-Pin Plastic QFJ (1150 × 1150 mils)

μPD70320L

μPD70320L-8



IC: Internally Connected

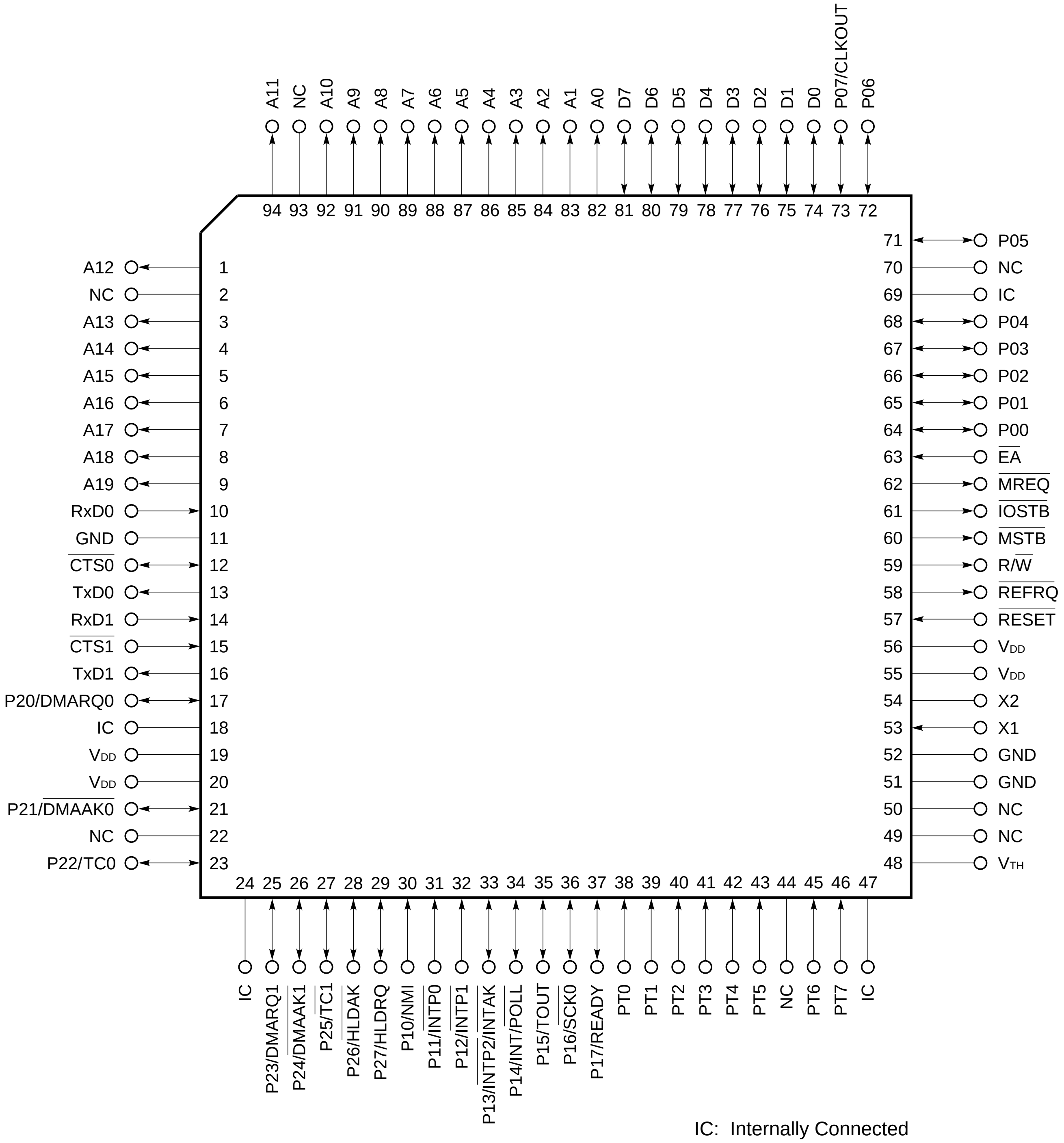
Cautions 1. Connect IC pin individually to V_{DD} via a resistor (3 to 10 kΩ).

2 Connect $\overline{EA}$ pin to GND via a resistor (3 to 10 kΩ).

★ **94-Pin Plastic QFP (20 × 20 mm)**

μPD70320GJ-5BG

μPD70320GJ-8-5BG



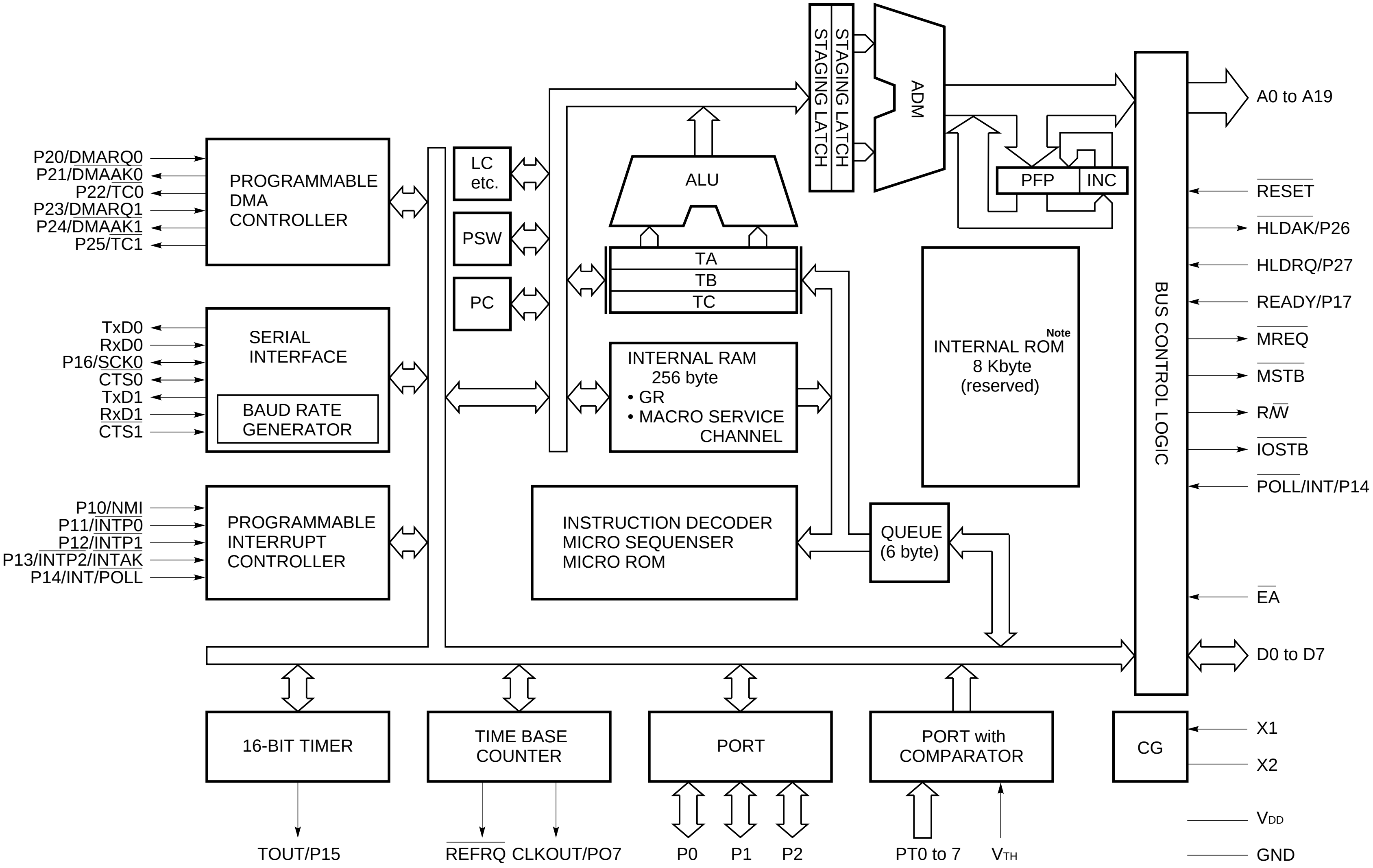
IC: Internally Connected

NC: Non-Connection

Cautions 1. Connect IC pin individually to V_{DD} via a resistor (3 to 10 kΩ).

2. Connect $\overline{EA}$ pin to GND via a resistor (3 to 10 kΩ).

INTERNAL BLOCK DIAGRAM



Note Not user-accessible.

CONTENTS

1. PIN FUNCTIONS

1.1 Port Pins

Pin Name	Input/Output	Port Function	Control Function
P00 to P06	Input & output	8-bit input/output ports, each to be specified bit-by-bit	—
P07/CLKOUT	Input & output/output		System clock output
P10/NMI	Input	Used as non-maskable interrupt request input (input port)	—
P11/$\overline{\text{INTP0}}$		Used as both external interrupt request input and input port	
P12/$\overline{\text{INTP1}}$			
P13/$\overline{\text{INTP2}}$/$\overline{\text{INTAK}}$	Input/input/output		INT acknowledge signal output
P14/$\overline{\text{POLL}}$/INT	Input & output/input/input	Used as both specifiable input/output port and $\overline{\text{POLL}}$ input	External interrupt request input
P15/TOUT	Input & output/output	Input/output port specifiable bit-by-bit	Timer output
P16/$\overline{\text{SCK0}}$			Serial clock output
P17/READY	Input & output/input		READY input
P20/DMARQ0	Input & output/input	8-bit input/output port specifiable bit-by-bit	DMA request input (CH0)
P21/$\overline{\text{DMAAK0}}$	Input & output/output		DMA acknowledge output (CH0)
P22/$\overline{\text{TC0}}$			DMA end output (CH0)
P23/DMARQ1	Input & output/input		DMA request input (CH1)
P24/$\overline{\text{DMAAK1}}$	Input & output/output		DMA acknowledge output (CH1)
P25/$\overline{\text{TC1}}$			DMA end output (CH1)
P26/$\overline{\text{HLDAK}}$	Input & output/output		HOLD acknowledge output
P27/HLDRQ	Input & output/input		HOLD input
PT0 to PT7	Input	8-bit input port with comparator	—

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

1.2 Non-port Pins

Pin Name	Input/Output	Function
TxD0	Output	Serial data output
TxD1		
RxD0	Input	Serial data input
RxD1		
$\overline{\text{CTS0}}$	Input & output	CTS input in asynchronous mode, receive clock input/output in I/O interface mode
$\overline{\text{CTS1}}$	Input	CTS input
$\overline{\text{REFRQ}}$	Output	DRAM refresh pulse output
V_{TH}	Input	Comparator reference voltage input
$\overline{\text{RESET}}$		Reset signal input
$\overline{\text{EA}}$		External memory access (connect to GND via a resistor (3 to 10 kΩ))
X1	Input	Used to connect crystal resonator/ceramic resonator for oscillating system clock. External clock is entered by entering reverse phase clock to both X1 and X2 pins.
X2		
D0 to D7	Input & output	8-bit data bus
A0 to A19	Output	20-bit address output
$\overline{\text{MREQ}}$		Output used to indicate that memory bus cycle has been started
$\overline{\text{MSTB}}$		Memory read/memory write strobe output
R/$\overline{\text{W}}$		Read cycle/write cycle ID signal output
$\overline{\text{IOSTB}}$		I/O read/I/O write strobe output
V_{DD}		Positive power supply pins (all pins should be connected)
GND		GND pins (all pins should be connected)
IC		Internally connected (connect individually to V_{DD} via a resistor (3 to 10 kΩ))

2. INSTRUCTION SETS

The μPD70320 instruction sets are upward-compatible with those of μPD70108/70116 in native mode.

2.1 Instructions Added to μPD70108/70116

The following instructions are newly added to the μPD70108/70116.

(1) Conditional branch instruction

- BTCLR ······· Bit test instruction used for special function registers

  If, when this BTCLR is executed, the target special function register bit status is “1”, the bit is reset (0) and the program is branched to short-label described in the operand. If the target bit status is “0”, the program is moved to the next instruction. PSW is not changed in this instruction.

(Descriptive format)

Mnemonic	Operand		
	Special Function Register Address	Special Function Register Bit	Branch Address
BTCLR	sfr	imm3	short-label

(2) Interrupt instructions

- RETRBI ······ Return instruction used for register banks

  This instruction is used to return the program from the interrupt service routine in which the register bank switching function is used. It cannot be used for returning from vectored interrupt servicing.

(Descriptive format)

Mnemonic	Operand
RETRBI	None

- FINT ··········· This instruction is used to report the interrupt controller that interrupt servicing has ended.

  If an interrupt other than NMI, INT, and software interrupt is used, this instruction must be executed prior to the instruction for returning from interrupt servicing. It should not be used for NMI, INT and software interrupts.

(Descriptive format)

Mnemonic	Operand
FINT	None

(3) CPU instruction

- STOP ········· Instruction for transition to STOP state

(Descriptive format)

Mnemonic	Operand
STOP	None

(4) Register bank switch instructions

- BRKCS ······ Used to switch register banks

  A register bank is switched to the register bank indicated by the lower 3 bits in the 16-bit register described in the operand. The program is also branched with this instruction to the address obtained from the PS stored in advance in the new register bank and the vector PC.

  The RETRBI instruction is used to return the program from the new register bank.

(Descriptive format)

Mnemonic	Operand
BRKCS	reg16

- TSKSW ······ Used to switch register banks

  Just like the BRKCS instruction, this instruction is also executed to select a register bank. The program is branched to the address obtained from the PS stored in advance in the new register bank and the address obtained from the PC save area.

(Descriptive format)

Mnemonic	Operand
TSKSW	reg16

(5) Data transfer instructions

- MOVSPA ··· Used to transfer SS and SP values

  This instruction is executed to transfer both SS and SP values before the register bank is switched to SS and SP of the current (post-switching) register bank.

(Descriptive format)

Mnemonic	Operand
MOVSPA	None

- MOVSPB ··· Used to transfer SS and SP values

  This instruction is executed to transfer the SS and SP values of the current (pre-switching) register bank to the SS and SP of the new register bank indicated by the lower 3 bits in the 16-bit register described in the operand.

(Descriptive format)

Mnemonic	Operand
MOVSPB	reg16

Some μPD70108/70116 instructions should be much cared as shown below when used for the μPD70320.

- I/O instruction, primitive I/O instruction

  If PSW $\overline{\text{IBRK}}$ flag is reset (0), an interrupt is generated without executing this instruction. Be sure to set (1) the $\overline{\text{IBRK}}$ flag when using the I/O instruction.

- FPO instruction

  An interrupt is generated without executing this instruction.

2.2 Instruction Set Operation

Table 2-1. Operand Identifier

Identifier	Description
reg	8-/16-bit general register
reg8	8-bit general register
reg16	16-bit general 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 within 0 to FFFFH
imm3	Constant within 0 to 7
imm4	Constant within 0 to FH
imm8	Constant within 0 to FFH
imm16	Constant within 0 to FFFFH
acc	Register AW or AL
sreg	Segment register
src-table	256-byte conversion table name
src-block	Register IX-addressed block name
dst-block	Register IY-addressed block name
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 within end of instruction to –128 to +127 bytes
far-label	Label in another program segment
memptr16	Word including location offset in the current program segment to which control is to be passed
memptr32	Double-word including location offset in another program segment to which control is to be passed and segment base address
regptr16	16-bit general register including location offset in another program segment to which control is to be passed
pop-value	Number of bytes to be abandoned from stack (0 to 64K, normally even number)
fp-op	Immediate value to judge instruction code of external floating point operation chip
R	Register set

Table 2-2. Operation Code Identifier

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

Table 2-3. Operation Identifier (1/2)

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
V	Overflow flag
BRK	Break flag
MD	Mode flag
(···)	Contents in memory shown in ( )
disp	Displacement (8/16 bits)
ext-disp8	16 bits obtained by extending sign of 8-bit displacement

Table 2-3. Operation Identifier (2/2)

Identifier	Description
temp	Temporary register (8/16/32 bits)
tmpcy	Temporary carry flag (1 bit)
seg	Immediate segment data (16 bits)
offset	Immediate offset data (16 bits)
←	Transfer direction
+	Addition
–	Subtraction
×	Multiplication
÷	Division
%	Modulo
∧	AND
∨	OR
∀	Exclusive OR
××H	2-digit hexadecimal number
××××H	4-digit hexadecimal number

Table 2-4. Flag Operation Identifier

Identifier	Description
(Blank)	No change
0	Cleared to 0
1	Set to 1
×	Set or cleared according to the result
U	Not defined
R	The previously saved value is restored.

Table 2-5. 8/16-Bit General Register Selection

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

The number of clocks, for memory operand, differs among addressing modes. So, use the following values for "EA" items shown in **Table 2-8 Number of Clocks**.

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

mod mem	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

"T" indicates the number of wait states. Use any number of waits starting at "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

Group	Mnemonic	Operand	Operation Code		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 to 4	(mem) ← reg						
		reg,mem	1 0 0 0 1 0 1 W	mod reg mem	2 to 4	reg ← (mem)						
		mem,imm	1 1 0 0 0 1 1 W	mod 0 0 0 mem	3 to 6	(mem) ← imm						
		reg,imm	1 0 1 1 W reg		2 to 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, (dmem) ← 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 to 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 to 4	(mem16) ← sreg						
		DS0,reg16, mem32	1 1 0 0 0 1 0 1	mod reg mem	2 to 4	reg16 ← (mem32) DS0 ← (mem32 + 2)						
		DS1,reg16, mem32	1 1 0 0 0 1 0 0	mod reg mem	2 to 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	×	×		×	×	×
	LDEA	reg16,mem16	1 0 0 0 1 1 0 1	mod reg mem	2 to 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 to 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	New register bank SS and SP ← old register bank SS and SP						
	MOVSPB Note	reg16	0 0 0 0 1 1 1 1	1 0 0 1 0 1 0 1	3	SS and SP of reg16-indicated new register bank ← old register bank SS and SP						
			1 1 1 1 1 reg									

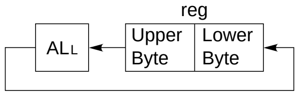
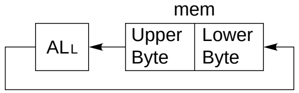
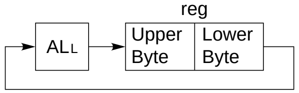
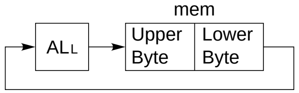
Note These instructions are newly added to the µPD70108/70116.

Group	Mnemonic	Operand	Operation Code		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
Repeat prefix	REPC		0 1 1 0 0 1 0 1		1	Executes the primitive block transfer instruction in the continued byte while CW ≠ 0, and decrements CW by one. If any interruption is held at this time, it is processed. The program exits the loop when CY ≠ 1.						
	REPNC		0 1 1 0 0 1 0 0		1	Same as above. The program exits the loop when CY ≠ 0.						
	REP REPE REPZ		1 1 1 1 0 0 1 1		1	Executes the primitive block transfer instruction in the continued byte while CW ≠ 0, and decrements CW by one. If any interruption is held at this time, it is processed. The program exits the loop when the primitive block transfer instruction is CMPBK or CMPM, and when Z ≠ 1.						
	REPNE REPNZ		1 1 1 1 0 0 1 0		1	Same as above. The program exits the loop when Z ≠ 0.						
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 + 1 DIR = 1: IX ← IX – 1, IY ← IY – 1 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 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						
	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						

Group	Mnemonic	Operand	Operation Code		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
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									
I/O	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 I/O	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 is generated automatically and the instruction is not executed.

Group	Mnemonic	Operand	Operation Code		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
Addition/subtraction	ADD	reg,reg	0 0 0 0 0 0 1 W	1 1 reg reg	2	reg ← reg + reg	×	×	×	×	×	×
		mem,reg	0 0 0 0 0 0 0 W	mod reg mem	2 to 4	(mem) ← (mem) + reg	×	×	×	×	×	×
		reg,mem	0 0 0 0 0 0 1 W	mod reg mem	2 to 4	reg ← reg + (mem)	×	×	×	×	×	×
		reg,imm	1 0 0 0 0 0 s W	1 1 0 0 0 reg	3 to 4	reg ← reg + imm	×	×	×	×	×	×
		mem,imm	1 0 0 0 0 0 s W	mod 0 0 0 mem	3 to 6	(mem) ← (mem) + imm	×	×	×	×	×	×
		acc,imm	0 0 0 0 0 1 0 W		2 to 3	When W = 0, AL ← AL + imm When W = 1, AW ← AW + imm	×	×	×	×	×	×
	ADDC	reg,reg	0 0 0 1 0 0 1 W	1 1 reg reg	2	reg ← reg + reg + CY	×	×	×	×	×	×
		mem,reg	0 0 0 1 0 0 0 W	mod reg mem	2 to 4	(mem) ← (mem) + reg + CY	×	×	×	×	×	×
		reg,mem	0 0 0 1 0 0 1 W	mod reg mem	2 to 4	reg ← reg + (mem) + CY	×	×	×	×	×	×
		reg,imm	1 0 0 0 0 0 s W	1 1 0 1 0 reg	3 to 4	reg ← reg + imm + CY	×	×	×	×	×	×
		mem,imm	1 0 0 0 0 0 s W	mod 0 1 0 mem	3 to 6	(mem) ← (mem) + imm + CY	×	×	×	×	×	×
		acc,imm	0 0 0 1 0 1 0 W		2 to 3	When W = 0, AL ← AL + imm + CY When W = 1, AW ← AW + imm + CY	×	×	×	×	×	×
	SUB	reg,reg	0 0 1 0 1 0 1 W	1 1 reg reg	2	reg ← reg – reg	×	×	×	×	×	×
		mem,reg	0 0 1 0 1 0 0 W	mod reg mem	2 to 4	(mem) ← (mem) – reg	×	×	×	×	×	×
		reg,mem	0 0 1 0 1 0 1 W	mod reg mem	2 to 4	reg ← reg – (mem)	×	×	×	×	×	×
		reg,imm	1 0 0 0 0 0 s W	1 1 1 0 1 reg	3 to 4	reg ← reg – imm	×	×	×	×	×	×
		mem,imm	1 0 0 0 0 0 s W	mod 1 0 1 mem	3 to 6	(mem) ← (mem) – imm	×	×	×	×	×	×
		acc,imm	0 0 1 0 1 1 0 W		2 to 3	When W = 0, AL ← AL – imm When W = 1, AW ← AW – imm	×	×	×	×	×	×
	SUBC	reg,reg	0 0 0 1 1 0 1 W	1 1 reg reg	2	reg ← reg – reg – CY	×	×	×	×	×	×
		mem,reg	0 0 0 1 1 0 0 W	mod reg mem	2 to 4	(mem) ← (mem) – reg – CY	×	×	×	×	×	×
		reg,mem	0 0 0 1 1 0 1 W	mod reg mem	2 to 4	reg ← reg – (mem) – CY	×	×	×	×	×	×
		reg,imm	1 0 0 0 0 0 s W	1 1 0 1 1 reg	3 to 4	reg ← reg – imm – CY	×	×	×	×	×	×
		mem,imm	1 0 0 0 0 0 s W	mod 0 1 1 mem	3 to 6	(mem) ← (mem) – imm – CY	×	×	×	×	×	×
		acc,imm	0 0 0 1 1 1 0 W		2 to 3	When W = 0, AL ← AL – imm – CY When W = 1, AW ← AW – imm – CY	×	×	×	×	×	×

Group	Mnemonic	Operand	Operation Code		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	×	U	U	U	×
	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	×	U	U	U	×
	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	×	U	U	U	×
	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							
		mem8	0 0 0 0 1 1 1 1 mod 0 0 0 mem	0 0 1 0 1 0 0 0	3 to 5							
	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							
		mem8	0 0 0 0 1 1 1 1 mod 0 0 0 mem	0 0 1 0 1 0 1 0	3 to 5							
Increment/decrement	INC	reg8	1 1 1 1 1 1 1 0	1 1 0 0 0 reg	2	reg8 ← reg8 + 1	×		×	×	×	×
		mem	1 1 1 1 1 1 1 W	mod 0 0 0 mem	2 to 4	(mem) ← (mem) + 1	×		×	×	×	×
		reg16	0 1 0 0 0 reg		1	reg16 ← reg16 + 1	×		×	×	×	×
	DEC	reg8	1 1 1 1 1 1 1 0	1 1 0 0 1 reg	2	reg8 ← reg8 – 1	×		×	×	×	×
		mem	1 1 1 1 1 1 1 W	mod 0 0 1 mem	2 to 4	(mem) ← (mem) – 1	×		×	×	×	×
		reg16	0 1 0 0 1 reg		1	reg16 ← reg16 – 1	×		×	×	×	×

n: 1/2 of the number of BCD digits

Note The number of BCD digits is given in the CL register. The value can be set within 1 to 254.

Group	Mnemonic	Operand	Operation Code		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
Multipli-cation	MULU	reg8	1 1 1 1 0 1 1 0	1 1 1 0 0 reg	2	AW ← AL × reg8 AH = 0: CY ← 0, V ← 0 AH ≠ 0: CY ← 1, V ← 1	U	×	×	U	U	U
		mem8	1 1 1 1 0 1 1 0	mod 1 0 0 mem	2 to 4	AW ← AL × (mem8) AH = 0: CY ← 0, V ← 0 AH ≠ 0: CY ← 1, V ← 1	U	×	×	U	U	U
		reg16	1 1 1 1 0 1 1 1	1 1 1 0 0 reg	2	DW, AW ← AW × reg16 DW = 0: CY ← 0, V ← 0 DW = 1: CY ← 1, V ← 1	U	×	×	U	U	U
		mem16	1 1 1 1 0 1 1 1	mod 1 0 0 mem	2 to 4	DW, AW ← AW × (mem16) DW = 0: CY ← 0, V ← 0 DW = 1: CY ← 1, V ← 1	U	×	×	U	U	U
	MUL	reg8	1 1 1 1 0 1 1 0	1 1 1 0 1 reg	2	AW ← AL × reg8 Extension of AH = AL sign: CY ← 0, V ← 0 Extension of AH ≠ AL sign: CY ← 1, V ← 1	U	×	×	U	U	U
		mem8	1 1 1 1 0 1 1 0	mod 1 0 1 mem	2 to 4	AW ← AL × (mem8) Extension of AH = AL sign: CY ← 0, V ← 0 Extension of AH ≠ AL sign: CY ← 1, V ← 1	U	×	×	U	U	U
		reg16	1 1 1 1 0 1 1 1	1 1 1 0 1 reg	2	DW, AW ← AW × reg16 Extension of DW = AW sign: CY ← 0, V ← 0 Extension of DW ≠ AW sign: CY ← 1, V ← 1	U	×	×	U	U	U
		mem16	1 1 1 1 0 1 1 1	mod 1 0 1 mem	2 to 4	DW, AW ← AW × (mem16) Extension of DW = AW sign: CY ← 0, V ← 0 Extension of DW ≠ AW sign: CY ← 1, V ← 1	U	×	×	U	U	U
		reg16, (reg16,)[Note] imm8	0 1 1 0 1 0 1 1	1 1 reg reg	3	reg16 ← reg16 × imm8 Product ≤ 16 bits: CY ← 0, V ← 0 Product > 16 bits: CY ← 1, V ← 1	U	×	×	U	U	U
		reg16, mem16, imm8	0 1 1 0 1 0 1 1	mod reg mem	3 to 5	reg16 ← (mem16) × imm8 Product ≤ 16 bits: CY ← 0, V ← 0 Product > 16 bits: CY ← 1, V ← 1	U	×	×	U	U	U
		reg16, (reg16,)[Note] imm16	0 1 1 0 1 0 0 1	1 1 reg reg	4	reg16 ← reg16 × imm16 Product ≤ 16 bits: CY ← 0, V ← 0 Product > 16 bits: CY ← 1, V ← 1	U	×	×	U	U	U
		reg16, mem16, imm16	0 1 1 0 1 0 0 1	mod reg mem	4 to 6	reg16 ← (mem16) × imm16 Product ≤ 16 bits: CY ← 0, V ← 0 Product > 16 bits: CY ← 1, V ← 1	U	×	×	U	U	U

Note The 2nd operand is omissible. If omitted, the 1st operand is assumed.

Group	Mnemonic	Operand	Operation Code		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
Unsign-ed 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 to 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 to 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		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
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 to 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 to 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		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	×	×	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	×	×	U	×	×	×
	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	×	×	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	×	×	U	×	×	×
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	×	×	×
	CVTDB		1 1 0 1 0 1 0 1	0 0 0 0 1 0 1 0	2	AL ← AH × 0AH + AL, AH ← 0	U	U	U	×	×	×
	CVTBW		1 0 0 1 1 0 0 0		1	When AL < 80H, AH ← 0. In other cases, AH ← FFH.						
	CVTWL		1 0 0 1 1 0 0 1		1	When AW < 8000H, DW ← 0. In other cases, DW ← FFFFH.						
Compare	CMP	reg,reg	0 0 1 1 1 0 1 W	1 1 reg reg	2	reg – reg	×	×	×	×	×	×
		mem,reg	0 0 1 1 1 0 0 W	mod reg mem	2 to 4	(mem) – reg	×	×	×	×	×	×
		reg,mem	0 0 1 1 1 0 1 W	mod reg mem	2 to 4	reg – (mem)	×	×	×	×	×	×
		reg,imm	1 0 0 0 0 0 s W	1 1 1 1 1 reg	3 to 4	reg – imm	×	×	×	×	×	×
		mem,imm	1 0 0 0 0 0 s W	mod 1 1 1 mem	3 to 6	(mem) – imm	×	×	×	×	×	×
		acc,imm	0 0 1 1 1 1 0 W		2 to 3	When W = 0, AL – imm When W = 1, AW – imm	×	×	×	×	×	×
Complement operation	NOT	reg	1 1 1 1 0 1 1 W	1 1 0 1 0 reg	2	reg ← $\overline{\text{reg}}$						
		mem	1 1 1 1 0 1 1 W	mod 0 1 0 mem	2 to 4	(mem) ← $(\overline{\text{mem}})$						
	NEG	reg	1 1 1 1 0 1 1 W	1 1 0 1 1 reg	2	reg ← $\overline{\text{reg}}$ + 1	×	×	×	×	×	×
		mem	1 1 1 1 0 1 1 W	mod 0 1 1 mem	2 to 4	(mem) ← $(\overline{\text{mem}})$ + 1	×	×	×	×	×	×

Group	Mnemonic	Operand	Operation Code		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
Logical opera-tion	TEST	reg,reg	1 0 0 0 0 1 0 W	1 1 reg reg	2	reg ∧ reg	U	0	0	×	×	×
		mem,reg reg,mem	1 0 0 0 0 1 0 W	mod reg mem	2 to 4	(mem) ∧ reg	U	0	0	×	×	×
		reg,imm	1 1 1 1 0 1 1 W	1 1 0 0 0 reg	3 to 4	reg ∧ imm	U	0	0	×	×	×
		mem,imm	1 1 1 1 0 1 1 W	mod 0 0 0 mem	3 to 6	(mem) ∧ imm	U	0	0	×	×	×
		acc,imm	1 0 1 0 1 0 0 W		2 to 3	When W = 0, AL ∧ imm8 When W = 1, AW ∧ imm16	U	0	0	×	×	×
	AND	reg,reg	0 0 1 0 0 0 1 W	1 1 reg reg	2	reg ← reg ∧ reg	U	0	0	×	×	×
		mem,reg	0 0 1 0 0 0 0 W	mod reg mem	2 to 4	(mem) ← (mem) ∧ reg	U	0	0	×	×	×
		reg,mem	0 0 1 0 0 0 1 W	mod reg mem	2 to 4	reg ← reg ∧ (mem)	U	0	0	×	×	×
		reg,imm	1 0 0 0 0 0 0 W	1 1 1 0 0 reg	3 to 4	reg ← reg ∧ imm	U	0	0	×	×	×
		mem,imm	1 0 0 0 0 0 0 W	mod 1 0 0 mem	3 to 6	(mem) ← (mem) ∧ imm	U	0	0	×	×	×
		acc,imm	0 0 1 0 0 1 0 W		2 to 3	When W = 0, AL ← AL ∧ imm8 When W = 1, AW ← AW ∧ imm16	U	0	0	×	×	×
	OR	reg,reg	0 0 0 0 1 0 1 W	1 1 reg reg	2	reg ← reg ∨ reg	U	0	0	×	×	×
		mem,reg	0 0 0 0 1 0 0 W	mod reg mem	2 to 4	(mem) ← (mem) ∨ reg	U	0	0	×	×	×
		reg,mem	0 0 0 0 1 0 1 W	mod reg mem	2 to 4	reg ← reg ∨ (mem)	U	0	0	×	×	×
		reg,imm	1 0 0 0 0 0 0 W	1 1 0 0 1 reg	3 to 4	reg ← reg ∨ imm	U	0	0	×	×	×
		mem,imm	1 0 0 0 0 0 0 W	mod 0 0 1 mem	3 to 6	(mem) ← (mem) ∨ imm	U	0	0	×	×	×
		acc,imm	0 0 0 0 1 1 0 W		2 to 3	When W = 0, AL ← AL ∨ imm8 When W = 1, AW ← AW ∨ imm16	U	0	0	×	×	×
	XOR	reg,reg	0 0 1 1 0 0 1 W	1 1 reg reg	2	reg ← reg ∀ reg	U	0	0	×	×	×
		mem,reg	0 0 1 1 0 0 0 W	mod reg mem	2 to 4	(mem) ← (mem) ∀ reg	U	0	0	×	×	×
		reg,mem	0 0 1 1 0 0 1 W	mod reg mem	2 to 4	reg ← reg ∀ (mem)	U	0	0	×	×	×
		reg,imm	1 0 0 0 0 0 0 W	1 1 1 1 0 reg	3 to 4	reg ← reg ∀ imm	U	0	0	×	×	×
		mem,imm	1 0 0 0 0 0 0 W	mod 1 1 0 mem	3 to 6	(mem) ← (mem) ∀ imm	U	0	0	×	×	×
		acc,imm	0 0 1 1 0 1 0 W		2 to 3	When W = 0, AL ← AL ∀ imm8 When W = 1, AW ← AW ∀ imm16	U	0	0	×	×	×

Group	Mnemonic	Operand	Operation Code		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
Bit manipulation	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	×
		mem8,CL	0 0 0 0	mod 0 0 0 mem	3 to 5	(mem8) bit No. CL = 0: Z ← 1 (mem8) bit No. CL = 1: Z ← 0	U	0	0	U	U	×
		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	×
		mem16,CL	0 0 0 1	mod 0 0 0 mem	3 to 5	(mem16) bit No. CL = 0: Z ← 1 (mem16) bit No. CL = 1: Z ← 0	U	0	0	U	U	×
		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	×
		mem8,imm3	1 0 0 0	mod 0 0 0 mem	4 to 6	(mem8) bit No. imm3 = 0: Z ← 1 (mem8) bit No. imm3 = 1: Z ← 0	U	0	0	U	U	×
		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	×
		mem16,imm4	1 0 0 1	mod 0 0 0 mem	4 to 6	(mem16) bit No. imm4 = 0: Z ← 1 (mem16) bit No. imm4 = 1: Z ← 0	U	0	0	U	U	×
	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 to 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 to 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 to 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. imm4 ← $\overline{\text{reg16 bit No. imm4}}$						
		mem16,imm4	1 1 1 1	mod 0 0 0 mem	4 to 6	(mem16) bit No. imm4 ← $\overline{\text{(mem16) bit No. imm4}}$						

2nd byte [Note] 3rd byte [Note]

Note 1st byte = 0FH

	NOT1	CY	1 1 1 1 0 1 0 1		1	CY ← $\overline{\text{CY}}$		×				

Group	Mnemonic	Operand	Operation Code		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
Bit manipu-lation	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 to 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 to 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 to 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 to 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 to 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 to 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 to 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 to 6	(mem16) bit No. imm4 ← 1						
			2nd byte [Note]	3rd byte [Note]								

Note 1st 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		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 × 2 When reg MSB ≠ CY, V ← 1 When reg MSB = CY, V ← 0	U	×	×	×	×	×
		mem,1	1 1 0 1 0 0 0 W	mod 1 0 0 mem	2 to 4	CY ← (mem) MSB, (mem) ← (mem) × 2 When (mem) MSB ≠ CY, V ← 1 When (mem) MSB = CY, V ← 0	U	×	×	×	×	×
		reg,CL	1 1 0 1 0 0 1 W	1 1 1 0 0 reg	2	The following operations are repeated while temp ← CL and temp ≠ 0. CY ← reg MSB, reg ← reg × 2 temp ← temp – 1	U	×	U	×	×	×
		mem,CL	1 1 0 1 0 0 1 W	mod 1 0 0 mem	2 to 4	The following operations are repeated while temp ← CL and temp ≠ 0. CY ← (mem) MSB, (mem) ← (mem) × 2 temp ← temp – 1	U	×	U	×	×	×
		reg,imm8	1 1 0 0 0 0 0 W	1 1 1 0 0 reg	3	The following operations are repeated while temp ← imm8 and temp ≠ 0. CY ← reg MSB, reg ← reg × 2 temp ← temp – 1	U	×	U	×	×	×
		mem,imm8	1 1 0 0 0 0 0 W	mod 1 0 0 mem	3 to 5	The following operations are repeated while temp ← imm8 and temp ≠ 0. CY ← (mem) MSB, (mem) ← (mem) × 2 temp ← temp – 1	U	×	U	×	×	×

Group	Mnemonic	Operand	Operation Code		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 ≠ bit following reg MSB: V ← 1 reg MSB = bit following reg MSB: V ← 0	U	×	×	×	×	×
		mem,1	1 1 0 1 0 0 0 W	mod 1 0 1 mem	2 to 4	CY ← (mem) LSB, (mem) ← (mem) ÷ 2 (mem) MSB ≠ bit following (mem) MSB: V ← 1 (mem) MSB = bit following (mem) MSB: V ← 0	U	×	×	×	×	×
		reg,CL	1 1 0 1 0 0 1 W	1 1 1 0 1 reg	2	The following operations are repeated while temp ← CL and temp ≠ 0. CY ← reg LSB, reg ← reg ÷ 2 temp ← temp – 1	U	×	U	×	×	×
		mem,CL	1 1 0 1 0 0 1 W	mod 1 0 1 mem	2 to 4	The following operations are repeated while temp ← CL and temp ≠ 0. CY ← (mem) LSB, (mem) ← (mem) ÷ 2 temp ← temp – 1	U	×	U	×	×	×
		reg,imm8	1 1 0 0 0 0 0 W	1 1 1 0 1 reg	3	The following operations are repeated while temp ← imm8 and temp ≠ 0. CY ← reg LSB, reg ← reg ÷ 2 temp ← temp – 1	U	×	U	×	×	×
		mem,imm8	1 1 0 0 0 0 0 W	mod 1 0 1 mem	3 to 5	The following operations are repeated while temp ← imm8 and temp ≠ 0. CY ← (mem) LSB, (mem) ← (mem) ÷ 2 temp ← temp – 1	U	×	U	×	×	×
	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 The operand MSB remains the same status.	U	×	0	×	×	×
		mem,1	1 1 0 1 0 0 0 W	mod 1 1 1 mem	2 to 4	CY ← (mem) LSB, (mem) ← (mem) ÷ 2, V ← 0 The operand MSB remains the same status.	U	×	0	×	×	×
		reg,CL	1 1 0 1 0 0 1 W	1 1 1 1 1 reg	2	The following operations are repeated while temp ← CL and temp ≠ 0. CY ← reg LSB, reg ← reg ÷ 2 temp ← temp – 1 The operand MSB remains the same status.	U	×	U	×	×	×
		mem,CL	1 1 0 1 0 0 1 W	mod 1 1 1 mem	2 to 4	The following operations are repeated while temp ← CL and temp ≠ 0. CY ← (mem) LSB, (mem) ← (mem) ÷ 2 temp ← temp – 1 The operand MSB remains the same status.	U	×	U	×	×	×
		reg,imm8	1 1 0 0 0 0 0 W	1 1 1 1 1 reg	3	The following operations are repeated while temp ← imm8 and temp ≠ 0. CY ← reg LSB, reg ← reg ÷ 2 temp ← temp – 1 The operand MSB remains the same status.	U	×	U	×	×	×
		mem,imm8	1 1 0 0 0 0 0 W	mod 1 1 1 mem	3 to 5	The following operations are repeated while temp ← imm8 and temp ≠ 0. CY ← (mem) LSB, (mem) ← (mem) ÷ 2 temp ← temp – 1 The operand MSB remains the same status.	U	×	U	×	×	×

Group	Mnemonic	Operand	Operation Code		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	ROL	reg,1	1 1 0 1 0 0 0 W	1 1 0 0 0 reg	2	CY ← reg MSB, reg ← reg × 2 + CY reg MSB ≠ CY: V ← 1 reg MSB = CY: V ← 0		×	×			
		mem,1	1 1 0 1 0 0 0 W	mod 0 0 0 mem	2 to 4	CY ← (mem) MSB, (mem) ← (mem) × 2 + CY (mem) MSB ≠ CY: V ← 1 (mem) MSB = CY: V ← 0		×	×			
		reg,CL	1 1 0 1 0 0 1 W	1 1 0 0 0 reg	2	The following operations are repeated while temp ← CL and temp ≠ 0. CY ← reg MSB, reg ← reg × 2 + CY temp ← temp – 1		×	U			
		mem,CL	1 1 0 1 0 0 1 W	mod 0 0 0 mem	2 to 4	The following operations are repeated while temp ← CL and temp ≠ 0. CY ← (mem) MSB, (mem) ← (mem) × 2 + CY temp ← temp – 1		×	U			
		reg,imm8	1 1 0 0 0 0 0 W	1 1 0 0 0 reg	3	The following operations are repeated while temp ← imm8 and temp ≠ 0. CY ← reg MSB, reg ← reg × 2 + CY temp ← temp – 1		×	U			
		mem,imm8	1 1 0 0 0 0 0 W	mod 0 0 0 mem	3 to 5	The following operations are repeated while temp ← imm8 and temp ≠ 0. CY ← (mem) MSB, (mem) ← (mem) × 2 + CY temp ← temp – 1		×	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 ≠ bit following reg MSB: V ← 1 reg MSB = bit following reg MSB: V ← 0		×	×			
		mem,1	1 1 0 1 0 0 0 W	mod 0 0 1 mem	2 to 4	CY ← (mem) LSB, (mem) ← (mem) ÷ 2 (mem) MSB ← CY (mem) MSB ≠ bit following (mem) MSB: V ← 1 (mem) MSB = bit following (mem) MSB: V ← 0		×	×			
		reg,CL	1 1 0 1 0 0 1 W	1 1 0 0 1 reg	2	The following operations are repeated while temp ← CL and temp ≠ 0. CY ← reg LSB, reg ← reg ÷ 2 reg MSB ← CY temp ← temp – 1		×	U			
		mem,CL	1 1 0 1 0 0 1 W	mod 0 0 1 mem	2 to 4	The following operations are repeated while temp ← CL and temp ≠ 0. CY ← (mem) LSB, (mem) ← (mem) ÷ 2 (mem) MSB ← CY temp ← temp – 1		×	U			
		reg,imm8	1 1 0 0 0 0 0 W	1 1 0 0 1 reg	3	The following operations are repeated while temp ← imm8 and temp ≠ 0. CY ← reg LSB, reg ← reg ÷ 2 reg MSB ← CY temp ← temp – 1		×	U			
		mem,imm8	1 1 0 0 0 0 0 W	mod 0 0 1 mem	3 to 5	The following operations are repeated while temp ← imm8 and temp ≠ 0. CY ← (mem) LSB, (mem) ← (mem) ÷ 2 (mem) MSB ← CY temp ← temp – 1		×	U			

Group	Mnemonic	Operand	Operation Code		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	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 × 2 + tmpcy reg MSB ≠ CY: V ← 1 reg MSB = CY: V ← 0		×	×			
		mem,1	1 1 0 1 0 0 0 W	mod 0 1 0 mem	2 to 4	tmpcy ← CY, CY ← (mem) MSB (mem) ← (mem) × 2 + tmpcy (mem) MSB ≠ CY: V ← 1 (mem) MSB = CY: V ← 0		×	×			
		reg,CL	1 1 0 1 0 0 1 W	1 1 0 1 0 reg	2	The following operations are repeated while temp ← CL and temp ≠ 0. tmpcy ← CY, CY ← reg MSB reg ← reg × 2 + tmpcy temp ← temp – 1		×	U			
		mem,CL	1 1 0 1 0 0 1 W	mod 0 1 0 mem	2 to 4	The following operations are repeated while temp ← CL and temp ≠ 0. tmpcy ← CY, CY ← (mem) MSB (mem) ← (mem) × 2 + tmpcy temp ← temp – 1		×	U			
		reg,imm8	1 1 0 0 0 0 0 W	1 1 0 1 0 reg	3	The following operations are repeated while temp ← imm8 and temp ≠ 0. tmpcy ← CY, CY ← reg MSB reg ← reg × 2 + tmpcy temp ← temp – 1		×	U			
		mem,imm8	1 1 0 0 0 0 0 W	mod 0 1 0 mem	3 to 5	The following operations are repeated while temp ← imm8 and temp ≠ 0. tmpcy ← CY, CY ← (mem) MSB (mem) ← (mem) × 2 + tmpcy temp ← temp – 1		×	U			

Group	Mnemonic	Operand	Operation Code		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 ≠ bit following reg MSB: V ← 1 reg MSB = bit following reg MSB: V ← 0		×	×			
		mem,1	1 1 0 1 0 0 0 W	mod 0 1 1 mem	2 to 4	tmpcy ← CY, CY ← (mem) LSB (mem) ← (mem) ÷ 2 (mem) MSB ← tmpcy (mem) MSB ≠ bit following (mem) MSB: V ← 1 (mem) MSB = bit following (mem) MSB: V ← 0		×	×			
		reg,CL	1 1 0 1 0 0 1 W	1 1 0 1 1 reg	2	The following operations are repeated while temp ← CL and temp ≠ 0. tmpcy ← CY, CY ← reg LSB reg ← reg ÷ 2 reg MSB ← tmpcy temp ← temp – 1		×	U			
		mem,CL	1 1 0 1 0 0 1 W	mod 0 1 1 mem	2 to 4	The following operations are repeated while temp ← CL and temp ≠ 0. tmpcy ← CY, CY ← (mem) LSB (mem) ← (mem) ÷ 2 (mem) MSB ← tmpcy temp ← temp – 1		×	U			
		reg,imm8	1 1 0 0 0 0 0 W	1 1 0 1 1 reg	3	The following operations are repeated while temp ← imm8 and temp ≠ 0. tmpcy ← CY, CY ← reg LSB reg ← reg ÷ 2 reg MSB ← tmpcy temp ← temp – 1		×	U			
		mem,imm8	1 1 0 0 0 0 0 W	mod 0 1 1 mem	3 to 5	The following operations are repeated while temp ← imm8 and temp ≠ 0. tmpcy ← CY, CY ← (mem) LSB (mem) ← (mem) ÷ 2 (mem) MSB ← tmpcy temp ← temp – 1		×	U			

Group	Mnemonic	Operand	Operation Code		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
Sub-routine 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 to 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 to 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						

Group	Mnemonic	Operand	Operation Code		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
Stack manipulation	PUSH	mem16	1 1 1 1 1 1 1 1	mod 1 1 0 mem	2 to 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						
		imm8	0 1 1 0 1 0 1 0		2	(SP – 1, SP – 2) ← imm8 sign extension SP ← SP – 2						
		imm16	0 1 1 0 1 0 0 0		3	(SP – 1, SP – 2) ← imm16 SP ← SP – 2						
	POP	mem16	1 0 0 0 1 1 1 1	mod 0 0 0 mem	2 to 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: SS, DS0, DS1 sreg ← (SP – 1, SP – 2)						
		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 to 4	PC ← (memptr16)						
		far-label	1 1 1 0 1 0 1 0		5	PS ← seg PC ← offset						
		memptr32	1 1 1 1 1 1 1 1	mod 1 0 1 mem	2 to 4	PS ← (memptr32 + 2) PC ← (memptr32)						

Group	Mnemonic	Operand	Operation Code		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
Condi-tional 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) ∨ Z = 1 PC ← PC + ext-disp8						
	BGT	short-label	1 1 1 1		2	if (S ∀ 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 and (sfr) bit No. imm3 ← 0.						

Note This instruction is newly added to the μPD70108/70116.

Group	Mnemonic	Operand	Operation Code		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 × 4 + 3, n × 4 + 2), PC ← (n × 4 + 1, n × 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	Reports the CPU internal interrupt controller that interrupt service routine operation has ended.						
	CHKIND	reg16,mem32	0 1 1 0 0 0 1 0	mod reg mem	2 to 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 ← lower 3 bits of reg16, 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 ← lower 3 bits of reg16, Old register bank Save PSW and Save PC ← PSW and PC, PSW and PC ← New register bank Save PSW and Save PC	×	×	×	×	×	×
			1 1 1 1 1 reg									

Note These instructions are newly added to the μPD70108/70116.

Group	Mnemonic	Operand	Operation Code		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
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 to 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 to 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: and SS:

2. This instruction is newly added to the μPD70108/70116.

3. In the μPD70320, an interrupt is generated without executing these instructions.

Table 2-8. Number of Clocks (1/10)

Group	Mnemonic	Operands	Byte Processing		Word Processing	
			On-chip RAM Access Enable	On-chip RAM Access Disable	On-chip RAM Access Enable	On-chip RAM Access Disable
Data transfer	MOV	reg, reg	2	2	2	2
		mem, reg	EA + 4 + T	EA + 2	EA + 6 + 2·T	EA + 2
		reg, mem	EA + 6 + T	EA + 6 + T	EA + 8 + 2·T	EA + 8 + 2·T
		mem, imm	EA + 5 + T	EA + 5 + T	EA + 5 + 2·T	EA + 5 + T
		reg, imm	5	5	6	6
		acc, dmem	9 + T	9 + T	11 + 2·T	11 + 2·T
		dmem, acc	7 + T	5	9 + 2·T	5
		sreg, reg16	—	—	4	4
		sreg, mem16	—	—	EA + 10 + 2·T	EA + 10 + 2·T
		reg16, sreg	—	—	3	3
		mem16, sreg	—	—	EA + 7 + 2·T	EA + 3
		DS0, reg16, mem32	—	—	EA + 19 + 4·T	EA + 19 + 4·T
		DS1, reg16, mem32	—	—	EA + 19 + 4·T	EA + 19 + 4·T
		AH, PSW	2	2	—	—
		PSW, AH	3	3	—	—
	LDEA	reg16, mem16	—	—	EA + 2	EA + 2
	TRANS	src-table	10 + T	10 + T	—	—
	XCH	reg, reg	3	3	3	3
		mem, reg/ reg, mem	EA + 10 + 2·T	EA + 8 + 2·T	EA + 14 + 2·T	EA + 10 + 2·T
		AW, reg16/ reg16, AW	—	—	4	4
	MOVSPA		—	—	16	16
	MOVSPB	reg16	—	—	11	11
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	MOVKB[Note]	dst-block, src-block	20 + 2·T	16 + T	24 + 4·T	20 + 2·T
			16 + (16 + 2·T)·n	16 + (12 + T)·n	16 + (20 + 4·T)·n	16 + (12 + 2·T)·n
	CMPKB[Note]	dst-block, src-block	23 + 2·T	19 + T	27 + 4·T	21 + 4·T
			16 + (21 + 2·T)·n	16 + (21 + 2·T)·n	16 + (25 + 4·T)·n	16 + (25 + 2·T)·n

Note n ≥ 1

Table 2-8. Number of Clocks (2/10)

Group	Mnemonic	Operands	Byte Processing		Word Processing	
			On-chip RAM Access Enable	On-chip RAM Access Disable	On-chip RAM Access Enable	On-chip RAM Access Disable
Primitive block transfer	CMPM[Note 1]	dst-block src-block	17 + T	17 + T	19 + 2·T	19 + 2·T
			16 + (15 + T)·n	16 + (15 + T)·n	16 + (17 + 2·T)·n	16 + (17 + 2·T)·n
	LDM[Note 1]	src-block	12 + T	12 + T	14 + 2·T	14 + 2·T
			16 + (10 + T)·n	16 + (10 + T)·n	16 + (12 + 2·T)·n	16 + (12 + 2·T)·n
	STM[Note 1]	dst-block	12 + T	10	14 + 2·T	10
			16 + (8 + T)·n	16 + (6+ T)·n	16 + (10 + 2·T)·n	16 + (6 + 2·T)·n
Bit field manipulation	INS	reg8, reg8	63 to 155 (The processing differs among bit lengths.)			
		reg8, imm4	64 to 156 (The processing differs among bit lengths.)			
	EXT	reg8, reg8	41 to 121 (The processing differs among bit lengths.)			
		reg8, imm4	42 to 122 (The processing differs among bit lengths.)			
I/O	IN[Note 2]	acc, imm8	14 + T	14 + T	16 + 2·T	16 + 2·T
		acc, DW	13 + T	13 + T	15 + 2·T	15 + 2·T
	OUT[Note 2]	imm8, acc	10 + T	10 + T	10 + 2·T	10 + 2·T
		DW, acc	9 + T	9 + T	9 + 2·T	9 + 2·T
Primitive I/O	INM[Note 2]	dst-block, DW	19 + 2·T	17 + 2·T	21 + 4·T	17 + 4·T
			18 + (13 + 2·T)·n	18 + (11 + 2·T)·n	18 + (15 + 4·T)·n	18 + (11 + 4·T)·n
	OUTM[Note 2]	DW, src-block	19 + 2·T	17 + 2·T	21 + 4·T	17 + 4·T
			18 + (13 + 2·T)·n	18 + (11 + 2·T)·n	18 + (15 + 4·T)·n	18 + (11 + 4·T)·n
Addition/ subtraction	ADD	reg, reg	2	2	2	2
		mem, reg	EA + 8 + 2·T	EA + 6 + T	EA + 12 + 4·T	EA + 8 + 2·T
		reg, mem	EA + 6 + T	EA + 6 + T	EA + 8 + 2·T	EA + 8 + 2·T
		reg, imm	5	5	6	6
		mem, imm	EA + 9 + 2·T	EA + 7 + 2·T	EA + 14 + 4·T	EA + 10 + 4·T
		acc, imm	5	5	6	6
	ADDC	reg, reg	2	2	2	2
		mem, reg	EA + 8 + 2·T	EA + 6 + T	EA + 12 + 4·T	EA + 8 + 2·T
		reg, mem	EA + 6 + T	EA + 6 + T	EA + 8 + 2·T	EA + 8 + 2·T
		reg, imm	5	5	6	6
		mem, imm	EA + 9 + 2·T	EA + 7 + 2·T	EA + 14 + 4·T	EA + 10 + 4·T
		acc, imm	5	5	6	6

Notes **1.** $n \geq 1$

2. When $\overline{\text{IBRK}} = 1$

Table 2-8. Number of Clocks (3/10)

Group	Mnemonic	Operands	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/ subtraction	SUB	reg, reg	2	2	2	2
		mem, reg	EA + 8 + 2·T	EA + 6 + T	EA + 12 + 4·T	EA + 8 + 2·T
		reg, mem	EA + 6 + T	EA + 6 + T	EA + 8 + 2·T	EA + 8 + 2·T
		reg, imm	5	5	6	6
		mem, imm	EA + 9 + 2·T	EA + 7 + 2·T	EA + 14 + 4·T	EA + 10 + 4·T
		acc, imm	5	5	6	6
	SUBC	reg, reg	2	2	2	2
		mem, reg	EA + 8 + 2·T	EA + 6 + T	EA + 12 + 4·T	EA + 8 + 2·T
		reg, mem	EA + 6 + T	EA + 6 + T	EA + 8 + 2·T	EA + 8 + 2·T
		reg, imm	5	5	6	6
		mem, imm	EA + 9 + 2·T	EA + 7 + 2·T	EA + 14 + 4·T	EA + 10 + 4·T
		acc, imm	5	5	6	6
BCD operation	ADD4S^Note		22 + (27 + 3·T)·n	22 + (25 + 3·T)·n	—	—
	SUB4S^Note		22 + (27 + 3·T)·n	22 + (25 + 3·T)·n	—	—
	CMP4S^Note		22 + (23 + 3·T)·n	22 + (23 + 3·T)·n	—	—
	ROL4	reg8	17	17	—	—
		mem8	EA + 18 + 2·T	EA + 16 + 2·T	—	—
	ROR4	reg8	21	21	—	—
		mem8	EA + 24 + 2·T	EA + 22 + 2·T	—	—
Increment/ decrement	INC	reg8	5	5	—	—
		mem8	EA + 11 + 2·T	EA + 9 + 2·T	EA + 15 + 4·T	EA + 11 + 4·T
		reg16	—	—	2	2
	DEC	reg8	5	5	—	—
		mem8	EA + 11 + 2·T	EA + 9 + 2·T	EA + 15 + 4·T	EA + 11 + 4·T
		reg16	—	—	2	2
Multiplication	MULU	reg8	24	24	—	—
		mem8	EA + 26 + T	EA + 26 + T	—	—
		reg16	—	—	32	32
		mem16	—	—	EA + 34 + 2·T	EA + 34 + 2·T

Note n: 1/2 of the number of BCD digits.

Table 2-8. Number of Clocks (4/10)

Group	Mnemonic	Operands	Byte Processing		Word Processing	
			On-chip RAM Access Enable	On-chip RAM Access Disable	On-chip RAM Access Enable	On-chip RAM Access Disable
Multiplication	MUL	reg8	31 to 40	31 to 40	—	—
		mem8	EA + 33 + T to EA + 42 + T	EA + 33 + T to EA + 42 + T	—	—
		reg16	—	—	39 to 48	39 to 48
		mem16	—	—	EA + 43 + 2·T to EA + 52 + 2·T	EA + 43 + 2·T to EA + 52 + 2·T
		reg16, (reg16,) imm8	—	—	39 to 49	39 to 49
		reg16, mem16, imm8	—	—	EA + 43 + 2·T to EA + 53 + 2·T	EA + 43 + 2·T to EA + 53 + 2·T
		reg16, (reg16,) imm16	—	—	40 to 50	40 to 50
		reg16, mem16, imm16	—	—	EA + 44 + 2·T to EA + 54 + 2·T	EA + 44 + 2·T to EA + 54 + 2·T
Unsigned division	DIVU	reg8	31	31	—	—
		mem8	EA + 33 + T	EA + 33 + T	—	—
		reg16	—	—	39	39
		mem16	—	—	EA + 43 + 2·T	EA + 43 + 2·T
Signed division	DIV	reg8	46 to 56	46 to 56	—	—
		mem8	EA + 48 + T to EA + 58 + T	EA + 48 + T to EA + 58 + T	—	—
		reg16	—	—	54 to 64	54 to 64
		mem16	—	—	EA + 58 + 2·T to EA + 68 + 2·T	EA + 58 + 2·T to EA + 68 + 2·T
BCD adjustment	ADJBA		17	17	—	—
	ADJ4A		9	9	—	—
	ADJBS		17	17	—	—
	ADJ4S		9	9	—	—
Data conversion	CVTBD		19	19	—	—
	CVTDB		20	20	—	—
	CVTBW		3	3	—	—
	CVTWL		—	—	8	8
Compare	CMP	reg, reg	2	2	2	2
		mem, reg	EA + 6 + T	EA + 6 + T	EA + 8 + 2·T	EA + 8 + 2·T
		reg, mem	EA + 6 + T	EA + 6 + T	EA + 8 + 2·T	EA + 8 + 2·T
		reg, imm	5	5	6	6
		mem, imm	EA + 7 + T	EA + 7 + T	EA + 10 + 2·T	EA + 10 + 2·T
		acc, imm	5	5	6	6

Table 2-8. Number of Clocks (5/10)

Group	Mnemonic	Operands	Byte Processing		Word Processing	
			On-chip RAM Access Enable	On-chip RAM Access Disable	On-chip RAM Access Enable	On-chip RAM Access Disable
Complement operation	NOT	reg	5	5	5	5
		mem	EA + 11 + 2·T	EA + 9 + T	EA + 15 + 4·T	EA + 11 + 2·T
	NEG	reg	5	5	5	5
		mem	EA + 11 + 2·T	EA + 9 + T	EA + 15 + 4·T	EA + 11 + 2·T
Logical operation	TEST	reg, reg	4	4	4	4
		mem, reg/ reg, mem	EA + 8 + T	EA + 8 + T	EA + 10 + 2·T	EA + 10 + 2·T
		reg, imm	7	7	8	8
		mem, imm	EA + 11 + T	EA + 11 + T	EA + 11 + 2·T	EA + 11 + 2·T
		acc, imm	5	5	6	6
	AND	reg, reg	2	2	2	2
		mem, reg	EA + 8 + 2·T	EA + 6 + T	EA + 12 + 4·T	EA + 8 + 2·T
		reg, mem	EA + 6 + T	EA + 6 + T	EA + 8 + 2·T	EA + 8 + 2·T
		reg, imm	5	5	6	6
		mem, imm	EA + 9 + T	EA + 7 + T	EA + 14 + 4·T	EA + 10 + 4·T
		acc, imm	5	5	6	6
	OR	reg, reg	2	2	2	2
		mem, reg	EA + 8 + 2·T	EA + 6 + T	EA + 12 + 4·T	EA + 8 + 2·T
		reg, mem	EA + 6 + T	EA + 6 + T	EA + 8 + 2·T	EA + 8 + 2·T
		reg, imm	5	5	6	6
		mem, imm	EA + 9 + T	EA + 7 + T	EA + 14 + 4·T	EA + 10 + 4·T
		acc, imm	5	5	6	6
	XOR	reg, reg	2	2	2	2
		mem, reg	EA + 8 + 2·T	EA + 6 + T	EA + 12 + 4·T	EA + 8 + 2·T
		reg, mem	EA + 6 + T	EA + 6 + T	EA + 8 + 2·T	EA + 8 + 2·T
		reg, imm	5	5	6	6
		mem, imm	EA + 9 + T	EA + 7 + T	EA + 14 + 4·T	EA + 10 + 4·T
		acc, imm	5	5	6	6
Bit manipulation	TEST1	reg8, CL	7	7	—	—
		mem8, CL	EA + 11 + T	EA + 11 + T	—	—
		reg16, CL	—	—	7	7
		mem16, CL	—	—	EA + 13 + 2·T	EA + 13 + 2·T
		reg8, imm3	6	6	—	—
		mem8, imm3	EA + 8 + T	EA + 8 + T	—	—
		reg16, imm4	—	—	6	6
		mem16, imm4	—	—	EA + 10 + 2·T	EA + 10 + 2·T
	NOT1	reg8, CL	7	7	—	—
		mem8, CL	EA + 13 + 2·T	EA + 11 + T	—	—
		reg16, CL	—	—	7	7

Table 2-8. Number of Clocks (6/10)

Group	Mnemonic	Operands	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	mem16, CL	—	—	EA + 17 + 4·T	EA + 13 + 2·T
		reg8, imm3	6	6	—	—
		mem8, imm3	EA + 10 + 2·T	EA + 8 + T	—	—
		reg16, imm4	—	—	6	6
		mem16, imm4	—	—	EA + 14 + 4·T	EA + 10 + 2·T
	NOT1	CY	2	2	2	2
Bit manipulation	CLR1	reg8, CL	8	8	—	—
		mem8, CL	EA + 14 + 2·T	EA + 12 + T	—	—
		reg16, CL	—	—	8	8
		mem16, CL	—	—	EA + 18 + 4·T	EA + 14 + 2·T
		reg8, imm3	7	7	—	—
		mem8, imm3	EA + 11 + 2·T	EA + 9 + T	—	—
		reg16, imm4	—	—	7	7
		mem16, imm4	—	—	EA + 15 + 4·T	EA + 10 + 2·T
	SET1	reg8, CL	7	7	—	—
		mem8, CL	EA + 13 + 2·T	EA + 11 + T	—	—
		reg16, CL	—	—	7	7
		mem16, CL	—	—	EA + 17 + 4·T	EA + 13 + 2·T
		reg8, imm3	6	6	—	—
		mem8, imm3	EA + 10 + 2·T	EA + 8 + T	—	—
		reg16, imm4	—	—	6	6
		mem16, imm4	—	—	EA + 14 + 4·T	EA + 10 + 2·T
	CLR1	CY	2	2	2	2
		DIR	2	2	2	2
	SET1	CY	2	2	2	2
		DIR	2	2	2	2
Shift	SHL	reg,1	8	8	8	8
	Note	mem, 1	EA + 14 + 2·T	EA + 12 + T	EA + 18 + 4·T	EA + 14 + 2·T
		reg, CL	11 + 2·n	11 + 2·n	11 + 2·n	11 + 2·n
		mem, CL	EA + 17 + 2·T + 2·n	EA + 15 + T + 2·n	EA + 21 + 4·T + 2·n	EA + 17 + 2·T + 2·n
		reg, imm8	9 + 2·n	9 + 2·n	9 + 2·n	9 + 2·n
		mem, imm8	EA + 13 + 2·T + 2·n	EA + 11 + T + 2·n	EA + 17 + 4·T + 2·n	EA + 13 + 2·T + 2·n
	SHR	reg, 1	8	8	8	8
		mem, 1	EA + 14 + 2·T	EA + 12 + T	EA + 18 + 4·T	EA + 14 + 2·T

Note n: Shift count

Table 2-8. Number of Clocks (7/10)

Group	Mnemonic	Operands	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	SHR	reg, CL	11 + 2·n	11 + 2·n	11 + 2·n	11 + 2·n
	Note	mem, CL	EA + 17 + 2·T + 2·n	EA + 15 + T + 2·n	EA + 21 + 4·T + 2·n	EA + 17 + 2·T + 2·n
		reg, imm8	9 + 2·n	9 + 2·n	9 + 2·n	9 + 2·n
		mem, imm8	EA + 13 + 2·T + 2·n	EA + 11 + T + 2·n	EA + 17 + 4·T + 2·n	EA + 13 + 2·T + 2·n
	SHRA	reg,1	8	8	8	8
	Note	mem, 1	EA + 14 + 2·T	EA + 12 + T	EA + 18 + 4·T	EA + 14 + 2·T
		reg, CL	11 + 2·n	11 + 2·n	11 + 2·n	11 + 2·n
		mem, CL	EA + 17 + 2·T + 2·n	EA + 15 + T + 2·n	EA + 21 + 4·T + 2·n	EA + 17 + 2·T + 2·n
		reg, imm8	9 + 2·n	9 + 2·n	9 + 2·n	9 + 2·n
		mem, imm8	EA + 13 + 2·T + 2·n	EA + 11 + T + 2·n	EA + 17 + 4·T + 2·n	EA + 13 + 2·T + 2·n
Rotate	ROL	reg,1	8	8	8	8
	Note	mem, 1	EA + 14 + 2·T	EA + 12 + T	EA + 18 + 4·T	EA + 14 + 2·T
		reg, CL	11 + 2·n	11 + 2·n	11 + 2·n	11 + 2·n
		mem, CL	EA + 17 + 2·T + 2·n	EA + 15 + T + 2·n	EA + 21 + 4·T + 2·n	EA + 17 + 2·T + 2·n
		reg, imm8	9 + 2·n	9 + 2·n	9 + 2·n	9 + 2·n
		mem, imm8	EA + 13 + 2·T + 2·n	EA + 11 + T + 2·n	EA + 17 + 4·T + 2·n	EA + 13 + 2·T + 2·n
	ROR	reg,1	8	8	8	8
	Note	mem, 1	EA + 14 + 2·T	EA + 12 + T	EA + 18 + 4·T	EA + 14 + 2·T
		reg, CL	11 + 2·n	11 + 2·n	11 + 2·n	11 + 2·n
		mem, CL	EA + 17 + 2·T + 2·n	EA + 15 + T + 2·n	EA + 21 + 4·T + 2·n	EA + 17 + 2·T + 2·n
		reg, imm8	9 + 2·n	9 + 2·n	9 + 2·n	9 + 2·n
		mem, imm8	EA + 13 + 2·T + 2·n	EA + 11 + T + 2·n	EA + 17 + 4·T + 2·n	EA + 13 + 2·T + 2·n
	ROLC	reg,1	8	8	8	8
	Note	mem, 1	EA + 14 + 2·T	EA + 12 + T	EA + 18 + 4·T	EA + 14 + 2·T
		reg, CL	11 + 2·n	11 + 2·n	11 + 2·n	11 + 2·n
		mem, CL	EA + 17 + 2·T + 2·n	EA + 15 + T + 2·n	EA + 21 + 4·T + 2·n	EA + 17 + 2·T + 2·n
		reg, imm8	9 + 2·n	9 + 2·n	9 + 2·n	9 + 2·n
		mem, imm8	EA + 13 + 2·T + 2·n	EA + 11 + T + 2·n	EA + 17 + 4·T + 2·n	EA + 13 + 2·T + 2·n
	RORC	reg,1	8	8	8	8
		mem, 1	EA + 14 + 2·T	EA + 12 + T	EA + 18 + 4·T	EA + 14 + 2·T

Note n: Shift count

Table 2-8. Number of Clocks (8/10)

Group	Mnemonic	Operands	Byte Processing		Word Processing	
			On-chip RAM Access Enable	On-chip RAM Access Disable	On-chip RAM Access Enable	On-chip RAM Access Disable
Rotate	RORC[Note]	reg, CL	11 + 2·n	11 + 2·n	11 + 2·n	11 + 2·n
		mem, CL	EA + 17 + 2·T + 2·n	EA + 15 + T + 2·n	EA + 21 + 4·T + 2·n	EA + 17 + 2·T + 2·n
		reg, imm8	9 + 2·n	9 + 2·n	9 + 2·n	9 + 2·n
		mem, imm8	EA + 13 + 2·T + 2·n	EA + 11 + T + 2·n	EA + 17 + 4·T + 2·n	EA + 13 + 2·T + 2·n
Subroutine control	CALL	near-proc	—	—	22 + 2·T	18 + 2·T
		regptr16	—	—	22 + 2·T	18 + 2·T
		memptr16	—	—	EA + 26 + 4·T	EA + 24 + 4·T
		far-proc	—	—	38 + 4·T	34 + 4·T
		memptr32	—	—	EA + 36 + 8·T	EA + 24 + 8·T
	RET		—	—	20 + 2·T	20 + 2·T
		pop-value	—	—	20 + 2·T	20 + 2·T
			—	—	29 + 4·T	29 + 4·T
		pop-value	—	—	30 + 4·T	30 + 4·T
Stack manipulation	PUSH	mem16	—	—	EA + 18 + 4·T	EA + 14 + 4·T
		reg16	—	—	10 + 2·T	6
		sreg	—	—	11 + 2·T	7
		PSW	—	—	10 + 2·T	6
		R	—	—	82 + 16·T	50
		imm8	—	—	13 + 2·T	9
		imm16	—	—	14 + 2·T	10
	POP	mem16	—	—	EA + 16 + 4·T	EA + 12 + 2·T
		reg16	—	—	12 + 2·T	12 + 2·T
		sreg	—	—	13 + 2·T	13 + 2·T
		PSW	—	—	14 + 2·T	14 + 2·T
		R	—	—	82 + 16·T	58
	PREPARE	imm16, imm8	When imm8 = 0, 27 + 2·T When imm8 = 1, 39 + 4·T When imm8 = n, n > 1, 46 + 19(n – 1) + 4·T			
	DISPOSE		—	—	12 + 2·T	12 + 2·T

Note n: Shift count

Table 2-8. Number of Clocks (9/10)

Group	Mnemonic	Operands	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	near-label	—	—	12	12
		short-label	—	—	12	12
		regptr16	—	—	13	13
		memptr16	—	—	EA + 17 + 2·T	EA + 17 + 2·T
		far-label	—	—	15	15
		memptr32	—	—	EA + 25 + 4·T	EA + 25 + 4·T
Conditional branch	BV	short-label	—	—	15/8	15/8
	BNV	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
	DBNZ	short-label	—	—	17/8	17/8
	BCWZ	short-label	—	—	15/8	15/8
	BTCLR	sfr, imm3, short-label	29/21	29/21	—	—
Interrupt	BRK	3	—	—	55 + 10·T	43 + 10·T
		imm8 (≠3)	—	—	56 + 10·T	44 + 10·T
	BRKV		—	—	55 + 10·T	43 + 10·T
	RETI		—	—	45 + 6·T	37 + 2·T
	RETRBI		—	—	12	12
	FINT		2	2	2	2
	CHKIND	reg16, mem32	—	—	EA + 26 + 4·T	EA + 26 + 4·T

Table 2-8. Number of Clocks (10/10)

Group	Mnemonic	Operands	Byte Processing		Word Processing	
			On-chip RAM Access Enable	On-chip RAM Access Disable	On-chip RAM Access Enable	On-chip RAM Access Disable
Register bank switch	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	—	—	60 + 10·T	48 + 10·T
		fp-op, mem	—	—	60 + 10·T	48 + 10·T
	FPO2	fp-op	—	—	60 + 10·T	48 + 10·T
		fp-op, mem	—	—	60 + 10·T	48 + 10·T
	NOP		4	4	4	4
Segment override prefix (DS0:, DS1:, PS: and SS:)			2	2	2	2

3. ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS (T_A = 25°C)

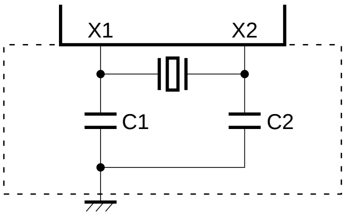
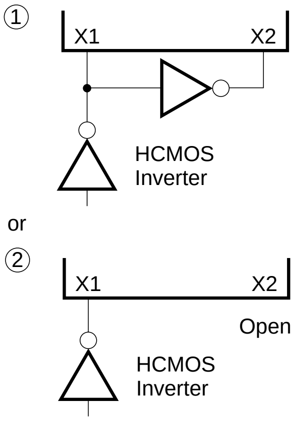
Parameter	Symbol	Test Conditions	Rating	Unit
Supply Voltage	V_{DD}		– 0.5 to +7.0	V
Input Voltage	V_{TH}		– 0.5 to V_{DD} + 0.5	V
	V_I		– 0.5 to V_{DD} + 0.5	V
Output Voltage	V_O		– 0.5 to V_{DD} + 0.5	V
Output Current Low	I_{OL}	Each output pin	4.0	mA
		Total	50	mA
Output Current High	I_{OH}	Each output 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 make direct connections of the output (or input/output) pins of the IC product with each other, and also avoid direct connections to V_{DD}, V_{CC} or GND. However, the open drain pins or the open collector pins can be directly connected with each other. For the external circuit designed with the timing specifications so that any collision of the outputs from the pins subject to high-impedance state may be prevented, direct connection can be also made.

2. Product quality may suffer if the absolute maximum ratings are exceeded for even a single parameter, or even momentarily. In other words, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions which ensure that the absolute maximum ratings are not exceeded. The normal operation and reliability of the product can be only assured with the specifications and the conditions indicated as the DC and AC characteristics.

★ OSCILLATOR CHARACTERISTICS

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

Resonator	Recommended Circuit	Parameter	μPD70320		μPD70320-8		Unit
			MIN.	MAX.	MIN.	MAX.	
Ceramic or Crystal Resonator		Oscillation frequency (f_{XX})	4	10	4	16	MHz
External Clock	 ① X1 X2 HCMOS Inverter or ② X1 X2 Open HCMOS Inverter	X1 input frequency (f_X)	4	10	4	16	MHz
		X1 rise/fall time (t_{XR}, t_{XF})	0	20	0	20	ns
		X1 input high-/low-level width (t_{WXH}, t_{WXL})	35	250	20	250	ns

Cautions 1. Mount the oscillation circuit as close to pins X1 and X2 as possible.

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

RECOMMENDED OSCILLATOR CONSTANT

Ceramic resonator

Manufacturer	Part Number	Recommended Constants	
		C1 [pF]	C2 [pF]
Kyocera Corp.	KBR-10.0M[Note 1]	33	33
Murata Mfg. Co., Ltd.	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.** The operating ambient temperature (T_A) is –10°C to +70°C when this resonator is used.

2. The operating ambient temperature (T_A) is –20°C to +80°C when this resonator is used.

Crystal resonator

Manufacturer	Part Number	Recommended Constants	
		C1 [pF]	C2 [pF]
Kinseki Co., Ltd.	HC-49/U(KR-100)	22	22
	HC-49/U(KR-160)	22	22

Remark For more details on the characteristics of the resonators, please contact the manufacturer.

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

Parameter	Symbol	Test Conditions	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 Conditions		MIN.	TYP.	MAX.	Unit
Input Voltage Low	V_{IL}			0		0.8	V
Input Voltage High	V_{IH1}	Except $\overline{\text{RESET}}$, P10/NMI, X1, X2		2.2		V_{DD}	V
	V_{IH2}	$\overline{\text{RESET}}$, P10/NMI, X1, X2		0.8V_{DD}		V_{DD}	V
Output Voltage Low	V_{OL}	I_{OL} = 1.6 mA				0.45	V
Output Voltage High	V_{OH}	I_{OH} = –0.4 mA		V_{DD} – 1.0			V
Input Current	I_I	$\overline{\text{EA}}$, P10/NMI; 0 ≤ V_I ≤ V_{DD}				±20	μA
Input Leakage Current	I_{LI}	Except $\overline{\text{EA}}$, P10/NMI; 0 ≤ V_I ≤ V_{DD}				±10	μA
Output Leakage Current	I_{LO}	0 ≤ V_O ≤ V_{DD}				±10	μA
V_{TH} Current	I_{TH}	0 V ≤ V_{TH} ≤ V_{DD}			0.5	1.0	mA
V_{DD} Supply Current	I_{DD1}	Operating mode	μPD70320		50	100	mA
			μPD70320-8		65	120	mA
	I_{DD2}	HALT mode	μPD70320		20	40	mA
			μPD70320-8		25	50	mA
	I_{DD3}	STOP mode			10	30	μA

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

Parameter	Symbol	Test Conditions	μPD70320		μPD70320-8		Unit
			MIN.	MAX.	MIN.	MAX.	
X1 Input Cycle Time	t_{CYX}		98	250	62	250	ns
X1 Input High-/Low-Level Width	t_{WXH}, t_{WXL}		35		20		ns
X1 Input Rise/Fall Time	t_{XR}, t_{XF}			20		20	ns
CLKOUT Output Cycle Time	t_{CYK}	f_X/2, T = t_{CYK}	200	2000	125	2000	ns
CLKOUT Output High-/Low-Level Width	t_{WKH}, t_{WKL}		0.5T – 15		0.5T – 15		ns
CLKOUT Output Rise/Fall Time	t_{KR}, t_{KF}			15		15	ns
Input Rise/Fall Time	t_{IR}, t_{IF}	Except $\overline{\text{RESET}}$, NMI, X1 and X2		20		20	ns
	t_{IRS}, t_{IFS}	$\overline{\text{RESET}}$, NMI		30		30	ns
Output Rise/Fall Time	t_{OR}, t_{OF}	Except CLKOUT		20		20	ns

Parameter	Symbol	Test Conditions	MIN.	MAX.	Unit
Address Delay Time from CLKOUT	t_{DKA}			90	ns
Data Input Delay Time from Address	t_{DADR}			(n + 1.5)T – 90	ns
Data Delay Time from $\overline{MREQ}$ ↓	t_{DMRD}			(n + 1)T – 75	ns
Data Delay Time from $\overline{MSTB}$ ↓	t_{DMSD}			(n+ 0.5)T – 75	ns
$\overline{MSTB}$ ↓ Delay Time from $\overline{MREQ}$ ↓	t_{DMRMS}		0.5T – 35	0.5T + 35	ns
$\overline{MREQ}$ Low-Level Width	t_{WMRL}		(n + 1)T – 30	(n + 1)T + 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
Data Output Delay Time from Address	t_{DADW}			0.5T + 50	ns
Address Setup Time (to $\overline{MREQ}$ ↓)	t_{DAMR}		0.5T – 30		ns
Address Setup Time (to $\overline{MSTB}$ ↓)	t_{DAMS}		T – 30		ns
$\overline{MSTB}$ Low-Level Width	t_{WMSL}		(n + 0.5)T – 30	(n + 0.5)T + 30	ns
Data Output Setup Time (to $\overline{MSTB}$ ↑)	t_{SDM}		(n + 1)T – 50		ns
Data Output Hold Time (from $\overline{MSTB}$ ↑)	t_{HMDW}		0.5T – 30		ns
Address Setup Time (to $\overline{IOSTB}$ ↓)	t_{DAIS}		0.5T – 30		ns
Data Delay Time from $\overline{IOSTB}$ ↓	t_{DISD}			(n + 1)T – 90	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 $\overline{IOREQ}$ ↑)	t_{HISDR}		0		ns
Data Output Setup Time (to $\overline{IOSTB}$ ↑)	t_{SDIS}		(n + 1)T – 50		ns
Data Output Hold Time (from $\overline{IOSTB}$ ↑)	t_{HISDW}		0.5T – 30		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 mode	(n + 1.5)T – 30		ns
$\overline{TC}$ ↓ Delay Time from $\overline{DMAAK}$ ↓	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 mode	(n + 1)T – 30		ns
Address Setup Time (to $\overline{REFRQ}$ ↓)	t_{DARF}		0.5T – 30		ns
$\overline{REFRQ}$ Low-Level Width	t_{WRFL}		(n + 1)T – 30		ns
Address Hold Time (from $\overline{REFRQ}$ ↑)	t_{HRFA}		0.5T – 30		ns
$\overline{RESET}$ Low-Level Width	t_{WRSL1}	STOP mode release/ power-ON reset	30		ms
	t_{WRSL2}	System reset	5		μs
READY Setup Time (to $\overline{MREQ}$ ↓, $\overline{IOSTB}$ ↓)	t_{SCRY0}	n ≥ 2		T – 100	ns
	t_{SCRY}	n ≥ 3		(n – 1)T – 100	ns

Parameter	Symbol	Test Conditions	MIN.	MAX.	Unit
READY Hold Time (from $\overline{\text{MREQ}}$ ↓, $\overline{\text{IOSTB}}$ ↓)	t_{HCRY0}	n = 2	1T		ns
	t_{HCRY}	n ≥ 3	(n – 1)T		ns
	t_{HCRY1}	n ≥ 3	(n – 2)T		ns
HLDRQ Setup Time (to CLKOUT ↑)	t_{SHQK}		30		ns
$\overline{\text{HLDAK}}$ ↓ Delay Time from CLKOUT ↑	t_{DKHA}			80	ns
$\overline{\text{HLDAK}}$ ↓ Delay Time from Bus Float	t_{CFHA}		1T – 50		ns
Bus Output Delay Time from $\overline{\text{HLDAK}}$ ↑	t_{DHAC}		1T – 50		ns
$\overline{\text{HLDAK}}$ ↑ Delay Time from HLDRQ ↓	t_{DHQHA}			3T + 160	ns
Bus Output Delay Time from HLDRQ ↓	t_{DHQC}		3T + 30		ns
HLDRQ Low-Level Width	t_{WHQL}		1.5T		ns
$\overline{\text{HLDAK}}$ Low-Level Width	t_{WHAL}		1T		ns
INT, DMARQ Setup Time (to CLKOUT ↑)	t_{SIQK}		30		ns
INT, DMARQ High-/Low-Level Width	t_{WIQH}, t_{WIQL}		8T		ns
$\overline{\text{POLL}}$ 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
$\overline{\text{INTAK}}$ ↓ Delay Time from CLKOUT ↓	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
Data Delay Time from $\overline{\text{INTAK}}$ ↓	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
TxD Delay Time from $\overline{\text{SCK0}}$ ↓	t_{DTKD}			210	ns
TxD Hold Time (from $\overline{\text{SCK0}}$ ↓)	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

Remark n indicates the number of wait states. No wait is “n = 0”.

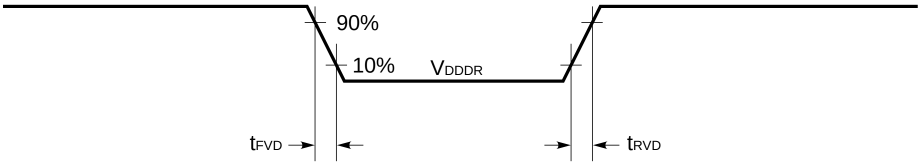
COMPARATOR CHARACTERISTICS (T_A = –40°C to +85°C, V_{DD} = +5.0 V ±10%)

Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Comparator Accuracy	V_{ACOMP}				±100	mV
Threshold Voltage	V_{TH}		0		V_{DD} + 0.1	V
Compare Time	t_{COMP}		64		65	t_{CYK}
PT Input Voltage	V_{IPT}		0		V_{DD}	V

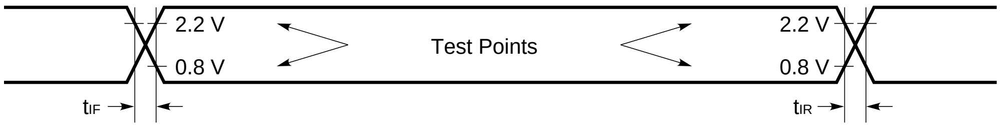
DATA MEMORY STOP MODE LOW SUPPLY VOLTAGE DATA HOLDING CHARACTERISTICS (T_A = –40 to +85°C)

Parameter	Symbol	Test Conditions	MIN.	MAX.	Unit
Data Hold Supply Voltage	V_{DDDR}		2.5	5.5	V
V_{DD} Rise/Fall Time	t_{RVD}, t_{FVD}		200		μs

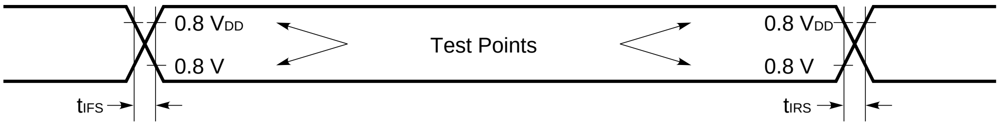
DATA HOLDING TIMING



AC TEST INPUT WAVEFORM (Except $\overline{\text{RESET}}$, NMI, X1 and X2)

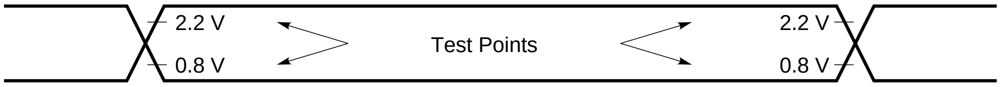


AC TEST INPUT WAVEFORM ($\overline{\text{RESET}}$, NMI, X1 and X2)

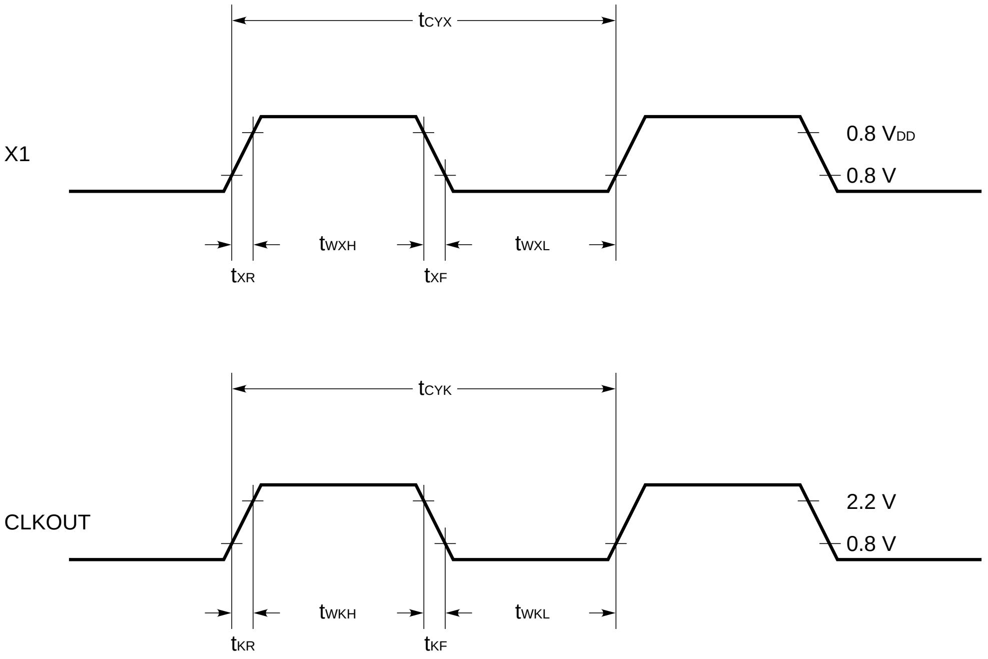


AC TEST OUTPUT TEST POINTS

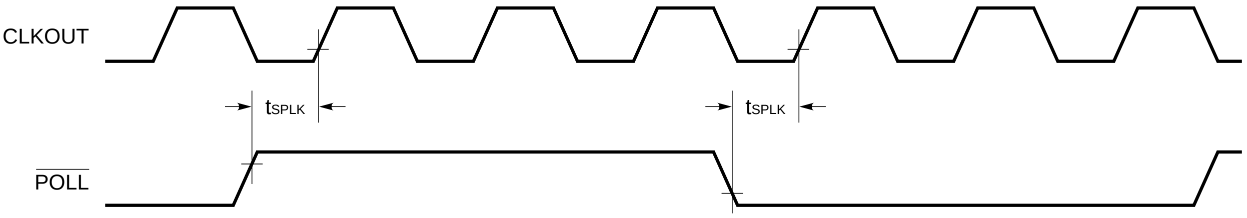
Output load condition: 100 pF



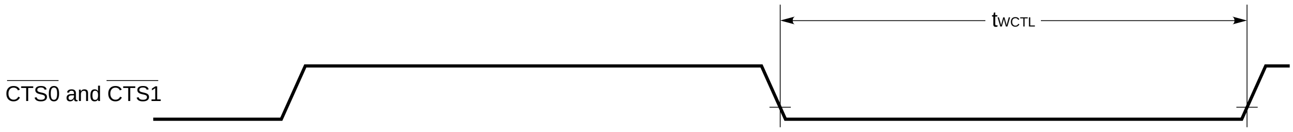
CLOCK TIMING



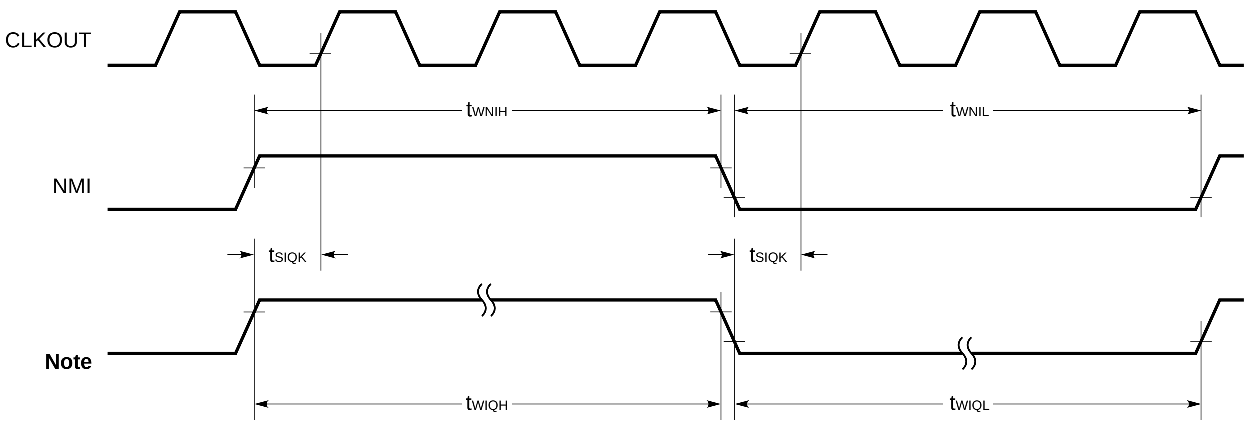
$\overline{\text{POLL}}$ INPUT TIMING



$\overline{\text{CTS0}}$ AND $\overline{\text{CTS1}}$ INPUT TIMING



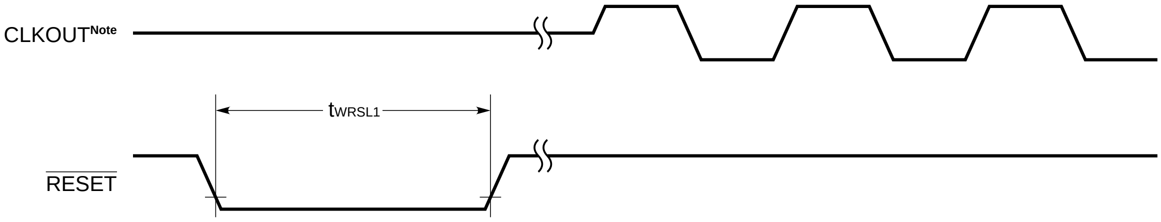
INTERRUPT INPUT/DMA INPUT TIMING



Note $\overline{\text{INTP0}}$ to $\overline{\text{INTP2}}$, DMARQ0 to DMARQ1

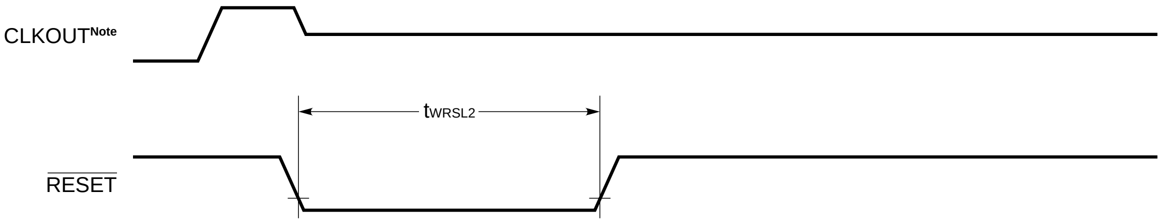
$\overline{\text{RESET}}$ INPUT TIMING

When STOP mode is released/at power-on reset:



Note CLKOUT signal is output after CLKOUT output is set.

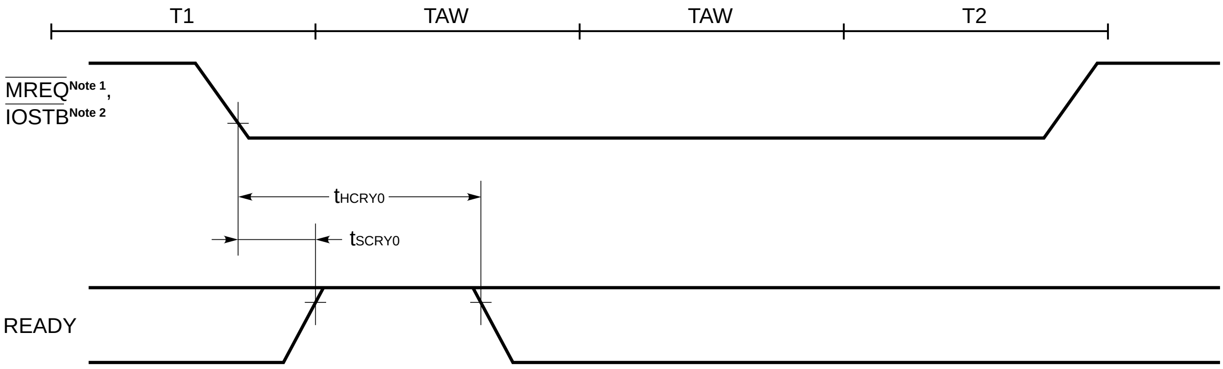
★ **When system is reset:**



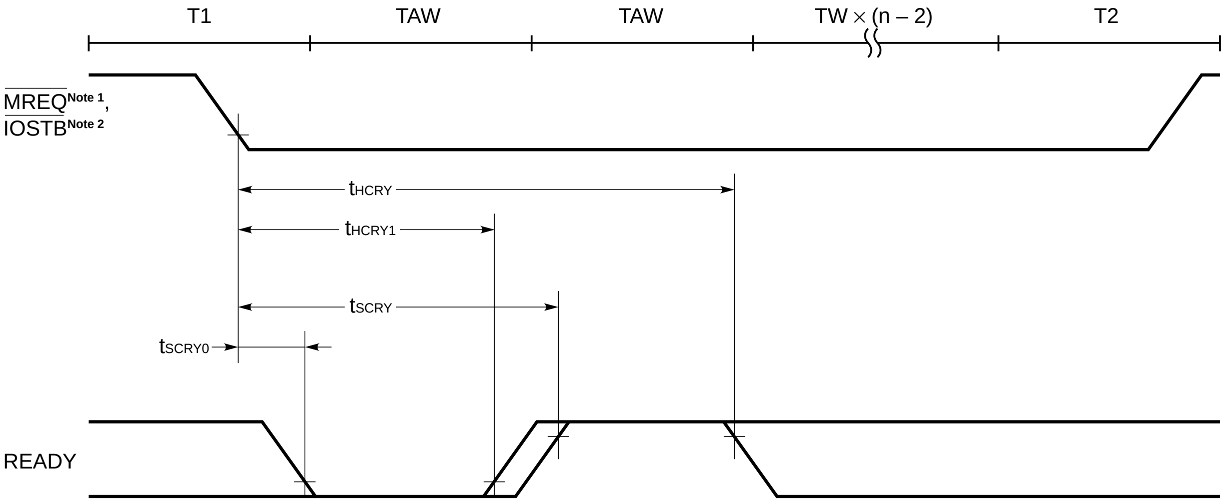
Note CLKOUT output is set to input port by $\overline{\text{RESET}}$ input.

READY TIMING

When 2 wait states are inserted:



When (n – 2) extra wait states are inserted [n ≥ 3]:

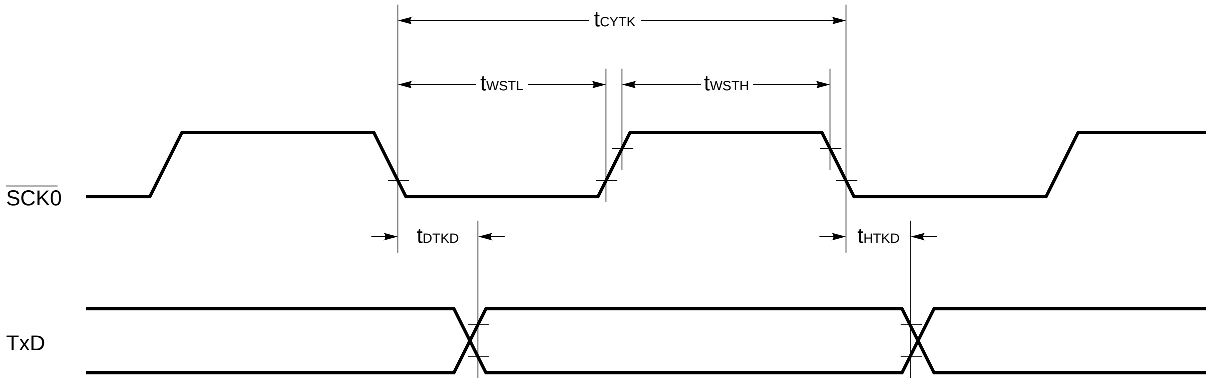


Notes **1.** In case of memory cycle

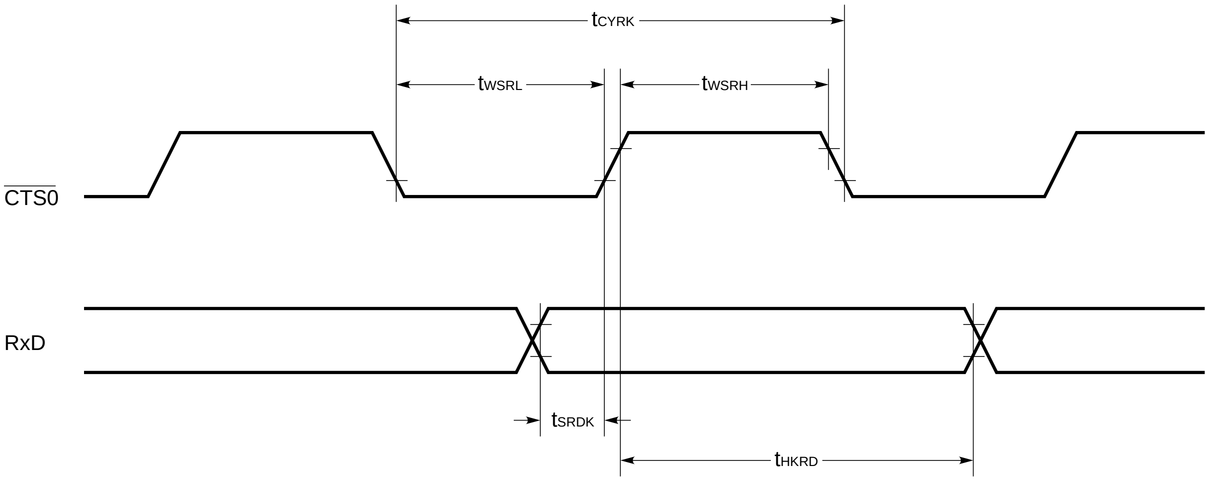
2. In case of I/O cycle

SERIAL OPERATION

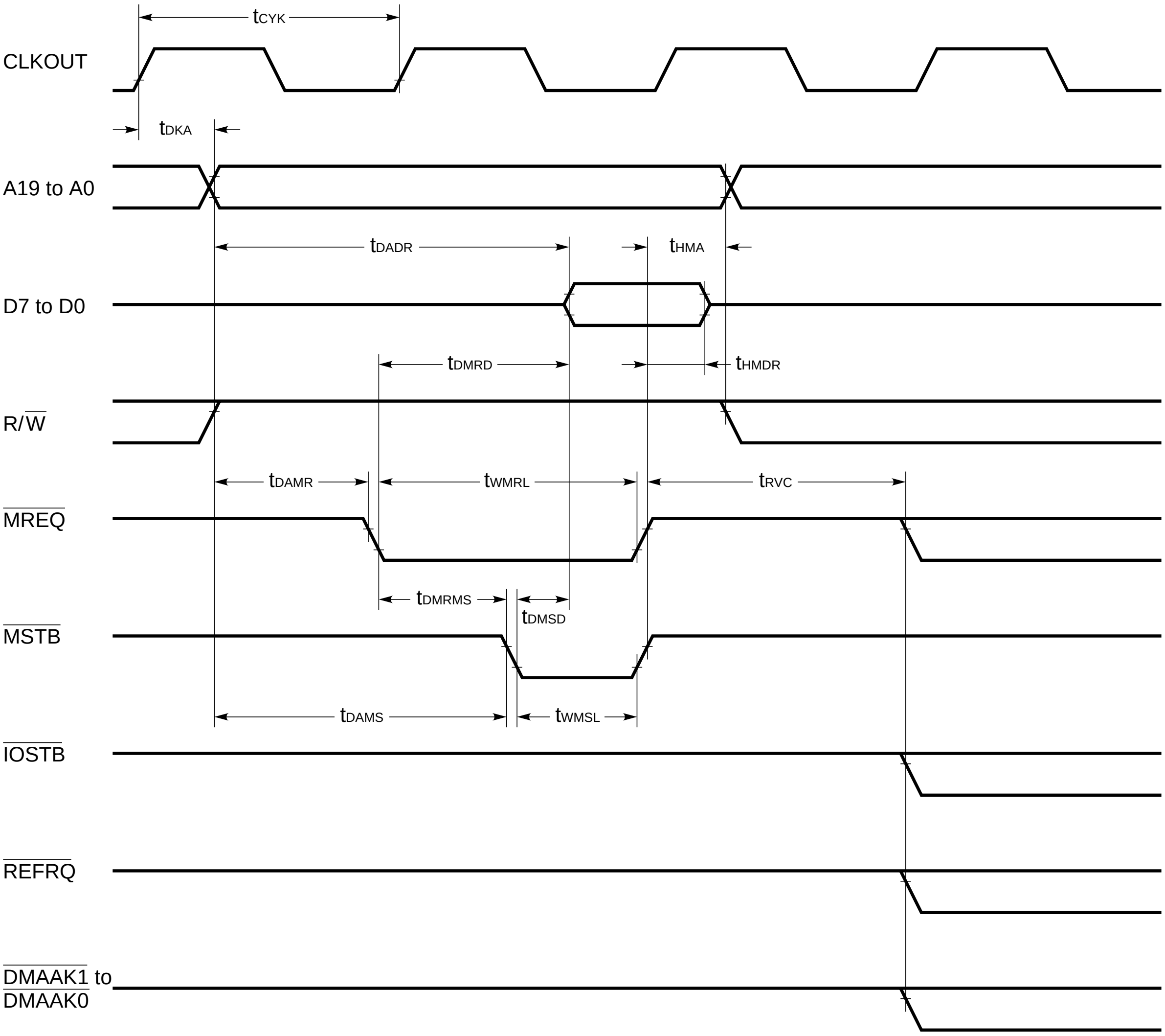
When transmitting data in I/O interface mode



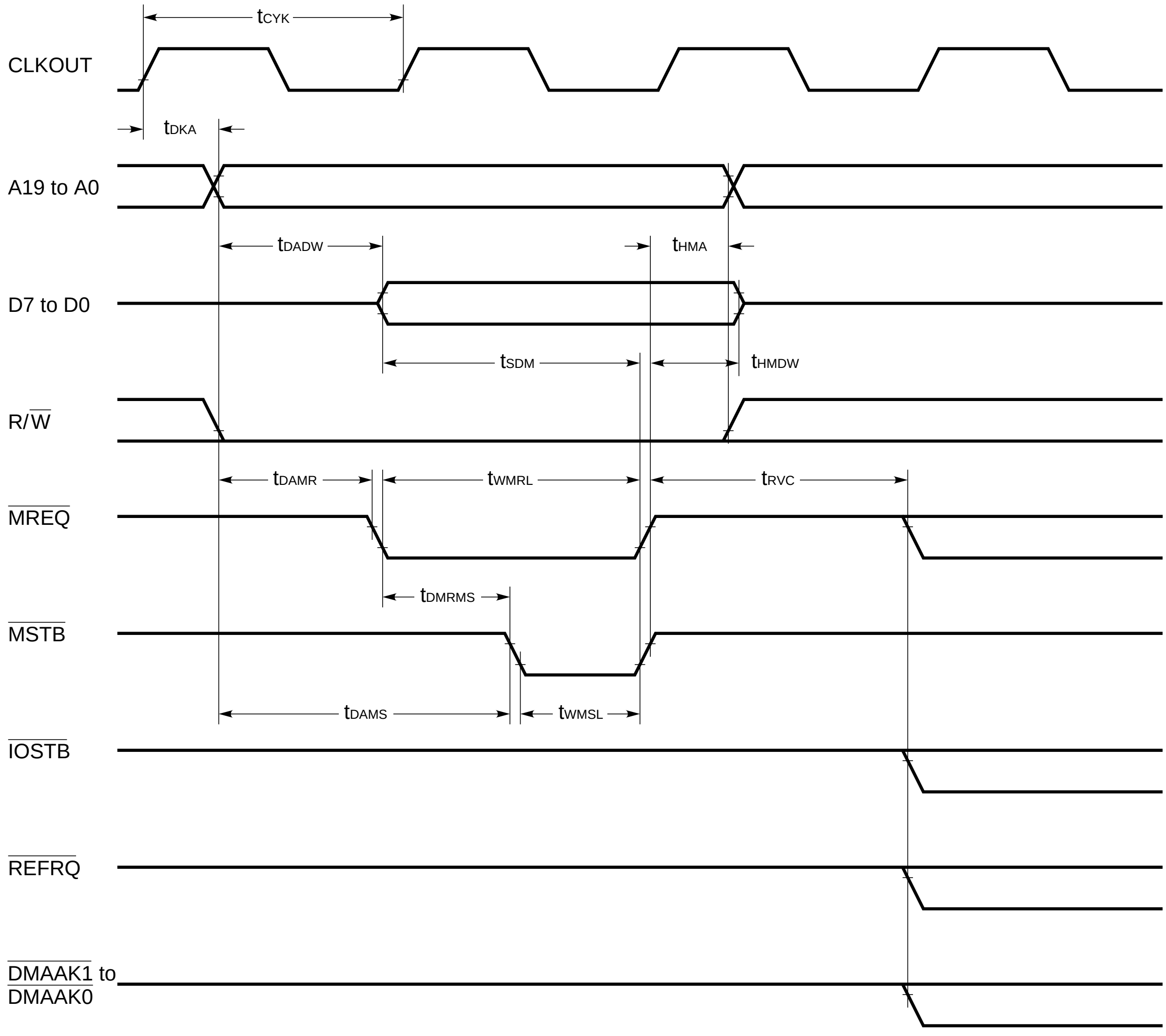
When receiving data in I/O interface mode



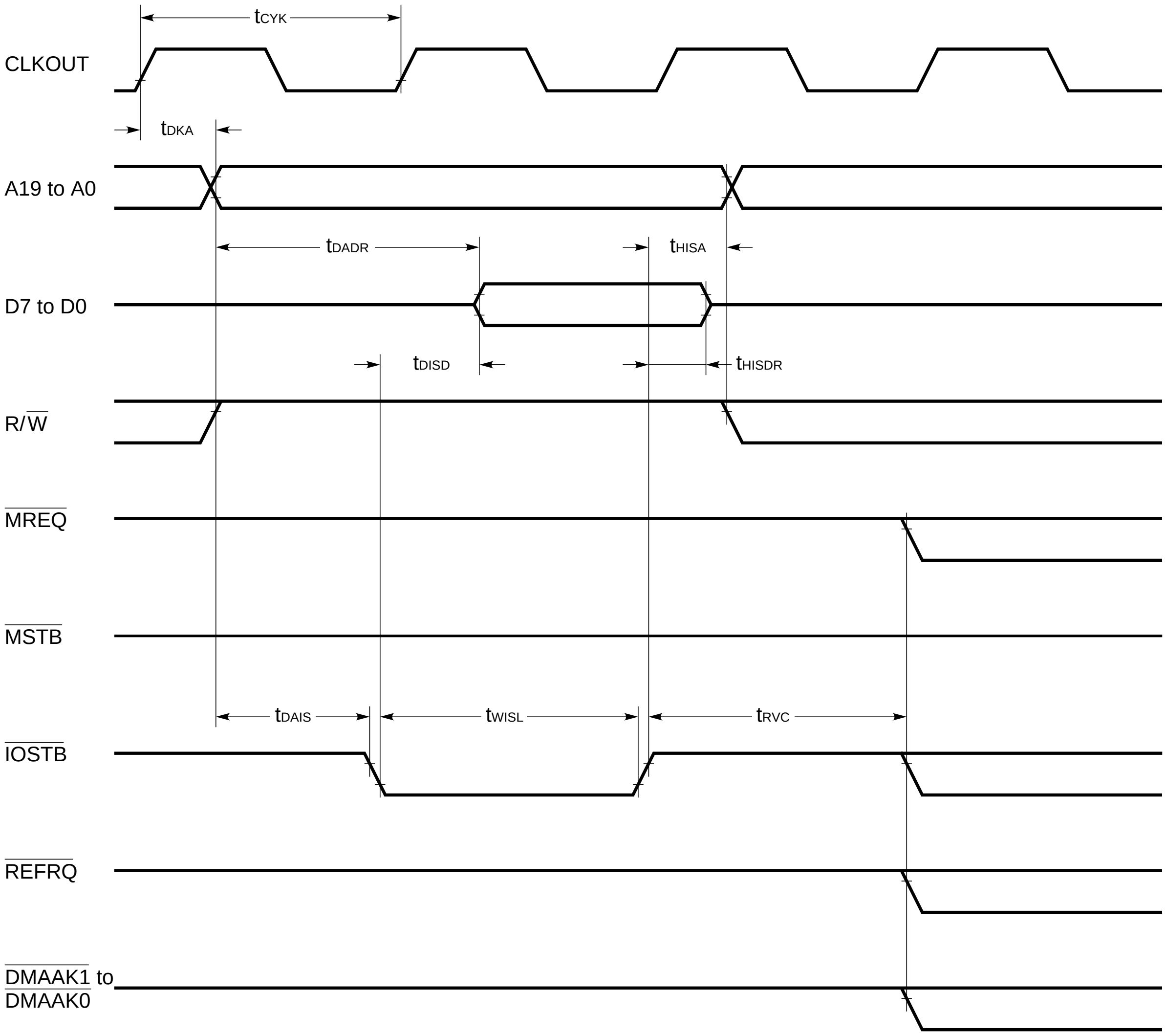
READ OPERATION



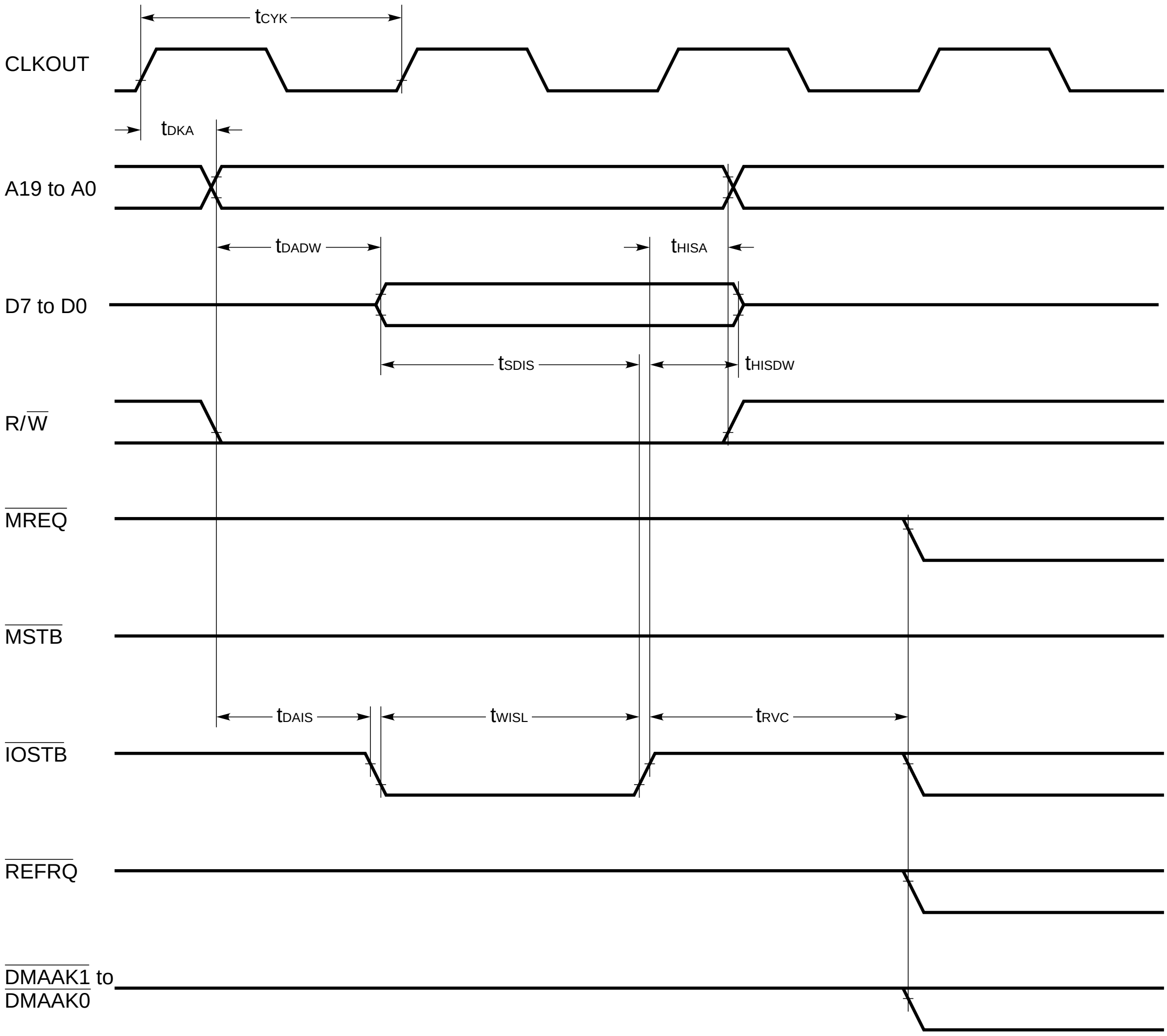
WRITE OPERATION



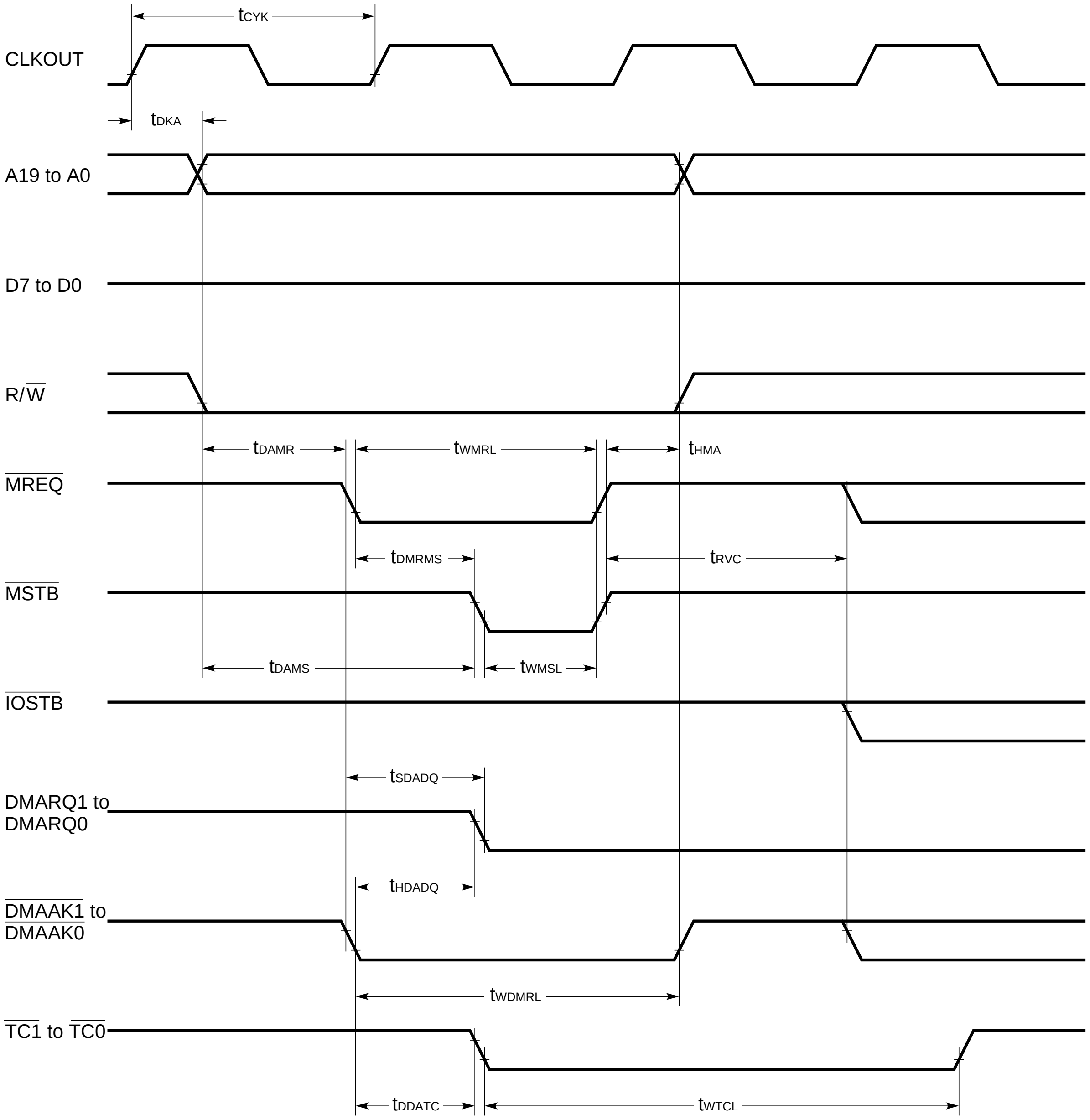
I/O READ TIMING



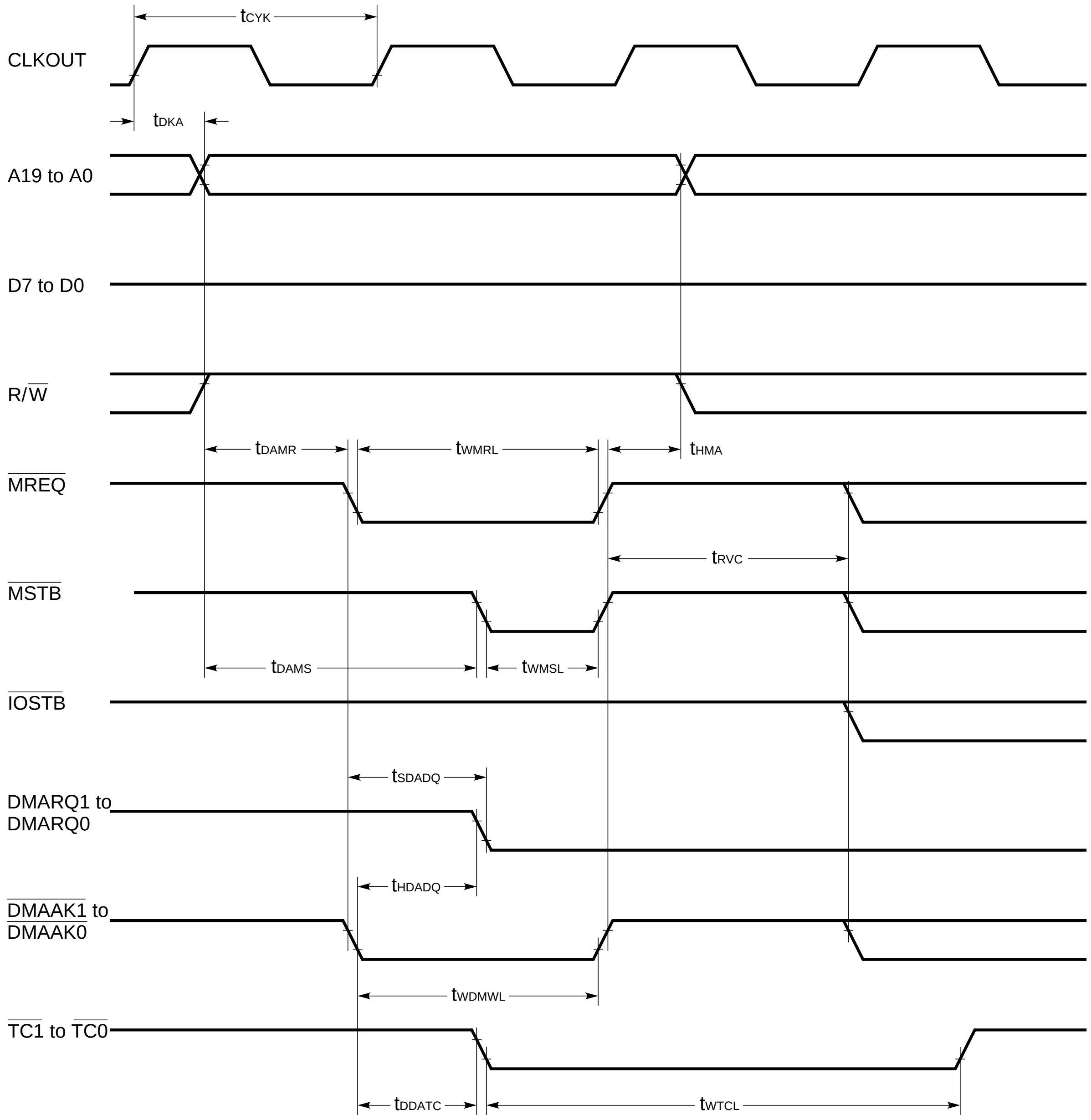
I/O WRITE TIMING



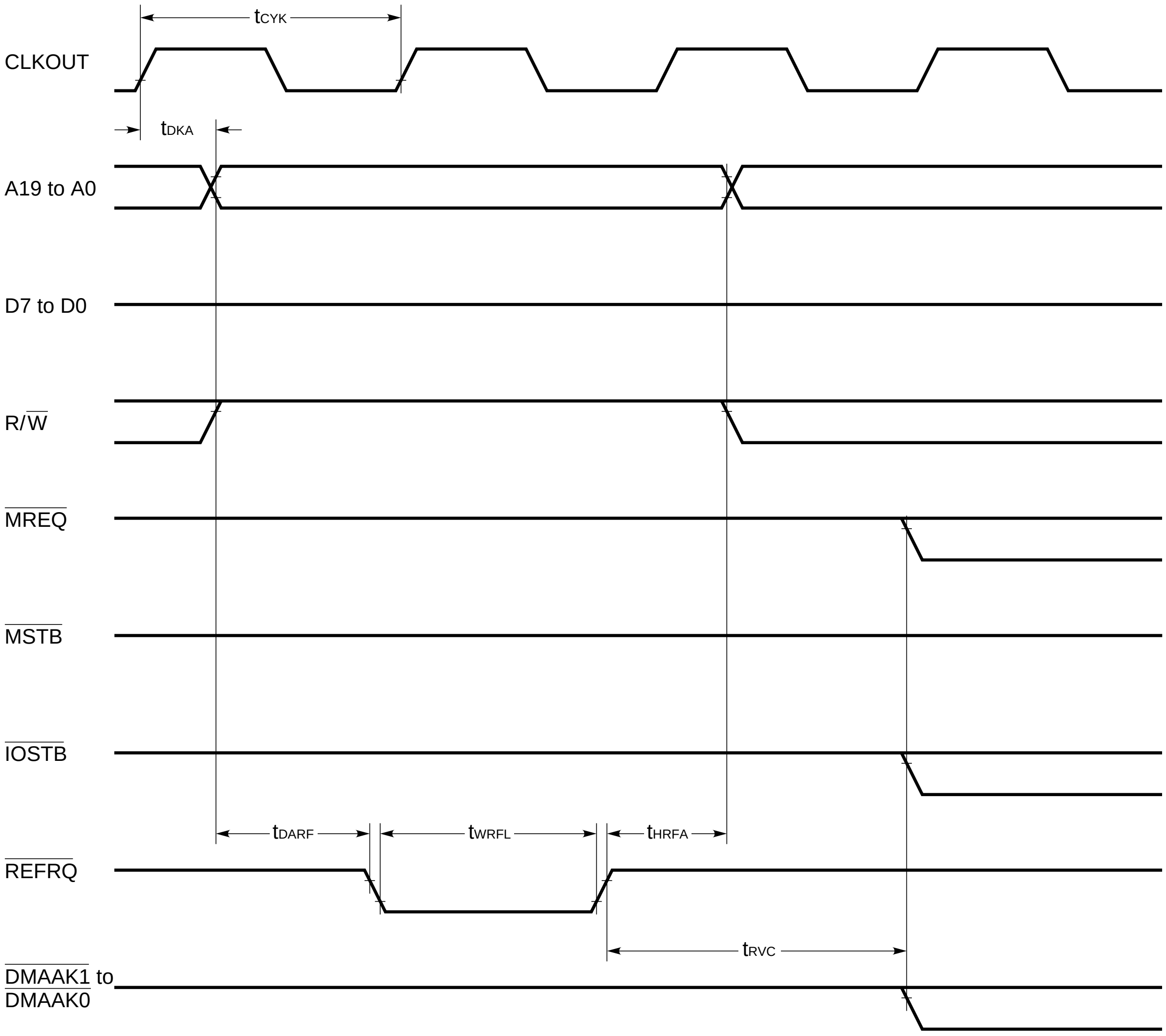
DMA (I/O → MEMORY) TIMING



DMA (MEMORY → I/O) TIMING

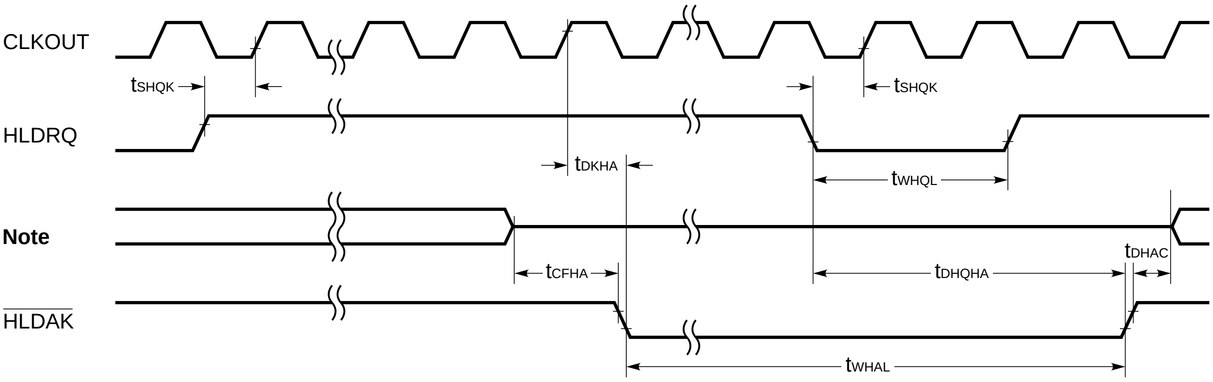


REFRESH TIMING

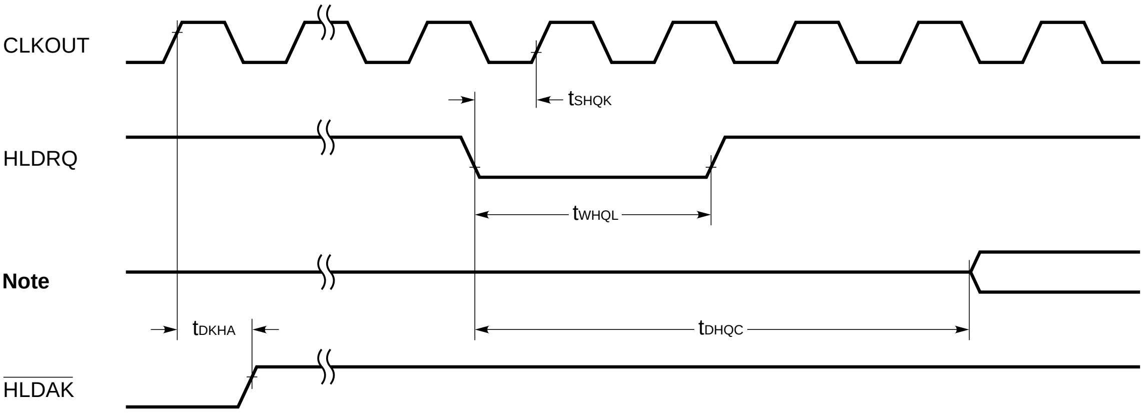


HOLD REQUEST/ACKNOWLEDGE TIMING

Normal mode

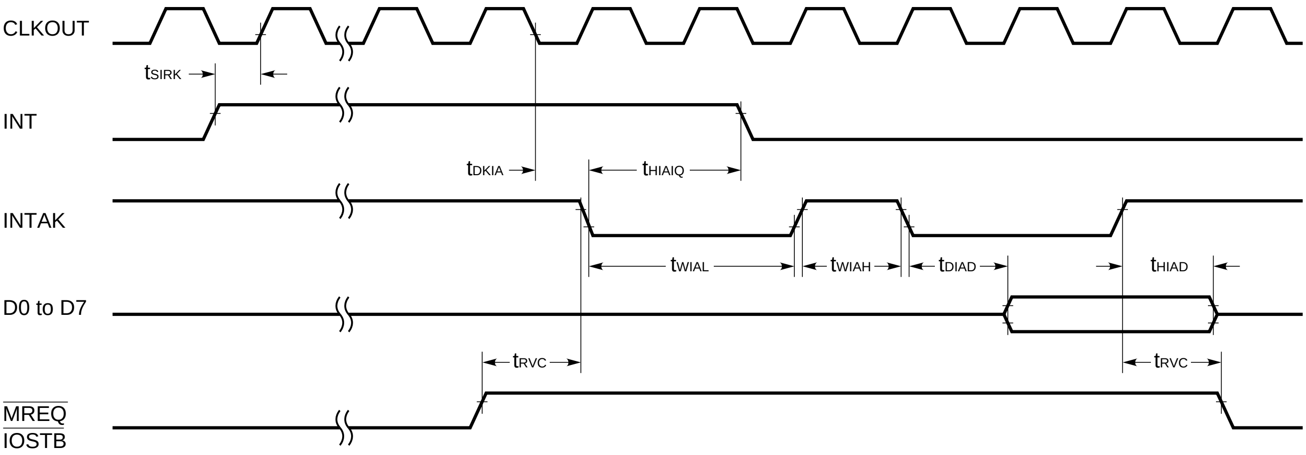


Releasing HOLD mode at refreshing time

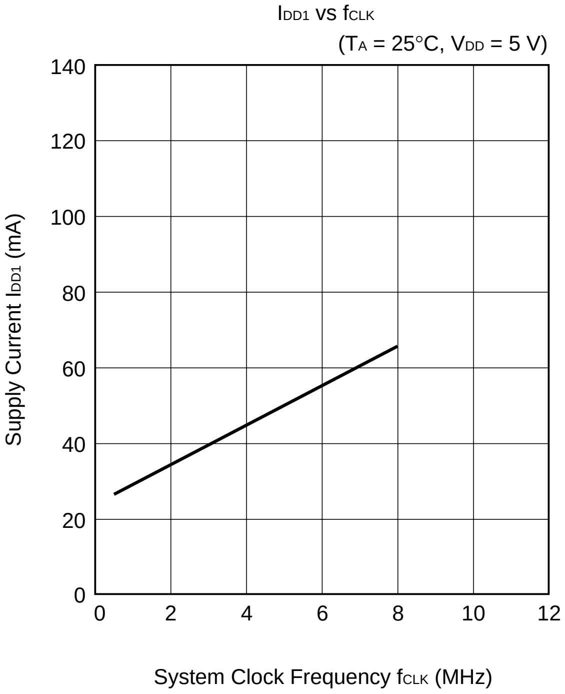


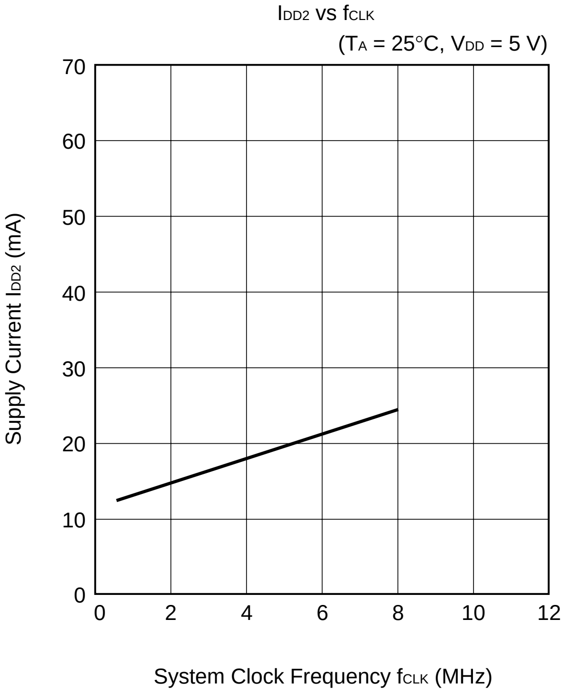
Note A19 to A0, D7 to D0, $\overline{\text{MREQ}}$, $\overline{\text{MSTB}}$, $\overline{\text{IOSTB}}$, R/$\overline{\text{W}}$

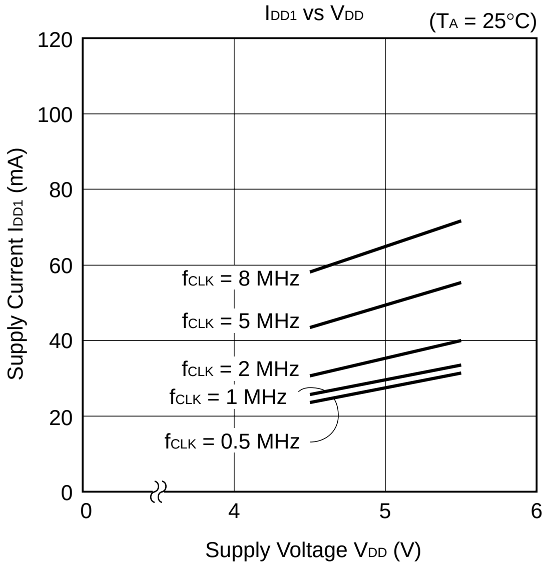
EXTERNAL INTERRUPT REQUEST/ACKNOWLEDGE TIMING

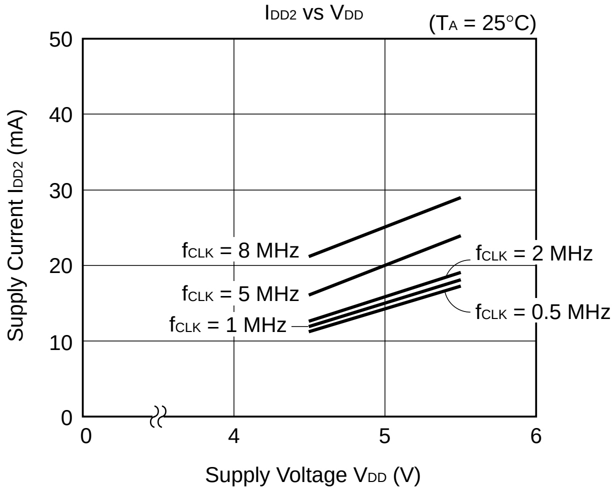


4. CHARACTERISTIC CURVES

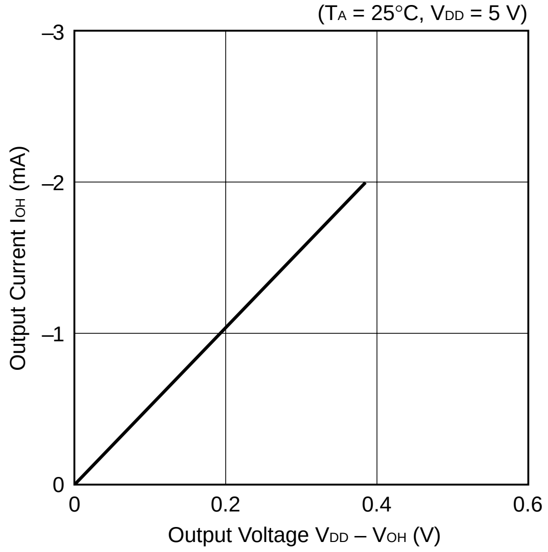




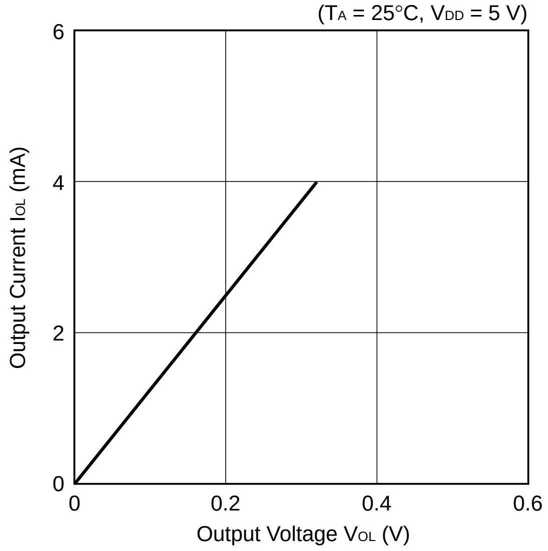

IDD1 vs VDD
(TA = 25°C)
Supply Current IDD1 (mA)
120
100
80
60
40
20
0
0
4
5
6
Supply Voltage VDD (V)
fCLK = 8 MHz
fCLK = 5 MHz
fCLK = 2 MHz
fCLK = 1 MHz
fCLK = 0.5 MHz


IDD2 vs VDD
(TA = 25°C)
Supply Current IDD2 (mA)
50
40
30
20
10
0
0
4
5
6
Supply Voltage VDD (V)
fCLK = 8 MHz
fCLK = 5 MHz
fCLK = 1 MHz
fCLK = 2 MHz
fCLK = 0.5 MHz

I_{OH} vs V_{OH}
(T_A = 25°C, V_{DD} = 5 V)

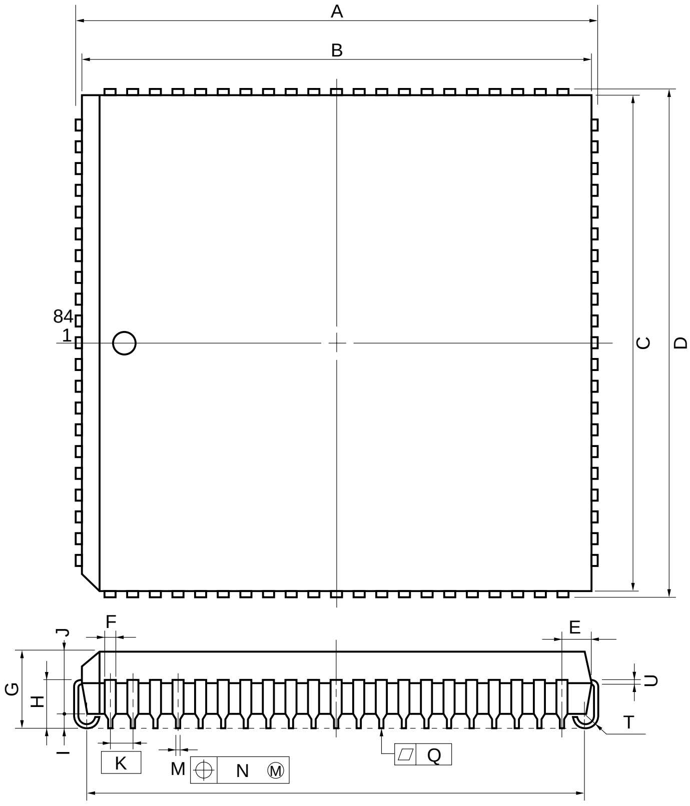


I_{OL} vs V_{OL}
(T_A = 25°C, V_{DD} = 5 V)



5. PACKAGE DRAWINGS

84 PIN PLASTIC QFJ (□1150 mil)



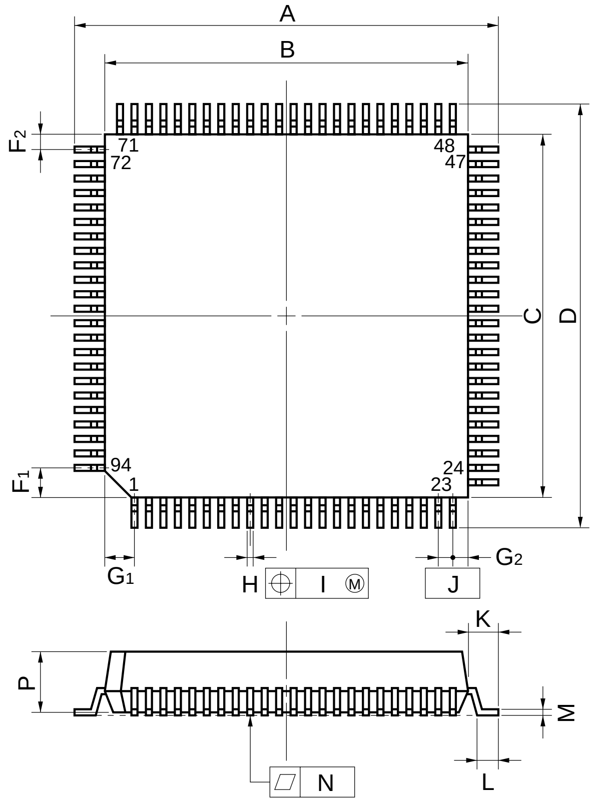
P84L-50A3-2

NOTE

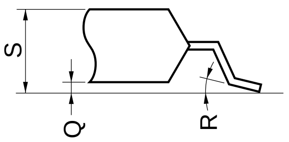
Each lead centerline is located within 0.12 mm (0.005 inch) of its true position (T.P.) at maximum material condition.

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	$0.20^{+0.10}_{-0.05}$	$0.008^{+0.004}_{-0.002}$

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

★ 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” (C10535E).**

Please consult with our sales office when using other soldering process or under different soldering conditions.

Table 6-1. Surface Mount Type Soldering Conditions

(1) μPD70320L : 84-pin plastic QFJ (1150 × 1150 mils)
μPD70320L-8 : 84-pin plastic QFJ (1150 × 1150 mils)

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

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

(2) μPD70320GJ-5BG : 94-pin plastic QFP (20 × 20 mm)
μPD70320GJ-8-5BG : 94-pin plastic QFP (20 × 20 mm)

Soldering Process	Soldering Conditions	Symbol
Infrared ray reflow	Package peak temperature: 235°C, Reflow time: 30 seconds or less, Number of reflow processes: 3 or less Exposure limit: 7 days[Note] (36 hours pre-baking is required at 125°C afterwards)	IR35-367-3
VPS	Package peak temperature: 215°C, Reflow time: 40 seconds or less, Number of reflow processes: 3 or less Exposure limit: 7 days[Note] (36 hours pre-baking is required at 125°C afterwards)	VP15-367-3
Wave soldering	Package peak temperature: 260°C, Reflow time: 10 seconds or less, Number of reflow processes: 1 Pre-heating temperature: 120°C max. (package surface temperature) Exposure limit: 7 days[Note] (36 hours pre-baking is required at 125°C afterwards)	WS60-367-1
Partial heating method	Pin temperature: 300°C or below, Flow time: 3 seconds or less (per side of device)	—

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

Caution Use of more than one soldering process should be avoided (except for partial heating method).

NOTES FOR CMOS DEVICES

① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note: Strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and quickly dissipate it once, when it has occurred. Environmental control must be adequate. When it is dry, humidifier should be used. It is recommended to avoid using insulators that easily build static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work bench and floor should be grounded. The operator should be grounded using wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with semiconductor devices on it.

② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note: No connection for CMOS device inputs can be cause of malfunction. If no connection is provided to the input pins, it is possible that an internal input level may be generated due to noise, etc., hence causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using a pull-up or pull-down circuitry. Each unused pin should be connected to V_{DD} or GND with a resistor, if it is considered to have a possibility of being an output pin. All handling related to the unused pins must be judged device by device and related specifications governing the devices.

③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note: Power-on does not necessarily define initial status of MOS device. Production process of MOS does not define the initial operation status of the device. Immediately after the power source is turned ON, the devices with reset function have not yet been initialized. Hence, power-on does not guarantee out-pin levels, I/O settings or contents of registers. Device is not initialized until the reset signal is received. Reset operation must be executed immediately after power-on for devices having reset function.

Regional Information

Some information contained in this document may vary from country to country. Before using any NEC product in your application, please contact the NEC office in your country to obtain a list of authorized representatives and distributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and other legal issues may also vary from country to country.

NEC Electronics Inc. (U.S.)
Santa Clara, California
Tel: 408-588-6000
800-366-9782
Fax: 408-588-6130
800-729-9288

NEC Electronics (Germany) GmbH
Duesseldorf, Germany
Tel: 0211-65 03 02
Fax: 0211-65 03 490

NEC Electronics (UK) Ltd.
Milton Keynes, UK
Tel: 01908-691-133
Fax: 01908-670-290

NEC Electronics Italiana s.r.1.
Milano, Italy
Tel: 02-66 75 41
Fax: 02-66 75 42 99

NEC Electronics (Germany) GmbH
Benelux Office
Eindhoven, The Netherlands
Tel: 040-2445845
Fax: 040-2444580

NEC Electronics (France) S.A.
Velizy-Villacoublay, France
Tel: 01-30-67 58 00
Fax: 01-30-67 58 99

NEC Electronics (France) S.A.
Spain Office
Madrid, Spain
Tel: 01-504-2787
Fax: 01-504-2860

NEC Electronics (Germany) GmbH
Scandinavia Office
Taeby, Sweden
Tel: 08-63 80 820
Fax: 08-63 80 388

NEC Electronics Hong Kong Ltd.
Hong Kong
Tel: 2886-9318
Fax: 2886-9022/9044

NEC Electronics Hong Kong Ltd.
Seoul Branch
Seoul, Korea
Tel: 02-528-0303
Fax: 02-528-4411

NEC Electronics Singapore Pte. Ltd.
United Square, Singapore 1130
Tel: 253-8311
Fax: 250-3583

NEC Electronics Taiwan Ltd.
Taipei, Taiwan
Tel: 02-719-2377
Fax: 02-719-5951

NEC do Brasil S.A.
Cumbica-Guarulhos-SP, Brasil
Tel: 011-6465-6810
Fax: 011-6465-6829

J97. 8

Related documents	V25, V35 User's Manual — Hardware	IEM-1220
	V25, V35 Family User's Manual — Instructions	U12120J (Japanese version)

The related documents indicated in this publication may include preliminary versions. However, preliminary versions are not marked as such.

V20, V25, V30, and V35 are trademarks of NEC Corporation.

No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Corporation. NEC Corporation assumes no responsibility for any errors which may appear in this document.
NEC Corporation does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from use of a device described herein or any other liability arising from use of such device. No license, either express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Corporation or others.
While NEC Corporation has been making continuous effort to enhance the reliability of its semiconductor devices, the possibility of defects cannot be eliminated entirely. To minimize risks of damage or injury to persons or property arising from a defect in an NEC semiconductor device, customers must incorporate sufficient safety measures in its design, such as redundancy, fire-containment, and anti-failure features.
NEC devices are classified into the following three quality grades:
"Standard", "Special", and "Specific". The Specific quality grade applies only to devices developed based on a customer designated "quality assurance program" for a specific application. The recommended applications of a device depend on its quality grade, as indicated below. Customers must check the quality grade of each device before using it in a particular application.

Standard: Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots

Special: Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support)

Specific: Aircrafts, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems or medical equipment for life support, etc.

The quality grade of NEC devices is "Standard" unless otherwise specified in NEC's Data Sheets or Data Books.
If customers intend to use NEC devices for applications other than those specified for Standard quality grade, they should contact an NEC sales representative in advance.
Anti-radioactive design is not implemented in this product.

M4 96.5