

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

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 \times 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.

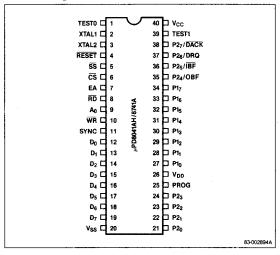
Features

- □ Complete single chip microcomputer
 - 8-bit CPU
 - 1K × 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 M Hz
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	RESET	Reset input
5	SS	Single step input
6	CS	Chip select input
7	EA	External access input
8	RD	Read strobe input
9	Α 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	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



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.

RESET (Reset)

An active low on RESET initializes the processor. RESET is also used for PROM programming, verification, and power-down.

SS (Single Step)

An active low on 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.

RD (Read)

RD will pulse low when the processor reads data and status words from the data bus buffer or status register.

WR (Write)

WR will pulse low when the processor writes data or status words to the data bus buffer or status register.

Dn-D7 (Data Bus)

 D_0 – D_7 is a three-state, bidirectional data bus. D_0 – D_7 interfaces the μ PD8041AH/8741A to the 8-bit master system's data bus.

P10-P17 (Port 1)

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

P2n-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 μ PD82C43 I/O port expander. P2₄–P2₇ can be used as port lines or interrupt requests (IBF and OBF) and DMA handshake signals (DRQ and DACK).

PROG (Program Pulse)

PROG is used in programming the μ PD8041AH/8741A. PROG is also used as an output pulse during a fetch when interfacing with the μ PD82C43 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 μ PD8041AH.

VDD (Programming Supply Voltage)

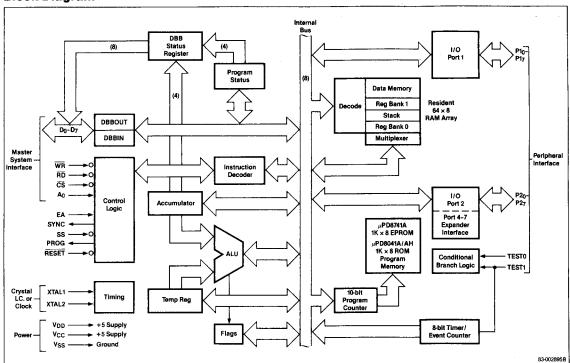
 V_{DD} is the programming supply voltage for programming the $\mu PD8741AH$. It is $+5\,V$ for normal operation of the $\mu PD8041AH/8741A$. V_{DD} is also the low power standby input for the ROM version.

Vss (Ground)

VSS is ground potential.



Block Diagram



Absolute Maximum Ratings

 $T_{\Lambda} = 25$ °C

1A - 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 ₀	-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			Test
Parameter	Symbol	Min	Тур	Max	Unit	Conditions
Input capacitance	C ₁			10	pF	
Output capacitance	C ₁₀			20	ρF	



DC Characteristics

 $T_{A} = 0 \text{ °C to } + 70 \text{ °C}, V_{CC} = V_{DD} = +5 \text{ V} \pm 10 \text{ %}; \\ \mu \text{PD8041AH: } V_{DD} = +5 \text{ V} \pm 5 \text{ %}; \\ \mu \text{PD8741A: } V_{SS} = 0 \text{ V}$

			Lin	nits			
		μPD8	741A	µPD80	41AH		Test
Parameter	Symbol	Min	Max	Min	Max	Unit	Conditions
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	٧	X1, X2, RESET
Input voltage high	V _{IH}	2.0	V _{CC}	2.0	V _{CC}	٧	Except X1, X2, and RESET
	V _{1H1}	3.8	V _{CC}	3.8	V _{CC}	٧	X1, X2, RESET
Output voltage low	V _{OL}		0.45		0.45	٧	D_0-D_7 , SYNC, $I_{0i} = 2.0 \text{ mA}$
output voltage ion	V _{OL1}		0.45		0.45	٧	Except PROG, I _{OL} = 1.0 mA
	V _{OL2}		0.45		0.45		PROG, $I_{OL} = 1.0 \text{ mA}$
Output voltage high	V _{OH}	2.4		2.4		٧	$D_0 - D_7$, $I_{0H} = -400 \mu\text{A}$
outpot tottage mg.	V _{OH1}	2.4		2.4		V	All other outputs: $1_{OH} = -50 \mu A$
Input current low	lu		0.5		0.5	mA	P1 ₀ -P1 ₇ , P2 ₀ -P2 ₇ : V _{IL} = 0.8 V
mpat barront ion	I _{LI1}		0.2		0.2	mA	\overline{SS} , \overline{RESET} ; $V_{IL} = 0.8 \text{ V}$
Input leakage current	IIL		±10		±10	μΑ	T0, T1, \overline{RD} , \overline{WR} , \overline{CS} , EA, A ₀ , $V_{SS} \le V_{IN} \le V_{CC}$
Output leakage current	I _{OL}		± 10		±10	μА	D_0 - D_7 , High Z state, V_{SS} +0.45 V \leq V _{IN} \leq V _{CC}
Supply current (total)	Ipp		15		15	mA	V _{DD}
	1DD +1CC		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

			Lin	nits			
		μ PD 8	741A	µPD80	41AH		Test
Parameter	Symbol	Min	Max	Min	Max	Unit	Conditions
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	μ PD8041A / 8741A: $C_L = 150 \text{ pF}$ μ PD8041AH: $C_L = 100 \text{ pF}$
RD ↓ to data out delay	t _{RD}		200	-	130	ns	μ PD8041A / 8741A: C_L = 150 pF μ PD8041AH: C_L = 100 pF
RD † to data float delay	t _{DF}		140		85		
Cycle time	tcy	2.5	15	1.36	15	ns	



AC Characteristics (cont) $T_A=0\,^{\circ}\text{C to }+70\,^{\circ}\text{C},\,V_{CC}=V_{DD}=+5\,\text{V}\pm10\,^{\circ}\text{V}_{SS}=0\,\text{V}$

DBB Write

			Lir	nits			
		μ PD 8	741A	µPD80)41AH		Test
Parameter	Symbol	Min	Max	Min	Max	Unit	Conditions
CS. A ₀ setup to WR ↓	t _{AW}	Ö		0		ns	
CS, A ₀ hold after WR ↑	t _{WA}	0		0		ns	
WR pulse width	t _{WW}	250		160		ns	μPD8041A / 8741A: t _{CY} = 2.5 μs
Data setup to WR ↑	t _{DW}	150		130		ns	
Data hold after WR ↑	two	0		0		ns	

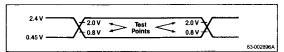
Port 2

			Lin	nits			
		µPD8	741A	μ PD 80	041AH		Test
Parameter	Symbol	Min	Max	Min	Max	Unit	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	μ PD8041AH: C _L = 20 pF
Input data setup to PROG ↓	t _{PR}		810		650	ns	μPD8041AH: C _L = 80 pF
Input data hold time	tpF	0	150	0	150	ns	μ PD8041AH: 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	tpp	1200		700		ns	

DMA

			Lin	nits			
		μPD8	3741A	μ PD 8	041AH		Test
Parameter	Symbol	Min	Max	Min	Max	Unit	Conditions
DACK setup time to	tacc	0		0		ns	
DACK hold time after RD, WR	†CAC	0		0		ns	
Data output delay after DACK	t _{ACD}		225		130	ns	μ PD8041A / 8741A: C _L = 150 pF
DRQ clear delay time after RD, WR	t _{CRQ}		200		130	ns	μPD8041AH: C _L = 100 pF

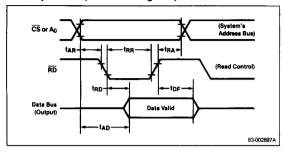
AC Timing Test Points



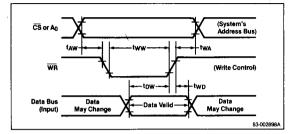


Timing Waveforms

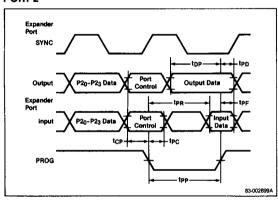
Read Operation (DBBOUT Register)



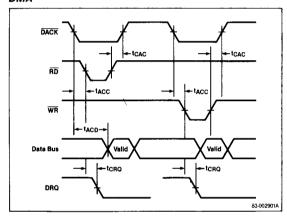
Write Operation (DBBIN Register)



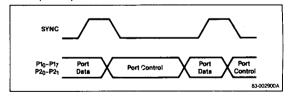
PORT 2



DMA



PORT(EA = 1)





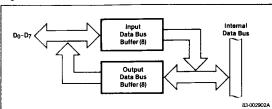
Functional Description

Two data bus buffers, an 8-bit status register, the $\overline{\text{RD}}$ and $\overline{\text{WR}}$ inputs, and expandable I/O lines enhance the $\mu\text{PD8041AH/8741A}$. 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.

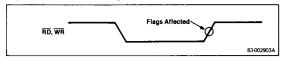
Figure 2. Status Register Format

	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

RD and 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. RD and 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 $\overline{\text{IBF}}$ 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 μ PD8041AH/8741A.

An EN FLAGS instruction execution also enables P25 to indicate that the μ PD8041AH/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.

 $P2_6$ 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 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	on Set															Flags			
			Ì			Perati	Operation Code			1				4	C N	ĭ	Į.	ORF	STST.
Mnemonic	Operand	Operation	5	ő	õ	4	ŝ	2	ā	8	Cycles	200	اد	-	2			1	
Accumulator										1	,	,	.						
ADD	A, # data	(A) ← (A) + data	o 4	၀ ဗိ	ဝမ်	0 \$	o &	၁ မိ	- 5	- 응	7	7	,						
ADD	A, Rr	$(A) \leftarrow (A) + (Rr)$ r = 0-7	0	-	-	0	-	_	_	_	-	-	•	ŀ					
ADD	A, @ Rr	(A) \leftarrow (A) + ((Rr)) r = 0-1	0	-	-	0	0	0	0	_	-	-	•						
ADDC	A, # data	(A) (A) + (C) + data	0 6	0 %	0 %	- 4	0 &	0 5	- 5	- 6	2	2	•						
ADDC	A, Br	(A) \leftarrow (A) + (C) + (Rr) r = 0-7	0	-	-	-	-	_	-	_	-	-	•						
ADDC	A, @ Rr	(A) \leftarrow (A) + (C) + ((Rr)) r = 0-1	0	-	-	-	0	0	0	-	-	-	•		ŀ				
ANL	A, # data	(A) ← (A) AND data	9	- %	o ç	d ₄	ဝန္	o &	- £	- 응	2	5		ļ				İ	
ANL	A, Rr	(A) \leftarrow (A) AND (Rr) $r = 0-7$	0	-	0	-	-	-	-	-	-	-	ļ					ļ	
ANL	A, @ Rr	(A) (A) AND ((Rr)) r = 0-1	0	-	0	-	0	0	0	-	-	-	ŀ		ŀ				
la	A	(A) ← NOT (A)	0	0	-	-	0	-	-	-	-	-							į
CI B	A	(A) ← 0	0	0	-	0	0	-	-	-	-	-				ŀ			
DA	A		0	-	0		0	-	-	-	-	- ,	•						
220	A	(A) ← (A) – 1	0	0	٥		0	-	-	- -	-	- ,							
INC	A	(A) ← (A) +1	0	0	0	- 1]	- 1		- -	-	-							
ORL.	A, # data	(A) (A) OR data	0 4p	- გ	0 \$	0 4	၀ ဗိ	o &	- 원	- 6	2	2							
ORL	A, Rr	(A) (A) 0R (Rr) r = 0-7	0	-	0	0	-	-	-	-	-	-							
ORL	A, @ Rr	(A) ← (A) OR ((Rr)) r = 0−1	0	-	0	0	0	0	0	-	-	-		ł					
균	A	$(AN + 1) \leftarrow (AN); N = 0-6$ $(A_0) \leftarrow (A_7)$	-	-		0	0	-	-	-	-	-		ļ					

NEC

Instruct	Instruction Set (cont)																	
					ဝီ	Operation Code	Code								Flags	_		
Mnemonic	Operand	Operation	7	ے	Š	2	_ 	20	6	D7 D6 D5 D4 D3 D2 D1 D0 Cycles Bytes C AC F0 F1 IBF OBF ST4-ST7	Bytes	ပ	ş	<u>د</u>	E	9	3F ST4	-ST,
Accumulator (cont)	cont)																	
RLC	∢	$(AN + 1) \leftarrow (AN)$; $N = 0-6$ $(A_0) \leftarrow (C)$ $(C) \leftarrow (A_7)$	-	-	-	-	0	<u>.</u>	_	-	-	•						
RR	A	$(AN) \leftarrow (AN + 1); N = 0-6$ $(A7) \leftarrow (A_0)$	0	-	-	1 0 1	0	-	_	-	-							
RRC	A	$(AN) \leftarrow (AN + 1); N = 0-6$ $(A_7) \leftarrow (C)$ $(C) \leftarrow (A_0)$	0	-	-	0 0	0		_	-	-	•						
SWAP	A	$(A_4-A_7) \leftrightarrow (A_0-A_3)$	0	-	0	0	0	-		-	-							
XRL	A, # data	(A) ← (A) XOR data	- 4	1 1 0 d ₇ d ₆ d ₅	1	- 4	0 0 d ₃ d ₂	0 d ₂ d ₁	- 8	2	2							
XRL	A, Rr	(A) ← (A) XOR (Rr) r = 0-7	-	-	0	-	-	_	_	-	,							
XBL	A, @ Rr	(A) ← (A) XOR ((Rr)) r = 0-1	-	1 0 1	0	1 0	0	0	_	-	-							

Branch Operation Operat	+- addr addr if B _b = 1 addr if B _b = 0 addr if C = 0 addr if C = 0 addr if F0 = 1 addr if F0 = 1 addr if F0 = 0 addr if F1 = 1 		Ds D		92 - 1 - 2 - 3 - 3 - 3 - 3 - 3 - 3 - 3 - 3 - 3	4 - 8 - 8 - 8 -	1 1	Cycles 2	Bytes	S	AC FO	F1 IBF 08F	F 514-517
Ri, addr (Rr) \leftarrow (Rr) - 1; r = 0-7 1 1 1 1 (Rr) \neq 0; (Pc ₀ -Pc ₇) \leftarrow addr if B ₀ = 1 b ₂ b ddr (Pc ₀ -Pc ₇) \leftarrow addr if B ₀ = 0 a ₇ a addr (Pc ₀ -Pc ₇) \leftarrow addr if C = 0 a ₇ a addr (Pc ₀ -Pc ₇) \leftarrow addr if F ₀ = 1 1 (Pc ₀ -Pc ₇) \leftarrow addr if F ₀ = 1 1 (Pc ₀ -Pc ₇) \leftarrow addr if F ₀ = 1 1 (Pc ₀ -Pc ₇) \leftarrow addr if F ₀ = 1 a ₇ a addr (Pc ₀ -Pc ₇) \leftarrow addr if F ₀ = 0 a ₇ a addr (Pc ₀ -Pc ₇) \leftarrow addr if F ₀ = 1 a ₇ a addr (Pc ₀ -Pc ₇) \leftarrow addr if C = 0 a ₇ a addr (Pc ₀ -Pc ₇) \leftarrow addr if C = 0 a ₇ a addr (Pc ₀ -Pc ₇) \leftarrow addr if C = 0 a ₇ a addr (Pc ₀ -Pc ₇) \leftarrow addr if C = 0 a ₇ a addr (Pc ₀ -Pc ₇) \leftarrow addr if C = 0 a ₇ a addr (Pc ₀ -Pc ₇) \leftarrow addr if OBF = 1 a ₇ a addr (Pc ₀ -Pc ₇) \leftarrow addr if OBF = 1 a ₇ a addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 1 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 1 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 1 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr if T ₀ = 0 a ₇ addr (Pc ₀ -Pc ₇) \leftarrow addr if T ₀ = 0 a ₇ addr if T ₀ = 0 a ₇ addr if T ₀ = 0 a ₇ addr if T ₀	+- addr if B _b = 1 +- addr if B _b = 1 +- addr if B _b = 0 +- addr if C = 1 +- addr if C = 1 +- addr if C = 1 +- addr if C = 0 +- addr if C = 1 +- addr if F = 0 addr if F = 0 addr if C = 0 +- addr if C = 0 addr if C = 0 (200 + 2 if BF = 1 (200 + 2 if BF = 1 addr if OBF = 1					- 8 - 8 -	_ %	2					
Riv. addr (Ri) + (Ri) - 1; r = 0-7	r) - 1; r = 0-7 +- addr if B _b = 1 +- addr if B _b = 1 c) + 2 if B _b = 0 +- addr if C = 1 c) + 2 if C = 0 +- addr if F = 1 c) + 2 if F = 0 y) + (addrg-addry) DBF y + (addrg-addr) y + (addrg-addr) y + (addrif E = 1 c) + 2 if G = 1 y + 2 if C = 1 y + 2 if C = 1 y + 2 if G = 1					- 8 - 8 -	~ %	7	•				
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	+- addr if B _b = 1 +- addr if B _b = 1 +- addr if C = 1 addr if C = 1 addr if C = 1 addr if F = 0 addr if F = 1 					8 -8 -8 -	ଟ		2				
addr $(PC_0 - PC_7) \leftarrow addr \ if B_b = 1$ P_2 P_3 P_4 P_4 P_4 P_5 P_4 P_5 P_5 P_5 P_6 P_5 P_6	(9)					- 8 - 8 -							
addr $(PC_0 - PC_7) \leftarrow addr \ if C = 1$ addr $(PC_0 - PC_7) \leftarrow addr \ if C = 1$ $(PC_0 - PC_7) \leftarrow addr \ if C = 0$ addr $(PC_0 - PC_7) \leftarrow addr \ if F = 1$ $(PC_0 - PC_7) \leftarrow addr \ if F = 0$ addr $(PC_0 - PC_7) \leftarrow addr \ if F = 0$ $(PC_0 - PC_7) \leftarrow addr \ if F = 0$ addr $(PC_0 - PC_7) \leftarrow addr \ if F = 0$ $(PC_0 - PC_7) \leftarrow addr \ if C = 0$ $(PC_0 - PC_7) \leftarrow addr \ if C = 0$ $(PC_0 - PC_7) \leftarrow addr \ if C = 0$ $(PC_0 - PC_7) \leftarrow addr \ if C = 0$ $(PC_0 - PC_7) \leftarrow addr \ if C = 0$ $(PC_0 - PC_7) \leftarrow addr \ if C = 0$ $(PC_0 - PC_7) \leftarrow addr \ if C = 0$ $(PC_0 - PC_7) \leftarrow addr \ if C = 1$ $(PC_0 - PC_7) \leftarrow addr \ if C = 0$ $(PC_0 - PC_1) \leftarrow addr \ if C = 0$ $(PC_0 - PC_1) \leftarrow addr \ if C = 0$ $(PC_0 - PC_1) \leftarrow addr \ if C = 0$ $(PC_0 - PC_1) \leftarrow addr \ i$	0 0 1 0					8-8-	0	2	2				
addr $(PC_0-PC_7) \leftarrow addr \ if C = 1$ 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	0 0 1 0					- & -	ଚ୍ଚ						
addr $(PC) \leftarrow (PC) + 2 \text{ if } C = 0$ a_7	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1					æ -	0	2	2				
addr $(PC_0-PC_7) \leftarrow addr \ if F0 = 1$ 1 ($PC_0 \rightarrow PC_1 \rightarrow PC_1 + 2 \ if F0 = 0$ 37 3 addr $(PC_0-PC_7) \leftarrow addr \ if F1 = 1$ 0 37 3 $(PC_0 \rightarrow PC_1) \leftarrow addr \ if F1 = 0$ 37 3 4 37 3 $(PC_0 \rightarrow PC_1) \leftarrow addr \ if F1 = 0$ 37 3 4 37 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3	100)					-	g ₀						
addr $(PC_0 - PC_7) \leftarrow addr \ if \ F1 = 1$ 0 and $(PC_0 - PC_7) \leftarrow addr \ if \ F1 = 0$ and $(PC_0 - PC_7) \leftarrow addr \ if \ F1 = 0$ and $(PC_0 - PC_7) \leftarrow (addr_0 - addr_0)$ and addr $(PC_0 - PC_7) \leftarrow (addr_0 - addr_0)$ and addr $(PC_0 - PC_7) \leftarrow (A)$ 1 addr $(PC_0 - PC_7) \leftarrow (A)$ 1 addr $(PC_0 - PC_7) \leftarrow addr \ if \ i$	(1) (1) (1) (1) (1) (1) (1) (1) (1) (1)						0	5	2				
addr $(PC_0 - PC_7) \leftarrow addr \ if F1 = 1$ 0 $PC_0 \leftarrow (PC_1) \leftarrow (PC_1) + 2 \ if F1 = 0$ 37 3 $(PC_1) \leftarrow (PC_1) \leftarrow (addr_0 - addr_0)$ 30 37 3 $(PC_1) \leftarrow (PC_1) \leftarrow (addr_0 - addr_0)$ 37 3 3 4 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4	100)				1 1	ક	30						
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	(10)				1 1 1	-	0	7	7				
addr $(PC_8 - PC_{10}) \leftarrow (addr_9 - addr_{10})$ a_{10}					1 1	ā	g ₀						
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	0 -		1 1			0	0	2	2				
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	((A)) addr if C = 0 + 2 if C = 1 addr if IBF = 0 + 2 if IBF = 1 addr if IBF = 1 + 2 if IBF = 1			1		æ	8						
addr $(PC_0 - PC_7) \leftarrow ((A))$ 1 addr $(PC_0 - PC_7) \leftarrow addr if C = 0$ 1 $(PC_0 - PC_7) \leftarrow addr if D = 0$ 1 $(PC_0 - PC_7) \leftarrow addr if BF = 0$ 1 $(PC_0 - PC_7) \leftarrow addr if BF = 0$ 1 $(PC_0 - PC_7) \leftarrow addr if DBF = 1$ 1 $(PC_0 - PC_7) \leftarrow addr if DBF = 1$ 1 $(PC_0 - PC_7) \leftarrow addr if DBF = 1$ 37 $(PC_0 - PC_7) \leftarrow addr if D = 0$ 37 $(PC_0 - PC_7) \leftarrow addr if T = 0$ 0 $(PC_0 - PC_7) \leftarrow addr if T = 0$ 0 $(PC_0 - PC_7) \leftarrow addr if T = 0$ 37 addr $(PC_0 - PC_7) \leftarrow addr if T = 0$ 37 $(PC_0 - PC_7) \leftarrow addr if T = 1$ 37 addr $(PC_0 - PC_7) \leftarrow addr if T = 1$ 37 $(PC_0 - PC_7) \leftarrow addr if T = 1$ 37 $(PC_0 - PC_7) \leftarrow addr if T = 1$ 37	0 -		1		ļ					ŀ			
addr $(PC_0 - PC_7) \leftarrow addr \ if C = 1$ a_7 $addr$ $(PC_0 + 2 \ if C = 1)$ a_7 $addr$ $(PC_0 - PC_7) \leftarrow addr \ if \ i$	0 -					-	-	2	-				
addr $(PC) \leftarrow (PC) + 2 \text{ if } C = 1$ ay addr $(PC) \leftarrow (PC) + 2 \text{ if } BF = 0$ 1 $(PC) \leftarrow (PC) + 2 \text{ if } 1BF = 1$ ay addr $(PC) \leftarrow (PC) + 2 \text{ if } 1BF = 1$ 1 $(PC) \leftarrow (PC) + 2 \text{ if } 1BF = 1$ 1 $(PC) \leftarrow (PC) + 2 \text{ if } 0BF = 1$ ay addr $(PC) \leftarrow (PC) + 2 \text{ if } 0BF = 0$ ay $(PC) \leftarrow (PC) + 2 \text{ if } 0BF = 0$ ay addr $(PC) \leftarrow (PC) + 2 \text{ if } 1T = 1$ ay addr $(PC) \leftarrow (PC) + 2 \text{ if } 1T = 1$ ay addr $(PC) \leftarrow (PC) + 2 \text{ if } 1T = 1$ ay addr $(PC) \leftarrow (PC) + 2 \text{ if } 1T = 1$ ay addr $(PC) \leftarrow (PC) + 2 \text{ if } 1F = 1$ ay addr $(PC) \leftarrow (PC) + 2 \text{ if } 1F = 1$ ay addr $(PC) \leftarrow (PC) + 2 \text{ if } 1F = 1$ ay addr $(PC) \leftarrow (PC) + 2 \text{ if } 1F = 0$ ay	0 - 0					-	0	2	2				
addr $(PC_0 - PC_7) \leftarrow addr \text{ if } \text{IBF} = 0$ 1 $(PC_0 - PC_7) \leftarrow addr \text{ if } \text{IBF} = 1$ 37 6 $(PC_0 - PC_7) \leftarrow addr \text{ if } \text{OBF} = 1$ 1 $(PC_0 - PC_7) \leftarrow addr \text{ if } \text{OBF} = 0$ 37 8 $(PC_0 - PC_7) \leftarrow addr \text{ if } \text{IO} = 0$ 0 $(PC_0 - PC_7) \leftarrow addr \text{ if } \text{IO} = 0$ 0 $(PC_0 - PC_7) \leftarrow addr \text{ if } \text{IO} = 1$ 37 addr $(PC_0 - PC_7) \leftarrow addr \text{ if } \text{II} = 0$ 0 $(PC_0 - PC_7) \leftarrow addr \text{ if } \text{II} = 1$ 37 (PC_0 + 2 if $\text{II} = 1$ 37 addr $(PC_0 - PC_7) \leftarrow addr \text{ if } \text{II} = 1$ 37 (PC_0 + 2 if $\text{II} = 1$ 37 addr $(PC_0 - PC_7) \leftarrow addr \text{ if } \text{IF} = 1$ 37 addr $(PC_0 - PC_7) \leftarrow addr \text{ if } \text{IF} = 1$ 37 addr $(PC_0 - PC_7) \leftarrow addr \text{ if } \text{IF} = 1$ 37 addr $(PC_0 - PC_7) \leftarrow addr \text{ if } \text{IF} = 1$ 37	0 1.0			4 a3	a ₂		a ₀						
addr (PC) \leftarrow (PC) + 2 if iBF = 1 a 7 addr (PC) \leftarrow (PC) + 2 if iBF = 1 1 (PC) \leftarrow (PC) + 2 if iBF = 0 a 7 addr (PC) \leftarrow (PC) + 2 if iBF = 0 a 7 addr (PC) \leftarrow (PC) + 2 if iD = 1 a 7 addr (PC) \leftarrow (PC) + 2 if iD = 1 a 7 addr (PC) \leftarrow (PC) + 2 if iD = 1 a 7 addr (PC) \leftarrow (PC) + 2 if iD = 1 a 7 addr (PC) \leftarrow (PC) + 2 if iD = 1 a 7 addr (PC) \leftarrow (PC) + 2 if iD = 1 a 7 addr (PC) \leftarrow (PC) + 2 if iD = 1 a 7 addr (PC) \leftarrow (PC) + 2 if iD = 1 a 7 addr (PC) \leftarrow (PC) + 2 if iD = 1 a 7 addr (PC) \leftarrow (PC) + 2 if iD = 0 a 7 and 7 and 7 and 7 and 7 addr (PC) \leftarrow (PC) + 2 if iD = 0 a 7 and 7 and 7 and 7 addr (PC) \leftarrow (PC) + 2 if iD = 0 a 7 and 7 an	- 0	-		0	-	-	0	2	2				
addr $(PC_0 - PC_7) \leftarrow addr if 0BF = 1$ 1 $(PC_0 - PC_7) \leftarrow addr if 0BF = 0$ 37 8 $(PC_0 - PC_7) \leftarrow addr if 10 = 0$ 0 $(PC_0 - PC_7) \leftarrow addr if 10 = 0$ 37 addr $(PC_0 - PC_7) \leftarrow addr if 11 = 0$ 37 $(PC_0 - PC_7) \leftarrow addr if 11 = 0$ 37 $(PC_0 - PC_7) \leftarrow addr if 11 = 0$ 37 addr $(PC_0 - PC_7) \leftarrow addr if A = 0$ 1 $(PC_0 - PC_7) \leftarrow addr if A = 0$ 1 $(PC_0 - PC_7) \leftarrow addr if A = 1$ 37 $(PC_0 - PC_7) \leftarrow addr if TF = 1$ 37 $(PC_0 - PC_7) \leftarrow addr if TF = 1$ 37 addr $(PC_0 - PC_7) \leftarrow addr if TF = 1$ 37 $(PC_0 - PC_7) \leftarrow addr if TF = 1$ 37 $(PC_0 - PC_7) \leftarrow addr if TF = 0$ 37 $(PC_0 \leftarrow (PC_7) \leftarrow addr if TF = 0$ 37	_	89	a5 a	a4 a3		£	g _o						
addr $(PC) \leftarrow (PC) + 2 \text{ if } 0BF = 0$ a7 addr $(PC) \leftarrow (PC) + 2 \text{ if } 0BF = 0$ b a7 addr $(PC) \leftarrow (PC) + 2 \text{ if } 10 = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } 11 = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } 11 = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } 11 = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } 1A = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and addr $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) \leftarrow (PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) \leftarrow (PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) \leftarrow (PC) \leftarrow (PC) \leftarrow (PC) + 2 \text{ if } A = 1$ and $(PC) \leftarrow (PC) $		0		0 0	-	-	0	7	2				
addr $(PC_0 - PC_7) \leftarrow addr if 10 = 0$ 0 $(PC_0 - PC_7) \leftarrow addr if 10 = 1$ a7 addr $(PC_0 - PC_7) \leftarrow addr if 11 = 0$ 0 $(PC_0 - PC_7) \leftarrow addr if 11 = 1$ a7 addr $(PC_0 - PC_7) \leftarrow addr if A = 0$ 1 $(PC_0 - PC_7) \leftarrow addr if A = 0$ 1 $(PC_0 - PC_7) \leftarrow addr if A = 1$ a7 $(PC_0 - PC_7) \leftarrow addr if A = 1$ a7 $(PC_0 - PC_7) \leftarrow addr if A = 1$ a7 addr $(PC_0 - PC_7) \leftarrow addr if A = 1$		89	a5 a				8						
addr $(PC) \leftarrow (PC) + 2 \text{ if } T0 = 1$ a7 $PC_0 - PC_7) \leftarrow addr \text{ if } T1 = 0$ $PC_0 - PC_7) \leftarrow addr \text{ if } T1 = 1$ addr $(PC_0 - PC_7) \leftarrow addr \text{ if } A = 0$ $(PC) \leftarrow (PC) + 2 \text{ if } T1 = 1$ addr $(PC_0 - PC_7) \leftarrow addr \text{ if } A = 1$ addr $(PC_0 - PC_7) \leftarrow addr \text{ if } TF = 1$ addr $(PC_0 - PC_7) \leftarrow addr \text{ if } TF = 1$ addr $(PC_0 - PC_7) \leftarrow addr \text{ if } TF = 1$ addr $(PC_0 - PC_7) \leftarrow addr \text{ if } TF = 1$ addr $(PC_0 - PC_7) \leftarrow addr \text{ if } TF = 1$ and $(PC_0 - PC_7) \leftarrow addr \text{ if } TF = 0$		0	-	0 0	-		0	7	2				
addr $(PC_0 - PC_7) \leftarrow addr \ if \ T1 = 0$ 0 $(PC) \leftarrow (PC) + 2 \ if \ T1 = 1$ a_7 addr $(PC_0 - PC_7) \leftarrow addr \ if \ A = 0$ 1 a_7 $(PC) \leftarrow (PC) + 2 \ if \ A = 1$ a_7 addr $(PC_0 - PC_7) \leftarrow addr \ if \ T = 1$ a_7 $addr$ $(PC_0 - PC_7) \leftarrow addr \ if \ T = 1$ a_7 $addr$ $(PC_0 - PC_7) \leftarrow addr \ if \ T = 1$ a_7 $addr$ $(PC_0 - PC_7) \leftarrow addr \ if \ T = 1$ a_7 $addr$ $(PC_0 - PC_7) \leftarrow addr \ if \ T = 0$ a_7		9 6				Ì	8						
$(PC) \leftarrow (PC) + 2 \text{ if } T1 = 1 \qquad a_7$ $addr \qquad (PC_0 - PC_7) \leftarrow addr \text{ if } A = 0 \qquad 1$ $(PC) \leftarrow (PC) + 2 \text{ if } A = 1 \qquad a_7$ $(PC_0 - PC_7) \leftarrow addr \text{ if } TF = 1 \qquad 0$ $(PC_0 - PC_7) \leftarrow addr \text{ if } TF = 0 \qquad a_7$	0	-	0	0 0	-	-	0	7	2				
addr $(PC_0 - PC_7) \leftarrow$ addr if $A = 0$ 1 $(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ 37 $(PC_0 - PC_7) \leftarrow$ addr if $TF = 1$ 0 $(PC_0 - PC_7) \leftarrow$ addr if $TF = 1$ 0		કુ	İ			- 1	କୃ						
$(PC) \leftarrow (PC) + 2 \text{ if } A = 1 \qquad 47$ $addr \qquad (PC_0 - PC_7) \leftarrow addr \text{ if } TF = 1 \qquad 0$ $(PC) \leftarrow (PC) + 2 \text{ if } TF = 0 \qquad 37$		ο,	۰,	- 3	- 6	- 3	o ,	2	7				
addr $(PC_0-PC_7) \leftarrow addr \ if \ F = 1 0$ $(PC) \leftarrow (PC) + 2 \ if \ FF = 0 $ a7		g.			l	1	3	c	·				
47		0 8	' ن o	ر ب ر	- 4	- 4	> &	7	7				
		90	Ì			ł	3 €	,					
170 addr $(PC_0-PC_7) \leftarrow addr \text{ if } T0 = 1 0 0$		0		1 0		Ψ,	0,	2	7.				
$(PC) \leftarrow (PC) + 2 \text{ if } T0 = 0$ a7		ge ge	.g.	a4 a	3 a2	-	ફ						
1=1	-	-	0	1 0	-	-	0	2	5				
$(PC) \leftarrow (PC) + 2 \text{ if } T1 = 0$		ge	1	34 9.	- 1	ŀ	S.						
addr if $A = 0$		-	0	0 0	- ,	- ,	0 8	5	5				
$(PC) \leftarrow (PC) + 2 \text{ if } A = 1$ a7		ae 9e		1	1		Ş						

NEC

					°	Operation Code	900						1	Flags	.		
Mnemonic	Operand	Operation	4	å	ď	4	5	2	2	Do Cycles	Bytes	C	2	E		OBF 3	ST4-ST7
Control																	
ENI	Enable the external interrupt input		0	0	0	0	0		0	-	-						
DIS	Disable the external interrupt input		0	0	0	-	0	-	0	1	-						
SEL RB0	(BS) ← 0		-	-	0	0	0	-	0	1	-						
SEL RB	(BS) ← 1		-	-	0	-	0	-	0	-	-						
EN DMA	Enable DMA handshake	(e	-	-	-	-	0	-	0	-	-						
EN FLAGS	Enable interrupt to mas device	naster	-	-	-	0	0	-	0	1	-						
Data Moves																	
MOV	A, # data	(A) ← data	0 42	o %	- გ	0 2	ဝမ္	0 q ₂	1 d ₁ d	1 2 d ₀	2						
MOV	A, Br	(A) \leftarrow (Rr); r = 0-7	-	-	-	-	-	L	_	- -	-						
MOV	A, @ Rr	(A) \leftarrow ((Rr)); r = 0-1	-	-	-	-	0	0	0	r 1	-						
MOV	A, PSW	(A) ← (PSW)	-	-	0	0	0	-	-	1	-						
MOV	Rr, # data	(Rr) ← data; r = 0-7	- 4	ဝန	- ન્ટ	- 광	– হু	- 8	- 2	r 2 do	2						
MOV	Rr, A	$(Rr) \leftarrow (A); r = 0-7$	-	0	-	0	-	L	_	٦ ٦	1						
MOV	@ Rr, A	$((Rr)) \leftarrow (A); r = 0-1$	-	0	-	0	0	0	0	-	-						
MOV	@ Rr, # data	((Rr)) \leftarrow data; r = 0-1	- 4	0 ಕಿ	- 운	- 4	0 చి	0 &	0 &	ر م	2						
MOV	PSW, A	(PSW) ← (A)	-	-	0	-	0	-	-	-	-						
MOVP	A, @ A	$(PC_0-PC_7) \leftarrow (A)$ $(A) \leftarrow ((PC))$	-	0	-	0	0	0	-	1 2	1						
MOVP3	A, @ A	$(PC_0-PC_7) \leftarrow (A)$ $(PC_8-PC_{10}) \leftarrow 011$ $(A) \leftarrow ((PC))$	-	-	-	0	0	0		1 2	-						
XCH	A, Rr	(A) ↔ (Rr); r = 0-7	0	0	1	0	-	ŗ	1	r 1	-						
XCH	A, @ Rr	(A) \leftrightarrow ((Rr)); $r = 0-1$	0	0	-	0	0	0	0	r 1	-						
хснр	A, @ Br	$(A_0-A_3) \leftrightarrow ((Rr))_0-((Rr))_3;$ r = 0-1	0	0	-	-	0	0	0	+	-						



Instruction Set (cont)

						Operat	Operation Code									Flags	5		
Mnemonic	Operand	Operation	4	٥	č	ð	ద్	2	ō	മ്	Cycles	Bytes	ပ	¥¢	2	E	100	0BF	ST4-ST7
Flags																			
CPL C		(C) NOT (C)	-	0	-	0	0	-	-	-	-	٠	•						
CPL F0		(F0) NOT (F0)	-	0	0	-	0	-	0	-	-	-			•				
CPL F1		(F1) ← NOT (F1)	-	0	-	-	0	-	0	-	-	-				•			
CLRC		0 - (0)	-	0	0	-	0	-	-	-	-	-	•						
CLR F0		(F0) ← 0	-	0	0	0	0	-	0	-	_	-			•				
CLR F1		(F1) 0	-	0	-	0	0	-	0	-	-	-				•			
MOV STS, A		ST4-ST7 A4-A7	-	0	0	-	0	0	0	0	-	+							
Input / Output																			
ANL	Pp, # data	(Pp) \leftarrow (Pp) AND data p = 1-2	t 6	0 %	0 %	- q	†	o &	o 2	a op	2	2							Ì
ANLD	Pp, A	(Pp) \leftarrow (Pp) AND (A ₀ -A ₃); p = 4-7	-	0	0	-	-	-	م	۵	2	-							
Z	A, Pp	(A) \leftarrow (Pp); p = 1-2	0	0	0	0	-	0	a	۵	2	-							
<u>N</u>	A, DBB	(A) ← (DBB)	0	0	-	0	0	0	-	0	-	-					•		
MOVD	A, Pp	$(A_0-A_3) \leftarrow (Pp); p = 4-7$ $(A_4-A_7) \leftarrow 0$	0	0	0	0	-	-	ď	d	2	•							
MOVD	Pp, A	$(Pp) \leftarrow (A_0 - A_3); p = 4-7$	0	0	-	-	-	-	۵	۵	-	-							
ORLD	Pp, A	(Pp) \leftarrow (Pp) OR (A ₀ -A ₃); p = 4-7	-	0	0	0		-	۵	۵	-	-							
ORL	Pp, # data	(Pp) ← (Pp) 0R data p = 1-2	1 4	0 %	0 &	o \$	~ చ్	90	o 2.	م ئ	2	2							
00T	DBB, A	(DBB) ← (A)	0	0	0	0	0	0	-	0	-	-							
OUTL	Pp, A	$(Pp) \leftarrow (A); p = 1-2$	0		-	-	-	0	۵	۵	-	-							
Registers																			
DEC	Rr (Rr)	$(Rr) \leftarrow (Rr) - 1; r = 0-7$	-	-	0	0	-	L	٦	_	1	+							
INC	Rr	$(Rr) \leftarrow (Rr) + 1; r = 0-7$	0	0	0	1	-	٦	ı	٦	1	1							
INC	@ Rr	$((Rr)) \leftarrow ((Rr)) + 1;$ r = 0-1	0	0	0	-	0	0	0	_	-	-							

NEC

Instruction Set (cont)

					ō	peratk	Operation Code									Flags	_		
Mnemonic	Operand	Operation	ō	۵	õ	4	ے	D2	ā	ت 6	Cycles	Bytes	ပ	Ş	2	E	出	OBF 9	ST4-ST7
Subroutine		The state of the s					1												
CALL	addr	((SP)) ← (PC),	a ₁₀	å	ag B	-	0	-	0	0	2	2							
		(PSW4-PSW7),	47	ge 9e	a 5	4	g 3	3 2	P.	a ₀									
		$(SP) \leftarrow (SP) + 1$																	
		(PC ₈ -PC ₁₀) - (addr ₈ -addr ₁₀)																	
		(PC_0-PC_7) — $(addr_0-addr_7)$																	
		$(PC_{11}) \leftarrow DBF$																	
RET		(SP) ← (SP) = 1	-	0	0	0	0	0	-		2	-							
		(PC) ← ((SP))																	
RETR		(SP) ← (SP) = 1	-	0	6	-	c	6	_	-	2	-							
		$(PC) \leftarrow ((SP))$,		,	,											
		$(PSW_4-PSW_7) \leftarrow ((SP))$																	
Timer / Counter																			
EN TCNTI	Enable internal interrupt		0	0	-	-		-	0	_	-	-							
	flag for timer / counter																		
	output.																		
DIS TCNTI	Disable internal interrupt		0	0	-	-	0	-	0	-	-	-							
	flag for timer / counter output																		
MOVAT	(A) ← (I)		-	-		-	-	-	-		-	-							
	(.) (.)		,	-	,	,	,	,	-		-	-							
MOV T, A	(T) ← (A)		0	•	-	0	0	0		0	_	-							
STOP TCNT	Stop count for event		0	-	-	0	0	-	0	_	-	-							
	counter.																		
STRT CNT	Start count for event																		
	counter.		0	-	0	0	0	-	0	_	_	-							
STRT T	Start count for timer.		0	-	0	-	0	_	0	_	_	-							
Miscellaneous																			
NOP		No operation performed.	0	0	0	0	0	0	0		-	-							
Made													l						

(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
Α	Accumulator
AC	Auxiliary carry flag
addr	Program memory address (12 bits)
B _b	Bit designator (b = 0-7)
BS	Bank switch
BUS	Bus port
С	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
l	Interrupt
Р	In-page operation designator
IBF	Input buffer full flag
Рр	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
Х	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