To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: “Standard”, “High Quality”, and “Specific”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as “Specific” without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as “Specific” or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is “Standard” unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   “Specific”: Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

USER'S MANUAL

RENESAS

Phase-out/Discontinued

AS17005 DEVICE FILE

PC-9800 SERIES(MS-DOS™)BASED
IBM-PC™(PC DOS™)BASED

Version 2

Document No. EEU−1344A
(O. D. No. EEU−725A)
Date Published August 1991 P
Printed in Japan

USER'S MANUAL

NEC

Phase-out/Discontinued

AS17005 DEVICE FILE

PC-9800 SERIES(MS-DOS)BASED
IBM-PC(PC DOS)BASED

Version 2

Phase-out/Discontinued

PREFACE

The AS17005 Device File is used together with the AS17K assembler to enable μPD17005 program.

During assembly, the AS17005 provides μPD17005 program memory capacitance, data memory capacitance, usable instructions, reserved symbols, etc.

The AS17005 (V2) is supplied with both IRQ.MAC (V2.1) and IFCSET.LIB files.

This manual describes the above target device and the files to be supplied.

For the operation procedure for the AS17K assembler and the AS17005 Device File, please refer to the AS17K User's Manual (EEU-1287).

CONTENTS

CHAPTER 1. DEVICE INFORMATION

The AS17005 Device File provides the following μPD17005-related information during assembly.

1.1 PROGRAM MEMORY (ROM) CAPACITY

7932 steps (1EFBH)

1.2 DATA MEMORY (RAM) CAPACITY

BANK0 to BANK3

1.3 USABLE INSTRUCTIONS

See **Chapter 2.**

1.4 REGISTER FILE, PORT REGISTER AND PERIPHERAL REGISTER READ/WRITE INFORMATION

See **Chapter 3.**

1.5 RESERVED SYMBOLS

See **Chapter 3.**

CHAPTER 2. μPD17005 INSTRUCTIONS

2.1 INSTRUCTION SET

b15 b14 b13 b12 b11		0		1	
0 0 0 0	0	ADD	r, m	ADD	m, #i
0 0 0 1	1	SUB	r, m	SUB	m, #i
0 0 1 0	2	ADDC	r, m	ADDC	m, #i
0 0 1 1	3	SUBC	r, m	SUBC	m, #i
0 1 0 0	4	AND	r, m	AND	m, #i
0 1 0 1	5	XOR	r, m	XOR	m, #i
0 1 1 0	6	OR	r, m	OR	m, #i
0 1 1 1	7	INC INC MOVT BR CALL RET RETSK EI DI RETI PUSH POP GET PUT PEEK POKE RORC STOP HALT NOP	AR IX DBF, @AR @AR @AR AR AR DBF, p p, DBF WR, rf rf, WR r 0 h 		
1 0 0 0	8	LD	r, m	ST	m, r
1 0 0 1	9	SKE	m, #i	SKGE	m, #i
1 0 1 0	A	MOV	@r, m	MOV	m, @r
1 0 1 1	B	SKNE	m, #i	SKLT	m, #i
1 1 0 0	C	BR	addr (page0)	CALL	addr (page0)
1 1 0 1	D	BR	addr (page1)	MOV	m, #i
1 1 1 0	E	BR	addr (page2)	SKT	m, #n
1 1 1 1	F	BR	addr (page3)	SKF	m, #n

2.2 LIST OF INSTRUCTIONS

Legend

M	:	Data memory address
m	:	Data memory address excluding bank
m_H	:	Data memory row address
m_L	:	Data memory column address
R	:	General register address
r	:	General register column address
RP	:	General register pointer
RF	:	Register file
rf	:	Register file address
rf_H	:	Register file address (high-order 3 bits)
rf_L	:	Register file address (low-order 3 bits)
AR	:	Address register
IX	:	Index register
IXE	:	Index enable flag
DBF	:	Data buffer
WR	:	Window register
MP	:	Data memory row address pointer
MPE	:	Memory pointer enable flag
PE	:	Peripheral register
P	:	Peripheral address
P_H	:	Peripheral address (high-order 3 bits)
P_L	:	Peripheral address (low-order 4 bits)
PC	:	Program memory counter
SP	:	Stack pointer
STACK	:	Stack value indicated by stack pointer
$STACK_{PC}$	:	Program counter value indicated by stack pointer
BANK	:	Bank register
$(ROM)_{PC}$	:	Program memory data indicated by program memory counter
INTEF	:	Interrupt enable flag
i	:	Immediate data (4 bits)
n	:	Bit position (4 bits)
addr	:	Program memory address (11 bits)
PAGE	:	High-order 2 bits of program counter
CY	:	Carry flag
h	:	Halt release condition
[ ]	:	Data memory or register address
( )	:	Data memory or register value

Instruction Group	Mnemonic	Operand	Operation	Machine Code			
				Op Code			
Add instruction	ADD	r, m	(R) ← (R) + (M)	00000	m_H	m_L	r
		m, #i	(M) ← (M) + i	10000	m_H	m_L	i
	ADDC	r, m	(R) ← (R) + (M) + (CY)	00010	m_H	m_L	r
		m, #i	(M) ← (M) + i + (CY)	10010	m_H	m_L	i
	INC	AR	(AR) ← (AR) + 1	00111	000	1001	0000
		IX	(IX) ← (IX) + 1	00111	000	1000	0000
Subtract instruction	SUB	r, m	(R) ← (R) − (M)	00001	m_H	m_L	r
		m, #i	(M) ← (M) − i	10001	m_H	m_L	i
	SUBC	r, m	(R) ← (R) − (M) − (CY)	00011	m_H	m_L	r
		m, #i	(M) ← (M) − i − (CY)	10011	m_H	m_L	i
Compare instruction	SKE	m, #i	(M) − i, skip if zero	01001	m_H	m_L	i
	SKGE	m, #i	(M) − i, skip if not borrow	11001	m_H	m_L	i
	SKLT	m, #i	(M) − i, skip if borrow	11011	m_H	m_L	i
	SKNE	m, #i	(M) − i, skip if not zero	01011	m_H	m_L	i
Logical operation instruction	AND	m, #i	(M) ← (M) AND i	10100	m_H	m_L	i
		r, m	(R) ← (R) AND (M)	00100	m_H	m_L	r
	OR	m, #i	(M) ← (M) OR i	10110	m_H	m_L	i
		r, m	(R) ← (R) OR (M)	00110	m_H	m_L	r
	XOR	m, #i	(M) ← (M) XOR i	10101	m_H	m_L	i
		r, m	(R) ← (R) XOR (M)	00101	m_H	m_L	r
Transfer instruction	LD	r, m	(R) ← (M)	01000	m_H	m_L	r
	ST	m, r	(M) ← (R)	11000	m_H	m_L	r

	MOV	@r, m	if MPE= 1 : [(MP), (R)] ← (M)
			if MPE= 0 : [(m_H), (R)] ← (M)
				01010	m_H	m_L	r

		m, @r	if MPE= 1 : (M) ← [(MP), (R)]
			if MPE= 0 : (M) ← [(m_H), (R)]
				11010	m_H	m_L	r

		m, #i	(M) ← i	11101	m_H	m_L	i
	MOVT	DBF, @AR	(STACK_{PC}) ← (PC) , (PC) ← (AR) , (DBF) ← $(\text{ROM})_{PC}$, (PC) ← (STACK_{PC})	00111	000	0001	0000
	PUSH	AR	(SP) ← (SP) − 1, (STACK_{PC}) ← (AR)	00111	000	1101	0000
	POP	AR	(AR) ← (STACK_{PC}), (SP) ← (SP) + 1	00111	000	1100	0000
	PEEK	WR, rf	(WR) ← (RF)	00111	rf_H	0011	rf_L
	POKE	rf, WR	(RF) ← (WR)	00111	rf_H	0010	rf_L
	GET	DBF, p	(DBF) ← (PE)	00111	p_H	1011	p_L
	PUT	p, DBF	(PE) ← (DBF)	00111	p_H	1010	p_L
Decision instruction	SKT	m, #n	if $(M)_n$=all " 1 ", then skip	11110	m_H	m_L	n
	SKF	m, #n	if $(M)_n$=all " 0 ", then skip	11111	m_H	m_L	n

Instruction Group	Mnemonic	Operand	Operation	Machine Code			
				Op Code			
Branch instruction	BR	addr	(PC) ← addr, PAGE ← 0	01100	addr (11 bits)		
			(PC) ← addr, PAGE ← 1	01101			
			(PC) ← addr, PAGE ← 2	01110			
			(PC) ← addr, PAGE ← 3	01111			
		@AR	(PC) ← (AR)	00111	000	0100	0000
Shift	RORC	r	(CY) ← → $(R)_{b3}$ → $(R)_{b2}$ → $(R)_{b1}$ → $(R)_{b0}$	00111	000	0111	r
Subroutine instruction	CALL	addr	(SP) ← (SP) − 1, ($STACK_{PC}$) ← ((PC) + 1), (PC) $_{\#11}$ ← 0, (PC) ← addr	11100	addr (11 bits)		
		@AR	(SP) ← (SP) − 1, ($STACK_{PC}$) ← ((PC) + 1), (PC) ← (AR)	00111	000	0101	0000
	RET		(PC) ← ($STACK_{PC}$), (SP) ← (SP} + 1	00111	000	1110	0000
	RETSK		(PC) ← ($STACK_{PC}$), (SP) ← (SP) + 1, skip	00111	001	1110	0000
	RETI		(PC), (BANK), (IXE) ← (STACK), (SP) ← (SP) + 1	00111	100	1110	0000
Interrupt	EI		INTEF ← 1	00111	000	1111	0000
	DI		INTEF ← 0	00111	001	1111	0000
Others	STOP	0	stop clock if CE = low	00111	010	1111	0000
	HALT	h	halt	00111	011	1111	h
	NOP		No operation	00111	100	1111	0000

2.3 ASSEMBLER (AS17K) INTRINSIC MACRO INSTRUCTIONS

Legend		
flag	:	One of flag1 through flagn
flag1 to flagn	:	Flag name indicated by reserved word
n	:	Number
< >	:	May be omitted

	Mnemonic	Operand	n	Operation
	SKTn	flag1, ···flagn	1 ≦n≦ 4	if (flag1) ~ (flagn) = all " 1 ", then skip
	SKFn	flag1, ···flagn	1 ≦n≦ 4	if (flag1) ~ (flagn) = all " 0 ", then skip
	SETn	flag1, ···flagn	1 ≦n≦ 4	(flag1) ~ (flagn) ← 1
	CLRn	flag1, ···flagn	1 ≦n≦ 4	(flag1) ~ (flagn) ← 0

	NOTn	flag1, ···flagn	1 ≦n≦ 4	if (flag) = " 0 ", then (flag) ← 1 &
				if (flag) = " 1 ", then (flag) ← 0

	INITFLG	<NOT>flag1,
		···<NOT>flagn
			1 ≦n≦ 4	if description = NOT flag, (flag) ← 0
				if description = flag, (flag) ← 1

	BANKn		0 ≦n≦ 3	(BANK) ← n, 0 ≦n≦ 3

CHAPTER 3. μPD17005 RESERVED SYMBOLS

3.1 LIST OF RESERVED SYMBOLS

3.1.1 System Registers (SYSREG)

Reserved Word	Type	Address	Read/ Write	Outline of Function
AR3	MEM	0.74H	R/W	Address register bits b_{15} to b_{12}
AR2	MEM	0.75H	R/W	Address register bits b_{11} to b_8
AR1	MEM	0.76H	R/W	Address register bits b_7 to b_4
AR0	MEM	0.77H	R/W	Address register bits b_3 to b_0
WR	MEM	0.78H	R/W	Window register
BANK	MEM	0.79H	R/W	Bank register
IXH	MEM	0.7AH	R/W	Index register-high
MPH	MEM	0.7AH	R/W	Memory pointer-high
MPE	FLG	0.7AH.3	R/W	Memory pointer enable flag
IXM	MEM	0.7BH	R/W	Index register-middle
MPL	MEM	0.7BH	R/W	Memory pointer-low
IXL	MEM	0.7CH	R/W	Index register-low
RPH	MEM	0.7DH	R/W	General register pointer-high
RPL	MEM	0.7EH	R/W	General register pointer-low
PSW	MEM	0.7FH	R/W	Program status word
BCD	FLG	0.7EH.0	R/W	BCD flag
CMP	FLG	0.7FH.3	R/W	Compare flag
CY	FLG	0.7FH.2	R/W	Carry flag
Z	FLG	0.7FH.1	R/W	Zero flag
IXE	FLG	0.7FH.0	R/W	Index enable flag

3.1.2 Data Buffer (DBF)

Reserved Word	Type	Address	Read/ Write	Outline of Function
DBF3	MEM	0.0CH	R/W	DBF bits b_{15} to b_{12}
DBF2	MEM	0.0DH	R/W	DBF bits b_{11} to b_8
DBF1	MEM	0.0EH	R/W	DBF bits b_7 to b_4
DBF0	MEM	0.0FH	R/W	DBF bits b_3 to b_0

3.1.3 LCD Dot Data Registers

Reserved Word	Type	Address	Read/ Write	Outline of Function
LCDD0	MEM	0.60H	R/W	LCD data register
LCDD1	MEM	0.61H	R/W	LCD data register
LCDD2	MEM	0.62H	R/W	LCD data register
LCDD3	MEM	0.63H	R/W	LCD data register
LCDD4	MEM	0.64H	R/W	LCD data register
LCDD5	MEM	0.65H	R/W	LCD data register
LCDD6	MEM	0.66H	R/W	LCD data register
LCDD7	MEM	0.67H	R/W	LCD data register
LCDD8	MEM	0.68H	R/W	LCD data register
LCDD9	MEM	0.69H	R/W	LCD data register
LCDD10	MEM	0.6AH	R/W	LCD data register
LCDD11	MEM	0.6BH	R/W	LCD data register
LCDD12	MEM	0.6CH	R/W	LCD data register
LCDD13	MEM	0.6DH	R/W	LCD data register
LCDD14	MEM	0.6EH	R/W	LCD data register

3.1.4 General Port Registers

Reserved Word	Type	Address	Read/ Write	Outline of Function
P0A3	FLG	0.70H.3	R/W	Port 0A bit b_3
P0A2	FLG	0.70H.2	R/W	Port 0A bit b_2
P0A1	FLG	0.70H.1	R/W	Port 0A bit b_1
P0A0	FLG	0.70H.0	R/W	Port 0A bit b_0
P0B3	FLG	0.71H.3	R/W	Port 0B bit b_3
P0B2	FLG	0.71H.2	R/W	Port 0B bit b_2
P0B1	FLG	0.71H.1	R/W	Port 0B bit b_1
P0B0	FLG	0.71H.0	R/W	Port 0B bit b_0
P0C3	FLG	0.72H.3	R/W	Port 0C bit b_3
P0C2	FLG	0.72H.2	R/W	Port 0C bit b_2
P0C1	FLG	0.72H.1	R/W	Port 0C bit b_1
P0C0	FLG	0.72H.0	R/W	Port 0C bit b_0
P0D3	FLG	0.73H.3	R	Port 0D bit b_3
P0D2	FLG	0.73H.2	R	Port 0D bit b_2
P0D1	FLG	0.73H.1	R	Port 0D bit b_1
P0D0	FLG	0.73H.0	R	Port 0D bit b_0
P0XL3	FLG	0.68H.3	R/W	Port 0X bit b_1
P0XL2	FLG	0.68H.2	R/W	Port 0X bit b_0
P0XL1	FLG	0.68H.1	R/W	Dummy
P0XL0	FLG	0.68H.0	R/W	Dummy
P0XH3	FLG	0.69H.3	R/W	Port 0X bit b_5
P0XH2	FLG	0.69H.2	R/W	Port 0X bit b_4
P0XH1	FLG	0.69H.1	R/W	Port 0X bit b_3
P0XH0	FLG	0.69H.0	R/W	Port 0X bit b_2
P0E3	FLG	0.6BH.3	R/W	Port 0E bit b_3
P0E2	FLG	0.6BH.2	R/W	Port 0E bit b_2
P0E1	FLG	0.6BH.1	R/W	Port 0E bit b_1
P0E0	FLG	0.6BH.0	R/W	Port 0E bit b_0
P0F3	FLG	0.6DH.3	R/W	Port 0F bit b_3
P0F2	FLG	0.6DH.2	R/W	Port 0F bit b_2
P0F1	FLG	0.6DH.1	R/W	Port 0F bit b_1
P0F0	FLG	0.6DH.0	R/W	Port 0F bit b_0
P1A3	FLG	1.70H.3	R/W	Port 1A bit b_3
P1A2	FLG	1.70H.2	R/W	Port 1A bit b_2
P1A1	FLG	1.70H.1	R/W	Port 1A bit b_1
P1A0	FLG	1.70H.0	R/W	Port 1A bit b_0

Reserved Word	Type	Address	Read/ Write	Outline of Function
P1B3	FLG	1.71H.3	R/W	Port 1B bit b_3
P1B2	FLG	1.71H.2	R/W	Port 1B bit b_2
P1B1	FLG	1.71H.1	R/W	Port 1B bit b_1
P1B0	FLG	1.71H.0	R/W	Port 1B bit b_0
P1C3	FLG	1.72H.3	R/W	Port 1C bit b_3
P1C2	FLG	1.72H.2	R/W	Port 1C bit b_2
P1C1	FLG	1.72H.1	R/W	Port 1C bit b_1
P1C0	FLG	1.72H.0	R/W	Port 1C bit b_0
P1D3	FLG	1.73H.3	R/W	Port 1D bit b_3
P1D2	FLG	1.73H.2	R/W	Port 1D bit b_2
P1D1	FLG	1.73H.1	R/W	Port 1D bit b_1
P1D0	FLG	1.73H.0	R/W	Port 1D bit b_0
P2A3	FLG	2.70H.3	R/W	Dummy
P2A2	FLG	2.70H.2	R/W	Dummy
P2A1	FLG	2.70H.1	R/W	Dummy
P2A0	FLG	2.70H.0	R/W	Port 2A bit b_0

3.1.5 Register File (Control Registers)

Reserved Word	Type	Address	Read/ Write	Outline of Function
SP	MEM	0.81H	R/W	Stack pointer
SIO2TS	FLG	0.82H.3	R/W	SIO2 start flag
SIO2HIZ	FLG	0.82H.2	R/W	SO2/POB1 select flag
SIO2CK1	FLG	0.82H.1	R/W	SIO2 clock select bit b_1
SIO2CK0	FLG	0.82H.0	R/W	SIO2 clock select bits b_0
IFCG	FLG	0.84H.0	R	IF counter gate status flag
PLLUL	FLG	0.85H.0	R	PLL unlock IF flag
ADCCMP	FLG	0.86H.0	R	ADC judge flag
CE	FLG	0.87H.0	R	CE pin status flag
SIO1CH	FLG	0.88H.3	R/W	SIO1 mode select flag
SB	FLG	0.88H.2	R/W	SB/SBI select flag
SIO1MS	FLG	0.88H.1	R/W	SIO1 clock mode select flag
SIO1TX	FLG	0.88H.0	R/W	SIO1 TX/RX select flag
TMMD3	FLG	0.89H.3	R/W	Timer interrupt mode select flag
TMMD2	FLG	0.89H.2	R/W	Timer interrupt mode select flag
TMMD1	FLG	0.89H.1	R/W	Timer carry FF mode select flag
TMMD0	FLG	0.89H.0	R/W	Timer carry FF mode select flag
INT1	FLG	0.8FH.1	R	INT1 pin status flag
INT0	FLG	0.8FH.0	R	INT0 pin status flag
KSEN	FLG	0.90H.1	R/W	Key source decoder enable flag
LCDEN	FLG	0.90H.0	R/W	LCD driver enable flag
P0YON	FLG	0.91H.3	R/W	Port 0Y enable flag
P0XON	FLG	0.91H.2	R/W	Port 0X enable flag
P0EON	FLG	0.91H.1	R/W	Port 0E enable flag
P0FON	FLG	0.91H.0	R/W	Port 0F enable flag
IFCMD1	FLG	0.92H.3	R/W	IF counter mode select flag
IFCMD0	FLG	0.92H.2	R/W	IF counter mode select flag
IFCCK1	FLG	0.92H.1	R/W	IF counter clock select flag
IFCCK0	FLG	0.92H.0	R/W	IF counter clock select flag
PWM2ON	FLG	0.93H.3	R/W	PWM2 enable flag
PWM1ON	FLG	0.93H.2	R/W	PWM1 enable flag
PWM0ON	FLG	0.93H.1	R/W	PWM0 enable flag
CGPON	FLG	0.93H.0	R/W	CGP enable flag

Reserved Word	Type	Address	Read/ Write	Outline of Function
ADCCH3	FLG	0.94H.3	R/W	AD mode select flag (dummy:0)
ADCCH2	FLG	0.94H.2	R/W	AD mode select flag
ADCCH1	FLG	0.94H.1	R/W	AD mode select flag
ADCCH0	FLG	0.94H.0	R/W	AD mode select flag
PLULDLY3	FLG	0.95H.3	R/W	PLL unlock time select flag (dummy:0)
PLULDLY2	FLG	0.95H.2	R/W	PLL unlock time select flag (dummy:0)
PLULDLY1	FLG	0.95H.1	R/W	PLL unlock time select flag
PLULDLY0	FLG	0.95H.0	R/W	PLL unlock time select flag
KEYJ	FLG	0.96H.0	R	Key input judge flag
TMCY	FLG	0.97H.0	R	Timer carry FF status flag
SBACK	FLG	0.98H.3	R/W	SB acknowledge flag
SIO1NWT	FLG	0.98H.2	R/W	SI01 not wait flag
SIO1WRQ1	FLG	0.98H.1	R/W	SI01 wait mode flag
SIO1WRQ0	FLG	0.98H.0	R/W	SI01 wait mode flag
IEG1	FLG	0.9FH.1	R/W	INT1 interrupt edge select flag
IEG0	FLG	0.9FH.0	R/W	INT0 interrupt edge select flag
PLLMD3	FLG	0.0A1H.3	R/W	PLL mode select flag (dummy:0)
PLLMD2	FLG	0.0A1H.2	R/W	PLL mode select flag (dummy:0)
PLLMD1	FLG	0.0A1H.1	R/W	PLL mode select flag
PLLMD0	FLG	0.0A1H.0	R/W	PLL mode select flag
IFCSTRT	FLG	0.0A3H.1	W	IF counter start flag
IFCRES	FLG	0.0A3H.0	W	IF counter reset flag
P0CGIO	FLG	0.0A7H.0	R/W	Port 0C I/O select flag
SIO1SF8	FLG	0.0A8H.3	R/W	SI01 clock counter status flag
SIO1SF9	FLG	0.0A8H.2	R/W	SI01 clock counter status flag
SBSTT	FLG	0.0A8H.1	R/W	SB start condition status flag
SBBSY	FLG	0.0A8H.0	R/W	SB start/stop codition status flag
IPIFC	FLG	0.0AEH.0	R/W	IF counter interrupt permission flag
IPSIO1	FLG	0.0AFH.3	R/W	SI01 interrupt permission flag
IPTM	FLG	0.0AFH.2	R/W	Timer interrupt permission flag
IP1	FLG	0.0AFH.1	R/W	INT1 interrupt permission flag
IP0	FLG	0.0AFH.0	R/W	INT0 interrupt permission flag
PLLRFMD3	FLG	0.0B1H.3	R/W	PLL reference clock select flag
PLLRFMD2	FLG	0.0B1H.2	R/W	PLL reference clock select flag
PLLRFMD1	FLG	0.0B1H.1	R/W	PLL reference clock select flag
PLLRFMD0	FLG	0.0B1H.0	R/W	PLL reference clock select flag

Reserved Word	Type	Address	Read/ Write	Outline of Function
P1ABIO3	FLG	0.0B5H.3	R/W	P1A3 I/O select flag
P1ABIO2	FLG	0.0B5H.2	R/W	P1A2 I/O select flag
P1ABIO1	FLG	0.0B5H.1	R/W	P1A1 I/O select flag
P1ABIO0	FLG	0.0B5H.0	R/W	P1A0 I/O select flag
P0BBIO3	FLG	0.0B6H.3	R/W	P0B3 I/O select flag
P0BBIO2	FLG	0.0B6H.2	R/W	P0B2 I/O select flag
P0BBIO1	FLG	0.0B6H.1	R/W	P0B1 I/O select flag
P0BBIO0	FLG	0.0B6H.0	R/W	P0B0 I/O select flag
P0ABIO3	FLG	0.0B7H.3	R/W	P0A3 I/O select flag
P0ABIO2	FLG	0.0B7H.2	R/W	P0A2 I/O select flag
P0ABIO1	FLG	0.0B7H.1	R/W	P0A1 I/O select flag
P0ABIO0	FLG	0.0B7H.0	R/W	P0A0 I/O select flag
SIO1IMD3	FLG	0.0B8H.3	R/W	SIO1 interrupt mode select flag (dummy:0)
SIO1IMD2	FLG	0.0B8H.2	R/W	SIO1 interrupt mode select flag (dummy:0)
SIO1IMD1	FLG	0.0B8H.1	R/W	SIO1 interrupt mode select flag
SIO1IMD0	FLG	0.0B8H.0	R/W	SIO1 interrupt mode select flag
SIO1CK3	FLG	0.0B9H.3	R/W	SIO1 shift clock select flag (dummy:0)
SIO1CK2	FLG	0.0B9H.2	R/W	SIO1 shift clock select flag (dummy:0)
SIO1CK1	FLG	0.0B9H.1	R/W	SIO1 shift clock select flag
SIO1CK0	FLG	0.0B9H.0	R/W	SIO1 shift clock select flag
IRQIFC	FLG	0.0BEH.0	R/W	IF counter interrupt request flag
IRQSIO1	FLG	0.0BFH.3	R/W	SIO1 interrupt request flag
IRQTM	FLG	0.0BFH.2	R/W	Timer interrupt request flag
IRQ1	FLG	0.0BFH.1	R/W	INT1 interrupt request flag
IRQ0	FLG	0.0BFH.0	R/W	INT0 interrupt request flag

3.1.6 Peripheral Hardware Addresses

Reserved Word	Type	Address	Read/ Write	Outline of Function
DBF	DAT	0FH	R/W	GET/PUT instruction data buffer address
IX	DAT	01H	R/W	INC instruction index register address
ADCR	DAT	02H	R/W	A/D converter V_{REF} data register
SIO2SFR	DAT	03H	R/W	SIO2 presettable shift register
SIO1SFR	DAT	04H	R/W	SIO1 presettable shift register
PWMR0	DAT	05H	R/W	PWM0 data register
PWMR1	DAT	06H	R/W	PWM1 data register
PWMR2	DAT	07H	R/W	PWM2 data register
LCDR0	DAT	08H	W	LCD group data register 0
LCDR1	DAT	09H	W	LCD group data register 1
LCDR2	DAT	0AH	W	LCD group data register 2
LCDR3	DAT	0BH	W	LCD group data register 3
LCDR4	DAT	0CH	W	LCD group data register 4
P0X	DAT	0CH	W	Port 0X data register
LCDR5	DAT	0DH	W	LCD group data register 5
LCDR6	DAT	0EH	W	LCD group data register 6
LCDR7	DAT	0FH	W	LCD group data register 7
CGPR	DAT	20H	R/W	CGP data register
AR	DAT	40H	R/W	GET/PUT/PUSH/CALL /BR/MOVT/INC instruction address register address
PLLR	DAT	41H	R/W	PLL data register
KSR	DAT	42H	R/W	Key source data register
P0Y	DAT	42H	R/W	Port 0Y data register
IFC	DAT	43H	R	IF counter data register

3.2 LIST OF RESERVED WORDS

3.2.1 Instructions/Pseudoinstructions

ADD	ADDC	AND	BANK0
BANK1	BANK2	BANK3	BELOW
BR	BYTE	C14344	C4444
CALL	CASE	CLR1	CLR2
CLR3	CLR4	CSEG0	DAT
DB	DI	DW	DWORD
EI	EJECT	ELSE	END
ENDCASE	ENDIF	ENDIFC	ENDIFNC
ENDM	ENDP	ENDR	EOF
EXIT	EXITR	EXTRN	FLG
GET	GLOBAL	HALT	IF
IFCHAR	IFNCHAR	INC	INCLUDE
INITFLG	IRP	LAB	LBMAC
LD	IFCOND	LIST	LITERAL
LMAC	MACRO	MEM	MOV
MOVT	NIBBLE	NIBBLE1	NIBBLE2
NIBBLE2V	NIBBLE3	NIBBLE3V	NIBBLE4
NIBBLE4V	NIBBLE5	NIBBLE5V	NIBBLE6
NIBBLE6V	NIBBLE7	NIBBLE7V	NIBBLE8
NIBBLE8V	NOBMAC	NOLIST	NOMAC
NOP	NOT1	NOT2	NOT3
NOT4	OBMAC	OMAC	OR
ORG	OTHER	PEEK	POKE
POP	PUBLIC	PURGE	PUSH
PUT	REPT	RET	RETI
RETSK	RORC	SBMAC	SET
SET1	SET2	SET3	SET4
SFCOND	SKE	SKF	SKF1
SKF2	SKF3	SKF4	SKGE
SKLT	SKNE	SKT	SKT1
SKT2	SKT3	SKT4	SMAC
ST	STOP	SUB	SUBC
SUMMARY	TAG	TITLE	WORD
XOR	ZZZERROR	ZZZMCHK	ZZZMSG
ZZZOPT			

3.2.2 Registers/Flags

ADCCH0	ADCCH1	ADCCH2	ADCCH3
ADCCMP	ADCR	AR	AR0
AR1	AR2	AR3	AR_EPA0
AR_EPA1	BANK	BCD	CE
CGPON	CGPR	CMP	CY
DBF	DBF0	DBF1	BDF2
DBF3	IEG0	IEG1	IFC
IFCCK0	IFCCK1	IFCG	IFCMD0
IFCMD1	IFCRES	IFCSTRT	INT0
INT1	IP0	IP1	IPIFC
IPSIO1	IPTM	IRQ0	IRQ1
IRQIFC	IRQSIO1	IRQTM	IX
IXE	IXH	IXL	IXM
KEYJ	KSEN	KSR	LCDD0
LCDD1	LCDD10	LCDD11	LCDD12
LCDD13	LCDD14	LCDD2	LCDD3
LCDD4	LCDD5	LCDD6	LCDD7
LCDD8	LCDD9	LCDEN	LCDR0
LCDR1	LCDR2	LCDR3	LCDR4
LCDR5	LCDR6	LCDR7	MPE
MPH	MPL	P0A0	P0A1
P0A2	P0A3	P0ABIO0	P0ABIO1
P0ABIO2	P0ABIO3	P0B0	P0B1
P0B2	P0B3	P0BBIO0	P0BBIO1
P0BBIO2	P0BBIO3	P0C0	P0C1
P0C2	P0C3	P0CGIO	P0D0
P0D1	P0D2	P0D3	P0E0
P0E1	P0E2	P0E3	P0EON
P0F0	P0F1	P0F2	P0F3
P0FON	P0X	P0XH0	P0XH1
P0XH2	P0XH3	P0XL0	P0XL1
P0XL2	P0XL3	P0XON	P0Y
P0YON	P1A0	P1A1	P1A2
P1A3	P1ABIO0	P1ABIO1	P1ABIO2
P1ABIO3	P1B0	P1B1	P1B2
P1B3	P1C0	P1C1	P1C2
P1C3	P1D0	P1D1	P1D2
P1D3	P2A0	P2A1	P2A2
P2A3	PLLMD0	PLLMD1	PLLMD2
PLLMD3	PLLR	PLLRFMD0	PLLRFMD1
PLLRFMD2	PLLRFMD3	PLLUL	PLULDLY0
PLULDLY1	PLULDLY2	PLULDLY3	PSW
PWM0ON	PWM1ON	PWM2ON	PWMR0
PWMR1	PWMR2	RPH	RPL
SB	SBACK	SBBSY	SBSTT
SIO1CH	SIO1CK0	SIO1CK1	SIO1CK2

SIO1CK3
SIO1IMD0
SIO1IMD1
SIO1IMD2
SIO1IMD3
SIO1MS
SIO1NWT
SIO1SF8
SIO1SF9
SIO1SFR
SIO1TX
SIO1WRQ0
SIO1WRQ1
SIO2CK0
SIO2CK1
SIO2HIZ
SIO2SFR
SIO2TS
SP
TMCY
TMMD0
TMMD1
TMMD2
TMMD3
WR
Z
ZZZ0
ZZZ1
ZZZ2
ZZZ3
ZZZ4
ZZZ5
ZZZ6
ZZZ7
ZZZ8
ZZZ9
ZZZDEVID
ZZZEPA
ZZZLSARG
ZZZPRINT
ZZZSKIP
ZZZSYD0C
ZZZALBMAC
ZZZALMAC
ZZZARGC
ZZZLINE

CHAPTER 4. NOTES ON USE OF DEVICE FILE

4.1 USE OF WRITE INSTRUCTION (POKE) ON IRQXXX FLAG

4.1.1 Point to Note

An error is generated in a write instruction (POKE) for register file address 3EH (BEH) which holds the IRQxxx (IRQ0, IRQ1, IRQTM, IRQSIO, IRQIFC) flags.

Therefore, a similar error is generated in an intrinsic macro instruction (SETn, CLRn, NOTn, INITFLG) for a flag in this address.

4.1.2 Reason for Prohibition of Use of Write Instruction

An IRQxxx flag is set when an interrupt request is issued, and reset when the interrupt is acknowledged.

If the IRQxxx flag is to be set or reset by the program at this time, for instance when interrupt request initialization or interrupt request polling processing is performed, the program shown in the following example is used:

Example: To reset IRQSIO flag

```
PEEK WR, . MF. IRQSIO SHR 4              ; ①
AND WR,  . DF. (NOT IRQSIO) AND 0FH      ; ②
POKE. MF . IRQSIO SHR 4, WR              ; ③
```

In the above example, even assuming that IRQxxx flags (IRQ0, IRQ1, IRQTM, IRQSIO) in the same address are modified during execution of instructions ① and ②, since the value read by instruction ① is written by instruction ③, the IRQxxx flag status is rewritten.

4.1.3 Methods of Performing a Write

(1) Use the following macros by INCLUDing the IRQ.MAC file provided with the device file (see Chapter 5):
SETIRQn
CLRIRQn
NOTIRQn
INITIRQ

(2) Define each instruction by a DW pseudo instruction.

4.2 USE OF READ INSTRUCTION (PEEK) ON IFCRES OR IFCSTRT FLAG

4.2.1 Point to Note

An error is generated in a read instruction (PEEK) on register file address 23H (A3H) which ho·ds the IFCRES and IFCSTRT flags.

Therefore, a similar error is generated in an intrinsic macro instruction (SETn, CLRn, NOTn, INITFLG) for a flag in this address.

4.2.2 Reason for Prohibition of Use of Read Instruction (PEEK)

IFCRES and IFCSTRT are write-only flags: If read, an indeterminate value will be read. If a SETn instruction is used at this time, it is expanded as shown in the example below:

Example: To set the IFCRES flag only

```
SET1 IFCRES
    PEEK WR, . MF. IFCRES SHR 4      ; ①
    OR WR,   . DF. IFCRES AND 0FH    ; ②
    POKE. MF . IFCRES SHR 4, WR      ; ③
```

In the above example, when instruction ① is executed, an indeterminate value is read for the IFCSTRT flag held in the same address. Therefore, an indeterminate value is written to the IFCSTRT flag by instruction ③.

4.2.3 Methods of Performing a Write

(1) Use the following macros by INCLUDing the IFCSET.LIB file provided with the device file (see Chapter 6):
IFC_RES
IFC_START
IFC_RES_AND_START

(2) Use a POKE instruction to perform the write.

CHAPTER 5. IRQ.MAC FILE

5.1 OUTLINE

The IRQ.MAC file provides macro instructions for manipulating the IRQxxx flags (see **Chapter 4**).
The macro instructions provided by the IRQ.MAC file can also be used on flags other than the IRQxxx flags.

5.2 MACRO INSTRUCTIONS PROVIDED

SETIRQn
CLRIRQn
NOTIRQn
INITIRQ

5.3 USING THE IRQ.MAC FILE

Specify "/MAC=25" as an assembly option.
The macro instructions can be used by INCLUDing the IRQ.MAC file in the source program.

5.4 SETTING ASSEMBLY-TIME VARIABLE "ZZZ9"

The IRQ.MAC file uses the assembly-time variable "ZZZ9".
Therefore, "ZZZ9" should not be used in the source program.
If "ZZZ9" is used with the default value (0), an error will be generated as shown below.
This is related to the point to note described in **Chapter 4,** and the object is output normally.
When ordering mask ROM, assembly option "/ZZZ9=−1" should be specified.

(1) When ZZZ9 is used with the default value (0) or when a value other than −1 is set
If an IRQxxx flag is specified by these macro instructions, an error will be generated as shown below.
   (i) PC-9800 series (MS-DOS) based
      CAUTION: Unexpected IRQ may be canceled. See user's manual.
   (ii) IBM-PC (PC DOS) based
      CAUTION: Unexpected IRQ may be canceled. See user's manual.
However, since the object is output normally, debugging can be performed.
No error is generated for a flag other than IRQxxx.

(2) When ZZZ9 is set to −1
No error is generated when an IRQxxx flag is specified by these macro instructions.
Assemble with this setting when ordering mask ROM.

5.5 ASSEMBLY CONTROL INSTRUCTIONS SET IN IRQ.MAC FILE

The following assembly control instructions are set in the IRQ.MAC file:

(1) OMAC: Macro expansion print control instruction
If "/NOG[EN]" is set as an assembly option, only object-generating statements in the macro expansion block will be printed.

(2) NOLIST : List output control instruction
LIST : List output control instruction
"NOLIST" is set at the start of the IRQ.MAC file, and "LIST" is set at the end.
If "/LIST" is set as an assembly option, the INCLUDEd IRQ.MAC contents will not be output to the list.
Note that if "NOLIST" is set before INCLUDing the IRQ.MAC file, a "LIST" specification will be effected.

5.6 POINTS TO NOTE WHEN USING IRQ.MAC FILE

(1) If these macro instructions are used on addresses other than the IRQxxx flag address, they are expanded to the corresponding assembler macro instructions (SETn, CLRn, NOTn, INITFLG).

(2) When both an IRQxxx flag and a non-IRQxxx-flag address are written, expansion to the assembler macro instruction (SETn, CLRn, NOTn, INITFLG) is performed for the non-IRQxxx-flag address.

(3) If a skip instruction is written directly before these macro instructions, a branch instruction is generated automatically. However, a warning (of the possibility that the branch instruction may be shortened) is not issued.

(4) If the assembly control instruction "NOLIST" is set before the IRQ.MAC file is INCLUDEd, "LIST" will be set again when the INCLUDE is performed.

5.7 DESCRIPTION OF MACRO INSTRUCTIONS

SETIRQn **SET nFLAGS**
CLRIRQn **CLEAR nFLAGS**

Symbol Field	Mnemonic Field	Operand Field	Comment Field
[Label:]	SETIRQ1	<Symbol (FLG type)>	[;Comment]
[Label:]	SETIRQ2	<Symbol (FLG type)>, <Symbol (FLG type)>	[;Comment]
[Label:]	SETIRQ3	<Symbol (FLG type)>, <Symbol (FLG type)>,	
		<Symbol (FLG type)>	[;Comment]
[Label:]	SETIRQ4	<Symbol (FLG type)>, <Symbol (FLG type)>,	
		<Symbol (FLG type)>, <Symbol (FLG type)>	[;Comment]
[Label:]	CLRIRQ1	<Symbol (FLG type)>	[;Comment]
[Label:]	CLRIRQ2	<Symbol (FLG type)>, <Symbol (FLG type)>	[;Comment]
[Label:]	CLRIRQ3	<Symbol (FLG type)>, <Symbol (FLG type)>,	
		<Symbol (FLG type)>	[;Comment]
[Label:]	CLRIRQ4	<Symbol (FLG type)>, <Symbol (FLG type)>,	
		<Symbol (FLG type)>, <Symbol (FLG type)>	[;Comment]

[Function]

SETIRQn : Sets (1) all flags specified in the operand field.
CLRIRQn : Resets (0) all flags specified in the operand field.

[Use]

Used for flag operations.

[Description]

(1) If anything other than <Symbol (FLG type)> is written in the operand field, an T error (Invalid type) will be generated.

(2) If flags in different data memory banks are mixed in the operand field, a B error (BANK unmatch) will be generated.

(3) The operand field may contain flags defined in data memory or flags defined in the register file, or both. The portion of the register file which overlaps the data memory (40H to 7FH) is processed as data memory.

(4) Up to approximately 20 nesting levels in total are permitted for repeat pseudo instructions, IF statements, intrinsic macro instructions, and macro reference statements.

(5) An error will be generated if the number of operands is different from the value of "n" in SETIRQn or CLRIRQn.

NOTIRQn **NOT nFLAGS**

Symbol Field	Mnemonic Field	Operand Field	Comment Field
[Label:]	NOTIRQ1	<Symbol (FLG type)>	[;Comment]
[Label:]	NOTIRQ2	<Symbol (FLG type)>, <Symbol (FLG type)>	[;Comment]
[Label:]	NOTIRQ3	<Symbol (FLG type)>, <Symbol (FLG type)>, <Symbol (FLG type)>	[;Comment]
[Label:]	NOTIRQ4	<Symbol (FLG type)>, <Symbol (FLG type)>, <Symbol (FLG type)>, <Symbol (FLG type)>	[;Comment]

[Function]

Complements all flags specified in the operand field.

[Use]

Used for flag complementing.

[Description]

(1) If anything other than <Symbol (FLG type)> is written in the operand field, an T error (Invalid type) will be generated.

(2) If flags in different data memory banks are mixed in the operand field, a B error (BANK unmatch) will be generated.

(3) The operand field may contain flags defined in data memory or flags defined in the register file, or both. The portion of the register file which overlaps the data memory (40H to 7FH) is processed as data memory.

(4) Up to approximately 20 nesting levels in total are permitted for repeat pseudo instructions, IF statements, intrinsic macro instructions, and macro reference statements.

(5) An error will be generated if the number of operands is different from the value of "n" in NOTIRQn.

INITIRQ   INITIALIZE 4 FLAGS

Symbol Field	Mnemonic Field	Operand Field	Comment Field
[Label:]	INITIRQ	[NOT] <Symbol (FLG type)>, [NOT] <Symbol (FLG type)>, [NOT] <Symbol (FLG type)>, [NOT] <Symbol (FLG type)>	[;Comment]

[Function]

Initializes the 4 flags specified in the operand field to set (1) or reset (0).

[Use]

Used for flag initialization.

[Description]

(1) If anything other than <Symbol (FLG type)> is written in the operand field, an T error (Invalid type) will be generated.

(2) If flags in different data memory banks are mixed in the operand field, a B error (BANK unmatch) will be generated.

(3) The operand field may contain flags defined in data memory or flags defined in the register file, or both. The portion of the register file which overlaps the data memory (40H to 7FH) is processed as data memory.

(4) Up to approximately 20 nesting levels in total are permitted for repeat pseudo instructions, IF statements, intrinsic macro instructions, and macro reference statements.

(5) An error will be generated if the number of operands is not 4.

(6) If "NOT" is written before <Symbol (FLG type)> in the operand field, that flag is reset (0), and if "NOT" is omitted, the flag is set (1).

CHAPTER 6. IFCSET.LIB FILE

6.1 OUTLINE

The IFCSET.LIB file provides macro instructions for manipulating the IFCRES and IFCSTRT flags (See **Chapter 4**).

The macro instructions provided by the IFCSET.LIB file can only be used for the IFCRES and IFCSTRT flags.

6.2 MACRO INSTRUCTIONS PROVIDED

IFC_RES
IFC_START
IFC_RES_AND_START

6.3 USING THE IFCSET.LIB FILE

The macro instructions can be used by INCLUDing the IRQ.MAC file in the source program.

6.4 ASSEMBLY CONTROL INSTRUCTIONS SET IN IFCSET.LIB FILE

The following assembly control instructions are set in the IFCSET.LIB file:

NOLIST : List output control instruction
LIST : List output control instruction

"NOLIST" is set at the start of the IFCSET.LIB file, and "LIST" is set at the end.

If "/LIST" is set as an assembly option, list output will stop at the point at which the IFCSET.LIB file is INCLUDEd.

Note that if "NOLIST" is set before INCLUDing the IFCSET.LIB file, a "LIST" specification will be effected.

6.5 DESCRIPTION OF MACRO INSTRUCTIONS

IFC_RES **RESET IF counter**

Symbol Field	Mnemonic Field	Operand Field	Comment Field
[Label:]	IFC_RES		[;Comment]

[Function]

Sets (1) the IFCRES flag.

[Use]

Resets (0000H) the IF counter.

[Description]

(1) "0" is written to the IFCSTRT flag.

(2) Up to approximately 40 nesting levels in total are permitted for repeat pseudo instructions, IF statements, intrinsic macro instructions, and macro reference statements.

IFC_START　　START IF counter

Symbol Field	Mnemonic Field	Operand Field	Comment Field
[Label:]	IFC_START		[;Comment]

[Function]

Sets (1) the IFCSTRT flag.

[Use]

Used to start the IF counter.

[Description]

(1) "0" is written to the IFCRES flag.

(2) Up to approximately 40 nesting levels in total are permitted for repeat pseudo instructions, IF statements, intrinsic macro instructions, and macro reference statements.

IFC_RES_AND_START RESET and START IF counter

Symbol Field	Mnemonic Field	Operand Field	Comment Field
[Label:]	IFC_RES_AND_START		[;Comment]

[Function]

Sets (1) the IFCRES and IFCSTRT flags.

[Use]

Used to reset and start the IF counter.

[Description]

(1) Up to approximately 40 nesting levels in total are permitted for repeat pseudo instructions, IF statements, intrinsic macro instructions, and macro reference statements.

NEC