

Description

The μ PD8041AH and μ PD8741A are programmable peripheral interface controllers intended for use in master/slave configurations with 8048, 8080A, 8085A, 8086, and other 8- and 16-bit microprocessors. The μ PD8041AH/8741A functions as a totally self-sufficient controller with its own program and data memory to effectively unburden the master CPU from I/O handling and peripheral control functions.

The bus structure and data and status registers of the μ PD8041AH/8741A allow easy interface to the master processor bus. This enables the processor to perform control tasks which offload main system processing and more efficiently distribute processing functions.

The μ PD8041AH/8741A contains an 8-bit CPU, $1K \times 8$ program memory, 64×8 data memory, 18 I/O lines, a counter/timer, and a clock generator. The program memory for the μ PD8041AH is factory mask-programmed, while program memory for the μ PD8741A is UV EPROM for more flexibility.

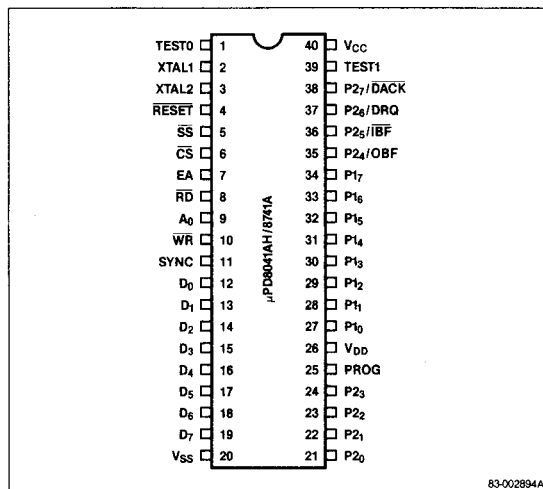
Features

- Complete single chip microcomputer
 - 8-bit CPU
 - $1K \times 8$ ROM
 - 64×8 RAM
 - 8-bit timer/counter
 - 18 I/O lines
- 8048-, 8080A-, 8085A-, 8086-compatible bus structure
- Asynchronous slave-to-master interface
 - 8-bit status register
 - Two data registers
- Interrupt, DMA, or polled operation
- Expandable I/O
- Single +5 V power supply

Ordering Information

Part Number	Package Type	Max Frequency of Operation
μ PD8041AHC	40-pin plastic DIP	11 MHz
μ PD8741AD	40-pin cerdip with quartz window	6 MHz

Pin Configuration



Pin Identification

No.	Symbol	Function
1	T0	Testable input 0
2	XTAL1	Crystal input 1
3	XTAL2	Crystal input 2
4	RESET	Reset input
5	SS	Single step input
6	CS	Chip select input
7	EA	External access input
8	RD	Read strobe input
9	A ₀	Address input 0
10	WR	Write strobe output
11	SYNC	SYNC output
12-19	D ₀ -D ₇	Bidirectional data bus
20	V _{SS}	Ground potential
21-24, 35-38	P ₂₀ -P ₂₇	Quasi-bidirectional Port 2
25	PROG	Program pulse output
26	V _{DD}	Programming supply voltage
27-34	P ₁₀ -P ₁₇	Quasi-bidirectional Port 1
39	T1	Testable input 1
40	V _{CC}	Primary power supply

Pin Functions**XTAL1 (Crystal 1)**

XTAL1 is one side of the crystal or external oscillator or external frequency source.

XTAL2 (Crystal 2)

XTAL2 is the other side of the crystal or frequency source.

T0 (Test 0)

T0 is the testable input using conditional transfer functions JT0, and JNT0. T0 can also be used during programming as a testable flag.

T1 (Test 1)

T1 is the testable input using conditional transfer functions JT1 and JNT1. T1 can be made the counter/timer input using the STRT CNT instruction.

 $\overline{\text{RESET}}$ (Reset)

An active low on $\overline{\text{RESET}}$ initializes the processor. $\overline{\text{RESET}}$ is also used for PROM programming, verification, and power-down.

 $\overline{\text{SS}}$ (Single Step)

An active low on $\overline{\text{SS}}$, together with the SYNC output, allows the processor to single step through each instruction in program memory.

EA (External Access)

An active high on EA disables internal program memory and fetches and accesses external program memory.

 $\overline{\text{RD}}$ (Read)

$\overline{\text{RD}}$ will pulse low when the processor reads data and status words from the data bus buffer or status register.

 $\overline{\text{WR}}$ (Write)

$\overline{\text{WR}}$ will pulse low when the processor writes data or status words to the data bus buffer or status register.

D₀-D₇ (Data Bus)

D₀-D₇ is a three-state, bidirectional data bus. D₀-D₇ interfaces the μPD8041AH/8741A to the 8-bit master system's data bus.

P₁₀-P₁₇ (Port 1)

P₁₀-P₁₇ is an 8-bit quasi-bidirectional port.

P₂₀-P₂₇ (Port 2)

P₂₀-P₂₇ is an 8-bit quasi-bidirectional port. P₂₀-P₂₃ output the high-order four bits of the address during an external program memory fetch. P₂₀-P₂₃ also function as a 4-bit I/O bus for the μPD82C43 I/O port expander. P₂₄-P₂₇ can be used as port lines or interrupt requests ($\overline{\text{IBF}}$ and OBF) and DMA handshake signals (DRQ and DACK).

PROG (Program Pulse)

PROG is used in programming the μPD8041AH/8741A. PROG is also used as an output pulse during a fetch when interfacing with the μPD82C43 I/O port expander.

V_{CC} (Primary Power Supply)

V_{CC} is the primary power supply. V_{CC} must be +5 V during programming and operation of the μPD8041AH.

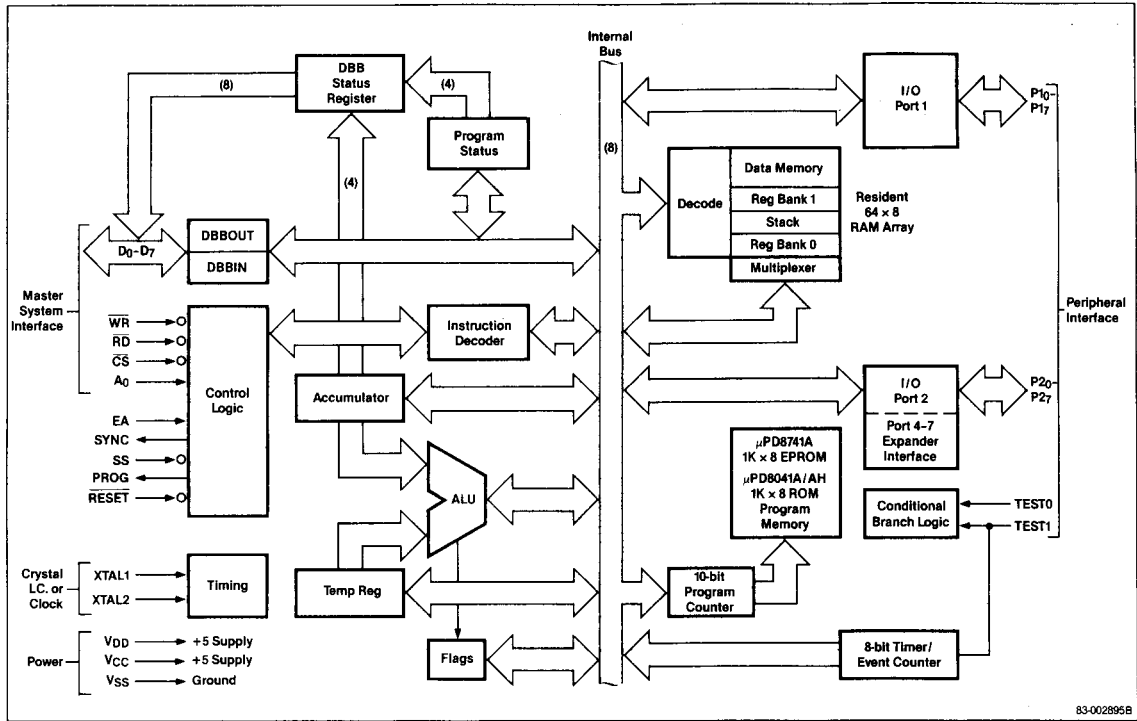
V_{DD} (Programming Supply Voltage)

V_{DD} is the programming supply voltage for programming the μPD8741AH. It is +5 V for normal operation of the μPD8041AH/8741A. V_{DD} is also the low power standby input for the ROM version.

V_{SS} (Ground)

V_{SS} is ground potential.

Block Diagram



83-002895B

Absolute Maximum Ratings

$T_A = 25^\circ\text{C}$

Power supply voltage, V_{CC}	-0.5 V to +7.0 V
Power supply voltage, V_{DD}	-0.5 V to +7.0 V
Input voltage, V_{IN}	-0.5 V to +7.0 V
Output voltage, V_O	-0.5 V to +7.0 V
Operating temperature, T_{OPT}	0°C to $+70^\circ\text{C}$
Storage temperature, T_{STG}	-65°C to $+150^\circ\text{C}$

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 the specification. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Capacitance

$T_A = 25^\circ\text{C}$

Parameter	Symbol	Limits			Unit	Test Conditions
		Min	Typ	Max		
Input capacitance	C_I			10	pF	
Output capacitance	C_{IO}			20	pF	

DC Characteristics

$T_A = 0^\circ\text{C to } +70^\circ\text{C}$, $V_{CC} = V_{DD} = +5\text{ V} \pm 10\%$; μPD8041AH: $V_{DD} = +5\text{ V} \pm 5\%$; μPD8741A: $V_{SS} = 0\text{ V}$

Parameter	Symbol	Limits				Unit	Test Conditions
		μPD8741A		μPD8041AH			
		Min	Max	Min	Max		
Input voltage low	V_{IL}	-0.5	0.8	-0.5	0.8	V	All except X1, X2, and RESET
	V_{IL1}	-0.5	0.6	-0.5	0.6	V	X1, X2, RESET
Input voltage high	V_{IH}	2.0	V_{CC}	2.0	V_{CC}	V	Except X1, X2, and RESET
	V_{IH1}	3.8	V_{CC}	3.8	V_{CC}	V	X1, X2, RESET
Output voltage low	V_{OL}		0.45		0.45	V	D ₀ -D ₇ , SYNC, $I_{OL} = 2.0\text{ mA}$
	V_{OL1}		0.45		0.45	V	Except PROG, $I_{OL} = 1.0\text{ mA}$
	V_{OL2}		0.45		0.45	V	PROG, $I_{OL} = 1.0\text{ mA}$
Output voltage high	V_{OH}	2.4		2.4		V	D ₀ -D ₇ , $I_{OH} = -400\text{ μA}$
	V_{OH1}	2.4		2.4		V	All other outputs: $I_{OH} = -50\text{ μA}$
Input current low	I_{LI}		0.5		0.5	mA	P1 ₀ -P1 ₇ , P2 ₀ -P2 ₇ : $V_{IL} = 0.8\text{ V}$
	I_{LI1}		0.2		0.2	mA	SS, RESET, $V_{IL} = 0.8\text{ V}$
Input leakage current	I_{IL}		±10		±10	μA	T ₀ , T ₁ , RD, WR, CS, EA, A ₀ , $V_{SS} \leq V_{IN} \leq V_{CC}$
Output leakage current	I_{OL}		±10		±10	μA	D ₀ -D ₇ , High Z state, $V_{SS} + 0.45\text{ V} \leq V_{IN} \leq V_{CC}$
Supply current (total)	I_{DD}		15		15	mA	V_{DD}
	$I_{DD} + I_{CC}$		135		125	mA	

AC Characteristics

$T_A = 0^\circ\text{C to } +70^\circ\text{C}$, $V_{CC} = V_{DD} = +5\text{ V} \pm 10\%$, $V_{SS} = 0\text{ V}$

DBB Read

Parameter	Symbol	Limits				Unit	Test Conditions
		μPD8741A		μPD8041AH			
		Min	Max	Min	Max		
CS, A ₀ setup to RD ↓	t_{AR}	300		0		ns	
CS, A ₀ hold after RD ↑	t_{RA}	30		0		ns	
RD pulse width	t_{RR}	300		160		ns	
CS, A ₀ , to data out delay	t_{AD}		370		130	ns	μPD8041A / 8741A: $C_L = 150\text{ pF}$ μPD8041AH: $C_L = 100\text{ pF}$
RD ↓ to data out delay	t_{RD}		200		130	ns	μPD8041A / 8741A: $C_L = 150\text{ pF}$ μPD8041AH: $C_L = 100\text{ pF}$
RD ↑ to data float delay	t_{DF}		140		85		
Cycle time	t_{CY}	2.5	15	1.36	15	ns	

AC Characteristics (cont)

$T_A = 0^\circ\text{C}$ to $+70^\circ\text{C}$, $V_{CC} = V_{DD} = +5\text{V} \pm 10\%$, $V_{SS} = 0\text{V}$

DBB Write

Parameter	Symbol	Limits				Unit	Test Conditions
		μPD8741A		μPD8041AH			
		Min	Max	Min	Max		
$\overline{\text{CS}}$, A_0 setup to $\text{WR} \downarrow$	t_{AW}	0		0		ns	
$\overline{\text{CS}}$, A_0 hold after $\text{WR} \uparrow$	t_{WA}	0		0		ns	
WR pulse width	t_{WW}	250		160		ns	μPD8041A / 8741A: $t_{CY} = 2.5\ \mu\text{s}$
Data setup to $\text{WR} \uparrow$	t_{DW}	150		130		ns	
Data hold after $\text{WR} \uparrow$	t_{WD}	0		0		ns	

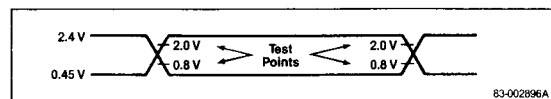
Port 2

Parameter	Symbol	Limits				Unit	Test Conditions
		μPD8741A		μPD8041AH			
		Min	Max	Min	Max		
Port control setup to $\text{PROG} \downarrow$	t_{CP}	110		100		ns	μPD8041AH: $C_L = 80\ \text{pF}$
Port control hold after $\text{PROG} \downarrow$	t_{PC}	100		60		ns	μPD8041AH: $C_L = 20\ \text{pF}$
Input data setup to $\text{PROG} \downarrow$	t_{PR}		810		650	ns	μPD8041AH: $C_L = 80\ \text{pF}$
Input data hold time	t_{PF}	0	150	0	150	ns	μPD8041AH: $C_L = 20\ \text{pF}$
Output data setup time	t_{DP}	250		200		ns	μPD8041AH: $C_L = 80\ \text{pF}$
Output data hold time	t_{PD}	65		65		ns	μPD8041AH: $C_L = 20\ \text{pF}$
PROG pulse width	t_{pp}	1200		700		ns	

DMA

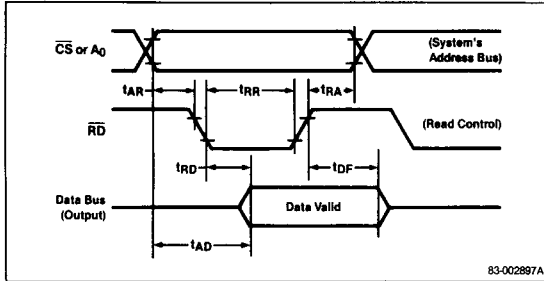
Parameter	Symbol	Limits				Unit	Test Conditions
		μPD8741A		μPD8041AH			
		Min	Max	Min	Max		
$\overline{\text{DACK}}$ setup time to RD , WR	t_{ACC}	0		0		ns	
$\overline{\text{DACK}}$ hold time after RD , WR	t_{CAC}	0		0		ns	
Data output delay after $\overline{\text{DACK}}$	t_{ACD}		225		130	ns	μPD8041A / 8741A: $C_L = 150\ \text{pF}$
DRQ clear delay time after RD , WR	t_{CRQ}		200		130	ns	μPD8041AH: $C_L = 100\ \text{pF}$

AC Timing Test Points

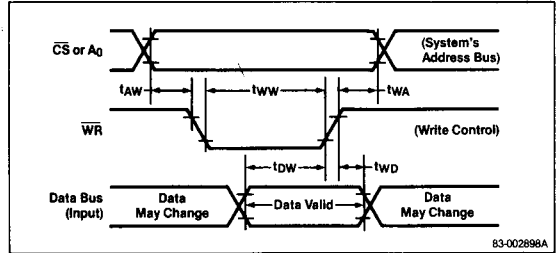


Timing Waveforms

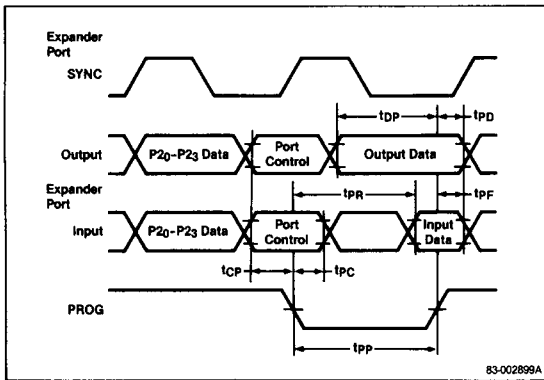
Read Operation (DBBOUT Register)



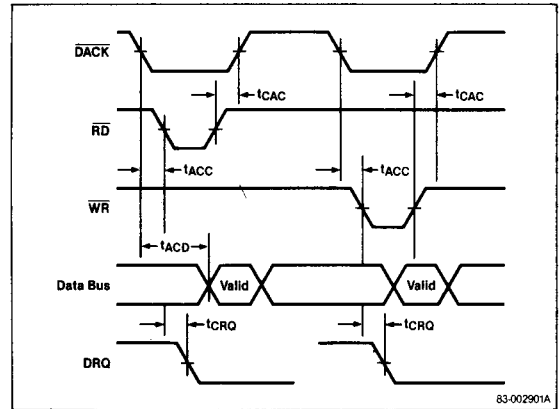
Write Operation (DBBIN Register)



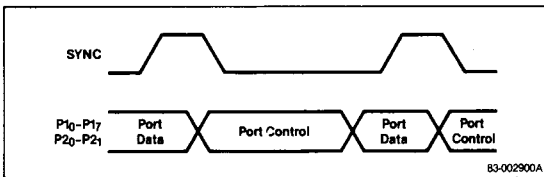
PORT 2



DMA



PORT (EA = 1)



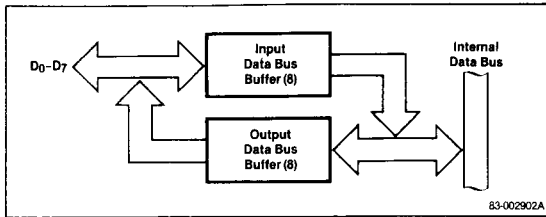
Functional Description

Two data bus buffers, an 8-bit status register, the \overline{RD} and \overline{WR} inputs, and expandable I/O lines enhance the μPD8041AH/8741A. These features enable easier master/slave interface and increased functionality.

Data Bus Buffers

Figure 1 shows how the input and output data bus buffers enable a smooth data flow to and from the master processors.

Figure 1. Data Bus Buffers

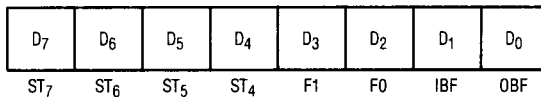


Status Register

The 8-bit status register includes four user-definable bits, ST₄-ST₇. Use the MOV STS, A instruction (90H) to define bits ST₄-ST₇ by moving accumulator bits 4-7 to bits 4-7 of the status register. Bits ST₀-ST₃ are not affected.

Figure 2 shows the format of the status register.

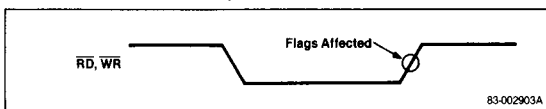
Figure 2. Status Register Format



\overline{RD} and \overline{WR}

The \overline{RD} and \overline{WR} inputs are edge-sensitive. Figure 3 shows that status bits \overline{IBF} , OBF, F1, and F0 are affected on the trailing edge at \overline{RD} or \overline{WR} .

Figure 3. \overline{RD} and \overline{WR} Inputs



Port 24-Port 27

P₂₄ and P₂₅ can be used as either port lines or buffer status flag lines. This allows you to make OBF and \overline{IBF} status available externally to interrupt the master processor. Upon execution of the EN FLAGS instruction (F5H), P₂₄ becomes the OBF pin. When a 1 is written to P₂₄, the OBF pin is enabled and the status of OBF is output. A₀ to P₂₄ disables the OBF pin AND the pin remains low. This pin indicates valid data is available from the μPD8041AH/8741A.

An EN FLAGS instruction execution also enables P₂₅ to indicate that the μPD8041AH/8741A is ready to accept data. A₁ written to P₂₅ enables the \overline{IBF} pin and the status of \overline{IBF} is available on P₂₅. A₀ written to P₂₅ disables the \overline{IBF} pin. If OBF is not true, the data at the data bus is invalid.

P₂₆ and P₂₇ can be used as either port lines or DMA handshake lines to allow DMA interface. The EN DMA instruction (E5H) enables P₂₆ and P₂₇ to be used as DRQ (DMA request) and \overline{DACK} (DMA acknowledge), respectively.

When a 1 is written to P₂₆, DRQ is activated and a DMA request is issued. The EN DMA instruction deactivates DRQ. You can also deactivate DRQ by adding \overline{DACK} with \overline{RD} or \overline{WR} . Execution of the EN DMA instruction enables P₂₇ (\overline{DACK}) to function as a chip select input for the data bus buffer registers during DMA transfers.

Instruction Set

Mnemonic	Operand	Operation	Operation Code																Flags			
			D7	D6	D5	D4	D3	D2	D1	D0	Cycles	Bytes	C	AC	F0	F1	IBF	OBF	ST4	ST7		
Accumulator																						
ADD	A, # data	(A) ← (A) + data	0	0	0	0	0	0	0	1	1	2	2	•								
		d7	d6	d5	d4	d3	d2	d1	d0													
ADD	A, Rr	(A) ← (A) + (Rr) r = 0-7	0	1	1	0	1	r	r	r	r	1	1	•								
ADD	A, @ Rr	(A) ← (A) + ((Rr)) r = 0-1	0	1	1	0	0	0	0	0	r	1	1	•								
ADDC	A, # data	(A) ← (A) + (C) + data	0	0	0	1	0	0	1	1	2	2	•									
		d7	d6	d5	d4	d3	d2	d1	d0													
ADDC	A, Rr	(A) ← (A) + (C) + (Rr) r = 0-7	0	1	1	1	1	1	r	r	r	1	1	•								
ADDC	A, @ Rr	(A) ← (A) + (C) + ((Rr)) r = 0-1	0	1	1	1	0	0	0	0	r	1	1	•								
ANL	A, # data	(A) ← (A) AND data	0	1	0	1	0	0	1	1	2	2										
		d7	d6	d5	d4	d3	d2	d1	d0													
ANL	A, Rr	(A) ← (A) AND (Rr) r = 0-7	0	1	0	1	1	r	r	r	1	1	1									
ANL	A, @ Rr	(A) ← (A) AND ((Rr)) r = 0-1	0	1	0	1	0	0	0	0	r	1	1									
CPL	A	(A) ← NOT(A)	0	0	1	1	0	1	1	1	1	1	1									
CLR	A	(A) ← 0	0	0	1	0	0	1	1	1	1	1	1									
DA	A	(A) ← (A) - 1	0	1	0	1	0	1	1	1	1	1	1	•								
DEC	A	(A) ← (A) - 1	0	0	0	0	0	1	1	1	1	1	1									
INC	A	(A) ← (A) + 1	0	0	0	0	0	1	1	1	1	1	1									
ORL	A, # data	(A) ← (A) OR data	0	1	0	0	0	1	1	1	2	2										
		d7	d6	d5	d4	d3	d2	d1	d0													
ORL	A, Rr	(A) ← (A) OR (Rr) r = 0-7	0	1	0	0	1	r	r	r	1	1	1									
ORL	A, @ Rr	(A) ← (A) OR ((Rr)) r = 0-1	0	1	0	0	0	0	0	0	r	1	1									
RL	A	(AN + 1) ← (AN); N = 0-6 (A ₀) ← (A ₇)	1	1	1	0	0	1	1	1	1	1	1									

Instruction Set (cont)

Mnemonic Accumulator (cont)	Operand	Operation	Operation Code										Flags							
			D7	D6	D5	D4	D3	D2	D1	D0	Cycles	Bytes	C	AC	F0	F1	IBF	OBF	ST ₄ -ST ₇	
RLC	A	(AN + 1) ← (AN); N = 0-6 (A ₀) ← (C) (C) ← (A ₇)	1	1	1	1	0	1	1	1	1	1	1	1	1	1	•			
RR	A	(AN) ← (AN + 1); N = 0-6 (A ₇) ← (A ₀)	0	1	1	1	0	1	1	1	1	1	1	1	1	1	•			
RRC	A	(AN) ← (AN + 1); N = 0-6 (A ₇) ← (C) (C) ← (A ₀)	0	1	1	1	0	0	1	1	1	1	1	1	1	1	•			
SWAP	A	(A ₄ -A ₇) ↔ (A ₀ -A ₃)	0	1	0	0	0	1	1	1	1	1	1	1	1	1				
XRL	A, # data	(A) ← (A) XOR data	1	1	0	1	0	0	1	1	1	1	2	2						
XRL	A, Rr	(A) ← (A) XOR (Rr) r = 0-7	1	1	0	1	1	1	r	r	r	r	1	1	1					
XRL	A, @Rr	(A) ← (A) XOR ((Rr)) r = 0-1	1	1	0	1	0	0	0	0	0	r	1	1	1					

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code										Flags						
			D7	D6	D5	D4	D3	D2	D1	D0	C	AC	F0	F1	IBF	OBF	ST4	ST7	
Branch																			
DJNZ	Rr, addr	(Rr) ← (Rr) - 1; r = 0-7 if (Rr) ≠ 0, (PC0-PC7) ← addr	1	1	1	0	1	1	0	1	1	r	r	r	r	2	2		
JBb	addr	(PC0-PC7) ← addr if Bb = 1 (PC) ← (PC) + 2 if Bb = 0	b2	b1	b0	1	0	0	1	0	1	0	0	1	0	2	2		
JC	addr	(PC0-PC7) ← addr if C = 1 (PC) ← (PC) + 2 if C = 0	1	1	1	1	0	1	1	0	1	0	1	0	0	2	2		
JFO	addr	(PC0-PC7) ← addr if FO = 1 (PC) ← (PC) + 2 if FO = 0	1	0	1	1	0	1	0	1	1	0	1	0	0	2	2		
JF1	addr	(PC0-PC7) ← addr if F1 = 1 (PC) ← (PC) + 2 if F1 = 0	0	1	1	1	0	1	0	1	1	0	1	0	0	2	2		
JMP	addr	(PC0-PC10) ← (addr; addr; r; 0) (PC0-PC7) ← (addr; 0-addr; r) (PC11) ← DBF	a10	a9	a8	0	0	1	0	0	1	0	0	0	0	2	2		
JMPP	@ A	(PC0-PC7) ← ((A))	1	0	1	1	0	0	1	0	1	1	1	1	0	2	1		
JNC	addr	(PC0-PC7) ← addr if C = 0 (PC) ← (PC) + 2 if C = 1	1	1	1	0	0	1	1	0	1	1	0	0	0	2	2		
JNIBF	addr	(PC0-PC7) ← addr if IBF = 0 (PC) ← (PC) + 2 if IBF = 1	1	1	0	1	0	1	0	1	1	0	1	0	0	2	2		
JOBFB		(PC0-PC7) ← addr if OBF = 1 (PC) ← (PC) + 2 if OBF = 0	1	0	0	0	0	1	1	1	1	0	1	0	0	2	2		
JNTO	addr	(PC0-PC7) ← addr if TO = 0 (PC) ← (PC) + 2 if TO = 1	0	0	1	0	0	1	0	0	1	1	0	0	0	2	2		
JNT1	addr	(PC0-PC7) ← addr if T1 = 0 (PC) ← (PC) + 2 if T1 = 1	0	1	0	0	0	1	1	1	1	0	1	0	0	2	2		
JNZ	addr	(PC0-PC7) ← addr if A = 0 (PC) ← (PC) + 2 if A = 1	1	0	0	1	0	1	0	1	1	0	1	0	0	2	2		
JTF	addr	(PC0-PC7) ← addr if TF = 1 (PC) ← (PC) + 2 if TF = 0	0	0	0	1	0	1	0	1	1	0	1	0	0	2	2		
JTO	addr	(PC0-PC7) ← addr if TO = 1 (PC) ← (PC) + 2 if TO = 0	0	0	1	1	0	1	0	1	1	0	1	0	0	2	2		
JT1	addr	(PC0-PC7) ← addr if T1 = 1 (PC) ← (PC) + 2 if T1 = 0	0	1	0	1	0	1	0	1	1	0	1	0	0	2	2		
JZ	addr	(PC0-PC7) ← addr if A = 0 (PC) ← (PC) + 2 if A = 1	1	1	0	0	0	1	1	0	1	1	0	0	0	2	2		

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code													Flags														
			D7	D6	D5	D4	D3	D2	D1	D0	Cycles	Bytes	C	AC	F0	F1	IBF	OBF	ST4-	ST7										
Control																														
EN I		Enable the external interrupt input	0	0	0	0	0	1	0	1	0	1	0	1	1	1														
DIS I		Disable the external interrupt input	0	0	0	1	0	1	0	1	0	1	0	1	1	1														
SEL RBO	(BS) ← 0		1	1	0	0	0	1	0	1	0	1	0	1	1	1														
SEL RB	(BS) ← 1		1	1	0	1	0	1	0	1	0	1	0	1	1	1														
EN DMA		Enable DMA handshake	1	1	1	1	0	1	0	1	0	1	0	1	1	1														
EN FLAGS		Enable interrupt to master device	1	1	1	0	0	1	0	1	0	1	0	1	1	1														
Data Moves																														
MOV	A, # data	(A) ← data	0	0	1	0	0	0	1	1	0	1	1	2	2		d7	d6	d5	d4	d3	d2	d1	d0						
MOV	A, Rr	(A) ← (Rr); r = 0-7	1	1	1	1	1	1	r	r	r	r	r	1	1	1														
MOV	A, @ Rr	(A) ← ((Rr)); r = 0-1	1	1	1	1	0	0	0	0	0	0	1	1	1															
MOV	A, PSW	(A) ← (PSW)	1	1	0	0	0	1	1	1	1	1	1	1	1															
MOV	Rr, # data	(Rr) ← data; r = 0-7	1	0	1	1	1	1	r	r	r	r	r	2	2		d7	d6	d5	d4	d3	d2	d1	d0						
MOV	Rr, A	(Rr) ← (A); r = 0-7	1	0	1	0	1	1	r	r	r	r	r	1	1															
MOV	@ Rr, A	((Rr)) ← (A); r = 0-1	1	0	1	0	0	0	0	0	0	0	1	1	1															
MOV	@ Rr, # data	((Rr)) ← data; r = 0-1	1	0	1	1	0	0	0	0	0	0	r	2	2		d7	d6	d5	d4	d3	d2	d1	d0						
MOV	PSW, A	(PSW) ← (A)	1	1	0	1	0	1	1	1	1	1	1	1	1															
MOVIP	A, @ A	(PC ₀ -PC ₇) ← (A) (A) ← ((PC))	1	0	1	0	0	0	1	1	1	1	1	2	1															
MOVIP3	A, @ A	(PC ₀ -PC ₇) ← (A) (PC ₈ -PC ₁₀) ← 011 (A) ← ((PC))	1	1	1	0	0	0	1	1	1	1	1	2	1															
XCH	A, Rr	(A) ↔ (Rr); r = 0-7	0	0	1	0	1	r	r	r	r	r	r	1	1															
XCH	A, @ Rr	(A) ↔ ((Rr)); r = 0-1	0	0	1	0	0	0	0	0	0	0	r	1	1															
XCHD	A, @ Rr	(A ₀ -A ₃) ↔ ((Rr)) ₀ -((Rr)) ₃ ; r = 0-1	0	0	1	1	0	0	0	0	0	0	r	1	1															

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code										Flags							
			D7	D6	D5	D4	D3	D2	D1	D0	Cycles	Bytes	C	AC	F0	F1	IBF	OBF	ST ₄ -ST ₇	
Flags																				
CPL C		(C) ← NOT (C)	1	0	1	0	0	1	1	1	1	1	1	1	1	1	•			
CPL F0		(F0) ← NOT (F0)	1	0	0	1	0	1	0	1	0	1	1	1	1	1	•			
CPL F1		(F1) ← NOT (F1)	1	0	1	1	0	1	0	1	0	1	1	1	1	1	•			
CLR C		(C) ← 0	1	0	0	1	0	1	1	1	1	1	1	1	1	1	•			
CLR F0		(F0) ← 0	1	0	0	0	0	1	0	1	0	1	1	1	1	1	•			
CLR F1		(F1) ← 0	1	0	1	0	0	1	0	1	0	1	1	1	1	1	•			
MOV ST ₅ , A		ST ₄ -ST ₇ ← A ₄ -A ₇	1	0	0	1	0	0	0	0	0	0	1	1	1	1				
Input / Output																				
ANL	Pp, # data	(Pp) ← (Pp) AND data p = 1-2	d7	d6	d5	d4	d3	d2	d1	d0										2
ANLD	Pp, A	(Pp) ← (Pp) AND (A ₀ -A ₃); p = 4-7	1	0	0	1	1	1	1	p	p	p	2	1						
IN	A, Pp	(A) ← (Pp); p = 1-2	0	0	0	0	1	0	p	p	p	2	1							
IN	A, DBB	(A) ← (DBB)	0	0	1	0	0	0	1	0	1	1	1	1	•					
MOVD	A, Pp	(A ₀ -A ₃) ← (Pp); p = 4-7 (A ₄ -A ₇) ← 0	0	0	0	0	1	1	p	p	p	2	1							
MOVD	Pp, A	(Pp) ← (A ₀ -A ₃); p = 4-7	0	0	1	1	1	1	1	p	p	1	1							
ORLD	Pp, A	(Pp) ← (Pp) OR (A ₀ -A ₃); p = 4-7	1	0	0	0	1	1	p	p	p	1	1							
ORL	Pp, # data	(Pp) ← (Pp) OR data p = 1-2	d7	d6	d5	d4	d3	d2	d1	d0										2
OUT	DBB, A	(DBB) ← (A)	0	0	0	0	0	0	1	0	1	0	1	1						
OUTL	Pp, A	(Pp) ← (A); p = 1-2	0	0	1	1	1	0	p	p	p	1	1							
Registers																				
DEC	Rr (Rr)	(Rr) ← (Rr) - 1; r = 0-7	1	1	0	0	1	r	r	r	r	1	1							
INC	Rr	(Rr) ← (Rr) + 1; r = 0-7	0	0	0	1	1	r	r	r	r	1	1							
INC	@ Rr	((Rr)) ← ((Rr)) + 1; r = 0-1	0	0	0	1	0	0	0	0	r	1	1							

Instruction Set (cont)

Mnemonic Subroutine	Operand	Operation	Operation Code																Flags			
			D7	D6	D5	D4	D3	D2	D1	D0	Cycles	Bytes	C	AC	F0	F1	IRF	OBF	ST ₄	ST ₇		
CALL	addr	$((SP)) \leftarrow (PC),$ $(PSW_4-PSW_7),$ $(SP) \leftarrow (SP) + 1$ $(PC_8-PC_{10}) \leftarrow (addr_8-addr_{10})$ $(PC_0-PC_7) \leftarrow (addr_0-addr_7)$ $(PC_{11}) \leftarrow DBF$	a ₁₀	a ₉	a ₈	1	0	1	0	1	0	0	0	2	2							
RET		$(SP) \leftarrow (SP) = 1$ $(PC) \leftarrow ((SP))$	1	0	0	0	0	0	0	1	1	2	1									
RETR		$(SP) \leftarrow (SP) = 1$ $(PC) \leftarrow ((SP))$ $(PSW_4-PSW_7) \leftarrow ((SP))$	1	0	0	1	0	0	1	1	2	1										
Timer / Counter																						
EN TCNTI	Enable internal interrupt flag for timer / counter output.		0	0	1	0	0	1	0	1	0	1	1	1								
DIS TCNTI	Disable internal interrupt flag for timer / counter output.		0	0	1	1	0	1	0	1	0	1	1	1								
MOV A, T	(A) ← (T)		0	1	0	0	0	0	1	0	1	0	1	1								
MOV T, A	(T) ← (A)		0	1	1	0	0	0	1	0	1	0	1	1								
STOP TCNT	Stop count for event counter.		0	1	1	0	0	1	0	1	0	1	1	1								
STRT CNT	Start count for event counter.		0	1	0	0	0	1	0	1	0	1	1	1								
STRT T	Start count for timer.		0	1	0	1	0	1	0	1	0	1	1	1								
Miscellaneous																						
NOP	No operation performed.		0	0	0	0	0	0	0	0	0	0	1	1								

Note:

- (1) Operation code designations r and p form the binary representation of the registers and ports involved.
- (2) The dot under the appropriate flag bit indicates that its contents is subject to change by the instruction it appears in.
- (3) References to the address and data are specified in bytes 2 and/or 1 of the instruction.
- (4) Numerical subscripts appearing in the operation column reference the specific bits affected.

Instruction Set (cont)**Symbol Definitions**

Symbol	Description
A	Accumulator
AC	Auxiliary carry flag
addr	Program memory address (12 bits)
B _b	Bit designator (b = 0-7)
BS	Bank switch
BUS	Bus port
C	Carry flag
CLK	Clock signal
CNT	Event counter
D	Nibble designator (4 bits)
data	Number of expression (8 bits)
DBF	Memory bank flip-flop
F0, F1	Flags 0, 1
I	Interrupt
P	In-page operation designator
IBF	Input buffer full flag
Pp	Port designator (p = 1, 2 or 4-7)
PSW	Program status word

Symbol	Description
Rr	Register designator (r = 0, 1 or 0-7)
SP	Stack pointer
T	Timer
TF	Timer flag
T0, T1	Testable inputs 0, 1
X	External RAM
#	Prefix for immediate data
@	Prefix for indirect address
\$	Current value of program counter
(x)	Contents of external RAM location
((x))	Contents of memory location addressed by the contents of external RAM location
←	Replaced by
OBF	Output buffer full flag
DBB	Data bus buffer
AND	Logical product (logical AND)
OR	Logical sum (logical OR)
XOR	Exclusive-OR