

PRELIMINARY INFORMATION

Description

The μPD7807/μPD7808/μPD7809 single chip micro-computer augments the high-end NEC family of 8-bit microcomputers with on-chip peripheral functions. Like the μPD7811, the device has a fast internal 16-bit ALU and data paths, 256 bytes of RAM, a multifunctional 16-bit timer/event counter, two 8-bit timers, a USART, and two zero-cross detect inputs.

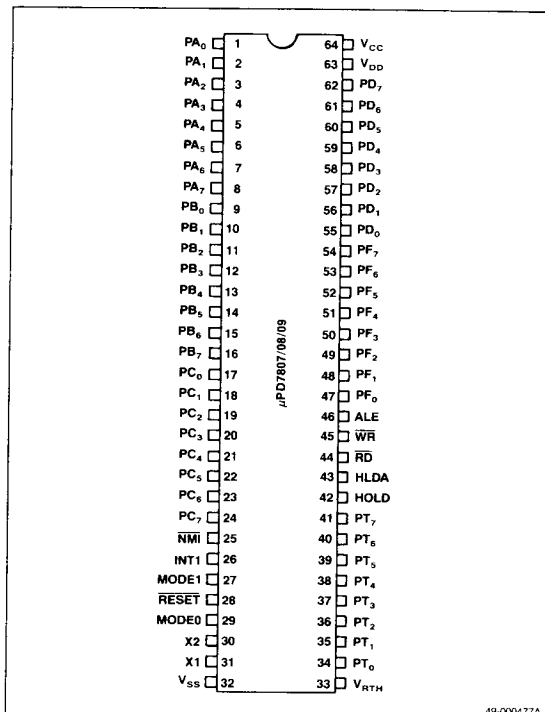
Other features are 8K ROM (4K ROM for the μPD7808), a programmable threshold comparator (8 inputs), a programmable WAIT function, a watchdog timer, hold and hold acknowledge for DMA interfaces, and bit test/write instructions for both RAM and I/O.

The μPD7809 and μPD7808 are mask-ROM versions with your program on chip. The μPD7807 is the ROM-less version for prototyping and small volume applications.

Features

- NMOS silicon gate technology requiring +5 V power supply
- Complete single-chip microcomputer
 - 16-bit ALU
 - 8K ROM (4K ROM for the μPD7808)
 - 256-byte RAM
- Large I/O capability
 - 40 I/O port lines (μPD7809 and μPD7808)
 - 28 I/O port lines (μPD7807)
 - 8 input lines
- Two zero-cross detect inputs
- Expansion capabilities (64K memory access total)
 - 8085A bus compatible
 - 56K-byte external memory address range (60K for the μPD7808)
- Programmable threshold comparator
 - 8 inputs, 1 of 16 software selectable levels
- Full duplex USART
 - Synchronous and asynchronous
- 165 instructions
 - 16-bit arithmetic, multiply and divide
- 1 μs instruction cycle time
- Prioritized interrupt structure
 - 3 external
 - 8 internal
- Hold, hold acknowledge for DMA interface
- Programmable WAIT function
- Watchdog timer
- Standby function
- On-chip clock generator
- 64-pin plastic straight or bent lead QUIP or plastic shrink DIP

Pin Configuration



Ordering Information

Part Number	Package Type	Max Freq. of Operation
μPD7807G-36	64-Pin plastic QUIP	12 MHz
μPD7808G-36		
μPD7809G-36		
μPD7807CW	64-Pin plastic shrink DIP	12 MHz
μPD7808CW		
μPD7809CW		

4

Pin Identification

No.	Symbol	Function
1-8	PA ₀ -PA ₇	Port A I/O
9-16	PB ₀ -PB ₇	Port B I/O
17	PC ₀ /TxD	Port C I/O line 0/Transmit data output
18	PC ₁ /Rx _D	Port C I/O line 1/Receive data input
19	PC ₂ /SCK	Port C I/O line 2/Serial clock I/O
20	PC ₃ /TI/ INT ₂	Port C I/O line 3/Timer input/Interrupt request 2 input
21	PC ₄ /TO	Port C I/O line 4/Timer output
22	PC ₅ /CI	Port C I/O line 5/Counter input
23,24	PC ₆ , PC ₇ / CO ₀ , CO ₁	Port C I/O lines 6, 7/Counter outputs 0, 1
25	NMI	Nonmaskable interrupt input
26	INT1	Interrupt request 1 input
27	MODE1/M ₁	Mode 1 input/memory cycle 1 output
28	RESET	Reset input
29	MODE0/ I/O/M	Mode 0 input/I/O/memory output
30,31	X2, X1	Crystal connections 1, 2
32	V _{SS}	Ground
33	V _{RTH}	Port T threshold voltage input
34-41	PT ₀ -PT ₇	Port T variable threshold input port
42	HOLD	Hold request input
43	HLDA	Hold acknowledge output
44	R _D	Read strobe output
45	W _R	Write strobe output
46	ALE	Address latch enable output
47-54	PF ₀ -PF ₇	Port F I/O
55-62	PD ₀ -PD ₇	Port D I/O
63	V _{DD}	RAM backup power supply
64	V _{CC}	5 V power supply

Pin Functions**PA₀-PA₇ [Port A]**

Port A is an 8-bit three-state port. Each bit is independently programmable as either input or output. Reset makes all lines of port A inputs.

PB₀-PB₇ [Port B]

Port B is an 8-bit three-state port. Each bit is independently programmable as either input or output. Reset makes all lines of port B inputs.

PC₀-PC₇ [Port C]

Port C is an 8-bit three-state port. Each bit is independently programmable as either input or output. Alternatively, the lines of port C can be used as control lines for the USART and timer. Reset puts all lines of port C in port mode, input.

TxD [Transmit Data]. Serial data output terminal.

RxD [Receive Data]. Serial data input terminal.

SCK [Serial Clock]. Output for the serial clock when internal clock is used. Input for serial clock when external clock is used.

TI [Timer Input]. Timer input terminal.

INT₂ [Interrupt Request 2]. Falling-edge-triggered, maskable interrupt input terminal and AC-input, zero-cross detection terminal.

TO [Timer Output]. The output of TO is a square wave with a frequency determined by the timer.

CI [Counter Input]. External pulse input to timer/event counter.

CO₀, CO₁ [Counter Outputs 0, 1]. Programmable rectangular wave outputs based on timer/event counter.

PD₀-PD₇ [Port D]

Port D is an 8-bit three-state port. It can be programmed as either 8 bits of input or 8 bits of output. When external expansion memory is used, port D acts as the multiplexed address/data bus.

PF₀-PF₇ [Port F]

Port F is an 8-bit three-state port. Each bit is independently programmable as an input or output. When external expansion memory is used, port F outputs the high-order address bits.

PT₀-PT₇ [Port T]

Port T is made up of 8 variable threshold inputs. The input of each line is compared to a threshold voltage.

V_{RTH} [Variable Threshold Reference Voltage]

V_{RTH} is the reference voltage that the port T threshold voltage is derived from.

NMI [Nonmaskable Interrupt]

Falling-edge-triggered nonmaskable interrupt input.

INT1 [Interrupt Request 1]

INT1 is a rising-edge-triggered, maskable interrupt input. It is also an AC-input, zero-cross detection terminal.

RESET [Reset]

When the RESET input is brought low, it initializes the PD7807/08/09.

MODE1, MODE0 [Mode 1, 0]

The MODE1 and MODE0 inputs select the memory expansion mode. MODE1 also outputs the M1 signal during each opcode fetch. MODE0 outputs the I/O/M signal.

HOLD [Hold Request]

When the HOLD input is high, the CPU is put in a hold state until HOLD is brought low.

HLDA [Hold Acknowledge]

The CPU brings the HLDA output high when it is in the hold state, and low when the hold is released.

RD [Read Strobe]

The RD output goes low to gate data from external devices onto the data bus. RD goes high during reset. Three-state.

WR [Write Strobe]

The WR output goes low to indicate that the data bus holds valid data. It is a strobe signal for external memory or I/O write operations. WR goes high during reset. Three-state.

ALE [Address Latch Enable]

The ALE output latches the address signal to the output of PD₀-PD₇.

X1, X2 [Crystal Connections 1, 2]

X1 and X2 are the system clock crystal oscillator terminals. X1 is the input for an external clock.

VSS [Ground]

Ground potential.

VDD [Backup Power]

Backup power for on-chip RAM.

VCC [Power Supply]

+5 V power supply.

Input/Output

The μPD7807/08/09 has 8 comparator input lines (port T) and 40 digital I/O lines; five 8-bit ports (port A, port B, port C, port D, port F).

Comparator Input Lines. PT₀-PT₇ are configured as variable threshold comparator input lines.

Port A, Port B, Port C, Port F. Each line of these ports can be individually programmed as an input or output. When used as I/O ports, all have latched outputs and high-impedance inputs.

Port D. Port D can be programmed as a byte input or a byte output.

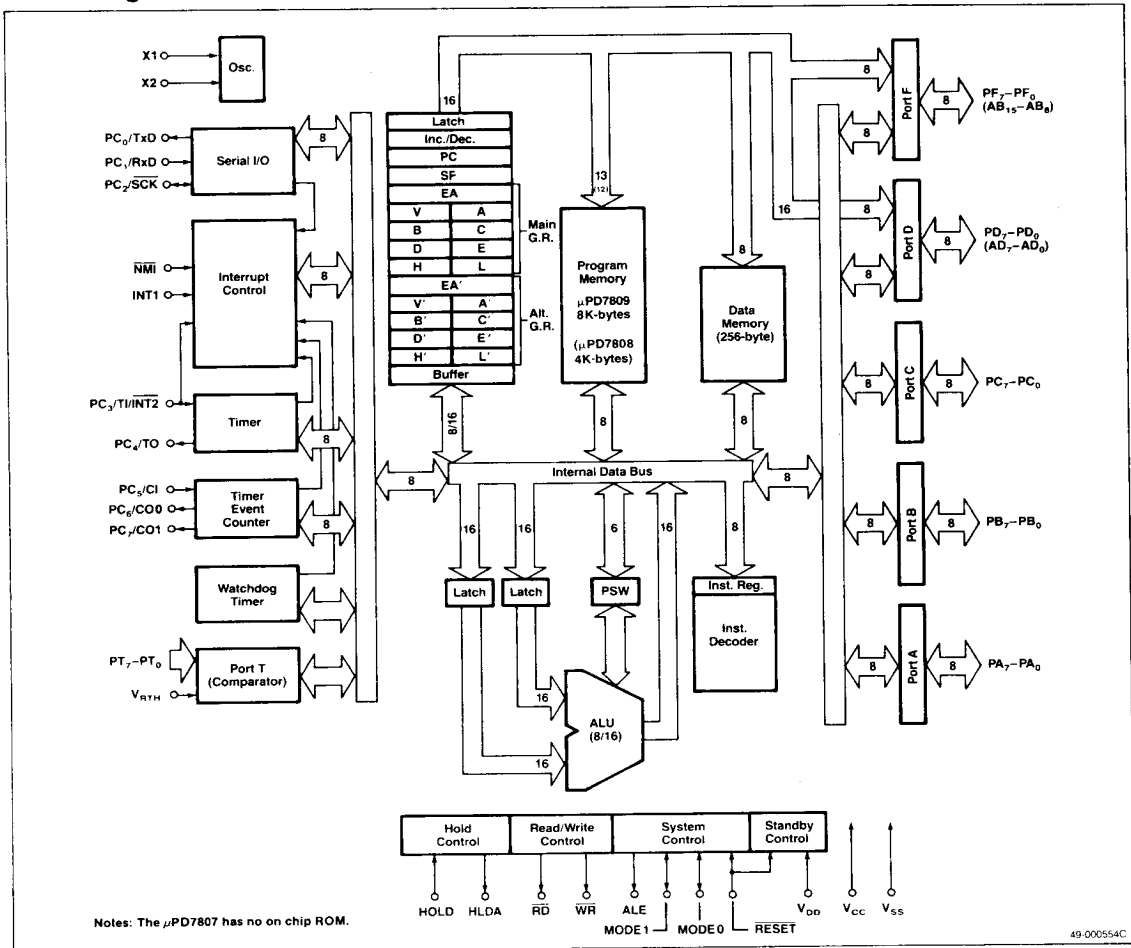
Control Lines. Under software control, each line of port C can be configured individually to provide control lines for the serial interface, timer, and timer/counter.

Memory Expansion. In addition to the single-chip operation mode, the μPD7809 has four memory expansion modes. Under software control, port D can provide a multiplexed low-order address and data bus; port F can provide a high-order address bus. Table 1 shows the relation between memory expansion modes and the pin configurations of port D and port F.

Table 1. Memory Expansion Modes and Port Configurations

Memory Expansion	Port Configuration	
None	Port D Port F	I/O port I/O port
256 Bytes	Port D Port F	Multiplexed address/data bus I/O port
4K Bytes	Port D Port F ₀ -F ₃ Port F ₄ -F ₇	Multiplexed address/data bus Address bus I/O port
16K Bytes	Port D Port F ₀ -F ₅ Port F ₆ -F ₇	Multiplexed address/data bus Address bus I/O port
56K Bytes	Port D Port F	Multiplexed address/data bus Address bus

Block Diagram



Timers

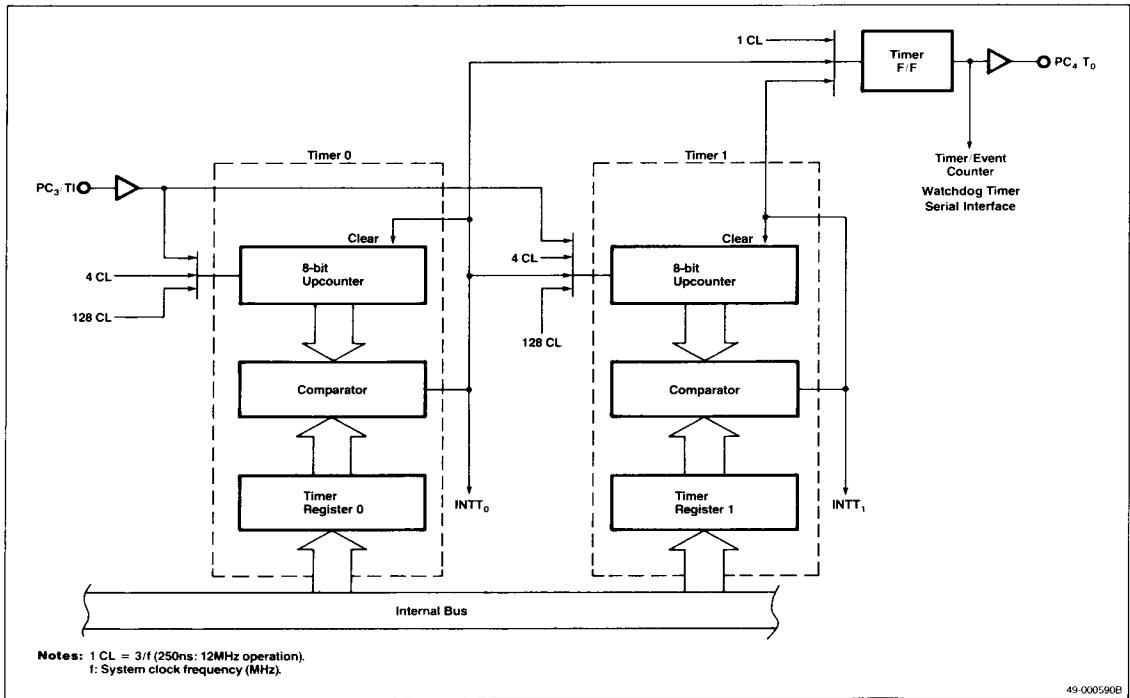
The timers consist of two 8-bit timers. The timers may be programmed independently or may be cascaded and used as a 16-bit timer. The timer can be software set to increment at intervals of four machine cycles (1 μs at 12 MHz operation) or 128 machine cycles (32 μs at 12 MHz), or to increment on receipt of a pulse at T1. Figure 1 shows the block diagram for the timer.

Timer/Event Counter

The 16-bit multifunctional timer/event counter (figure 2) can be used for the following operations:

- Interval timer
- External event counter
- Frequency measurement
- Pulse width measurement
- Programmable square-wave output

Figure 1. Timer Block Diagram



4

Interrupt Structure

There are 11 interrupt sources. Three are external interrupts and eight are internal. The following, table 2, shows 11 interrupt sources divided into six priority levels. See figure 3.

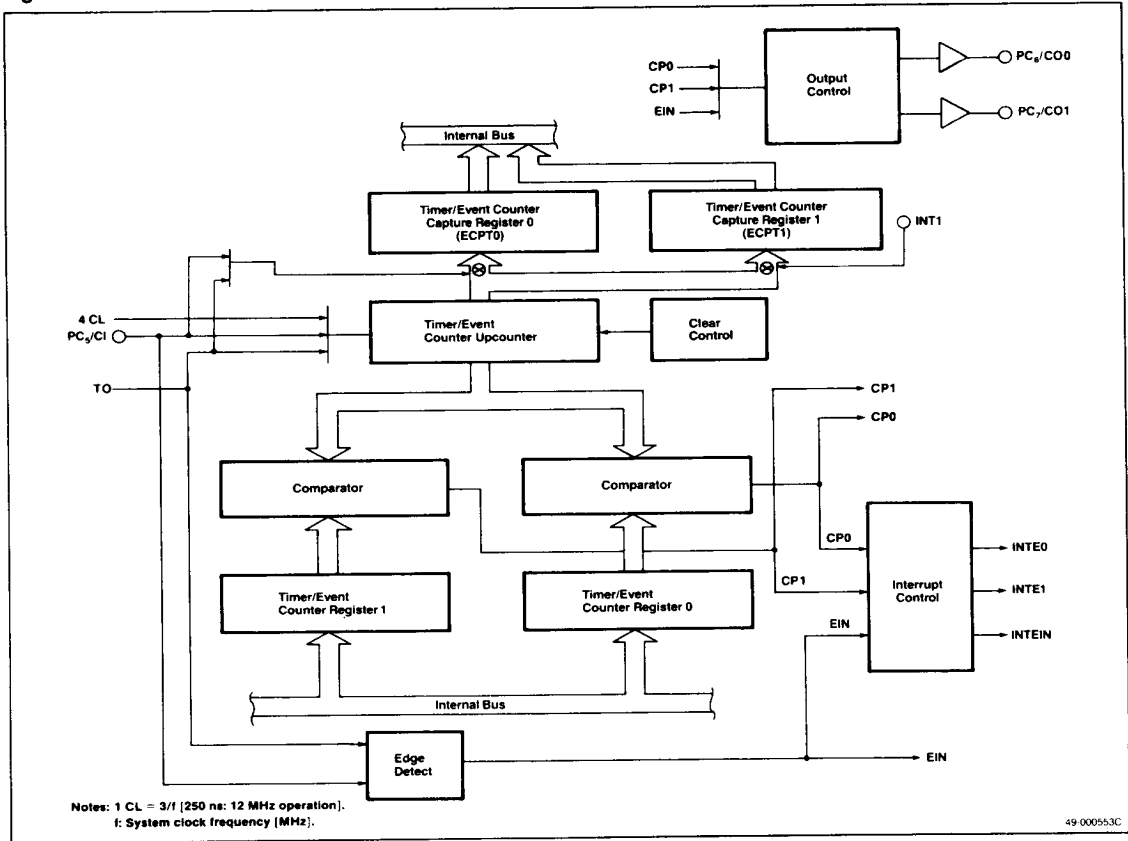
Standby Function

The standby function saves the top 32 bytes of RAM with backup power (V_{DD}) if the main power (V_{CC}) fails. On power up, you can check the standby flag to determine whether recovery was made from standby mode or from a cold start.

Table 2. Interrupt Sources

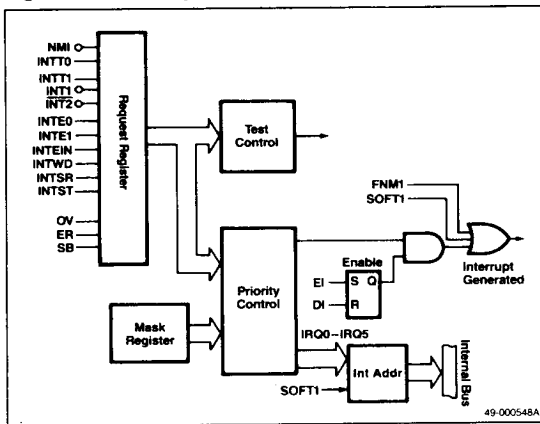
Interrupt Request	Interrupt Address	Type of Interrupt	Internal/External
IRQ0	4	NMI (Nonmaskable interrupt)	Ext
		INTWD (Watchdog timer)	Int
IRQ1	8	INTT0 (Coincidence signal from timer 0)	Int
		INTT1 (Coincidence signal from timer 1)	Int
IRQ2	16	INT1 (Maskable interrupt)	Ext
		INT2 (Maskable interrupt)	Ext
IRQ3	24	INTE0 (Coincidence signal from timer/event counter)	Int
		INTE1 (Coincidence signal from timer/event counter)	Int
IRQ4	32	INTEIN (Falling signal of CI and T0 counter)	Int/Ext
IRQ5	40	INTSR (Serial receive interrupt)	Int
		INST (Serial send interrupt)	Int

Figure 2. Timer/Event Counter Block Diagram



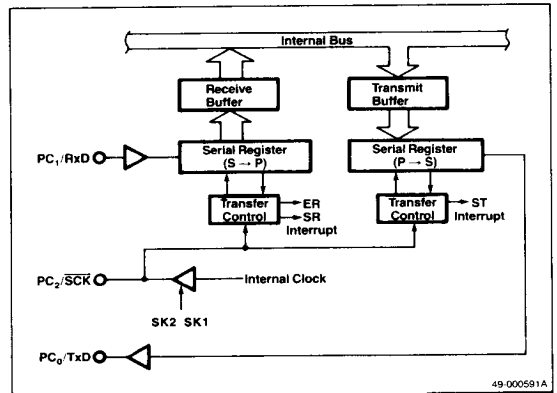
49-00053C

Figure 3. Interrupt Structure Block Diagram



49-000548A

Figure 4. Universal Serial Interface Block Diagram



49-000591A

Universal Serial Interface

The serial interface can operate in one of three modes: synchronous, asynchronous, and I/O interface. The I/O interface mode transfers data MSB first, for easy interfacing to certain NEC peripheral devices. Synchronous and asynchronous modes transfer data LSB first. Synchronous operation offers two modes of data reception: search and nonsearch. In the search mode, data is transferred one bit at a time from the serial register to the receive buffer. This allows a software search for a sync character. In the nonsearch mode, data transfer from the serial register to the transmit buffer occurs eight bits at a time. In asynchronous mode, the serial interface can act as a full-duplex USART with data transfer rates up to 125K bps. Figure 4 shows the universal serial interface block diagram.

Zero-Crossing Detector

The INT1 and INT2 terminals (used common to TI and PC₃) can detect the zero-crossing point of low-frequency AC signals. When driven directly, these pins respond as a normal digital input. Figure 5 shows the zero-crossing detection circuitry.

The zero-crossing detection capability allows you to make the 50-60 Hz power signal the basis for system timing and to control voltage phase sensitive devices.

To use the zero-cross detection mode, an AC signal of approximately 1-3 V AC (peak-to-peak) and a maximum frequency of 1 kHz is coupled through an external capacitor to the INT1 and INT2 pins.

For the INT1 pin, the internal digital state is sensed as a 0 until the rising edge crosses the average DC level, when it becomes a 1 and INT1 interrupt is generated.

For the INT2 pin, the state is sensed as a 1 until the falling edge crosses the average DC level, when it becomes a 0 and INT2 interrupt is generated.

Figure 5. Zero-Crossing Detection Circuitry

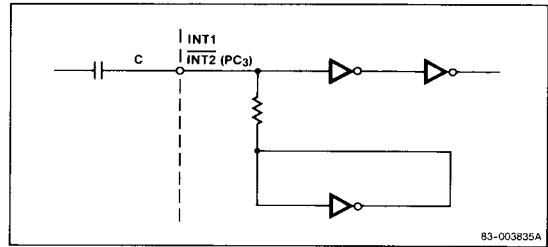
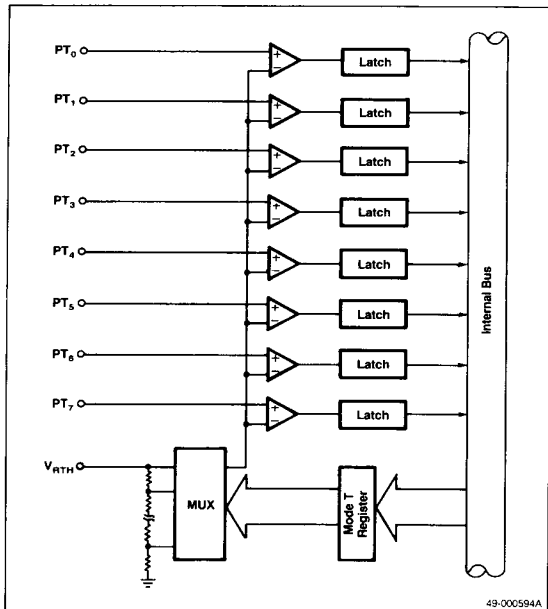


Figure 6. Threshold Variable Input Port



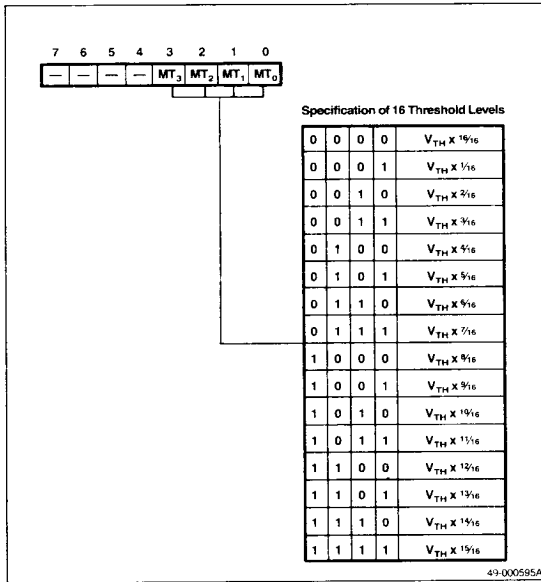
Variable Threshold Input Port [Port T]

Port T has the following features:

- 8 input lines
- 16 threshold levels from 1/16 to 16/16 of reference voltage (V_{RTH})
- Level selected by writing to mode T register (figure 7)
- Output of comparator reads 0 until voltage at pin exceeds selected level
- Comparison execution time: 12 μs

Figure 6 shows the block diagram for the threshold variable input port. Figure 7 shows the mode T register format.

Figure 7. Mode T Register Format



Watchdog Timer

Use the watchdog timer for software or overall performance safety checks. If the watchdog is enabled, it must be cleared at regular intervals in program execution to avoid watchdog interrupts. Intervals are software selectable via the WDM register. Figure 8 shows the block diagram for the watchdog timer.

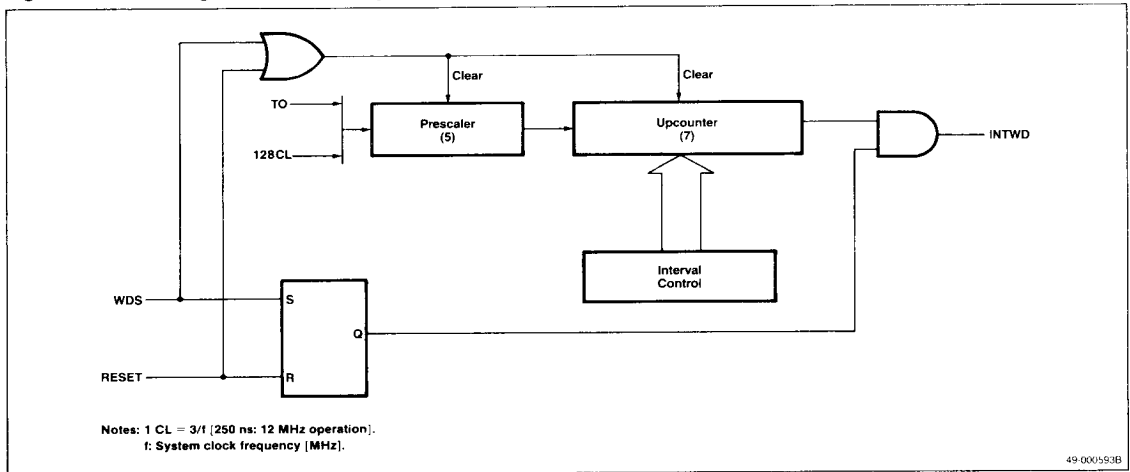
Bit Address Instructions

The following bits may be addressed directly with certain instructions:

- Any bit in the first 16 bytes of memory addressed by the V register
- Any bit in the five 8-bit I/O ports (A, B, C, D, F)
- Any bit in the comparator port
- Any bit in the following special registers: interrupt mask, serial mode high, timer mode, timer/event counter output control.

An addressed bit may be tested, set, cleared, or complemented. It also may be moved to or from the carry flag. An addressed bit may be ANDed, ORed, and XORed with the carry flag.

Figure 8. Watchdog Timer Block Diagram



Absolute Maximum Ratings

Power supply voltages, V_{CC}	-0.5 V to +7.0 V
V_{DD}	-0.5 V to +7.0 V
AV_{CC}	-0.5 V to +7.0 V
Input voltage, V_I	-0.5 V to +7.0 V
Output voltage, V_O	-0.5 V to +7.0 V
Reference input threshold voltage, V_{RTH}	-0.5 V to $V_{CC} + 0.1V$
Operating temperature, T_{OPR} 10 MHz $\leq f_{XTAL} \leq 12$ MHz	-10°C to +70°C
$f_{XTAL} \leq 10$ MHz (μ PD7807/09)	-10°C to +70°C
$f_{XTAL} \leq 10$ MHz (μ PD7808 only)	-40°C to +85°C
Storage temperature, T_{STG} (μ PD7807/09)	-40°C to +125°C
Storage temperature, (μ PD7808 only)	-65°C to +150°C

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

Operating Conditions

Oscillating Frequency	T_A	V_{CC}, AV_{CC}
10 MHz $\leq f_{XTAL} \leq 12$ MHz	-10°C to +70°C	+5.0 V \pm 5%
$f_{XTAL} \leq 10$ MHz (μ PD7807/09)	-10°C to +70°C	+5.0 V \pm 10%
$f_{XTAL} \leq 10$ MHz (μ PD7808)	-40°C to -85°C	+5.0 V \pm 10%

Capacitance

Parameter	Symbol	Limits			Unit	Test Conditions
		Min	Typ	Max		
Capacitance	C_I		10		pF	$Af_c = 1$ MHz.
Output capacitance	C_O		20		pF	Unmeasured pin returned
I/O capacitance	C_{IO}		20		pF	to 0 V.

DC Characteristics

$T_A = -10^\circ\text{C}$ to $+70^\circ\text{C}$; $V_{CC} = +5.0\text{ V} \pm 10\%$; $V_{SS} = 0\text{ V}$; $V_{CC} - 0.8\text{ V} \leq V_{DD} \leq V_{CC}$; $T_A = -40^\circ\text{C}$ to $+85^\circ\text{C}$ (μ PD7808)

Parameter	Symbol	Limits			Unit	Test Conditions
		Min	Typ	Max		
Input low voltage	V_{IL}	0		0.8	V	
Input high voltage	V_{IH1}	2.0		V_{CC}	V	All except SCK, RESET, and X1
	V_{IH2}	$0.8 V_{CC}$		V_{CC}	V	SCK, X1
	V_{IH3}	$0.8 V_{DD}$		V_{CC}	V	RESET
Output low voltage	V_{OL}			0.45	V	$I_{OL} = 2.0\text{ mA}$
Output high voltage	V_{OH}	2.4			V	$I_{OH} = -200\text{ }\mu\text{A}$
Input current	I_I			± 200	μA	INT1, T(PC3); $+0.45\text{ V} \leq V_{IN} \leq V_{CC}$
Input leakage current	I_{LI}			± 10	μA	All except INT1, T(PC3) $0\text{ V} \leq V_{IN} \leq V_{CC}$
Output leakage current	I_{LO}			± 10	μA	$+0.45\text{ V} \leq V_O \leq V_{CC}$
V_{RTH} input current	I_{RTH}	0.2(1)	0.6		mA	$V_{RTH} = V_{CC}$
V_{DD} supply current	I_{DD}	1.5(1)	3.5		mA	
V_{CC} supply current	I_{CC}	150(1)	220		mA	

Note:

(1) $T_A = 25^\circ\text{C}$; $V_{CC} = V_{DD} = +5.0\text{ V}$

Hold Operation

$T_A = -10^\circ\text{C}$ to $+70^\circ\text{C}$; $V_{CC} = +5.0\text{ V} \pm 10\%$; $V_{SS} = 0\text{ V}$; $V_{CC} - 0.8\text{ V} \leq V_{DD} \leq V_{CC}$; $T_A = -40^\circ\text{C}$ to $+85^\circ\text{C}$ (μ PD7808)

Parameter	Symbol	Limits			Unit	Test Conditions
		Min	Typ	Max		
HOLD \uparrow setup time to ALE \uparrow	t_{SHDL}	2T + 150			ns	
ALE \uparrow to HLDA \uparrow delay	t_{DLHA}			T + 150	ns	
HLDA \uparrow to bus floating	t_{FBHA}	0			ns	
HOLD \downarrow to HLDA \downarrow delay	t_{HDDA}	T - 50		4T + 150	ns	
HLDA \downarrow to bus enable time	t_{EHAB}	0			ns	
Bus setup time to ALE	t_{BL}	2T - 100			ns	



Comparator Characteristics

$T_A = -10^\circ\text{C}$ to $+70^\circ\text{C}$; $V_{CC} = +5.0\text{ V} \pm 10\%$; $V_{SS} = 0\text{ V}$; $V_{CC} - 0.8\text{ V} \leq V_{DD} \leq V_{CC}$; $T_A = -40^\circ\text{C}$ to $+85^\circ\text{C}$ (μPD7808)

Parameter	Symbol	Limits			Unit	Test Conditions
		Min	Typ	Max		
Comparison accuracy	V_{ACOMP}			± 100	mV	
Threshold voltage	V_{TH}	0		V_{CC}	V	
Comparison time	t_{COMP}	144		145	t_{CYC}	
PT input voltage	V_{IPT}	0		V_{CC}	V	

External Clock

$T_A = -10^\circ\text{C}$ to $+70^\circ\text{C}$; $V_{CC} = +5.0\text{ V} \pm 10\%$; $V_{SS} = 0\text{ V}$; $V_{CC} - 0.8\text{ V} \leq V_{DD} \leq V_{CC}$; $T_A = -40^\circ\text{C}$ to $+85^\circ\text{C}$ (μPD7808)

Parameter	Symbol	Limits			Unit	Test Conditions
		Min	Typ	Max		
High level width	$t_{\phi H}$	30		250	ns	
Low level width	$t_{\phi L}$	30		250	ns	
Rising time	t_r	0		30	ns	
Falling time	t_f	0		30	ns	

AC Characteristics

$V_{SS} = 0\text{ V}$, $V_{CC} - 0.8\text{ V} \leq V_{DD} \leq V_{CC}$; See Operating Conditions table

Parameter	Symbol	Limits				Unit	Test Conditions (1)
		$f_{XTAL} = 10\text{ MHz}$		$f_{XTAL} = 12\text{ MHz}$			
		Min	Max	Min	Max		
Read/Write Operation							
RESET pulse width	t_{RP}	6.0		5.0		μs	
Interrupt pulse width	t_{IP}	3.6		3.0		μs	
Counter input pulse width	t_{CI}	600		500		ns	Event counter mode
	t_{CI}	4.8		4.0		μs	Pulse width measurement mode
Timer input pulse width	t_{TI}	600		500		ns	
X1 Input cycle time	t_{CYC}	100	250	83	250	ns	
Address set-up to ALE ↓	t_{AL}	100		65		ns	
Address hold after ALE ↓	t_{LA}	70		50		ns	
Address to \overline{RD} ↓ delay time	t_{AR}	200		150		ns	
\overline{RD} ↓ to address floating	t_{AFR}		20		20	ns	
Address to data input	t_{AD}		480		360	ns	

Note:

(1) Load capacitance: $C_L = 150\text{ pF}$.

Data Retention Characteristics

$T_A = -10^\circ\text{C}$ to $+70^\circ\text{C}$; $V_{CC} = 0\text{ V}$, $V_{DD} = V_{DDDR}$; $T_A = -40^\circ\text{C}$ to $+85^\circ\text{C}$ (μPD7808)

Parameter	Symbol	Limits			Unit	Test Conditions
		Min	Typ	Max		
Data retention voltage	V_{DDDR}	3.2		5.5	V	RESET = V_{IL}
Data retention supply current	I_{DDDR}		1.3	3.0	mA	RESET = V_{IL} $V_{DDDR} = 3.2\text{ V}$

AC Characteristics (cont)

$V_{SS} = 0\text{ V}$, $V_{CC} - 0.8\text{ V} \leq V_{DD} \leq V_{CC}$; See Operating Conditions table

Parameter	Symbol	Limits				Unit	Test Conditions (1)
		$f_{XTAL} = 10\text{ MHz}$		$f_{XTAL} = 12\text{ MHz}$			
		Min	Max	Min	Max		
Read/Write Operation							
ALE ↓ to data input	t_{LDR}		300		215	ns	
\overline{RD} ↓ to data input	t_{RD}		250		180	ns	
ALE ↓ to \overline{RD} ↓ delay time	t_{LR}	50		35		ns	
Data hold time to \overline{RD} ↑	t_{RDH}	0		0		ns	
\overline{RD} ↑ to ALE ↑ delay time	t_{RL}	150		115		ns	
\overline{RD} width low	t_{RR}	350		280		ns	Data read
		650		530		ns	Opcode fetch
ALE width high	t_{LL}	160		125		ns	
$\overline{M1}$ setup time to ALE ↓	t_{ML}	100		65		ns	
$\overline{M1}$ hold time from ALE ↓	t_{LM}	70		50		ns	
$\overline{I0/M}$ setup time to ALE ↓	t_{iL}	100		65		ns	
$\overline{I0/M}$ hold time from ALE ↓	t_{lI}	70		50		ns	
Address to \overline{WR} ↓ delay	t_{AW}	200		150		ns	
ALE ↓ to data output	t_{LDW}		210		195	ns	
\overline{WR} ↓ to data output	t_{WD}		100		100	ns	
ALE ↓ to \overline{WR} ↓ delay	t_{LW}	50		35		ns	
Data set-up time to \overline{WR} ↑	t_{DW}	300		230		ns	
Data hold time to \overline{WR} ↑	t_{WDH}	130		95		ns	
\overline{WR} ↑ to ALE ↑ delay time	t_{WL}	150		115		ns	
\overline{WR} width low	t_{WW}	350		280		ns	

Note:

(1) Load capacitance: $C_L = 150\text{ pF}$.

4

Serial Operation

See Operating Conditions table

Parameter	Symbol	Limits				Unit	Test Conditions
		f _{XTAL} = 10 MHz		f _{XTAL} = 12 MHz			
		Min	Max	Min	Max		
SCK cycle time	t _{CYK}	1.2		1		μs	SCK input (4)
		500		500		ns	SCK input (5)
		2.4		2		μs	SCK output
SCK width low	t _{KKL}	500(6)		400(7)		ns	SCK input(4)
		200		200		ns	SCK input (5)
		1100		900		ns	SCK output
SCK width high	t _{KKH}	500(6)		400(7)		ns	SCK input (4)
		200		200		ns	SCK input (5)
		1100		900		ns	SCK output
RxD set-up time to SCK ↑	t _{RXK}	80		80		ns	(4)
RxD hold time after SCK ↑	t _{KRX}	80		80		ns	(4)
SCK ↓ TxD delay time	t _{KTX}		210		210	ns	(4)

Note:

- (4) 1x Baud rate in Asynchronous, Synchronous, or I/O Interface mode.
- (5) 16x Baud rate or 64x Baud rate in Asynchronous mode.
- (6) 505 ns min for μPD7808 only.
- (7) 420 ns min for μPD7808 only.

Zero-Cross Characteristics

Parameter	Symbol	Limits			Unit	Test Conditions
		Min	Typ	Max		
Zero-cross detection input	V _{ZX}	1		3(8)	VAC _{p-p}	AC coupled
Zero-cross accuracy	A _{ZX}			±135	mV	60 Hz sine wave
Zero-cross detection input frequency	f _{ZX}	0.05		1	kHz	

Note:

- (8) 1.8 VAC_{p-p} max for μPD7808 only.

Bus Timing Depending on t_{CYC}

Symbol	Calculating Expression	Min/Max
t _{RP}	60T	Min
t _{TI}	6T	Min
t _{CJ(2)}	6T	Min
t _{CJ(3)}	48T	Min
t _{IP}	36T	Min
t _{AL}	2T - 100	Min
t _{LA}	T - 30	Min
t _{AR}	3T - 100	Min
t _{AD}	7T - 220(4)	Max
t _{LDR}	5T - 200(4)	Max
t _{RD}	4T - 150(4)	Max
t _{LR}	T - 50	Min
t _{RL}	2T - 50	Min
t _{RR}	4T - 50 (Data Read)(4) 7T - 50 (Opcode Fetch)(4)	Min
t _{LL}	2T - 40	Min
t _{ML}	2T - 100	Min
t _{LM}	T - 30	Min
t _{IL}	2T - 100	Min
t _{LI}	T - 30	Min

Bus Timing Depending on t_{CYC} (cont)

Symbol	Calculating Expression	Min/Max
t _{AW}	3T - 100	Min
t _{LDW}	T + 110	Max
t _{LW}	T - 50	Min
t _{DW}	4T - 100(4)	Min
t _{WDH}	2T - 70	Min
t _{WL}	2T - 50	Min
t _{WW}	4T - 50(4)	Min
t _{CYK}	12T (SCK input)(1) 24T (SCK output)	Min
t _{KKL}	6T - 100 (SCK input)(1)(5) 12T - 100 (SCK output)	Min
t _{KKH}	6T - 100 (SCK input)(1)(5) 12T - 100 (SCK output)	Min

Note:

(1) 1x Baud rate in asynchronous, synchronous, or I/O interface mode.

$$T = t_{CYC} = \frac{1}{f_{XTAL}}$$

The items not included in this list are independent of oscillator frequency (f_{XTAL}).

(2) Event counter mode.

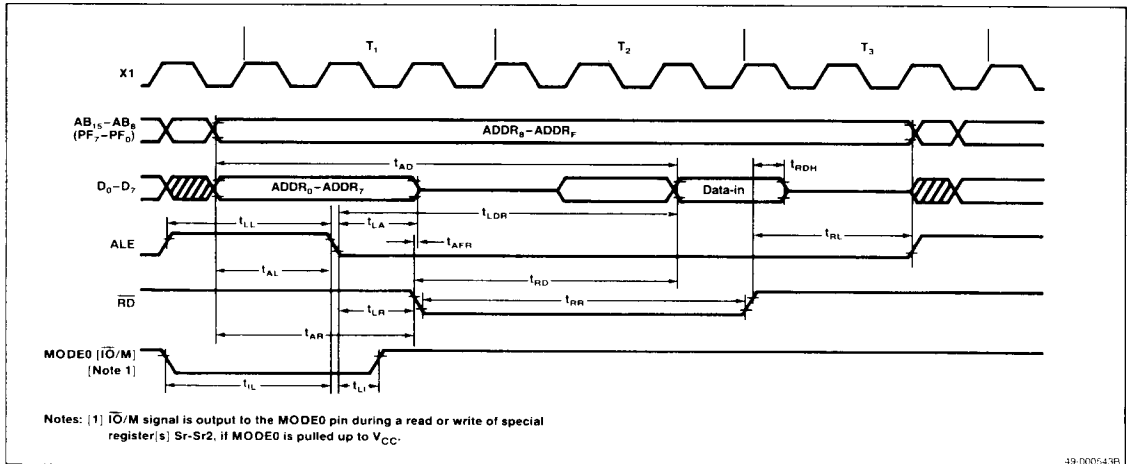
(3) Pulse width measurement mode.

(4) Add 3T when using external program memory with programmable WAIT function.

(5) 5T+5 (SCK input)(1) min for μPD7808 only.

Timing Waveforms

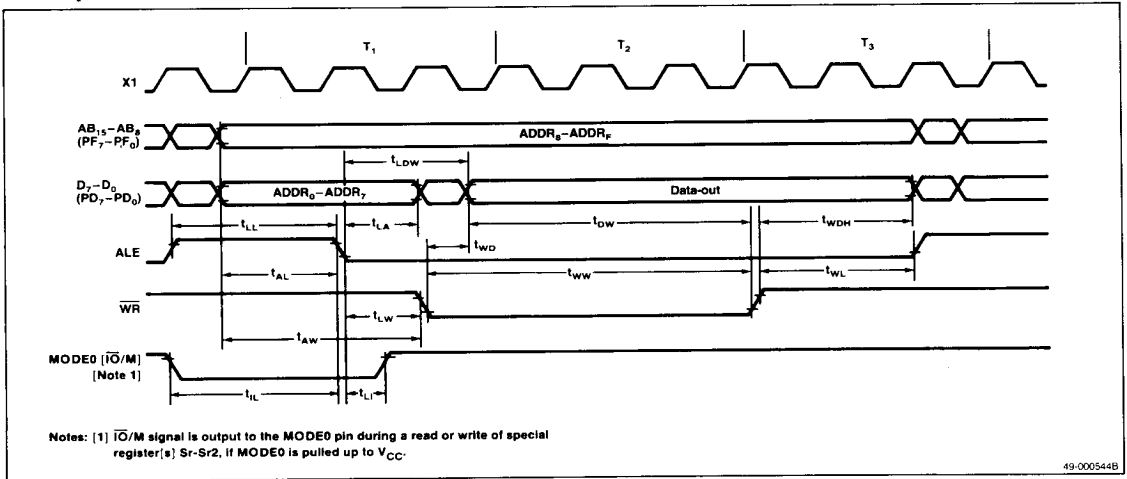
Read Operation



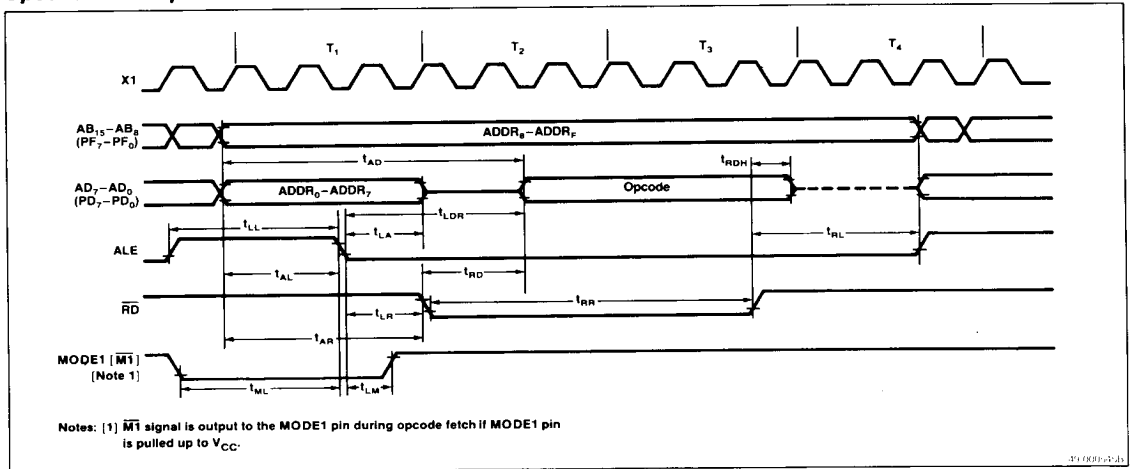
4

Timing Waveforms (cont)

Write Operation

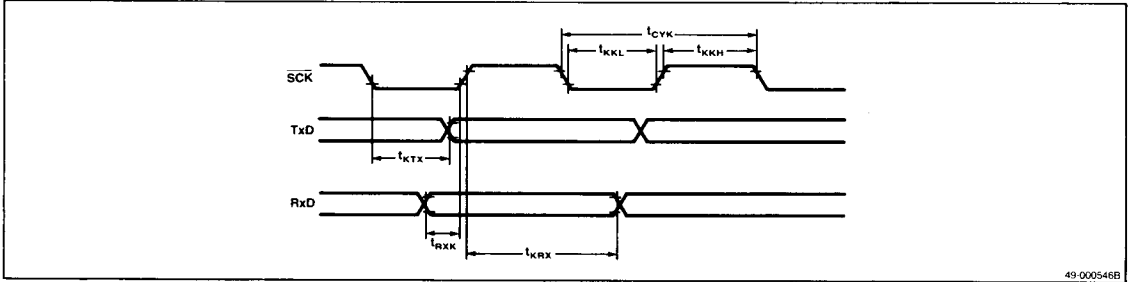


Opcode Fetch Operation

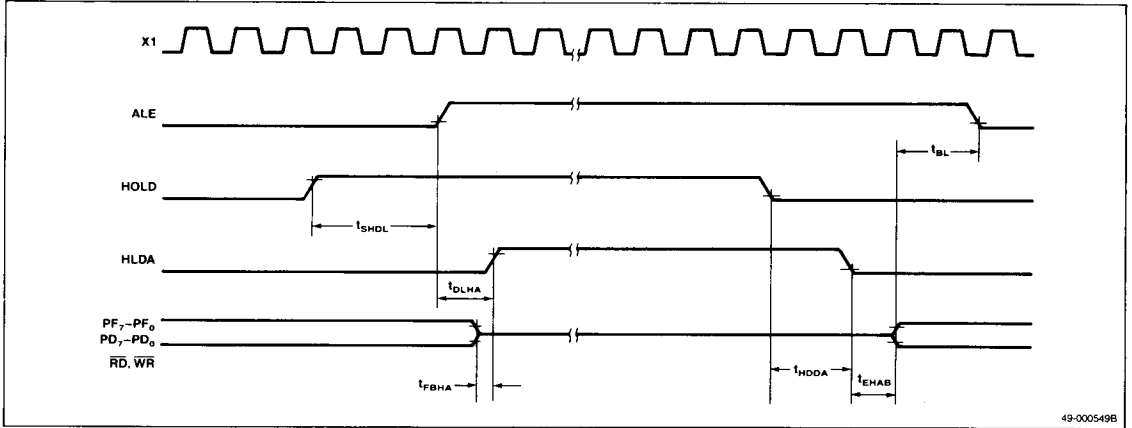


Timing Waveforms (cont)

Serial Operation Transmit/Receive Timing

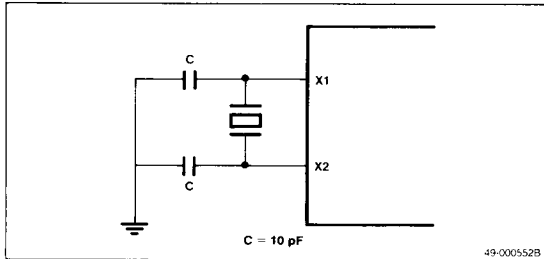


Hold Operation

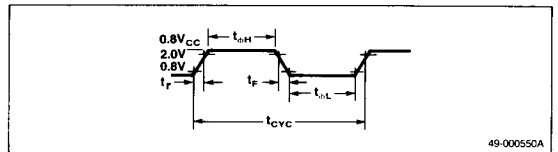


4

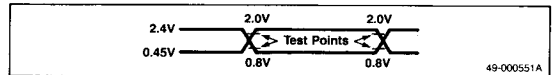
XTAL Oscillation Circuit



External Clock Timing



AC Timing Test Points



Instruction Set

In addition to the basic 7800 family instruction set, the μPD7807/08/09 executes the following types of instructions:

- 16-bit data transfers between memory, registers, and extended accumulator
- 16-bit addition and subtraction
- 16-bit comparison and skip
- 16-bit AND, OR, XOR operation
- 16-bit data shift and rotation
- Multiply; 8-bit by 8-bit, 16-bit product (less than 8 μs execution)
- Divide; 16-bit by 8-bit, 16-bit quotient, 8-bit remainder (less than 15 μs execution)
- Working register instruction for efficient RAM addressing, testing, and manipulating
- Direct bit addressing for code-efficient addressing, testing, and manipulating bits in RAM, port lines, and mode registers.

Operand Format/Description

Format	Description
r	V, A, B, C, D, E, H, L
r1	EAH, EAL, B, C, D, E, H, L
r2	A, B, C
sr	PA, PB, PC, PD, PF, MKH, MKL, SMH, SML, EOM, ETMM, TMM, MM, MCC, MA, MB, MC, MF, TxB, TM ₀ , TM ₁ , WDM, MT
sr1	PA, PB, PC, PD, PF, MKH, MKL, SMH, EOM, TMM, RxB, PT, WDM
sr2	PA, PB, PC, PD, PF, MKH, MKL, SMH, EOM, TMM
sr3	ETM ₀ , ETM ₁
sr4	ECNT, ECPT0, ECPT1
sr5	PA, PB, PC, PD, PF, MKH, MKL, SMH, EOM, TMM, PT
rp	SP, B, D, H
rp1	V, B, D, H, EA
rp2	SP, B, D, H, EA
rp3	B, D, H
rpa	B, D, H, D+, H+, D-, H-
rpa1	B, D, H
rpa2	B, D, H, D+, H+, D-, H-, D + byte, H + A, H + B, H + EA, H + byte
rpa3	D, H, D + +, H + +, D + byte, H + A, H + B, H + EA, H + byte
wa	8-Bit immediate data
word	16-Bit immediate data
byte	8-Bit immediate data
bit	8-Bit address of bit location
f	CY, HC, Z
irf	NMI, FT0, FT1, F1, F2, FE0, FE1, FEIN, FSR, FST, ER, OV, IFE2, SB

Instruction Set Symbol Definitions

Symbol	Description
←	Transfer direction, result
∧	Logical product (logical AND)
∨	Logical sum (logical OR)
⊕	Exclusive OR
—	Complement
•	Concatenation

Remarks

1. sr-sr5 (special register)

PA = Port A	ECNT = Timer/Event Counter Upcounter
PB = Port B	ECPT0 = Timer/Event Counter Capture 0
PC = Port C	ECPT1 = Timer/Event Counter Capture 1
PD = Port D	ETMM = Timer/Event Counter Mode
PF = Port F	EOM = Timer/Event Counter Output Mode
PT = Port T	WDM = Watchdog Timer Mode
MA = Mode A	TxB = Tx Buffer
MB = Mode B	RxB = Rx Buffer
MC = Mode C	SMH = Serial Mode High
MCC = Mode Control C	SML = Serial Mode Low
MF = Mode F	MKH = Mask High
MT = Mode T	MKL = Mask Low
MM = Memory Mapping	
TM ₀ = Timer Register 0	
TM ₁ = Timer Register 1	
TMM = Timer Mode	
ETM ₀ = Timer/Event Counter Register 0	
ETM ₁ = Timer/Event Counter Register 1	

2. rp-rp3 (register pair)

SP = Stack Pointer	H = HL
B = BC	V = VA
D = DE	EA = Extended Accumulator

3. rpa-rpa3 (rp addressing)

B = (BC)	D + + = (DE) + 2
D = (DE)	H + + = (HL) + 2
H = (HL)	D + byte = (DE) + byte
D + = (DE) + 1	H + A = (HL) + (A)
H - = (HL) + (B)	H + B = (HL) + (B)
D - = (DE) - 1	H + EA = (HL) + (EA)
H - = (HL) - 1	H + byte = (HL) + byte

4. f (flag)

CY = Carry	HC = Half Carry	Z = Zero
------------	-----------------	----------

5. irf (interrupt flag)

NMI = NMI input	FEIN = INTFEIN
FT0 = INTFT0	FSR = INTFSR
FT1 = INTFT1	FST = INTFST
F1 = INTF1	ER = Error
F2 = INTF2	OV = Overflow
FE0 = INTFE0	IE2 = Interrupt Enable F/F2
FE1 = INTFE1	SB = Standby

Instruction Set

Mnemonic	Operand	Operation	Operation Code																Bytes	Skip Condition					
			B1	B2	B3	B4	7	6	5	4	3	2	1	0	7	6	5	4			3	2	1	0	State(I)
8-Bit Data Transfer																									
MOV	r1,A (r1) ← (A)		0	0	0	1	1	T ₂	T ₁	T ₀													4		
	A,r1 (A) ← (r1)		0	0	0	0	1	T ₂	T ₁	T ₀													4		
	*sr,A (sr) ← (A)		0	1	0	0	1	1	0	1	1	1	S ₅	S ₄	S ₃	S ₂	S ₁	S ₀					10		
	*A,sr1 (A) ← (sr1)		0	1	0	0	1	1	0	0	1	1	S ₅	S ₄	S ₃	S ₂	S ₁	S ₀					10		
	r,word (r) ← (word)		0	1	1	0	0	0	0	0	0	1	1	0	1	R ₂	R ₁	R ₀					17		
	word,r (word) ← (r)		0	1	1	0	0	0	0	0	0	1	1	1	R ₂	R ₁	R ₀					17			
MVI	*r,byte (r) ← byte set L1 if r = A set L0 if r = L		0	1	1	0	1	R ₂	R ₁	R ₀													7	L1 = 1 and r = A L0 = 1 and r = L	
	sr2,byte (sr2) ← byte		0	1	1	0	0	1	0	0	S ₃	0	0	0	S ₂	S ₁	S ₀						14		
MVIW	*wa,byte ((V)*wa) ← byte		0	1	1	1	0	0	0	1													13		
	*rpa1,byte (rpa1) ← byte		0	1	0	0	1	0	A ₁	A ₀													10		
	*wa ((V)*wa) ← A		0	1	1	0	0	0	1	1													10		
	*wa (A) ← ((V)*wa)		0	0	0	0	0	0	0	1													10		
	*rpa2 (rpa2) ← (A)		A ₃	0	1	1	1	A ₂	A ₁	A ₀												7/13(3)			
	rpa2 (A) ← ((rpa2))		A ₃	0	1	0	1	A ₂	A ₁	A ₀												7/13(3)			
	(B) ↔ (B'),(C) ↔ (C'),(D) ↔ (D')		0	1	0	0	1	0	0	0	1	0	1	0	1	1	1	1						8	
	(E) ↔ (E'),(H) ↔ (H'),(L) ↔ (L')		0	1	0	0	1	0	0	0	1	0	1	0	1	1	1						8		
	(V) ↔ (V'),(A) ↔ (A'),(EA) ↔ (EA')		0	1	0	0	1	0	0	0	1	0	1	0	1	1	1						8		
	(H) ↔ (H'),(L) ↔ (L')		0	1	0	0	1	0	0	0	1	0	1	0	1	1	1						8		
	(V) ↔ (V'),(A) ↔ (A'),(B) ↔ (B')		0	1	0	0	1	0	0	0	1	0	1	0	1	1	1						8		
	(C) ↔ (C'),(D) ↔ (D'),(E) ↔ (E'),(H) ↔ (H'),(L) ↔ (L'),(EA) ↔ (EA')		0	1	0	0	1	0	0	0	1	0	1	0	1	1	1						8		
16-Bit Data Transfer																									
BLOCK	D + ((DE)) ← ((HL)),(DE) ← (DE) + 1, (HL) ← (HL) + 1, (C) ← (C) - 1 End if borrow		0	0	0	1	0	0	0	0													13 x (C+1)		
	D - ((DE)) ← ((HL)),(DE) ← (DE) - 1, (HL) ← (HL) - 1, (C) ← (C) - 1 End if borrow		0	0	0	1	0	0	0	1													13 x (C+1)		
DMOV	rp3,EA (rp3) ← (EA), (rp3H) ← (EAH)		1	0	1	1	0	1	P ₁	P ₀													4		
	EA,rp3 (EA) ← (rp3), (EAH) ← (rp3H)		1	0	1	0	0	1	P ₁	P ₀													4		

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code																Bytes	Skip Condition				
			B1	B2	B3	B4	7	6	5	4	3	2	1	0	State(1)	0								
16-Bit Data Transfer (cont)																								
DMOV	sr3, EA (sr3) ← (EA) EA, sr4 (EA) ← (sr4)		0	1	0	0	1	0	0	0	1	1	0	1	0	0	1	U ₀	14	2				
SBCD	word (word) ← (C), (word + 1) ← (B)		0	1	1	1	0	0	0	0	0	0	0	0	1	1	1	0	20	4				
		Low addr																High addr						
SDED	word (word) ← (E), (word + 1) ← (D)		0	1	1	1	0	0	0	0	0	0	0	1	0	1	1	0	20	4				
		Low addr																High addr						
SHLD	word (word) ← (L), (word + 1) ← (H)		0	1	1	1	0	0	0	0	0	0	1	1	1	1	0	20	4					
		Low addr																High addr						
SSPD	word (word) ← (SP _L), (word + 1) ← (SP _H)		0	1	1	1	0	0	0	0	0	0	0	0	1	1	1	0	20	4				
		Low addr																High addr						
STEAX	rpa3 ((rpa3)) ← (EAL), ((rpa3)) + 1 ← (EAH)		0	1	0	0	1	0	0	0	0	1	0	0	1	C ₃	C ₂	C ₁	C ₀	14/20(3)	3			
		Data(4)																						
LBCD	word (C) ← (word), (B) ← (word + 1)		0	1	1	1	0	0	0	0	0	0	0	1	1	1	1	1	20	4				
		Low addr																High addr						
LBED	word (E) ← (word), (D) ← (word + 1)		0	1	1	1	0	0	0	0	0	0	1	0	1	1	1	1	20	4				
		Low addr																High addr						
LHLD	word (L) ← (word), (H) ← (word + 1)		0	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	20	4				
		Low addr																High addr						
LSPD	word (SP _L) ← (word), (SP _H) ← ((word) + 1)		0	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	20	4				
		Low addr																High addr						
LDEAX	rpa3 (EAL) ← ((rpa3)), (EAH) ← ((rpa3) + 1)		0	1	0	0	1	0	0	0	0	1	0	0	0	C ₃	C ₂	C ₁	C ₀	14/20(3)	3			
		Data(4)																						
PUSH	rp1 ((SP) - 1) ← (rp1H), ((SP) - 2) ← (rp1L) (SP) ← (SP) - 2		1	0	1	1	0	Q ₂	Q ₁	Q ₀													13	1
POP	rp1 (rp1L) ← ((SP)), (rp1H) ← ((SP) + 1) (SP) ← (SP) + 2		1	0	1	0	0	Q ₂	Q ₁	Q ₀													10	1
LXI	*rp2, word (rp2) ← (word) set L0 fr rp2 = H		0	P ₂	P ₁	P ₀	0	1	0	0													10	3
		High byte																Low byte		L0 = 1 and rp2 = H				
TABLE	(C) ← ((PC)+3+(A)), (B) ← ((PC)+3+(A)+1)		0	1	0	0	1	0	0	0	1	0	1	0	1	0	0	0	17	2				
8-Bit Arithmetic (Register)																								
ADD	A,r (A) ← (A) + (r)		0	1	1	0	0	0	0	0	1	1	0	0	R ₂	R ₁	R ₀		8	2				
	r,A (r) ← (r) + (A)		0	1	1	0	0	0	0	0	0	1	0	0	R ₂	R ₁	R ₀		8	2				
ADC	A,r (A) ← (A) + (r) + (CY)		0	1	1	0	0	0	0	0	1	1	0	1	0	R ₂	R ₁	R ₀		8	2			
	r,A (r) ← (r) + (A) + (CY)		0	1	1	0	0	0	0	0	0	1	0	1	0	R ₂	R ₁	R ₀		8	2			

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code																Bytes	State(I)	Skip Condition
			B1				B3				B2				B4						
			7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0			
8-Bit Arithmetic (Register) (cont)																					
ADDNC	A,r (A) ← (A) + (r)		0	1	1	0	0	0	0	1	0	1	0	0	R ₂	R ₁	R ₀	8	2	No carry	
	r,A (r) ← (r) + (A)		0	1	1	0	0	0	0	0	0	1	0	0	R ₂	R ₁	R ₀	8	2	No carry	
SUB	A,r (A) ← (A) - (r)		0	1	1	0	0	0	0	1	1	1	0	0	R ₂	R ₁	R ₀	8	2		
	r,A (r) ← (r) - (A)		0	1	1	0	0	0	0	0	1	1	0	0	R ₂	R ₁	R ₀	8	2		
SBB	A,r (A) ← (A) - (r) - (CY)		0	1	1	0	0	0	0	1	1	1	1	0	R ₂	R ₁	R ₀	8	2		
	r,A (r) ← (r) - (A) - (CY)		0	1	1	0	0	0	0	0	1	1	1	0	R ₂	R ₁	R ₀	8	2		
SUBNB	A,r (A) ← (A) - (r)		0	1	1	0	0	0	0	1	0	1	1	0	R ₂	R ₁	R ₀	8	2	No borrow	
	r,A (r) ← (r) - (A)		0	1	1	0	0	0	0	0	1	0	1	0	R ₂	R ₁	R ₀	8	2	No borrow	
ANA	A,r (A) ← (A) ∧ (r)		0	1	1	0	0	0	0	1	0	0	0	1	R ₂	R ₁	R ₀	8	2		
	r,A (r) ← (r) ∧ (A)		0	1	1	0	0	0	0	0	0	0	0	1	R ₂	R ₁	R ₀	8	2		
ORA	A,r (A) ← (A) ∨ (r)		0	1	1	0	0	0	0	1	0	0	1	1	R ₂	R ₁	R ₀	8	2		
	r,A (r) ← (r) ∨ (A)		0	1	1	0	0	0	0	0	0	0	1	1	R ₂	R ₁	R ₀	8	2		
XRA	A,r (A) ← (A) ⊕ (r)		0	1	1	0	0	0	0	1	0	0	1	0	R ₂	R ₁	R ₀	8	2		
	r,A (r) ← (r) ⊕ (A)		0	1	1	0	0	0	0	0	0	0	1	0	R ₂	R ₁	R ₀	8	2		
GTA	A,r (A) - (r) - 1		0	1	1	0	0	0	0	1	0	1	0	1	R ₂	R ₁	R ₀	8	2	No borrow	
	r,A (r) - (A) - 1		0	1	1	0	0	0	0	0	0	1	0	1	R ₂	R ₁	R ₀	8	2	No borrow	
LTA	A,r (A) - (r)		0	1	1	0	0	0	0	1	0	1	1	1	R ₂	R ₁	R ₀	8	2	Borrow	
	r,A (r) - (A)		0	1	1	0	0	0	0	0	0	1	1	1	R ₂	R ₁	R ₀	8	2	Borrow	
NEA	A,r (A) - (r)		0	1	1	0	0	0	0	1	1	1	0	1	R ₂	R ₁	R ₀	8	2	No zero	
	r,A (r) - (A)		0	1	1	0	0	0	0	0	1	1	0	1	R ₂	R ₁	R ₀	8	2	No zero	
EOA	A,r (A) - (r)		0	1	1	0	0	0	0	1	1	1	1	1	R ₂	R ₁	R ₀	8	2	Zero	
	r,A (r) - (A)		0	1	1	0	0	0	0	0	1	1	1	1	R ₂	R ₁	R ₀	8	2	Zero	
ONA	A,r (A) ∧ (r)		0	1	1	0	0	0	0	1	1	0	0	1	R ₂	R ₁	R ₀	8	2	No zero	
	r,A (A) ∧ (r)		0	1	1	0	0	0	0	1	1	0	0	1	R ₂	R ₁	R ₀	8	2	Zero	
8-Bit Arithmetic (Memory)																					
ADDX	rpa (A) ← (A) + ((rpa))		0	1	1	1	0	0	0	1	1	0	0	0	A ₂	A ₁	A ₀	11	2		
ADCX	rpa (A) ← (A) + ((rpa)) + (CY)		0	1	1	1	0	0	0	1	1	0	1	0	A ₂	A ₁	A ₀	11	2		
ADDNCX	rpa (A) ← (A) + ((rpa))		0	1	1	1	0	0	0	1	0	1	0	0	A ₂	A ₁	A ₀	11	2	No carry	
SUBX	rpa (A) ← (A) - ((rpa))		0	1	1	1	0	0	0	1	1	1	0	0	A ₂	A ₁	A ₀	11	2		
SBBX	rpa (A) ← (A) - ((rpa)) - (CY)		0	1	1	1	0	0	0	1	1	1	1	0	A ₂	A ₁	A ₀	11	2		
SUBNBX	rpa (A) ← (A) - ((rpa))		0	1	1	1	0	0	0	1	0	1	1	0	A ₂	A ₁	A ₀	11	2	No borrow	
ANAX	rpa (A) ← (A) ∧ ((rpa))		0	1	1	1	0	0	0	1	0	0	0	1	A ₂	A ₁	A ₀	11	2		
OFAX	rpa (A) ← (A) ∨ ((rpa))		0	1	1	1	0	0	0	1	0	0	1	1	A ₂	A ₁	A ₀	11	2		

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code																Bytes	Skip Condition	
			B1	B3	B2	B4	7	6	5	4	3	2	1	0	State(1)						
8-Bit Arithmetic (Memory) (cont)																					
XRAX	rpa (A) ← (A) ∨ ((rpa))		0	1	1	1	0	0	0	0	1	0	0	1	0	A ₂	A ₁	A ₀	11	2	
GTAX	rpa (A) ← ((rpa)) - 1		0	1	1	1	0	0	0	0	1	0	1	0	1	A ₂	A ₁	A ₀	11	2	No borrow
LTAX	rpa (A) ← ((rpa))		0	1	1	1	0	0	0	0	1	0	1	1	1	A ₂	A ₁	A ₀	11	2	Borrow
NEAX	rpa (A) ← ((rpa))		0	1	1	1	0	0	0	0	1	1	1	0	1	A ₂	A ₁	A ₀	11	2	No zero
EQAX	rpa (A) ← ((rpa))		0	1	1	1	0	0	0	0	1	1	1	1	1	A ₂	A ₁	A ₀	11	2	Zero
ONAX	rpa (A) ∧ ((rpa))		0	1	1	1	0	0	0	0	1	1	0	1	1	A ₂	A ₁	A ₀	11	2	No zero
OFFAX	rpa (A) ∧ ((rpa))		0	1	1	1	0	0	0	0	1	1	0	1	1	A ₂	A ₁	A ₀	11	2	Zero
Immediate Data																					
ADI	*A.byte (A) ← (A) + byte		0	1	0	0	1	1	0	Data							7	2			
	r.byte (r) ← (r) + byte		0	1	1	1	0	1	0	0	Data							11	3		
	sr2.byte (sr2) ← (sr2) + byte		0	1	1	0	1	0	0	Data							20	3			
ACI	*A.byte (A) ← (A) + byte + (CY)		0	1	0	1	0	1	1	0	Data							7	2		
	r.byte (r) ← (r) + byte + (CY)		0	1	1	1	0	1	0	0	Data							11	3		
	sr2.byte (sr2) ← (sr2) + byte + (CY)		0	1	1	0	1	0	0	Data							20	3			
ADINC	*A.byte (A) ← (A) + byte		0	0	1	0	0	1	1	0	Data							7	2	No carry	
	r.byte (r) ← (r) + byte		0	1	1	1	0	1	0	0	Data							11	3	No carry	
	sr2.byte (sr2) ← (sr2) + byte		0	1	1	0	1	0	0	Data							20	3	No carry		
SUI	*A.byte (A) ← (A) - byte		0	1	1	0	0	1	1	0	Data							7	2		
	r.byte (r) ← (r) - byte		0	1	1	1	0	1	0	0	Data							11	3		
	sr2.byte (sr2) ← (sr2) - byte		0	1	1	0	1	0	0	Data							20	3			
SBI	*A.byte (A) ← (A) - byte - (CY)		0	1	1	0	1	1	0	Data							7	2			
	r.byte (r) ← (r) - byte - (CY)		0	1	1	1	0	1	0	0	Data							11	3		
	sr2.byte (sr2) ← (sr2) - byte - (CY)		0	1	1	0	1	0	0	Data							20	3			

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code																Bytes	Skip Condition
			B1	B2	B3	B4	7	6	5	4	3	2	1	0	State(1)					
Immediate Data (cont)																				
SUIB	*A.byte (A) ← (A) - byte		0	0	1	1	0	1	1	0	Data						7	2	No borrow	
	r.byte (r) ← (r) - byte		0	1	1	1	0	1	0	0	Data						11	3	No borrow	
	sr2.byte (sr2) ← (sr2) - byte		0	1	1	0	0	1	0	0	Data						20	3	No borrow	
ANI	*A.byte (A) ← (A) ∧ byte		0	0	0	0	1	1	1	Data						7	2			
	r.byte (r) ← (r) ∧ byte		0	1	1	1	0	1	0	Data						11	3			
	sr2.byte (sr2) ← (sr2) ∧ byte		0	1	1	0	1	0	0	Data						20	3			
ORI	*A.byte (A) ← (A) ∨ byte		0	0	1	0	1	1	1	Data						7	2			
	r.byte (r) ← (r) ∨ byte		0	1	1	1	0	1	0	Data						11	3			
	sr2.byte (sr2) ← (sr2) ∨ byte		0	1	1	0	1	0	0	Data						20	3			
XRI	*A.byte (A) ← (A) ⊕ byte		0	0	1	0	1	1	0	Data						7	2			
	r.byte (r) ← (r) ⊕ byte		0	1	1	1	0	1	0	Data						11	3			
	sr2.byte (sr2) ← (sr2) ⊕ byte		0	1	1	0	1	0	0	Data						20	3			
GTI	*A.byte (A) - byte - 1		0	0	1	0	0	1	1	Data						7	2	No borrow		
	r.byte (r) - byte - 1		0	1	1	1	0	1	0	Data						11	3	No borrow		
	sr5.byte (sr5) - byte - 1		0	1	1	0	0	1	0	Data						14	3	No borrow		
LTI	*A.byte (A) - byte		0	0	1	1	0	1	1	Data						7	2	Borrow		
	r.byte (r) - byte		0	1	1	1	0	1	0	Data						11	3	Borrow		
	sr5.byte (sr5) - byte		0	1	1	0	0	1	0	Data						14	3	Borrow		
NEI	*A.byte (A) - byte		0	1	1	0	0	1	1	Data						7	2	No zero		
	r.byte (r) - byte		0	1	1	1	0	1	0	Data						11	3	No zero		

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code																Skip Condition		
			B1	B2	B3	B4	7	6	5	4	3	2	1	0	State(I)	Bytes					
Immediate Data (cont)																					
NEI	sr5.byte (sr5) - byte		0	1	1	0	0	1	0	0	S ₃	1	1	0	1	S ₂	S ₁	S ₀	14	3	No Zero
Data																					
EQI	*A.byte (A) - byte		0	1	1	0	1	1	1	Data									7	2	Zero
	r.byte (r) - byte		0	1	1	0	1	0	0	0	1	1	1	1	R ₂	R ₁	R ₀	11	3	Zero	
Data																					
	sr5.byte (sr5) - byte		0	1	1	0	0	1	0	0	S ₃	1	1	1	1	S ₂	S ₁	S ₀	14	3	Zero
Data																					
ONI	*A.byte (A) ^ byte		0	1	0	0	1	1	1	Data									7	2	No Zero
	r.byte (r) ^ byte		0	1	1	0	1	0	0	0	1	0	0	1	R ₂	R ₁	R ₀	11	3	No Zero	
Data																					
	sr5.byte (sr5) ^ byte		0	1	1	0	0	1	0	0	S ₃	1	0	0	1	S ₂	S ₁	S ₀	14	3	No Zero
Data																					
OFFI	*A.byte (A) ^ byte		0	1	0	1	0	1	1	Data									7	2	Zero
	r.byte (r) ^ byte		0	1	1	0	1	0	0	0	1	0	1	1	R ₂	R ₁	R ₀	11	3	Zero	
Data																					
	sr5.byte (sr5) ^ byte		0	1	1	0	0	1	0	0	S ₃	1	0	1	1	S ₂	S ₁	S ₀	14	3	Zero
Data																					
Working Register																					
ADDW	wa (A) ← (A) + ((V)•(wa))		0	1	1	1	0	1	0	0	1	1	0	0	0	0	0	0	14	3	
Offset																					
ADCW	wa (A) ← (A) + ((V)•(wa)) + (CY)		0	1	1	1	0	1	0	0	1	1	0	1	0	0	0	0	14	3	
Offset																					
ADDNCW	wa (A) ← (A) + ((V)•(wa))		0	1	1	1	0	1	0	0	1	0	1	0	0	0	0	0	14	3	No carry
Offset																					
SUBW	wa (A) ← (A) - ((V)•(wa))		0	1	1	1	0	1	0	0	1	1	1	0	0	0	0	0	14	3	
Offset																					
SBBW	wa (A) ← (A) - ((V)•(wa)) - (CY)		0	1	1	1	0	1	0	0	1	1	1	1	0	0	0	0	14	3	
Offset																					
SUBNBW	wa (A) ← (A) - ((V)•(wa))		0	1	1	1	0	1	0	0	1	0	1	1	0	0	0	0	14	3	No borrow
Offset																					
ANAW	wa (A) ← (A) ^ ((V)•(wa))		0	1	1	1	0	1	0	0	1	0	0	0	1	0	0	0	14	3	
Offset																					

Instruction Set (cont)

Mnemonic Working Register (cont)	Operand	Operation	Operation Code																Bytes	Skip Condition				
			B1	B2	B3	B4	7	6	5	4	3	2	1	0	7	6	5	4			3	2	1	0
ORAW	wa	(A) ← (A) V ((V)•(wa))	0	1	1	1	0	1	0	0	1	0	0	1	0	0	1	0	0	0	0	14	3	
		Offset																						
XRAW	wa	(A) ← (A) V ((V)•(wa))	0	1	1	1	0	1	0	0	1	0	0	1	0	0	1	0	0	0	0	14	3	
		Offset																						
GTAW	wa	(A) ← ((V)•(wa)) - 1	0	1	1	1	0	1	0	0	1	0	1	0	1	0	0	0	0	0	0	14	3	No borrow
		Offset																						
LTAW	wa	(A) ← ((V)•(wa))	0	1	1	1	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0	14	3	Borrow
		Offset																						
NEAW	wa	(A) ← ((V)•(wa))	0	1	1	1	0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	14	3	No zero
		Offset																						
EQAW	wa	(A) ← ((V)•(wa))	0	1	1	1	0	1	0	0	1	1	1	1	0	0	0	0	0	0	0	14	3	Zero
		Offset																						
ONAW	wa	(A) ^ ((V)•(wa))	0	1	1	1	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	14	3	No zero
		Offset																						
OFFAW	wa	(A) ^ ((V)•(wa))	0	1	1	1	0	1	0	0	1	1	0	1	0	0	0	0	0	0	0	14	3	Zero
		Offset																						
ANIW	*wa.byte	((V)•(wa)) ← ((V)•(wa)) ^ byte	0	0	0	0	0	1	0	1	0	1										19	3	
		Data																						
ORIW	*wa.byte	((V)•(wa)) ← ((V)•(wa)) V byte	0	0	0	1	0	1	0	1												19	3	
		Data																						
GTIW	*wa.byte	((V)•(wa)) - byte - 1	0	0	1	0	0	1	0	1												13	3	No borrow
		Data																						
LTIW	*wa.byte	((V)•(wa)) - byte	0	0	1	1	0	1	0	1												13	3	Borrow
		Data																						
NEIW	*wa.byte	((V)•(wa)) - byte	0	1	1	0	0	1	0	1												13	3	No zero
		Data																						
EQIW	*wa.byte	((V)•(wa)) - byte	0	1	1	1	0	1	0	1												13	3	Zero
		Data																						
ONIW	*wa.byte	((V)•(wa)) ^ byte	0	1	0	0	0	1	0	1												13	3	No zero
		Data																						
OFFIW	*wa.byte	((V)•(wa)) ^ byte	0	1	0	1	0	1	0	1												13	3	Zero
		Data																						

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code																Skip Condition		
			B1				B3				B2										
			7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	State(t)	Bytes	
16-Bit Arithmetic																					
EADD	EA,r2	$(EA) \leftarrow (EA) + (r2)$	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	0	R ₁ R ₀	11	2
DADD	EA,rp3	$(EA) \leftarrow (EA) + (rp3)$	0	1	1	1	0	1	0	0	1	1	0	0	0	1	1	0	P ₁ P ₀	11	2
DADC	EA,rp3	$(EA) \leftarrow (EA) + (rp3) + (CY)$	0	1	1	1	0	1	0	0	1	1	0	1	0	1	1	0	P ₁ P ₀	11	2
DADDC	EA,rp3	$(EA) \leftarrow (EA) + (rp3)$	0	1	1	1	0	1	0	0	1	0	1	0	0	1	1	0	P ₁ P ₀	11	2
ESUB	EA,r2	$(EA) \leftarrow (EA) - (r2)$	0	1	1	1	0	0	0	0	0	1	1	0	0	0	0	0	R ₁ R ₀	11	2
DSUB	EA,rp3	$(EA) \leftarrow (EA) - (rp3)$	0	1	1	1	0	1	0	0	1	1	1	0	0	1	1	0	P ₁ P ₀	11	2
DSBB	EA,rp3	$(EA) \leftarrow (EA) - (rp3) - (CY)$	0	1	1	1	0	1	0	0	1	0	1	1	0	1	1	0	P ₁ P ₀	11	2
DSUBNB	EA,rp3	$(EA) \leftarrow (EA) - (rp3)$	0	1	1	1	0	1	0	0	1	0	1	1	0	1	1	0	P ₁ P ₀	11	2
DAN	EA,rp3	$(EA) \leftarrow (EA) \wedge (rp3)$	0	1	1	1	0	1	0	0	1	0	0	0	1	1	1	0	P ₁ P ₀	11	2
DOR	EA,rp3	$(EA) \leftarrow (EA) \vee (rp3)$	0	1	1	1	0	1	0	0	1	0	0	1	1	1	1	0	P ₁ P ₀	11	2
DXR	EA,rp3	$(EA) \leftarrow (EA) \oplus (rp3)$	0	1	1	1	0	1	0	0	1	0	0	1	0	1	1	0	P ₁ P ₀	11	2
DGT	EA,rp3	$(EA) - (rp3) - 1$	0	1	1	1	0	1	0	0	1	0	1	0	1	1	1	0	P ₁ P ₀	11	2
DLT	EA,rp3	$(EA) - (rp3)$	0	1	1	1	0	1	0	0	1	0	1	1	1	1	1	0	P ₁ P ₀	11	2
DNE	EA,rp3	$(EA) - (rp3)$	0	1	1	1	0	1	0	0	1	1	1	0	1	1	1	0	P ₁ P ₀	11	2
DEQ	EA,rp3	$(EA) - (rp3)$	0	1	1	1	0	1	0	0	1	1	1	1	1	1	1	0	P ₁ P ₀	11	2
DON	EA,rp3	$(EA) \wedge (rp3)$	0	1	1	1	0	1	0	0	1	1	0	0	1	1	1	0	P ₁ P ₀	11	2
DOFF	EA,rp3	$(EA) \wedge (rp3)$	0	1	1	1	0	1	0	0	1	1	0	1	1	1	1	0	P ₁ P ₀	11	2
Multiply/Divide																					
MUL	r2	$(EA) \leftarrow (A) \times (r2)$	0	1	0	0	1	0	0	0	0	0	0	1	0	1	1	0	R ₁ R ₀	32	2
DIV	r2	$(EA) \leftarrow (EA) \div (r2), (r2) \leftarrow \text{Remainder}$	0	1	0	0	1	0	0	0	0	0	0	1	1	1	1	0	R ₁ R ₀	59	2
Increment/Decrement																					
INR	r2	$(r2) \leftarrow (r2) + 1$	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	R ₁ R ₀	4	1
INRW	*wa	$((W) \leftarrow (wa)) \leftarrow ((W) \leftarrow (wa)) + 1$	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	Offset	16	2
INX	rp	$(rp) \leftarrow (rp) + 1$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Offset	7	1
	EA	$(EA) \leftarrow (EA) + 1$	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0		7	1
DCR	r2	$(r2) \leftarrow (r2) - 1$	0	1	0	1	0	0	0	0	0	1	0	1	0	0	0	0	R ₁ R ₀	4	1
DCRW	wa	$((W) \leftarrow (wa)) \leftarrow ((W) \leftarrow (wa)) - 1$	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	Offset	16	2
DCX	rp	$(rp) \leftarrow (rp) - 1$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		7	1
	EA	$(EA) \leftarrow (EA) - 1$	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0		7	1
Others																					
DAA		Decimal Adjust Accumulator	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0		4	1
STC	(CY) ← 1		0	1	0	0	1	0	0	0	0	0	1	0	0	1	0	1		8	2
CLC	(CY) ← 0		0	1	0	0	1	0	0	0	0	0	1	0	0	0	1	0		8	2
CMC	(CY) ← (CY)		0	1	0	0	1	0	0	0	1	0	1	0	1	0	1	0		8	2

Instruction Set (cont)

Mnemonic Others (cont)	Operand	Operation	Operation Code																Bytes	Skip Condition			
			7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0			State(I)		
NEGA	(A) ← (A) + 1		0	1	0	0	1	0	0	0	0	0	0	0	1	1	0	1	0	8	2		
Rotate and Shift																							
RLO	Rotate left digit		0	1	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	17	2		
RRO	Rotate right digit		0	1	0	0	1	0	0	0	0	0	0	1	1	0	0	1	17	2			
RLL	(R _{2n+1}) ← (R _{2m}), (R ₂₀) ← (CY), (CY) ← (R ₂₇)		0	1	0	0	1	0	0	0	0	0	0	1	0	1	0	1	R ₁ R ₀	8	2		
RLR	(R _{2n+1}) ← (R _{2m}), (R ₂₇) ← (CY), (CY) ← (R ₂₀)		0	1	0	0	1	0	0	0	0	0	0	1	1	0	0	1	R ₁ R ₀	8	2		
SLL	(R _{2n+1}) ← (R _{2m}), (R ₂₀) ← 0, (CY) ← (R ₂₇)		0	1	0	0	1	0	0	0	0	0	0	1	0	0	1	R ₁ R ₀	8	2			
SLR	(R _{2n+1}) ← (R _{2m}), (R ₂₇) ← 0, (CY) ← (R ₂₀)		0	1	0	0	1	0	0	0	0	0	0	1	0	0	0	R ₁ R ₀	8	2			
SLLC	(R _{2n+1}) ← (R _{2m}), (R ₂₀) ← 0, (CY) ← (R ₂₇)		0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	R ₁ R ₀	8	2			
SLRC	(R _{2n+1}) ← (R _{2m}), (R ₂₇) ← 0, (CY) ← (R ₂₀)		0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	R ₁ R ₀	8	2			
DRLL	EA ← (EA _n) + 1 ← (EA _n), (EA ₀) ← (CY), (CY) ← (EA ₁₅)		0	1	0	0	1	0	0	0	0	1	0	1	0	1	0	0		8	2		
DRLR	EA ← (EA _n) - 1 ← (EA _n), (EA ₁₅) ← (CY), (CY) ← (EA ₀)		0	1	0	0	1	0	0	0	1	0	1	0	0	0	0		8	2			
DSLL	EA ← (EA _n) + 1 ← (EA _n), (EA ₀) ← 0, (CY) ← (EA ₁₅)		0	1	0	0	1	0	0	0	1	0	1	0	0	1	0	0		8	2		
DSLRL	EA ← (EA _n) - 1 ← (EA _n), (EA ₁₅) ← 0, (CY) ← (EA ₀)		0	1	0	0	1	0	0	0	1	0	1	0	0	0	0		8	2			
Jump																							
JMP	*word (PC) ← word		0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	High addr	Low addr	10	3	
JB	(PC _H) ← (B), (PC _L) ← (C)		0	0	1	0	0	0	0	0	1									4	1		
JR	word (PC) ← (PC) + 1 + jdisp 1		1	1	←	jdisp 1	→													10	1		
JRE	*word (PC) ← (PC) + 2 + jdisp		0	1	0	0	1	1	1	←	jdisp	→									10	2	
JEA	(PC) ← (EA)		0	1	0	0	1	0	0	0	0	0	1	0	1	0	0	0			8	2	
Call																							
CALL	*word (SP - 1) ← ((PC) + 3) _H , (SP - 2) ← ((PC) + 3) _L , (PC) ← word, (SP) ← (SP) - 2		0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	High addr	Low addr	16	3	
CALB	(SP - 1) ← ((PC) + 2) _H , (SP - 2) ← ((PC) + 2) _L , (PC _H) ← (B), (PC _L) ← (C), (SP) ← (SP) - 2		0	1	0	0	1	0	0	0	0	0	0	1	0	1	0	0	1			17	2
CALF	*word (SP - 1) ← ((PC) + 2) _H , (SP - 2) ← ((PC) + 2) _L , (PC _{15:11}) ← 00001, (PC _{0:10}) ← fa, (SP) ← (SP) - 2		0	1	1	1	1	←	fa	→											13	2	

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code																Bytes	State(!)	Skip Condition
			7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0			
CALL	word	$((SP) - 1) \leftarrow (PC) + 1)_H$, $((SP) - 2) \leftarrow (PC) + 1)_L$, $(PC)_L \leftarrow (128 + 2ia), (PC)_H \leftarrow (128 + 2ia), (SP) \leftarrow (SP) - 2$	1	0	0	←	←	←	←	←	←	←	←	←	←	←	16	1			
SOFTI		$((SP) - 1) \leftarrow (PSW), (SP) - 2) \leftarrow ((PC) + 1)_H$, $((PC) + 1)_L, ((SP) - 3) \leftarrow ((PC) + 1)_L$, $(PC) \leftarrow 0060H, (SP) \leftarrow (SP) - 3$	0	1	1	1	0	0	1	0							16	1			
Return																					
RET		$(PC)_L \leftarrow ((SP), (PC)_H) \leftarrow ((SP) + 1)$, $(SP) \leftarrow (SP) + 2$	1	0	1	1	1	0	0	0							10	1			
RETS		$(PC)_L \leftarrow (SP), (PC)_H \leftarrow (SP) + 1)$, $(SP) \leftarrow (SP) + 2, (PC) \leftarrow (PC) + n$	1	0	1	1	1	0	0	1							10	1	Unconditional Skip		
RETI		$(PC)_L \leftarrow (SP), (PC)_H \leftarrow (SP) + 1)$, $(PSW) \leftarrow ((SP) + 2), (SP) \leftarrow (SP) + 3$	0	1	1	0	0	0	1	0							13	1			
Bit Manipulation																					
MOV	*CY, bit	$(CY) \leftarrow (bit)$	0	1	0	1	1	1	1	1	1						10	2	Bit Addr		
	*bit CY	$(bit) \leftarrow (CY)$	0	1	0	1	1	0	1	0							13	2	Bit Addr		
AND	*CY, bit	$(CY) \leftarrow (CY) \wedge (bit)$	0	0	1	1	0	0	0	1							10	2	Bit Addr		
OR	*CY, bit	$(CY) \leftarrow (CY) \vee (bit)$	0	1	0	1	1	1	0	0							10	2	Bit Addr		
XOR	*CY, bit	$(CY) \leftarrow (CY) \oplus (bit)$	0	1	0	1	1	1	1	0							10	2	Bit Addr		
SETB	*bit	$(bit) \leftarrow 1$	0	1	0	1	1	0	0	0							13	2	Bit Addr		
CLR	*bit	$(bit) \leftarrow 0$	0	1	0	1	1	0	1	1							13	2	Bit Addr		
NOT	*bit	$(bit) \leftarrow \overline{(bit)}$	0	1	0	1	1	0	0	1							13	2	Bit Addr		
SK	*bit	Skip if (bit) = 1	0	1	0	1	1	1	0	1							10	2	Bit Addr (bit) = 1		
SKN	*bit	Skip if (bit) = 0	0	1	0	1	0	0	0	0							10	2	Bit Addr (bit) = 0		

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code																Bytes	Skip Condition		
			B1				B3				B2										B4	
CPU Control			7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	State(1)			
SK	f	Skip if f = 1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	F ₂ F ₁ F ₀	8	2	f = 1
SKN	f	Skip if f = 0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	1	F ₂ F ₁ F ₀	8	2	f = 0
SKIT	irf	Skip if irf = 1, then reset irf	0	1	0	0	1	0	0	0	0	1	0	1	0	1	0	1	I ₂ I ₁ I ₀	8	2	irf = 1
SKNIT	irf	Skip if irf = 0 Reset irf if irf = 1	0	1	0	0	1	0	0	0	0	1	1	1	0	1	0	1	I ₂ I ₁ I ₀	8	2	irf = 0
NOP		No operation	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		4	1	
EI		Enable interrupt	1	0	1	0	1	0	1	0										4	1	
DI		Disable interrupt	1	0	1	1	1	0	1	0										4	1	
HLT		Halt	0	1	0	0	1	0	0	0	0	0	1	1	0	1	1	0		11	2	

Notes:

- (1) In the case of skip condition, the idle states are as follows:
 1-byte instruction: 4 states
 2-byte instruction (with *): 7 states
 3-byte instruction (with *): 10 states
 4-byte instruction: 14 states
- (2) B2 (Data): rpa2 = D + byte, H + byte
- (3) Right side of slash (/) in states indicates case rpa2, rpa3 = D + byte, H + A, H + B, H + EA, H + byte
- (4) B3 (Data): rpa3 = D + byte, H + byte