NEC Electronics Inc.

μPD7810/11
8-BIT, SINGLE-CHIP NMOS MICROCOMPUTERS WITH A/D CONVERTER

Description

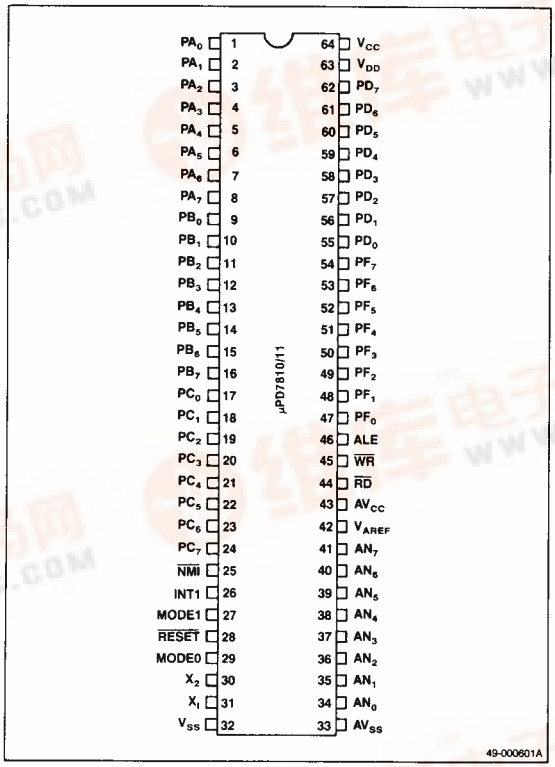
The μPD7810 and μPD7811 single-chip microcomputers integrate sophisticated on-chip peripheral functionality normally provided by external components. The device's internal 16-bit ALU and data paths, combined with a powerful instruction set and addressing, make the μPD7810/11 appropriate in data processing as well as control applications. The devices integrate a 16-bit ALU, 4K-ROM, 256-byte RAM with an 8-channel A/D converter, a multifunction 16-bit timer/event counter, two 8-bit timers, a USART, and two zero-cross detect inputs on a single die, allowing their use in fast, high end processing applications. This involves analog signal interface and processing.

The μPD7811 is the mask-ROM high volume production device embedded with custom customer program. The μPD7810 is a ROM-less version for prototyping and small volume production. The μPD78PG11E is a piggy-back EPROM version for design development.

Features

- ☐ NMOS silicon gate technology requiring +5 V power supply
- ☐ Complete single-chip microcomputer
  - — 16-bit ALU
  - — 4K x 8 ROM
  - — 256-byte RAM
- ☐ 44 I/O lines
- ☐ Two zero-cross detect inputs
- ☐ Two 8-bit timers
- ☐ Expansion capabilities
  - — 8085A bus-compatible
  - — 60K-byte external memory address range
- ☐ 8-channel, 8-bit A/D converter
  - — Autoscan mode
  - — Channel select mode
- ☐ Full duplex USART
  - — Synchronous and asynchronous
- ☐ 153 instructions
  - — 16-bit arithmetic, multiply and divide
- ☐ 1 μs instruction cycle time (12 MHz operation)
- ☐ Prioritized interrupt structure
  - — 3 external
  - — 8 internal
- ☐ Standby function
- ☐ On-chip clock generator
- ☐ 64-pin plastic QUIP or shrink DIP

Pin Configuration



Ordering Information

Part Number	Package Type	Max Frequency of Operation
μPD7810G-36 μPD7811G-36	64-pin plastic QUIP	12 MHz
μPD7810CW μPD7811CW	64-pin plastic shrink DIP	12 MHz

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/RxD	Port C I/O line 1/Receive data input
19	PC_2/$\overline{SCK}$	Port C I/O line 2/Serial clock I/O
20	PC_3/TI/$\overline{INT2}$	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	$\overline{NMI}$	Nonmaskable interrupt input
26	INT1	Interrupt request 1 input
27	MODE1/$\overline{M1}$	Mode 1 input/Memory cycle 1 output
28	RESET	Reset input
29	MODE0/$\overline{IO}$/M	Mode 0 input/I/O/Memory output
30, 31	X2, X1	Crystal connections 1, 2
32	V_{SS}	Ground
33	AV_{SS}	Port T threshold voltage input
34-41	AN_0-AN_7	A/D converter analog inputs 0-7
42	V_{AREF}	A/D converter reference voltage
43	AV_{CC}	A/D converter power supply
44	$\overline{RD}$	Read strobe output
45	$\overline{WR}$	Write strobe output
46	ALE	Address latch enable output
47-54	PF_0-PF_7	Port F I/O/Expansiom memory address bus (bits 8-15)
55-62	PD_0-PD_7	Port D I/O/Expansion memory address/data bus
63	V_{DD}	RAM backup power supply
64	V_{CC}	5 V power supply

Pin Functions

PA_0-PA_7 [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.

PB_0-PB_7 [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.

PC_0-PC_7 [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.

$\overline{SCK}$ [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.

$\overline{INT2}$ [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/counter.

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

CO_0, CO_1 [Counter Outputs 0, 1]. Programmable rectangular wave outputs based on timer/event counter.

PD_0-PD_7 [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.

PF_0-PF_7 [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.

AN_0-AN_7

These are the eight analog inputs to the A/D converter. AN_4-AN_7 can also be used as a digital input for falling edge detection.

AV_{SS} [A/D Converter Power Ground]

AV_{SS} is the ground potential for the A/D converter power supply.

$\overline{NMI}$ [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.

$\overline{\text{RESET}}$ [Reset]

When the $\overline{\text{RESET}}$ input is brought low, it initializes the μPD7810/11.

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 $\overline{\text{IO}}$/M signal.

V_{AREF} [A/D Converter Reference]

V_{AREF} set the upper limit for the A/D converter's conversion range.

AV_{CC} [A/D Converter Power]

This is the power supply voltage for the A/D converter.

$\overline{\text{RD}}$ [Read Strobe]

The $\overline{\text{RD}}$ output goes low to gate data from external devices onto the data bus. $\overline{\text{RD}}$ goes high during reset.

$\overline{\text{WR}}$ [Write Strobe]

The $\overline{\text{WR}}$ 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. $\overline{\text{WR}}$ goes high during reset.

ALE [Address Latch Enable]

The ALE output latches the address signal to the output of PD_0-PD_7.

X1, X2 [Crystal Connections 1, 2]

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

V_{SS} [Ground]

Ground potential.

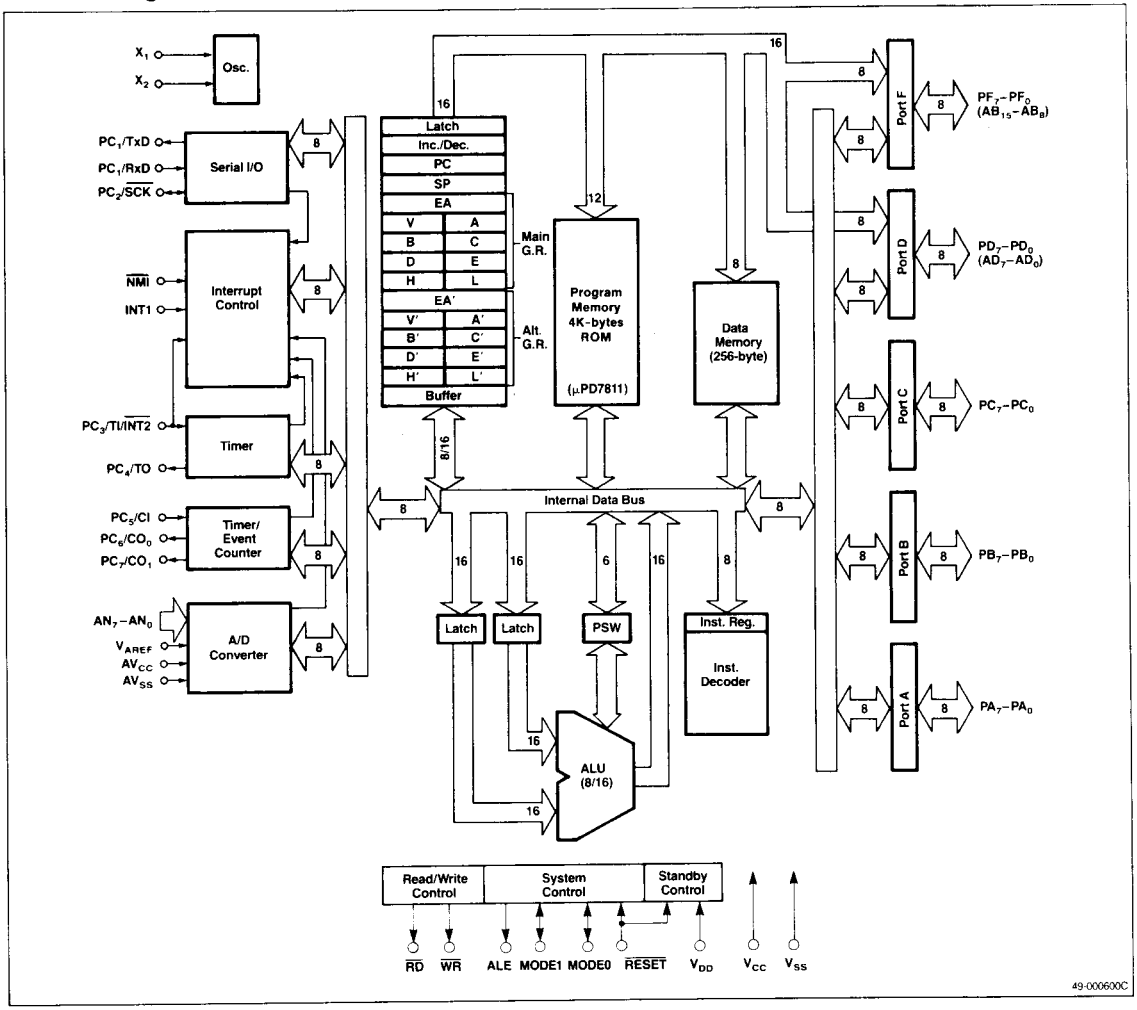
V_{DD} [Backup Power]

Backup power for on-chip RAM.

V_{CC} [Power Supply]

+5 V power supply.

Block Diagram



Functional Description

Memory Map

The μPD7811 can directly address up to 64K bytes of memory. Except for the on-chip ROM (0-4095) and RAM (65280-65535), any memory location can be used as ROM or RAM. The memory map, figure 1, defines the 0 to 64K byte memory space for the μPD7811.

Input/Output

The μPD7810/11 has 8 analog input lines (AN_0-AN_7), 44 digital I/O lines, five 8-bit ports (port A, port B, port C, port D, port F), and 4 input lines (AN_4-AN_7).

Analog Input Lines. AN_0-AN_7 are configured as analog input lines for on-chip A/D converter.

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.

AN_4-AN_7. The high order analog input lines, AN_4-AN_7, can be used as digital input lines for falling edge detection.

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 μPD7811 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	I/O port
	Port F	I/O port
256 Bytes	Port D	Multiplexed address/data bus
	Port F	I/O port
4K Bytes	Port D	Multiplexed address/data bus
	Port F_0-F_3	Address bus
	Port F_4-F_7	I/O port
16K Bytes	Port D	Multiplexed address/data bus
	Port F_0-F_5	Address bus
	Port F_6-F_7	I/O port
60K Bytes	Port D	Multiplexed address/data bus
	Port F	Address bus

Timers

There are two 8-bit timers. The timers may be programmed independently or may be cascaded and used as an 8-bit timer with 8-bit prescaler. 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 TI. Figure 2 shows the block diagram for the timer.

Timer/Event Counter

The 16-bit multifunctional timer/event counter (figure 3) can be used for the following operations:

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

Figure 1. Memory Map

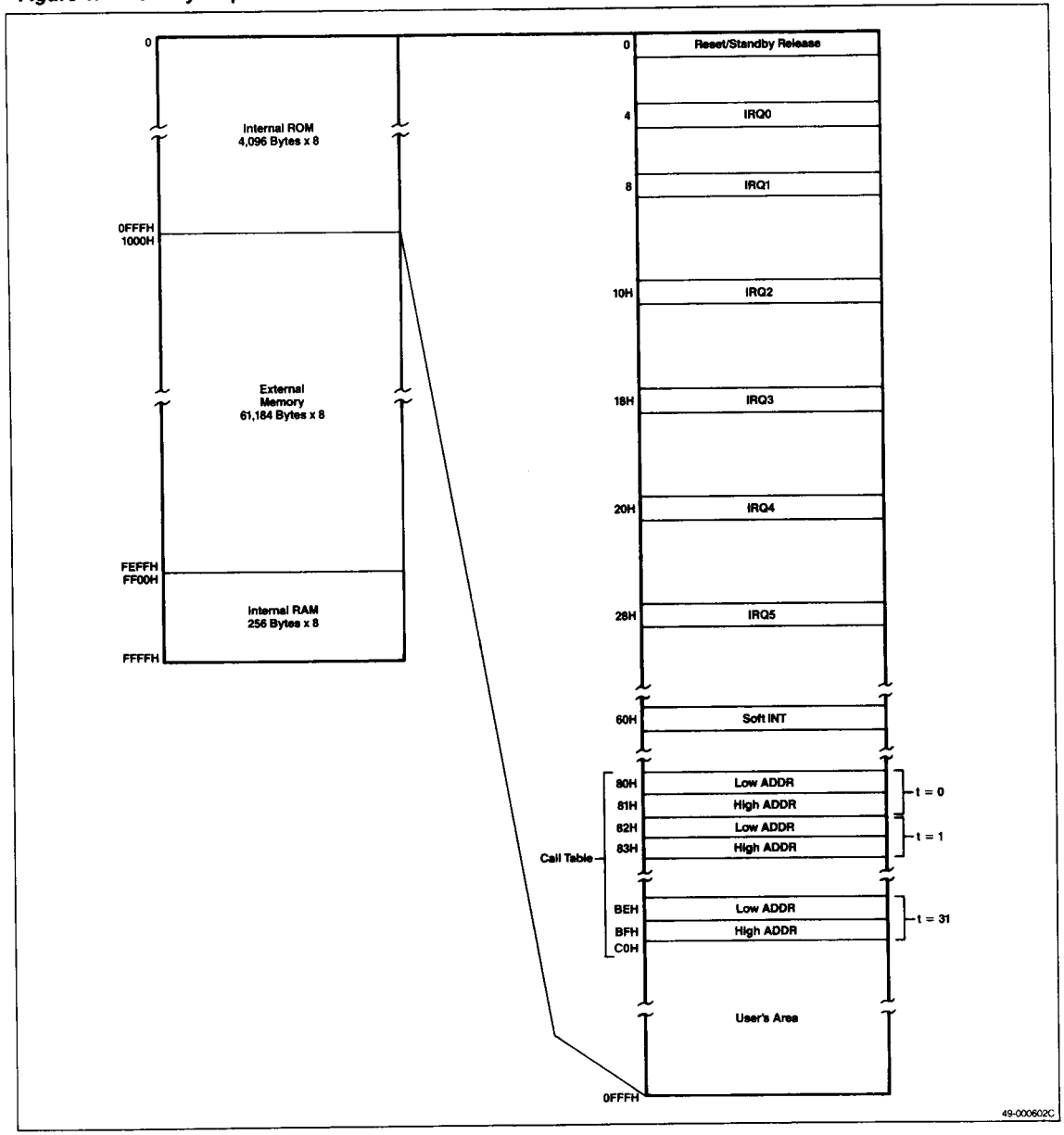
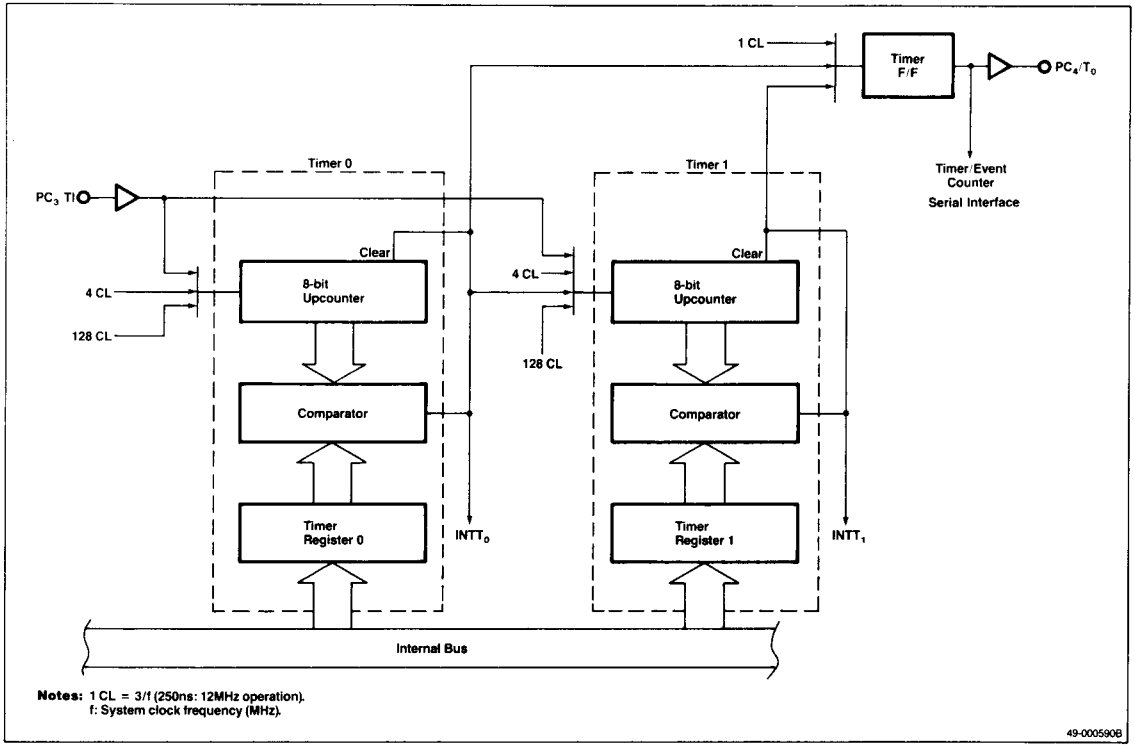


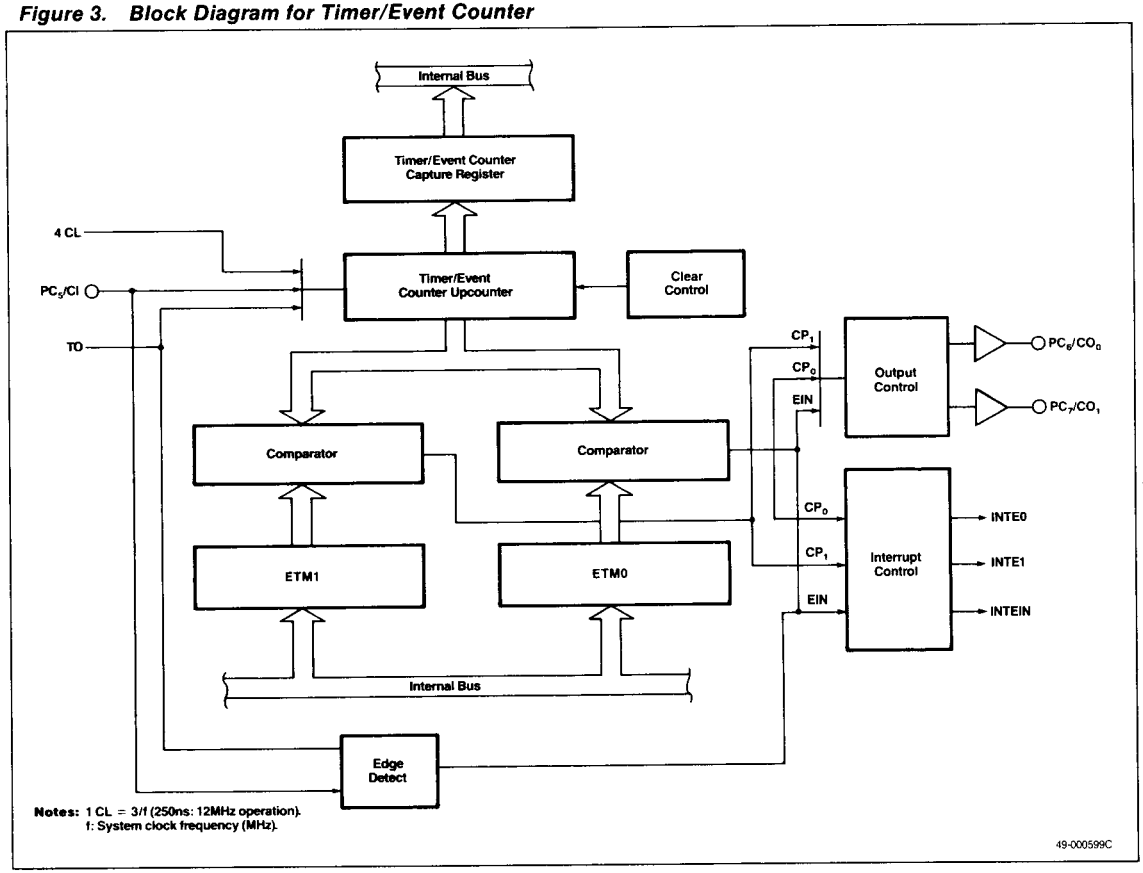
Figure 2. Timer Block Diagram



Notes: 1 CL = 3/f (250ns: 12MHz operation).
f: System clock frequency (MHz).

49-000590B

Figure 3. Block Diagram for Timer/Event Counter



8-Bit A/D Converter

- 8 input channels
- 4 conversion result registers
- 2 powerful operation modes
  - — Autoscan mode
  - — Channel select mode
- Successive approximation technique
- Absolute accuracy: ±1.5 LSB (±0.6%)
- Conversion range: 0 to 5 V
- Conversion time: 48 μs
- Interrupt generation

Analog/Digital Converter

The μPD7810/11 features an 8-bit, high speed, high accuracy A/D converter. The A/D converter is made up of a 256-resistor ladder and a successive approximation register (SAR). There are four conversion result registers (CR_0-CR_3). The 8-channel analog input may be operated in either of two modes. In the select mode, the conversion value of one analog input is sequentially stored in CR_0-CR_3. In the scan mode, the upper four channels or the lower four channels may be specified. Then those four channels will be consecutively selected and the conversion results stored sequentially in the four conversion result registers. Figure 4 shows the block diagram for the A/D converter.

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 5.

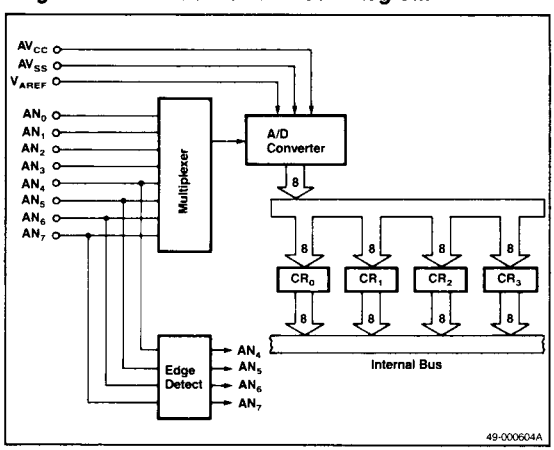
Standby Function

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 (SB) 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	$\overline{NMI}$ (Nonmaskable interrupt)	Ext
IRQ1	8	INTT0 (Coincidence signal from timer 0)	Int
		INTT1 (Coincidence signal from timer 1)	
IRQ2	16	INT1 (Maskable interrupt)	Ext
		$\overline{INT2}$ (Maskable interrupt)	
IRQ3	24	INTE0 (Coincidence signal from timer/event counter)	Int
		INTE1 (Coincidence signal from timer/event counter)	
IRQ4	32	INTEIN (Falling signal of CI and TO counter)	Int/Ext
		INTAD (A/D converter interrupt)	
IRQ5	40	INTSR (Serial receive interrupt)	Int
		INST (Serial send interrupt)	

Figure 4. A/D Converter Block Diagram



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. Figure 6 shows the universal serial interface block diagram.

Figure 5. Interrupt Structure Block Diagram

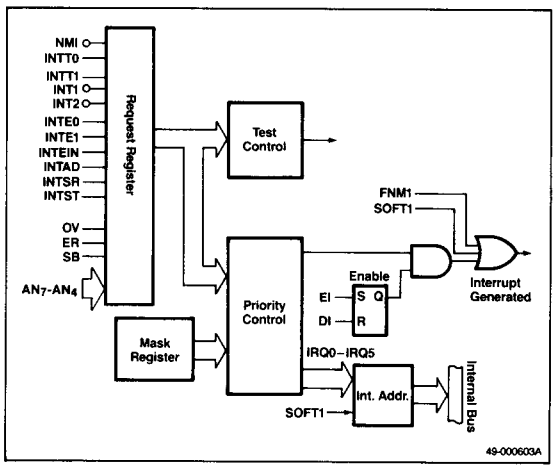
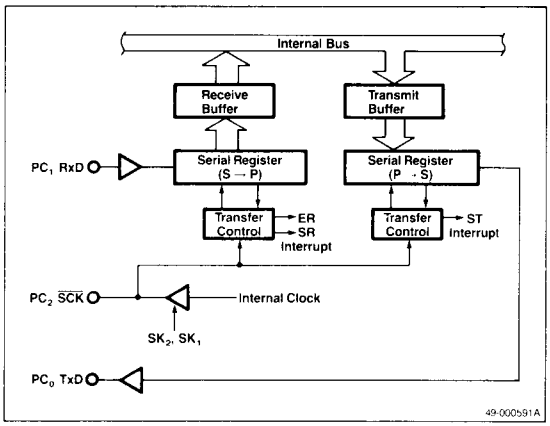


Figure 6. Universal Signal Interface Block Diagram



Zero-Crossing Detector

The INT1 and $\overline{INT2}$ terminals (used common to TI and PC_3) can detect the zero-crossing point of low-frequency AC signals. When driven directly, these pins respond as a normal digital input. Figure 7 shows the zero-crossing detection circuitry.

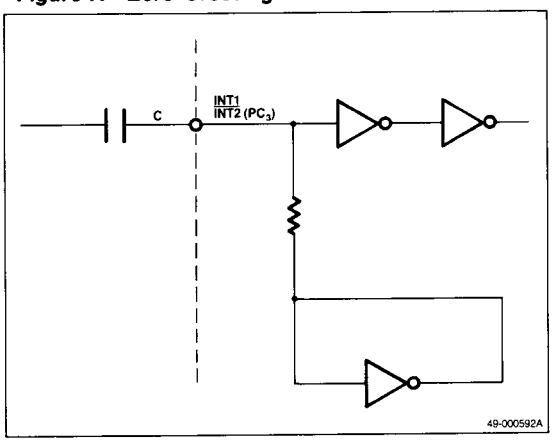
The zero-crossing detection capability allows you to make the 50-60 Hz power signal the basis for system timing and to control voltage phase-sensitive devices.

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

Figure 7. Zero-Crossing Detection Circuit



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
	AV_{SS}	−0.5 V to +0.5 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 voltage, V_{AREF}		−0.5 V to V_{CC}
Operating temperature, T_{OPR} 10 MHz ≤ f_{XTAL} ≤ 12 MHz		−10 °C to +70 °C
f_{XTAL} ≤ 10 MHz		−40 °C to −85 °C
Storage temperature, T_{STG}		−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

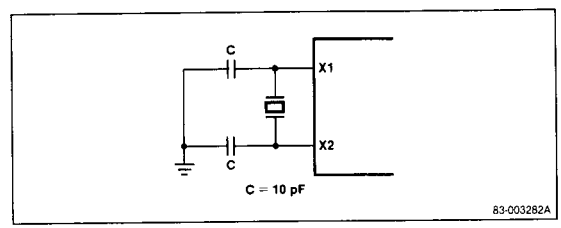
Oscillating Frequency	T_A	V_{CC}, AV_{CC}
f_{XTAL} ≤ 10 MHz	−40 °C to +85 °C	+5.0 V ±10%
10 MHz ≤ f_{XTAL} ≤ 12 MHz	−10 °C to +70 °C	+5.0 V ±5%

Capacitance

T_A = 25 °C; V_{CC} = V_{DD} = V_{SS} = 0 V

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

Recommended XTAL Oscillation Circuit



DC Characteristics

$T_A = -10°C$ to $+70°C$; $V_{CC} = +5.0$ V ±5%; $V_{SS} = 0$ V; $V_{DD} = V_{CC} - 0.8$ V to V_{CC}

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Input low voltage	V_{IL}	0		0.8	V	
Input high voltage	V_{IH1}	2.0		V_{CC}	V	All except $\overline{SCK}$, RESET, X1 and X2
	V_{IH2}	0.8 V_{CC}		V_{CC}	V	$\overline{SCK}$, X1, X2
	V_{IH3}	0.8 V_{DD}		V_{CC}	V	RESET
Output low voltage	V_{OL}			0.45	V	$I_{OL} = 2.0$ mA
Output high voltage	V_{OH}	2.4			V	$I_{OH} = -200$ μA
Data retention voltage	V_{DDDR}	3.2			V	$V_{CC} = 0$ V; $\overline{RESET} = V_{IL}$
Input current	I_I			±200	μA	INT1, TI(PC_3); $0.45\ V \le V_I < V_{CC}$
Input leakage current	I_{LI}			±10	μA	All except INT, TI(PC_3) $0\ V \le V_I \le V_{CC}$
Output leakage current	I_{LO}			±10	μA	$+0.45\ V \le V_O \le V_{CC}$
AV_{CC} supply current	AI_{CC}		6	12	mA	
V_{DD} supply current	I_{DD}		1.5	3.5	mA	$T_A = -40$ to $+85°C$
				3.2	mA	$V_{CC} = V_{DD} = 5$ V $T_A = -10$ to $+70°C$
V_{CC} supply current	I_{CC}		150	220	mA	$T_A = -40$ to $+85°C$; $V_{CC} = V_{DD} = 5$ V

Serial Operation

Parameter	Symbol	Limits Min	Max	Unit	Test Conditions
$\overline{SCK}$ cycle time	t_{CYK}	1		μs	$\overline{SCK}$ input (1)
		500		ns	(2)
		2		μs	$\overline{SCK}$ output
$\overline{SCK}$ width low	t_{KKL}	750		ns	$\overline{SCK}$ input(1)
		200		ns	$\overline{SCK}$ input (2)
		900		ns	$\overline{SCK}$ output
$\overline{SCK}$ width high	t_{KKH}	750		ns	$\overline{SCK}$ input (1)
		200		ns	$\overline{SCK}$ input (2)
		900		ns	$\overline{SCK}$ output
RxD set-up time to SCK ↑	t_{RXK}	80		ns	(1)
RxD hold time after $\overline{SCK}$ ↑	t_{KRX}		80	ns	(1)
$\overline{SCK}$ ↓ TxD delay time	t_{KTX}		210	ns	(1)

Note:

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

(2) 16x baud rate or 64x baud rate in asynchronous mode.

Zero-Cross Characteristics

Parameter	Symbol	Limits Min	Max	Unit	Test Conditions
Zero-cross detection input	V_{ZX}	1	3	V ac, 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	

AC Characteristics

Read/Write Operation

$V_{SS} = 0\ V$, $V_{CC} - 0.8\ V \leq V_{DD} \leq V_{CC}$

Parameter	Symbol	Limits $f_{XTAL} = 10$ MHz Min	Limits $f_{XTAL} = 10$ MHz Max	Limits $f_{XTAL} = 12$ MHz Min	Limits $f_{XTAL} = 12$ MHz Max	Unit	Test Conditions (1)
$\overline{RESET}$ 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	
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 after ALE ↓	t_{LM}	70		50		ns	
$\overline{IO}$/M setup time to ALE ↓	t_{IL}	100		65		ns	
$\overline{IO}$/M hold time after 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: $C_L = 150$ pF.

A/D Converter Characteristics

$T_A = -10°C$ to $+70°C$; $V_{CC} = AV_{CC} = 5.0\ V \pm 5\%$; $V_{SS} = AV_{SS} = 0\ V$; $V_{AREF} = AV_{CC} - 0.5\ V$ to AV_{CC}.

Parameter	Symbol	Limits Min	Typ	Max	Unit	Test Conditions
Resolution		8			Bits	
Absolute accuracy				0.4% ± 1/2	LSB	$T_A = -10°C$ to $+50°C$
				0.6% ±1/2	LSB	$T_A = -10°C$ to $+70°C$ (Note 1)
Conversion time	t_{CONV}	576			t_{CYC}	83 ns ≤ t_{CYC} ≤ 110 ns
		432			t_{CYC}	110 ns ≤ t_{CYC} ≤ 170 ns
Sampling time	t_{SAMP}	96			t_{CYC}	83 ns ≤ t_{CYC} ≤ 110 ns
		72			t_{CYC}	110 ns ≤ t_{CYC} ≤ 170 ns
Analog input voltage	V_{IA}	0		V_{AREF}	V	
Analog resistance	R_{AN}		1000		MΩ	
Analog reference current	I_{AREF}	0.2	0.5	1.5	mA	

Note:

(1) In case of $f_{XTAL} \leq 10$ MHz, $T_A = -40°C$ to $+85°C$.

Bus Timing Depending on t_{CYC}

Symbol	Calculating Expression	Min/Max
t_{RP}	60T	Min
t_{TI}	6T	Min
t_{CI}(2)	6T	Min
t_{CI}(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	Max
t_{LDR}	5T − 200	Max
t_{RD}	4T − 150	Max
t_{LR}	T − 50	Min
t_{RL}	2T − 50	Min
t_{RR}	4T − 50 (Data Read)	Min
	7T − 50 (Opcode Fetch)	
t_{LL}	2T − 40	Min
t_{AW}	3T −100	Min
t_{LDW}	T + 110	Max
t_{LW}	T − 50	Min
t_{DW}	4T − 100	Min
t_{WDH}	2T − 70	Min
t_{WL}	2T − 50	Min
t_{WW}	4T − 50	Min
t_{CYK}	20T ($\overline{SCK}$ input)(1)	Min
	24T ($\overline{SCK}$ output)	
t_{KKL}	10T − 80 ($\overline{SCK}$ input)(1)	Min
	12T − 100 ($\overline{SCK}$ output)	
t_{KKH}	10T − 80 ($\overline{SCK}$ input)(1)	Min
	12T − 100 ($\overline{SCK}$ output)	

Note:

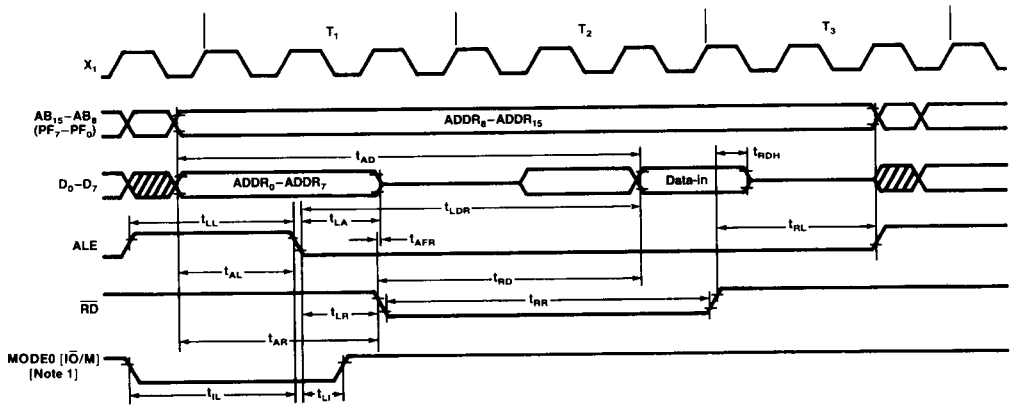
(1) 1x Baud rate in asynchronous, synchronous, or I/O interface mode.
$T = t_{CYC} = 1/f_{XTAL}$.
The items not included in this list are independent of oscillator frequency (f_{XTAL}).

(2) Event counter mode.

(3) Pulse width measurement mode.

Timing Waveforms

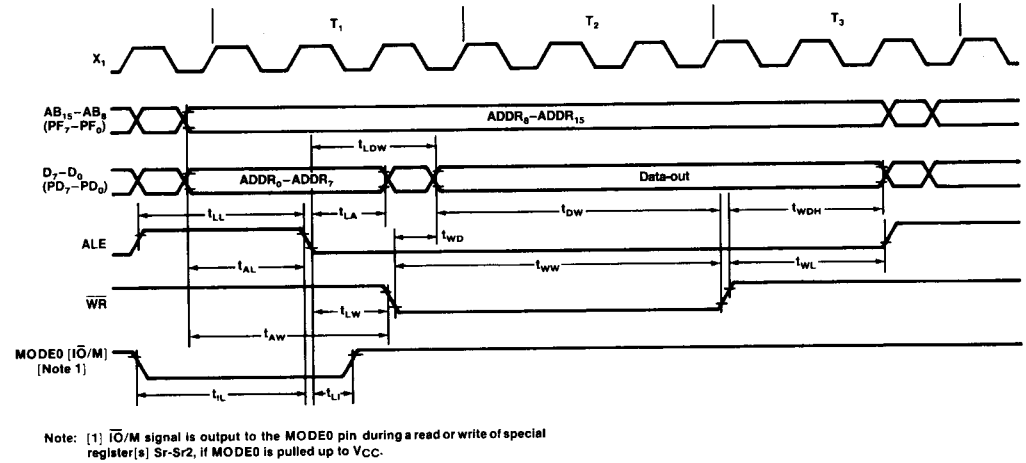
Read Operation



Note: [1] $\overline{IO}/M$ signal is output to the MODE0 pin during a read or write of special register[s] Sr-Sr2, if MODE0 is pulled up to V_{CC}.

49-000543B

Write Operation

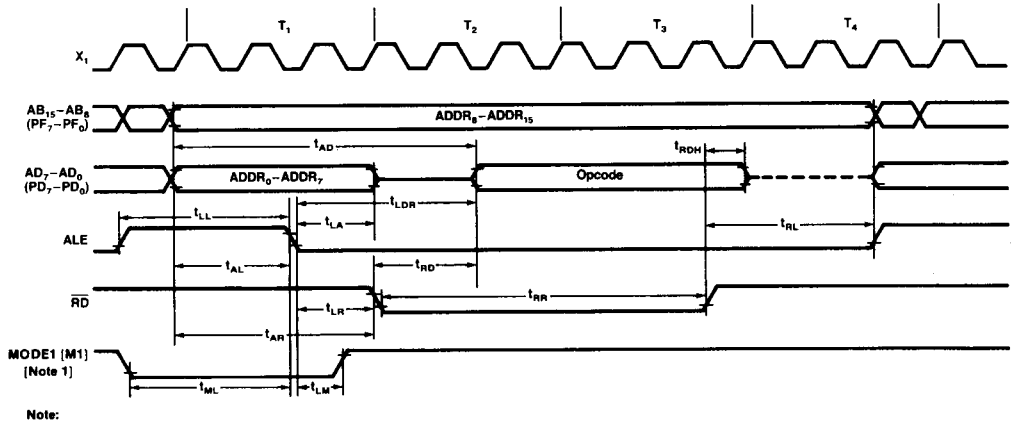


Note: [1] $\overline{IO}/M$ signal is output to the MODE0 pin during a read or write of special register[s] Sr-Sr2, if MODE0 is pulled up to V_{CC}.

49-000639B

Timing Waveforms (cont)

Opcode Fetch Operation

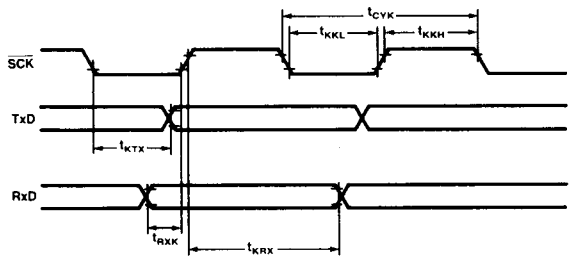


Note:

[1] M1 signal is output to the MODE1 pin during Opcode Fetch if MODE1 pin is pulled up to V_{CC}.

49-000545B

Serial Operation Transmit/Receive Timing



49-000546B

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, ANM, SMH, SML, EOM, ETMM, TMM, MM, MCC, MA, MB, MC, MF, TxB, TM_0, TM_1
sr1	PA, PB, PC, PD, PF, MKH, MKL, ANM, SMH, EOM, TMM, RXB, CR0, CR1, CR2, CR3
sr2	PA, PB, PC, PD, PF, MKH, ANM, MKL, SMH, EOM, TMM
sr3	ETM_0, ETM_1
sr4	ECNT, ECPT
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	3-Bit immediate data
f	CY, HC, Z
irf	FNMI, FT0, FT1, F1, F2, FE0, FE1, FEIN, FAD, FSR, FST, ER, OV, AN_4, AN_5, AN_6, AN_7, 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-sr4 (special register)

PA = Port A
PB = Port B
PC = Port C
PD = Port D
PF = Port F
MA = Mode A
MB = Mode B
MC = Mode C
MCC = Mode Control C
MF = Mode F

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

ECNT = Timer/Event Counter Upcounter
ECPT = Timer/Event Counter Capture

ETMM = Timer/Event Counter Mode
EOM = Timer/Event Counter Output Mode

TxB = TX Buffer
RxB = RX Buffer
SMH = Serial Mode High
SML = Serial Mode Low
MKH = Mask High
MKL = Mask Low
ANM = A/D Channel Mode
CR_0 = A/D Conversion Result 0-3 to CR_3

2. rp-rp3 (register pair)

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

3. rpa-rpa3 (rp addressing)

B = (BC)
D = (DE)
H = (HL)
D + = (DE) +
H − = (HL) +
D − = (DE) −
H − = (HL) −
D + + = (DE) ++
H + + = (HL) ++
D + byte = (DE) + byte
H + A = (HL) + (A)
H + B = (HL) + (B)
H + EA = (HL) + (EA)
H + byte = (HL) + byte

4. f (flag)

CY = Carry
HC = Half Carry
Z = Zero

5. irf (interrupt flag)

NMI = NMI* Input

FT0 = INTFT0
FT1 = INTFT1
F1 = INTF1
F2 = INTF2
FE0 = INTFE0
FE1 = INTFE1

FEIN = INTFEIN
FAD = INTFAD
FSR = INTFSR
FST = INTFST
ER = Error
OV = Overflow
AN_4 to AN_7 = Analog Input 4-7
SB = Standby

Instruction Set

Mnemonic	Operand	Operation	Operation Code B1 7	6	5	4	B3 3	2	1	0	B2 7	6	5	B4 4	3	2	1	0	State(1)	Bytes	Skip Condition
8-Bit Data Transfer																					
MOV	r1,A	(r1) ← (A)	0	0	0	1	1	T_2	T_1	T_0									4	1	
	A, r1	(A) ← (r1)	0	0	0	0	1	T_2	T_1	T_0									4	1	
	*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	2	
	*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	2	
	r,word	(r) ← (word)	0	1	1	1	0	0	0	0	0	1	1	0	1	R_2	R_1	R_0	17	4	
			Low addr								High addr										
	word,r	(word) ← (r)	0	1	1	1	0	0	0	0	0	1	1	1	1	R_2	R_1	R_0	17	4	
			Low addr								High addr										
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	Data								7	2	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	0	S_2	S_1	S_0	14	3	
			Data																		
MVIW	*wa, byte	((V)•(wa)) ← byte	0	1	1	1	0	0	0	1	Offset								13	3	
			Data																		
MVIX	*rpa1,byte	(rpa1) ← byte	0	1	0	0	1	0	A_1	A_0	Data								10	2	
STAW	*wa	((V)•(wa)) ← A	0	1	1	0	0	0	1	1	Offset								10	2	
LDAW	*wa	(A) ← ((V)•(wa))	0	0	0	0	0	0	0	1	Offset								10	2	
STAX	*rpa2	(rpa2) ← (A)	A_3	0	1	1	1	A_2	A_1	A_0	Data (2)								7/13(3)	2	
LDAX	*rpa2	(A) ← ((rpa2))	A_3	0	1	0	1	A_2	A_1	A_0	Data (2)								7/13(3)	2	
EXX		(B) ↔ (B'),(C) ↔ (C'),(D) ↔ (D') (E) ↔ (E'),(H) ↔ (H'),(L) ↔ (L')	0	0	0	1	0	0	0	1									4	1	
EXA		(V) ↔ (V'),(A) ↔ (A'),(EA) ↔ (EA')	0	0	0	1	0	0	0	0									4	1	
EXH		(H) ↔ (H'),(L) ↔ (L')	0	1	0	1	0	0	0	0									4	1	
16-Bit Data Transfer																					
BLOCK	D	((DE)) ← ((HL)),(DE) ← (DE + 1), (HL) ← (HL) + 1, (C) ← (C) − 1 End if borrow	0	0	1	1	0	0	0	1									13 x (C + 1)	1	
DMOV	rp3, EA	($rp3_L$) ← (EAL),($rp3_H$) ← (EAH)	1	0	1	1	0	1	P_1	P_0									4	1	
	EA,rp3	(EAL) ← ($rp3_L$),(EAH) ← ($rp3_H$)	1	0	1	0	0	1	P_1	P_0									4	1	

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code B1 B3								Operation Code B2 B4								State(1)	Bytes	Skip Condition
			7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0			
16-Bit Data Transfer (cont)																					
DMOV	sr3, EA	(sr3) ← (EA)	0	1	0	0	1	0	0	0	1	1	0	1	0	0	1	U_0	14	2	
	EA,sr4	(EA) ← (sr4)	0	1	0	0	1	0	0	0	1	1	0	0	0	0	V_1	V_0	14	2	
SBCD	word	(word) ← (C), (word + 1) ← (B)	0	1	1	1	0	0	0	0	0	0	0	1	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	1	0	1	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	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	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										
LDED	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	1	0	0	0	C_3	C_2	C_1	C_0	14/20(3)	3	
			Data(4)																		
PUSH	rp1	((SP) − 1) ← ($rp1_H$) ((SP) − 2) ← ($rp1_L$) (SP) ← (SP) − 2	1	0	1	1	0	Q_2	Q_1	Q_0									13	1	
POP	rp1	($rp1_L$) ← ((SP)),($rp1_H$) ← ((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 if rp2 = H	0	P_2	P_1	P_0	0	1	0	0	Low byte								10	3	L0 = 1 and rp2 = H
			High byte																		
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	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	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 B1 / B3								Operation Code B2 / B4								State(1)	Bytes	Skip Condition
			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	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	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	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	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	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	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	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	0	0	1	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	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	0	1	R_2	R_1	R_0	8	2	
ORA	A,r	(A) ← (A) ∨ (r)	0	1	1	0	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	0	1	1	R_2	R_1	R_0	8	2	
XRA	A,r	(A) ← (A) ∨̄ (r)	0	1	1	0	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	0	1	0	R_2	R_1	R_0	8	2	
GTA	A,r	(A) − (r) − 1	0	1	1	0	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	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	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	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	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	0	1	1	0	1	R_2	R_1	R_0	8	2	No zero
EQA	A,r	(A) − (r)	0	1	1	0	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	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	0	1	1	0	0	1	R_2	R_1	R_0	8	2	No zero
OFFA	A,r	(A) ∧ (r)	0	1	1	0	0	0	0	0	1	1	0	1	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	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	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	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	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	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	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	0	1	0	0	0	1	A_2	A_1	A_0	11	2	
ORAX	rpa	(A) ← (A) ∨ ((rpa))	0	1	1	1	0	0	0	0	1	0	0	1	1	A_2	A_1	A_0	11	2	

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code	State(1)	Bytes	Skip Condition
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 0 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 0 1 1 0 / Data	7	2	
	r,byte	(r) ← (r) + byte	0 1 1 1 0 1 0 0 / Data / 0 1 0 0 0 R_2 R_1 R_0	11	3	
	sr2, byte	(sr2) ← (sr2) + byte	0 1 1 0 0 1 0 0 / Data / S_3 1 0 0 0 S_2 S_1 S_0	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 / 0 1 0 1 0 R_2 R_1 R_0	11	3	
	sr2,byte	(sr2) ← (sr2) + byte + (CY)	0 1 1 0 0 1 0 0 / Data / S_3 1 0 1 0 S_2 S_1 S_0	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 / 0 0 1 0 0 R_2 R_1 R_0	11	3	No carry
	sr2,byte	(sr2) ← (sr2) + byte	0 1 1 0 0 1 0 0 / Data / S_3 0 1 0 0 S_2 S_1 S_0	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 / 0 1 1 0 0 R_2 R_1 R_0	11	3	
	sr2,byte	(sr2) ← (sr2) – byte	0 1 1 0 0 1 0 0 / Data / S_3 1 1 0 0 S_2 S_1 S_0	20	3	
SBI	*A,byte	(A) ← (A) – byte – (CY)	0 1 1 1 0 1 1 0 / Data	7	2	
	r,byte	(r) ← (r) – byte – (CY)	0 1 1 1 0 1 0 0 / Data / 0 1 1 1 0 R_2 R_1 R_0	11	3	
	sr2,byte	(sr2) ← (sr2) – byte – (CY)	0 1 1 0 0 1 0 0 / Data / S_3 1 1 1 0 S_2 S_1 S_0	20	3	

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code B1 7	6	5	4	B3 3	2	1	0	B2 7	6	5	4	B4 3	2	1	0	State(1)	Bytes	Skip Condition
Immediate Data (cont)																					
SUINB	*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	0	0	1	1	0	R_2	R_1	R_0	11	3	No borrow
			Data																		
	sr2,byte	(sr2) ← (sr2) − byte	0	1	1	0	0	1	0	0	S_3	0	1	1	0	S_2	S_1	S_0	20	3	No borrow
			Data																		
ANI	*A,byte	(A) ← (A) ∧ byte	0	0	0	0	0	1	1	1	Data								7	2	
	r,byte	(r) ← (r) ∧ byte	0	1	1	1	0	1	0	0	0	0	0	0	1	R_2	R_1	R_0	11	3	
			Data																		
	sr2,byte	(sr2) ← (sr2) ∧ byte	0	1	1	0	0	1	0	0	S_3	0	0	0	1	S_2	S_1	S_0	20	3	
			Data																		
ORI	*A,byte	(A) ← (A) V byte	0	0	0	1	0	1	1	1	Data								7	2	
	r,byte	(r) ← (r) V byte	0	1	1	1	0	1	0	0	0	0	0	1	1	R_2	R_1	R_0	11	3	
			Data																		
	sr2,byte	(sr2) ← (sr2) V byte	0	1	1	0	0	1	0	0	S_3	0	0	1	1	S_2	S_1	S_0	20	3	
			Data																		
XRI	*A,byte	(A) ← (A) ∀ byte	0	0	0	1	0	1	1	0	Data								7	2	
	r,byte	(r) ← (r) ∀ byte	0	1	1	1	0	1	0	0	0	0	0	1	0	R_2	R_1	R_0	11	3	
			Data																		
	sr2,byte	(sr2) ← (sr2) V byte	0	1	1	0	0	1	0	0	S_3	0	0	1	0	S_2	S_1	S_0	20	3	
			Data																		
GTI	*A,byte	(A) − byte − 1	0	0	1	0	0	1	1	1	Data								7	2	No borrow
	r,byte	(r) − byte − 1	0	1	1	1	0	1	0	0	0	0	1	0	1	R_2	R_1	R_0	11	3	No borrow
			Data																		
	sr2,byte	(sr2) − byte − 1	0	1	1	0	0	1	0	0	S_3	0	1	0	1	S_2	S_1	S_0	14	3	No borrow
			Data																		
LTI	*A,byte	(A) − byte	0	0	1	1	0	1	1	1	Data								7	2	Borrow
	r,byte	(r) − byte	0	1	1	1	0	1	0	0	0	0	1	1	1	R_2	R_1	R_0	11	3	Borrow
			Data																		
	sr2,byte	(sr2) − byte	0	1	1	0	0	1	0	0	S_3	0	1	1	1	S_2	S_1	S_0	14	3	Borrow
			Data																		
NEI	*A,byte	(A) − byte	0	1	1	0	0	1	1	1	Data								7	2	No zero
	r,byte	(r) − byte	0	1	1	1	0	1	0	0	0	1	1	0	1	R_2	R_1	R_0	11	3	No zero
			Data																		

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code B1 / B3 7 6 5 4 3 2 1 0	Operation Code B2 / B4 7 6 5 4 3 2 1 0	State[1]	Bytes	Skip Condition
Immediate Data (cont)							
NEI	sr2,byte	(sr2) − byte	0 1 1 0 0 1 0 0 Data	S_3 1 1 0 1 S_2 S_1 S_0	14	3	No zero
EQI	*A,byte	(A) − byte	0 1 1 1 0 1 1 1	Data	7	2	Zero
	r,byte	(r) − byte	0 1 1 1 0 1 0 0 Data	0 1 1 1 1 R_2 R_1 R_0	11	3	Zero
	sr2,byte	(sr2) − byte	0 1 1 0 0 1 0 0 Data	S_3 1 1 1 1 S_2 S_1 S_0	14	3	Zero
ONI	*A,byte	(A) ∧ byte	0 1 0 0 0 1 1 1	Data	7	2	No zero
	r,byte	(r) ∧ byte	0 1 1 1 0 1 0 0 Data	0 1 0 0 1 R_2 R_1 R_0	11	3	No zero
	sr2,byte	(sr2) ∧ byte	0 1 1 0 0 1 0 0 Data	S_3 1 0 0 1 S_2 S_1 S_0	14	3	No zero
OFFI	*A,byte	(A) ∧ byte	0 1 0 1 0 1 1 1	Data	7	2	Zero
	r,byte	(r) ∧ byte	0 1 1 1 0 1 0 0 Data	0 1 0 1 1 R_2 R_1 R_0	11	3	Zero
	sr2,byte	(sr2) ∧ byte	0 1 1 0 0 1 0 0 Data	S_3 1 0 1 1 S_2 S_1 S_0	14	3	Zero
Working Register							
ADDW	wa	(A) ← (A) + ((V)•(wa))	0 1 1 1 0 1 0 0 Offset	1 1 0 0 0 0 0 0	14	3	
ADCW	wa	(A) ← (A) + ((V)•(wa)) + (CY)	0 1 1 1 0 1 0 0 Offset	1 1 0 1 0 0 0 0	14	3	
ADDNCW	wa	(A) ← (A) + ((V)•(wa))	0 1 1 1 0 1 0 0 Offset	1 0 1 0 0 0 0 0	14	3	No carry
SUBW	wa	(A) ← (A) − ((V)•(wa))	0 1 1 1 0 1 0 0 Offset	1 1 1 0 0 0 0 0	14	3	
SBBW	wa	(A) ← (A) − ((V)•(wa)) − (CY)	0 1 1 1 0 1 0 0 Offset	1 1 1 1 0 0 0 0	14	3	
SUBNBW	wa	(A) ← (A) − ((V)•(wa))	0 1 1 1 0 1 0 0 Offset	1 0 1 1 0 0 0 0	14	3	No borrow
ANAW	wa	(A) ← (A) ∧ ((V)•(wa))	0 1 1 1 0 1 0 0 Offset	1 0 0 0 1 0 0 0	14	3	

Instruction Set (cont)

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

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code B1/B3 7	6	5	4	3	2	1	0	Operation Code B2/B4 7	6	5	4	3	2	1	0	State(1)	Bytes	Skip Condition
16-Bit Arithmetic																					
EADD	EA,r2	(EA) ← (EA) + (r2)	0	1	1	1	0	0	0	0	0	1	0	0	0	0	R_1	R_0	11	2	
DADD	EA,rp3	(EA) ← (EA) + (rp3)	0	1	1	1	0	1	0	0	1	1	0	0	0	1	P_1	P_0	11	2	
DADC	EA,rp3	(EA) ← (EA) + (rp3) + (CY)	0	1	1	1	0	1	0	0	1	1	0	1	0	1	P_1	P_0	11	2	
DADDNC	EA,rp3	(EA) ← (EA) + (rp3)	0	1	1	1	0	1	0	0	1	0	1	0	0	1	P_1	P_0	11	2	No carry
ESUB	EA,r2	(EA) ← (EA) − (r2)	0	1	1	1	0	0	0	0	0	1	1	0	0	0	R_1	R_0	11	2	
DSUB	EA,rp3	(EA) ← (EA) − (rp3)	0	1	1	1	0	1	0	0	1	1	1	0	0	1	P_1	P_0	11	2	
DSBB	EA,rp3	(EA) ← (EA) − (rp3) − (CY)	0	1	1	1	0	1	0	0	1	1	1	1	0	1	P_1	P_0	11	2	
DSUBNB	EA,rp3	(EA) ← (EA) − (rp3)	0	1	1	1	0	1	0	0	1	0	1	1	0	1	P_1	P_0	11	2	No borrow
DAN	EA,rp3	(EA) ← (EA) ∧ (rp3)	0	1	1	1	0	1	0	0	1	0	0	0	1	1	P_1	P_0	11	2	
DOR	EA,rp3	(EA) ← (EA) ∨ (rp3)	0	1	1	1	0	1	0	0	1	0	0	1	1	1	P_1	P_0	11	2	
DXR	EA,rp3	(EA) ← (EA) ∀ (rp3)	0	1	1	1	0	1	0	0	1	0	0	1	0	1	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	P_1	P_0	11	2	No borrow
DLT	EA,rp3	(EA) − (rp3)	0	1	1	1	0	1	0	0	1	0	1	1	1	1	P_1	P_0	11	2	Borrow
DNE	EA,rp3	(EA) − (rp3)	0	1	1	1	0	1	0	0	1	1	1	0	1	1	P_1	P_0	11	2	No zero
DEQ	EA,rp3	(EA) − (rp3)	0	1	1	1	0	1	0	0	1	1	1	1	1	1	P_1	P_0	11	2	Zero
DON	EA,rp3	(EA) ∧ (rp3)	0	1	1	1	0	1	0	0	1	1	0	0	1	1	P_1	P_0	11	2	No zero
DOFF	EA,rp3	(EA) ∧ (rp3)	0	1	1	1	0	1	0	0	1	1	0	1	1	1	P_1	P_0	11	2	Zero
Multiply/Divide																					
MUL	r2	(EA) ← (A) x (r2)	0	1	0	0	1	0	0	0	0	0	1	0	1	1	R_1	R_0	32	2	
DIV	r2	(EA) ← (EA) + (r2), (r2) ← Remainder	0	1	0	0	1	0	0	0	0	0	1	1	1	1	R_1	R_0	59	2	
Increment/Decrement																					
INR	r2	(r2) ← (r2) + 1	0	1	0	0	0	0	R_1	R_0									4	1	Carry
INRW	*wa	((V)•(wa)) ← ((V)•(wa)) + 1	0	0	1	0	0	0	0	0	Offset								16	2	Carry
INX	rp	(rp) ← (rp) + 1	0	0	P_1	P_0	0	0	1	0									7	1	
	EA	(EA) ← (EA) + 1	1	0	1	0	1	0	0	0									7	1	
DCR	r2	(r2) ← (r2) − 1	0	1	0	1	0	0	R_1	R_0									4	1	Borrow
DCRW	*wa	((V)•(wa)) ← ((V)•(wa)) − 1	0	0	1	1	0	0	0	0	Offset								16	2	Borrow
DCX	rp	(rp) ← (rp) − 1	0	0	P_1	P_0	0	0	1	1									7	1	
	EA	(EA) ← (EA) − 1	1	0	1	0	1	0	0	1									7	1	
Others																					
DAA		Decimal Adjust Accumulator	0	1	1	0	0	0	0	1									4	1	
STC		(CY) ← 1	0	1	0	0	1	0	0	0	0	0	1	0	1	0	1	1	8	2	
CLC		(CY) ← 0	0	1	0	0	1	0	0	0	0	0	1	0	1	0	1	0	8	2	

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code B1 / B3								Operation Code B2 / B4								State(1)	Bytes	Skip Condition
			7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0			
Others (cont)																					
NEGA		(A) ← $(\overline{A})$ + 1	0	1	0	0	1	0	0	0	0	0	1	1	1	0	1	0	8	2	
Rotate and Shift																					
RLD		Rotate left digit	0	1	0	0	1	0	0	0	0	0	1	1	1	0	0	0	17	2	
RRD		Rotate right digit	0	1	0	0	1	0	0	0	0	0	1	1	1	0	0	1	17	2	
RLL	r2	$(r2_{m+1})$ ← $(r2_m)$, $(r2_0)$ ← (CY), (CY) ← $(r2_7)$	0	1	0	0	1	0	0	0	0	0	1	1	0	1	R_1	R_0	8	2	
RLR	r2	$(r2_{m-1})$ ← $(r2_m)$,$(r2_7)$ ← (CY), (CY) ← $(r2_0)$	0	1	0	0	1	0	0	0	0	0	1	1	0	0	R_1	R_0	8	2	
SLL	r2	$(r2_{m+1})$ ← $(r2_m)$,$(r2_0)$ ← 0, (CY) ← $(r2_7)$	0	1	0	0	1	0	0	0	0	0	1	0	0	1	R_1	R_0	8	2	
SLR	r2	$(r2_{m-1})$ ← $(r2_m)$,$(r2_7)$ ← 0, (CY) ← $(r2_0)$	0	1	0	0	1	0	0	0	0	0	1	0	0	0	R_1	R_0	8	2	
SLLC	r2	$(r2_{m+1})$ ← $(r2_m)$,$(r2_0)$ ← 0, (CY) ← $(r2_7)$	0	1	0	0	1	0	0	0	0	0	0	0	0	1	R_1	R_0	8	2	Carry
SLRC	r2	$(r2_{m-1})$ ← $(r2_m)$,$(r2_7)$ ← 0, (CY) ← $(r2_0)$	0	1	0	0	1	0	0	0	0	0	0	0	0	0	R_1	R_0	8	2	Carry
DRLL	EA	(EA_{n+1}) ← (EA_n),(EA_0) ← (CY), (CY) ← (EA_{15})	0	1	0	0	1	0	0	0	1	0	1	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	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	
DSLR	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	0	8	2	
Jump																					
JMP	*word	(PC) ← word	0	1	0	1	0	1	0	0	Low addr								10	3	
			High addr																		
JB		(PC_H) ← (B),(PC_L) ← (C)	0	0	1	0	0	0	0	1									4	1	
JR	word	(PC) ← (PC) + 1 + jdisp 1	1	1	←		jdisp1			→									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	Low addr								16	3	
			High addr																		
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	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_{10-0}) ← fa, (SP) ← (SP) − 2	0	1	1	1	1	←			fa							→	13	2	

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code B1/B3 7	6	5	4	3	2	1	0	Operation Code B2/B4 7	6	5	4	3	2	1	0	State(1)	Bytes	Skip Condition
Call (cont)																					
CALT	word	$((SP)-1) \leftarrow ((PC)+1)_H$, $((SP)-2) \leftarrow ((PC)+1)_L$, $(PC_L) \leftarrow (128+2ta)$, $(PC_H) \leftarrow (129+2ta)$, $(SP) \leftarrow (SP)-2$	1	0	0	←	ta			→									16	1	
SOFTI		$((SP)-1) \leftarrow (PSW)$, $((SP)-2) \leftarrow ((PC)+1)_H$, $((SP)-3) \leftarrow ((PC)+1)_L$, $(PC) \leftarrow 0060H$, $(SP) \leftarrow (SP)-3$	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		
Skip																					
Bit		bit, wa	0	1	0	1	1	B_2	B_1	B_0	Offset								10	2	Bit Test
CPU Control																					
SK	f	Skip if f = 1	0	1	0	0	1	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	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	I_4	I_3	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	I_4	I_3	I_2	I_1	I_0	8	2	irf = 0
NOP		No operation	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	1	0	1	1	11	2	

Notes:

(1) In the case of skip condition, the idle states are as follows:
- 1-byte instruction: 4 states
- 2-byte instruction: 8 states
- 3-byte instruction: 11 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