NEC

NEC Electronics Inc.

μPD8041AH, μPD8741A 8-BIT, SINGLE-CHIP NMOS MICROCOMPUTERS WITH UNIVERSAL PPI

Description

The μPD8041AH and μPD8741A 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 μPD8041AH/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 μPD8041AH/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 μPD8041AH/8741A contains an 8-bit CPU, 1K×8 program memory, 64×8 data memory, 18 I/O lines, a counter/timer, and a clock generator. The program memory for the μPD8041AH is factory mask-programmed, while program memory for the μPD8741A is UV EPROM for more flexibility.

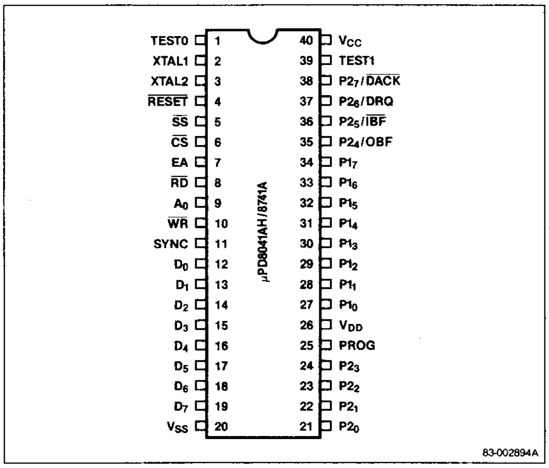
Features

- ☐ Complete single chip microcomputer
  - 8-bit CPU
  - 1K × 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
μPD8041AHC	40-pin plastic DIP	11 MHz
μPD8741AD	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	$\overline{\text{RESET}}$	Reset input
5	$\overline{\text{SS}}$	Single step input
6	$\overline{\text{CS}}$	Chip select input
7	EA	External access input
8	$\overline{\text{RD}}$	Read strobe input
9	A_0	Address input 0
10	$\overline{\text{WR}}$	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	$P2_0$–$P2_7$	Quasi-bidirectional Port 2
25	PROG	Program pulse output
26	V_{DD}	Programming supply voltage
27–34	$P1_0$–$P1_7$	Quasi-bidirectional Port 1
39	T1	Testable input 1
40	V_{CC}	Primary power supply

4

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.

D_0-D_7 (Data Bus)

D_0-D_7 is a three-state, bidirectional data bus. D_0-D_7 interfaces the μPD8041AH/8741A to the 8-bit master system's data bus.

$P1_0$-$P1_7$ (Port 1)

$P1_0$-$P1_7$ is an 8-bit quasi-bidirectional port.

$P2_0$-$P2_7$ (Port 2)

$P2_0$-$P2_7$ is an 8-bit quasi-bidirectional port. $P2_0$-$P2_3$ output the high-order four bits of the address during an external program memory fetch. $P2_0$-$P2_3$ also function as a 4-bit I/O bus for the μPD82C43 I/O port expander. $P2_4$-$P2_7$ can be used as port lines or interrupt requests ($\overline{\text{IBF}}$ and OBF) and DMA handshake signals (DRQ and $\overline{\text{DACK}}$).

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.

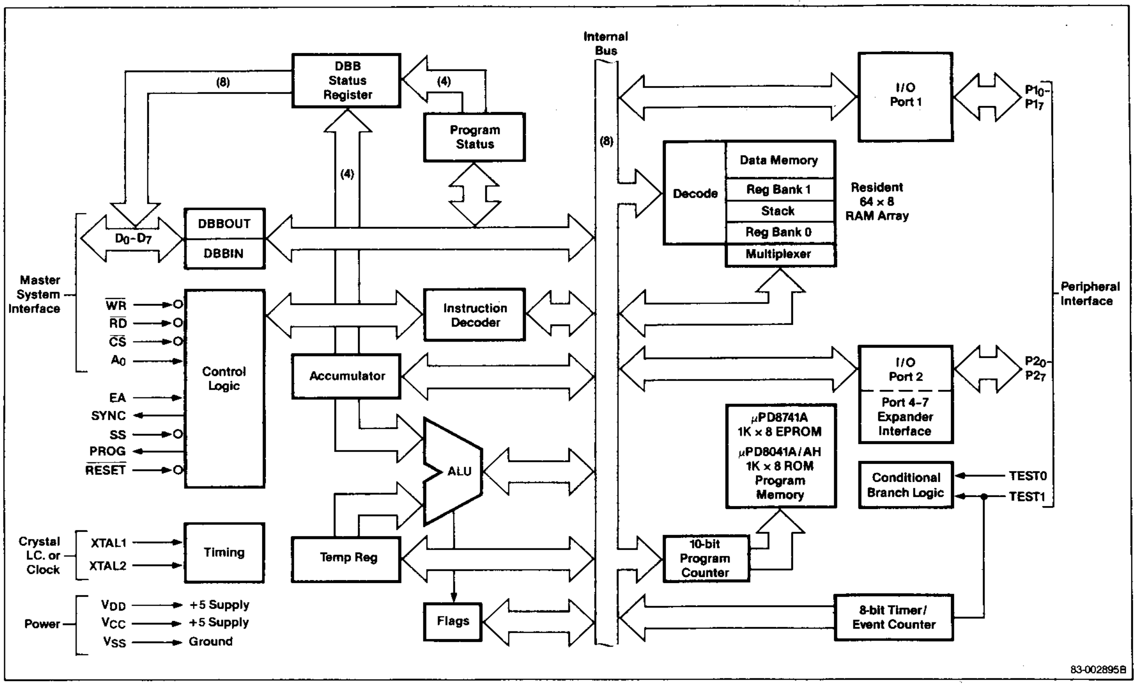
V_{DD} (Programming Supply Voltage)

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



Absolute Maximum Ratings

$T_A = 25°C$

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

Capacitance

$T_A = 25°C$

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

DC Characteristics

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

		Limits					
		μPD8741A		μPD8041AH			
Parameter	Symbol	Min	Max	Min	Max	Unit	Test Conditions
Input voltage low	V_{IL}	−0.5	0.8	−0.5	0.8	V	All except X1, X2, and $\overline{RESET}$
	V_{IL1}	−0.5	0.6	−0.5	0.6	V	X1, X2, $\overline{RESET}$
Input voltage high	V_{IH}	2.0	V_{CC}	2.0	V_{CC}	V	Except X1, X2, and $\overline{RESET}$
	V_{IH1}	3.8	V_{CC}	3.8	V_{CC}	V	X1, X2, $\overline{RESET}$
Output voltage low	V_{OL}		0.45		0.45	V	D_0–D_7, SYNC, $I_{OL} = 2.0\,mA$
	V_{OL1}		0.45		0.45	V	Except PROG, $I_{OL} = 1.0\,mA$
	V_{OL2}		0.45		0.45	V	PROG, $I_{OL} = 1.0\,mA$
Output voltage high	V_{OH}	2.4		2.4		V	D_0–D_7, $I_{OH} = -400\,\mu A$
	V_{OH1}	2.4		2.4		V	All other outputs: $I_{OH} = -50\,\mu A$
Input current low	I_{LI}		0.5		0.5	mA	$P1_0$–$P1_7$, $P2_0$–$P2_7$: $V_{IL} = 0.8\,V$
	I_{LI1}		0.2		0.2	mA	$\overline{SS}$, $\overline{RESET}$; $V_{IL} = 0.8\,V$
Input leakage current	I_{IL}		±10		±10	μA	T0, T1, $\overline{RD}$, $\overline{WR}$, $\overline{CS}$, 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, $V_{SS} + 0.45\,V \leq V_{IN} \leq V_{CC}$
Supply current (total)	I_{DD}		15		15	mA	V_{DD}
	$I_{DD} + I_{CC}$		135		125	mA	

AC Characteristics

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

DBB Read

		Limits					
		μPD8741A		μPD8041AH			
Parameter	Symbol	Min	Max	Min	Max	Unit	Test Conditions
$\overline{CS}$, A_0 setup to $\overline{RD}$ ↓	t_{AR}	300		0		ns	
$\overline{CS}$, A_0 hold after $\overline{RD}$ ↑	t_{RA}	30		0		ns	
$\overline{RD}$ pulse width	t_{RR}	300		160		ns	
$\overline{CS}$, A_0, to data out delay	t_{AD}		370		130	ns	μPD8041A / 8741A: $C_L = 150\,pF$ μPD8041AH: $C_L = 100\,pF$
$\overline{RD}$ ↓ to data out delay	t_{RD}		200		130	ns	μPD8041A / 8741A: $C_L = 150\,pF$ μPD8041AH: $C_L = 100\,pF$
$\overline{RD}$ ↑ to data float delay	t_{DF}		140		85		
Cycle time	t_{CY}	2.5	15	1.36	15	ns	

AC Characteristics (cont)

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

DBB Write

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

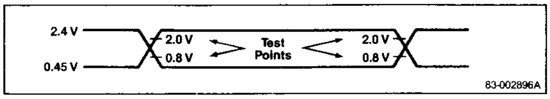
Port 2

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

DMA

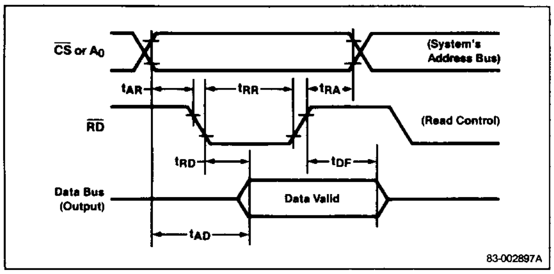
		Limits					
		μPD8741A		μPD8041AH			
Parameter	Symbol	Min	Max	Min	Max	Unit	Test Conditions
$\overline{DACK}$ setup time to $\overline{RD}$, $\overline{WR}$	t_{ACC}	0		0		ns	
$\overline{DACK}$ hold time after $\overline{RD}$, $\overline{WR}$	t_{CAC}	0		0		ns	
Data output delay after $\overline{DACK}$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

AC Timing Test Points

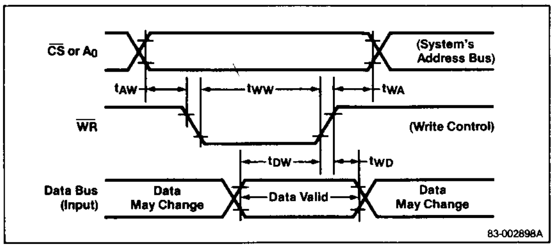


Timing Waveforms

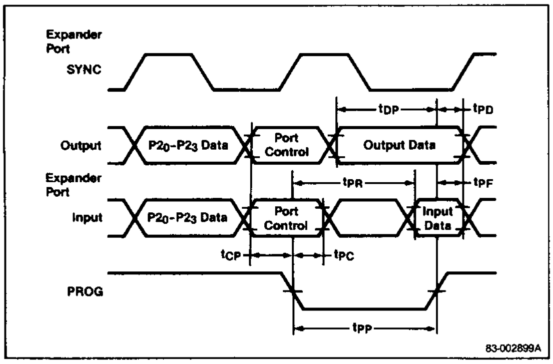
Read Operation (DBBOUT Register)



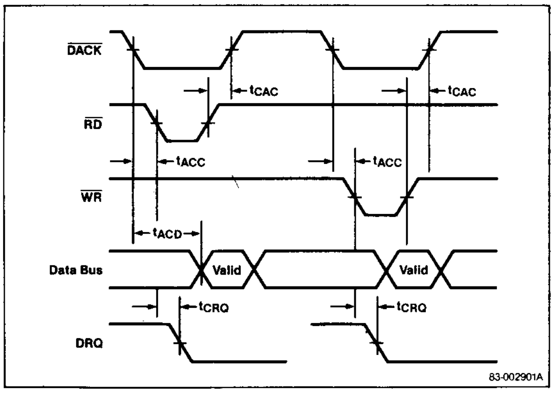
Write Operation (DBBIN Register)



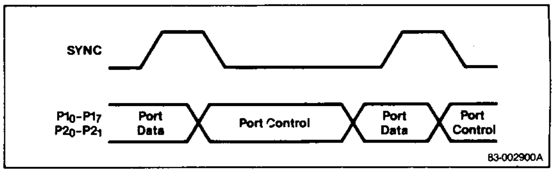
PORT 2



DMA



PORT (EA = 1)



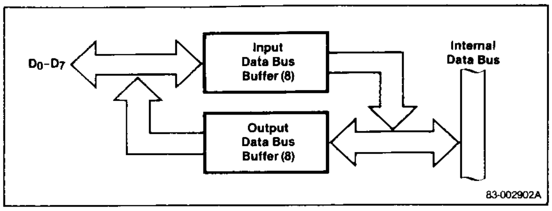
Functional Description

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, ST_4-ST_7. Use the MOV STS, A instruction (90H) to define bits ST_4-ST_7 by moving accumulator bits 4-7 to bits 4-7 of the status register. Bits ST_0-ST_3 are not affected.

Figure 2 shows the format of the status register.

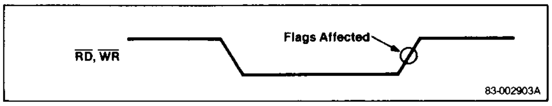
Figure 2. Status Register Format

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
ST_7	ST_6	ST_5	ST_4	F1	F0	IBF	OBF

$\overline{RD}$ and $\overline{WR}$

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

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



Port 2_4-Port 2_7

$P2_4$ and $P2_5$ 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), $P2_4$ becomes the OBF pin. When a 1 is written to $P2_4$, the OBF pin is enabled and the status of OBF is output. A_0 to $P2_4$ 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 $P2_5$ to indicate that the μPD8041AH/8741A is ready to accept data. A_1 written to $P2_5$ enables the $\overline{IBF}$ pin and the status of $\overline{IBF}$ is available on $P2_5$. A_0 written to $P2_5$ disables the $\overline{IBF}$ pin. If OBF is not true, the data at the data bus is invalid.

$P2_6$ and $P2_7$ can be used as either port lines or DMA handshake lines to allow DMA interface. The EN DMA instruction (E5H) enables $P2_6$ and $P2_7$ to be used as DRQ (DMA request) and $\overline{DACK}$ (DMA acknowledge), respectively.

When a 1 is written to $P2_6$, 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 $P2_7$ ($\overline{DACK}$) to function as a chip select input for the data bus buffer registers during DMA transfers.

Instruction Set

Mnemonic	Operand	Operation	Operation Code D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Cycles	Bytes	Flags C	AC	F0	F1	$\overline{\text{IBF}}$	OBF	ST_4-ST_7
Accumulator																			
ADD	A, # data	(A) ← (A) + data	0 d_7	0 d_6	0 d_5	0 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2	●						
ADD	A, Rr	(A) ← (A) + (Rr) r = 0–7	0	1	1	0	1	r	r	r	1	1	●						
ADD	A, @ Rr	(A) ← (A) + ((Rr)) r = 0–1	0	1	1	0	0	0	0	r	1	1	●						
ADDC	A, # data	(A) ← (A) + (C) + data	0 d_7	0 d_6	0 d_5	1 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2	●						
ADDC	A, Rr	(A) ← (A) + (C) + (Rr) r = 0–7	0	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	r	1	1	●						
ANL	A, # data	(A) ← (A) AND data	0 d_7	1 d_6	0 d_5	1 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2							
ANL	A, Rr	(A) ← (A) AND (Rr) r = 0–7	0	1	0	1	1	r	r	r	1	1							
ANL	A, @ Rr	(A) ← (A) AND ((Rr)) r = 0–1	0	1	0	1	0	0	0	r	1	1							
CPL	A	(A) ← NOT (A)	0	0	1	1	0	1	1	1	1	1							
CLR	A	(A) ← 0	0	0	1	0	0	1	1	1	1	1							
DA	A		0	1	0	1	0	1	1	1	1	1	●						
DEC	A	(A) ← (A) − 1	0	0	0	0	0	1	1	1	1	1							
INC	A	(A) ← (A) + 1	0	0	0	1	0	1	1	1	1	1							
ORL	A, # data	(A) ← (A) OR data	0 d_7	1 d_6	0 d_5	0 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2							
ORL	A, Rr	(A) ← (A) OR (Rr) r = 0–7	0	1	0	0	1	r	r	r	1	1							
ORL	A, @ Rr	(A) ← (A) OR ((Rr)) r = 0–1	0	1	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							

Instruction Set (cont)

			Operation Code										Flags						
Mnemonic	Operand	Operation	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Cycles	Bytes	C	AC	F0	F1	$\overline{\text{IBF}}$	OBF	ST_4-ST_7
Accumulator (cont)																			
RLC	A	(AN + 1) ← (AN); N = 0–6 (A_0) ← (C) (C) ← (A_7)	1	1	1	1	0	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							
RRC	A	(AN) ← (AN + 1); N = 0–6 (A_7) ← (C) (C) ← (A_0)	0	1	1	0	0	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							
XRL	A, # data	(A) ← (A) XOR data	1 d_7	1 d_6	0 d_5	1 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2							
XRL	A, Rr	(A) ← (A) XOR (Rr) r = 0–7	1	1	0	1	1	r	r	r	1	1							
XRL	A, @ Rr	(A) ← (A) XOR ((Rr)) r = 0–1	1	1	0	1	0	0	0	r	1	1							

NEC

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code D7	D6	D5	D4	D3	D2	D1	D0	Cycles	Bytes	Flags C	AC	F0	F1	IBF	OBF	ST_4-ST_7
Branch																			
DJNZ	Rr, addr	(Rr) ← (Rr) − 1; r = 0-7 If (Rr) ≠ 0; (PC_0-PC_7) ← addr	1 a_7	1 a_6	1 a_5	0 a_4	1 a_3	r a_2	r a_1	r a_0	2	2							
JB_b	addr	(PC_0-PC_7) ← addr if B_b = 1 (PC) ← (PC) + 2 if B_b = 0	b_2 a_7	b_1 a_6	b_0 a_5	1 a_4	0 a_3	0 a_2	1 a_1	0 a_0	2	2							
JC	addr	(PC_0-PC_7) ← addr if C = 1 (PC) ← (PC) + 2 if C = 0	1 a_7	1 a_6	1 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2							
JF0	addr	(PC_0-PC_7) ← addr if F0 = 1 (PC) ← (PC) + 2 if F0 = 0	1 a_7	0 a_6	1 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2							
JF1	addr	(PC_0-PC_7) ← addr if F1 = 1 (PC) ← (PC) + 2 if F1 = 0	0 a_7	1 a_6	1 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2							
JMP	addr	(PC_8-PC_{10}) ← ($addr_8$-$addr_{10}$) (PC_0-PC_7) ← ($addr_0$-$addr_7$) (PC_{11}) ← DBF	a_{10} a_7	a_9 a_6	a_8 a_5	0 a_4	0 a_3	1 a_2	0 a_1	0 a_0	2	2							
JMPP	@A	(PC_0-PC_7) ← ((A))	1	0	1	1	0	0	1	1	2	1							
JNC	addr	(PC_0-PC_7) ← addr if C = 0 (PC) ← (PC) + 2 if C = 1	1 a_7	1 a_6	1 a_5	0 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2							
JNIBF	addr	(PC_0-PC_7) ← addr if IBF = 0 (PC) ← (PC) + 2 if IBF = 1	1 a_7	1 a_6	0 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2							
JOBF		(PC_0-PC_7) ← addr if OBF = 1 (PC) ← (PC) + 2 if OBF = 0	1 a_7	0 a_6	0 a_5	0 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2							
JNT0	addr	(PC_0-PC_7) ← addr if T0 = 0 (PC) ← (PC) + 2 if T0 = 1	0 a_7	0 a_6	1 a_5	0 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2							
JNT1	addr	(PC_0-PC_7) ← addr if T1 = 0 (PC) ← (PC) + 2 if T1 = 1	0 a_7	1 a_6	0 a_5	0 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2							
JNZ	addr	(PC_0-PC_7) ← addr if A = 0 (PC) ← (PC) + 2 if A = 1	1 a_7	0 a_6	0 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2							
JTF	addr	(PC_0-PC_7) ← addr if TF = 1 (PC) ← (PC) + 2 if TF = 0	0 a_7	0 a_6	0 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2							
JT0	addr	(PC_0-PC_7) ← addr if T0 = 1 (PC) ← (PC) + 2 if T0 = 0	0 a_7	0 a_6	1 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2							
JT1	addr	(PC_0-PC_7) ← addr if T1 = 1 (PC) ← (PC) + 2 if T1 = 0	0 a_7	1 a_6	0 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2							
JZ	addr	(PC_0-PC_7) ← addr if A = 0 (PC) ← (PC) + 2 if A = 1	1 a_7	1 a_6	0 a_5	0 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2							

Instruction Set (cont)

			Operation Code										Flags						
Mnemonic	Operand	Operation	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Cycles	Bytes	C	AC	F0	F1	$\overline{IBF}$	OBF	ST_4-ST_7
Control																			
EN I	Enable the external interrupt input		0	0	0	0	0	1	0	1	1	1							
DIS I	Disable the external interrupt input		0	0	0	1	0	1	0	1	1	1							
SEL RB0	(BS) ← 0		1	1	0	0	0	1	0	1	1	1							
SEL RB	(BS) ← 1		1	1	0	1	0	1	0	1	1	1							
EN DMA	Enable DMA handshake		1	1	1	1	0	1	0	1	1	1							
EN FLAGS	Enable interrupt to master device		1	1	1	0	0	1	0	1	1	1							
Data Moves																			
MOV	A, # data	(A) ← data	0 d_7	0 d_6	1 d_5	0 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2							
MOV	A, Rr	(A) ← (Rr); r = 0–7	1	1	1	1	1	r	r	r	1	1							
MOV	A, @ Rr	(A) ← ((Rr)); r = 0–1	1	1	1	1	0	0	0	r	1	1							
MOV	A, PSW	(A) ← (PSW)	1	1	0	0	0	1	1	1	1	1							
MOV	Rr, # data	(Rr) ← data; r = 0–7	1 d_7	0 d_6	1 d_5	1 d_4	1 d_3	r d_2	r d_1	r d_0	2	2							
MOV	Rr, A	(Rr) ← (A); r = 0–7	1	0	1	0	1	r	r	r	1	1							
MOV	@ Rr, A	((Rr)) ← (A); r = 0–1	1	0	1	0	0	0	0	r	1	1							
MOV	@ Rr, # data	((Rr)) ← data; r = 0–1	1 d_7	0 d_6	1 d_5	1 d_4	0 d_3	0 d_2	0 d_1	r d_0	2	2							
MOV	PSW, A	(PSW) ← (A)	1	1	0	1	0	1	1	1	1	1							
MOVP	A, @ A	(PC_0–PC_7) ← (A) (A) ← ((PC))	1	0	1	0	0	0	1	1	2	1							
MOVP3	A, @ A	(PC_0–PC_7) ← (A) (PC_8–PC_{10}) ← 011 (A) ← ((PC))	1	1	1	0	0	0	1	1	2	1							
XCH	A, Rr	(A) ↔ (Rr); r = 0–7	0	0	1	0	1	r	r	r	1	1							
XCH	A, @ Rr	(A) ↔ ((Rr)); r = 0–1	0	0	1	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	r	1	1							

4

Instruction Set (cont)

			Operation Code										Flags						
Mnemonic	Operand	Operation	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Cycles	Bytes	C	AC	F0	F1	$\overline{IBF}$	OBF	ST_4-ST_7
Flags																			
CPL C		(C) ← NOT (C)	1	0	1	0	0	1	1	1	1	1	●						
CPL F0		(F0) ← NOT (F0)	1	0	0	1	0	1	0	1	1	1			●				
CPL F1		(F1) ← NOT (F1)	1	0	1	1	0	1	0	1	1	1				●			
CLR C		(C) ← 0	1	0	0	1	0	1	1	1	1	1	●						
CLR F0		(F0) ← 0	1	0	0	0	0	1	0	1	1	1			●				
CLR F1		(F1) ← 0	1	0	1	0	0	1	0	1	1	1				●			
MOV STS, A		ST_4-ST_7 ← A_4-A_7	1	0	0	1	0	0	0	0	1	1							
Input / Output																			
ANL	Pp, # data	(Pp) ← (Pp) AND data p = 1-2	1 d_7	0 d_6	0 d_5	1 d_4	1 d_3	0 d_2	p d_1	p d_0	2	2							
ANLD	Pp, A	(Pp) ← (Pp) AND (A_0-A_3); p = 4-7	1	0	0	1	1	1	p	p	2	1							
IN	A, Pp	(A) ← (Pp); p = 1-2	0	0	0	0	1	0	p	p	2	1							
IN	A, DBB	(A) ← (DBB)	0	0	1	0	0	0	1	0	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	2	1							
MOVD	Pp, A	(Pp) ← (A_0-A_3); p = 4-7	0	0	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	1	1							
ORL	Pp, # data	(Pp) ← (Pp) OR data p = 1-2	1 d_7	0 d_6	0 d_5	0 d_4	1 d_3	0 d_2	p d_1	p d_0	2	2							
OUT	DBB, A	(DBB) ← (A)	0	0	0	0	0	0	1	0	1	1							
OUTL	Pp, A	(Pp) ← (A); p = 1-2	0	0	1	1	1	0	p	p	1	1							
Registers																			
DEC	Rr (Rr)	(Rr) ← (Rr) − 1; r = 0-7	1	1	0	0	1	r	r	r	1	1							
INC	Rr	(Rr) ← (Rr) + 1; r = 0-7	0	0	0	1	1	r	r	r	1	1							
INC	@ Rr	((Rr)) ← ((Rr)) + 1; r = 0-1	0	0	0	1	0	0	0	r	1	1							

Instruction Set (cont)

Mnemonic	Operand	Operation	Operation Code D7	D6	D5	D4	D3	D2	D1	D0	Cycles	Bytes	Flags C	AC	F0	F1	$\overline{IBF}$	OBF	ST4-ST7
Subroutine																			
CALL	addr	((SP)) ← (PC), (PSW$_4$–PSW$_7$), (SP) ← (SP) + 1 (PC$_8$–PC$_{10}$) ← (addr$_8$–addr$_{10}$) (PC$_0$–PC$_7$) ← (addr$_0$–addr$_7$) (PC$_{11}$) ← DBF	a_{10} a_7	a_9 a_6	a_8 a_5	1 a_4	0 a_3	1 a_2	0 a_1	0 a_0	2	2							
RET		(SP) ← (SP) = 1 (PC) ← ((SP))	1	0	0	0	0	0	1	1	2	1							
RETR		(SP) ← (SP) = 1 (PC) ← ((SP)) (PSW$_4$–PSW$_7$) ← ((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	1	1							
DIS TCNTI	Disable internal interrupt flag for timer / counter output.		0	0	1	1	0	1	0	1	1	1							
MOV A, T	(A) ← (T)		0	1	0	0	0	0	1	0	1	1							
MOV T, A	(T) ← (A)		0	1	1	0	0	0	1	0	1	1							
STOP TCNT	Stop count for event counter.		0	1	1	0	0	1	0	1	1	1							
STRT CNT	Start count for event counter.		0	1	0	0	0	1	0	1	1	1							
STRT T	Start count for timer.		0	1	0	1	0	1	0	1	1	1							
Miscellaneous																			
NOP		No operation performed.	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