TM57 Series

Instruction Set

Application Note

tenx reserves the right to change or discontinue the manual and online documentation to this product herein to improve reliability, function or design without further notice. tenx does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. tenx products are not designed, intended, or authorized for use in life support appliances, devices, or systems. If Buyer purchases or uses tenx products for any such unintended or unauthorized application, Buyer shall indemnify and hold tenx and its officers, employees, subsidiaries, affiliates and distributors harmless against all claims, cost, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use even if such claim alleges that tenx was negligent regarding the design or manufacture of the part.

AMENDMENT HISTORY

Version	Date	Description
V1.0	Sept, 2011	New release

CONTENTS

Instruction Set

Each instruction is a 14-bit word divided into an OPCODE, which specifies the instruction type, and one or more operands, which further specify the operation of the instruction. The instructions can be categorized as byte-oriented, bit-oriented and literal operations listed in the following table.

For byte-oriented instructions, “f” or “r” represents the address designator and “d” represents the destination designator. The address designator is used to specify which address in Program memory is to be used by the instruction. The destination designator specifies where the result of the operation is to be placed. If “d” is “0”, the result is placed in the W register. If “d” is “1”, the result is placed in the address specified in the instruction.

For bit-oriented instructions, “b” represents a bit field designator, which selects the number of the bit affected by the operation, while “f” represents the address designator. For literal operations, “k” represents the literal or constant value.

Field / Legend	Description
f	F-Plane Register File Address
r	R-Plane Register File Address
b	Bit address
k	Literal. Constant data or label
d	Destination selection field, 0: Working register, 1: Register file
W	Working Register
Z	Zero Flag
C	Carry Flag
DC	Decimal Carry Flag
PC	Program Counter
TOS	Top Of Stack
GIE	Global Interrupt Enable Flag (i-Flag)
[]	Option Field
( )	Contents
.	Bit Field
B	Before
A	After
←	Assign direction

There are two types of instruction set in TM57 series. The differences between them are described in below table:

	INSTA Set	INSTB Set
IC body	except body of INSTB Set	TM57FLA80 series TM56MA40 series TM57ML40 series
Different Instruction Code	MOVWR XORLW CLRWDT SLEEP	
MOVRW Instruction	No	Yes
DPTR instruction (TBLWL/TBLWH)	TM57MA40 series TM57MA41 series	TM56MA40 series

Instruction Set List

The following table lists all the instruction set in TM57 series.

Label “A” means the instruction is included in instruction set A.

Label “B” means the instruction is included in instruction set B.

Most of the instructions are included in instruction set A and B, please notice the sign “◯” in the column.

Mnemonic		Op Code	Cycle	Flag Affect	Description	A	B
Byte-Oriented File Register Instruction							
ADDWF	f,d	00 0111 dfff ffff	1	C, DC, Z	Add W to f	◯	◯
ANDWF	f,d	00 0101 dfff ffff	1	Z	AND W to f	◯	◯
CLRF	f	00 0001 1fff ffff	1	Z	Clear f	◯	◯
CLRW		00 0001 0100 0000	1	Z	Clear W	◯	◯
COMF	f,d	00 1001 dfff ffff	1	Z	Invert F bit by bit	◯	◯
DECF	f,d	00 0011 dfff ffff	1	Z	Decrement of f	◯	◯
DECFSZ	f,d	00 1011 dfff ffff	1 or 2	-	Decrease f, skip if zero	◯	◯
INCF	f,d	00 1010 dfff ffff	1	Z	Increment of f	◯	◯
INCFSZ	f,d	00 1111 dfff ffff	1 or 2	-	Increase f, skip if zero	◯	◯
IORWF	f,d	00 0100 dfff ffff	1	Z	OR W to f	◯	◯
MOVFW	f	00 1000 0fff ffff	1	-	Move f to W	◯	◯
MOVWF	f	00 0000 1fff ffff	1	-	Move W to f	◯	◯
MOVWR	r	00 0000 00rr rrrr	1	-	Move W to r	◯	
MOVWR	r	01 1110 rrrr rrrr	1	-	Move W to r		◯
RLF	f,d	00 1101 dfff ffff	1	C	F rotate to left through C	◯	◯
RRF	f,d	00 1100 dfff ffff	1	C	F rotate to right through C	◯	◯
SUBWF	f,d	00 0010 dfff ffff	1	C, DC, Z	Substrate W from f	◯	◯
MOVRW	r	01 1111 rrrr rrrr	1	-	Move r to W		◯
SWAPF	f,d	00 1110 dfff ffff	1	-	Swap high and low nibble of f	◯	◯
TESTZ	f	00 1000 1fff ffff	1	Z	Test f if zero	◯	◯
XORWF	f,d	00 0110 dfff ffff	1	Z	XOR W to f	◯	◯
Bit-Oriented File Register Instruction							
BCF	f,b	01 000b bbff ffff	1	-	Bit clear f	◯	◯
BSF	f,b	01 001b bbff ffff	1	-	Bit set f	◯	◯
BTFSC	f,b	01 010b bbff ffff	1 or 2	-	Bit test f, skip if clear	◯	◯
BTFSS	f,b	01 011b bbff ffff	1 or 2	-	Bit test f, skip if set	◯	◯

Mnemonic	Op Code	Cycle	Flag Affect	Description	A	B
Literal and Control Instruction						
ADDLW k	01 1100 kkkk kkkk	1	C, DC, Z	Add literal to W	○	○
ANDLW k	01 1011 kkkk kkkk	1	Z	AND literal to W	○	○
CALL k	10 kkkk kkkk kkkk	2	-	Subroutine call	○	○
CLRWDT	00 0000 0000 0100	1	TO,PD	Clear watchdog timer	○	
CLRWDT	01 1110 0000 0100	1	TO,PD	Clear watchdog timer		○
GOTO k	11 kkkk kkkk kkkk	2	-	Unconditional branch	○	○
IORLW k	01 1010 kkkk kkkk	1	Z	OR literal to W	○	○
MOVLW k	01 1001 kkkk kkkk	1	-	Move literal to W	○	○
NOP	00 0000 0000 0000	1	-	No operation	○	○
RET	00 0000 0100 0000	2	-	Return from CALL	○	○
RETI	00 0000 0110 0000	2	-	Return from interrupt	○	○
RETLW k	01 1000 kkkk kkkk	2	-	Return with literal to W	○	○
TBLWL	00 0000 0101 0000	2	-	Return DPTR low byte to W	○	○
TBLWH	00 0000 0101 1000	2	-	Return DPTR high byte to W	○	○
SLEEP	00 0000 0000 0011	1	TO,PD	Power down	○	
SLEEP	01 1110 0000 0011	1	TO,PD	Power down		○
XORLW k	01 1111 kkkk kkkk	1	Z	XOR literal to W	○	
XORLW k	01 1101 kkkk kkkk	1	Z	XOR literal to W		○

ADDLW	Add Literal “k” and W
Syntax	ADDLW k
Operands	k : 00h ~ FFh
Operation	(W) ← (W) + k
Status Affected	C, DC, Z
OP-Code	01 1100 kkkk kkkk
Description	The contents of the W register are added to the eight-bit literal ‘k’ and the result is placed in the W register.
Cycle	1
Example	ADDLW 0x18 1> B : W = 0x2E ,C=?,DC=?,Z=? A : W = 0x46 ,C=0,DC=1,Z=0 2> B : W = 0xF7 ,C=?,DC=?, Z=? A : W = 0x0F ,C=1,DC=0,Z=0 3> B : W = 0XE8 ,C=?,DC=?, Z=? A : W = 0x00 C=1,DC=1,Z=1

ADDWF	Add W and “f”
Syntax	ADDWF f [,d]
Operands	f : 00h ~ 7Fh d : 0, 1
Operation	(Destination) ← (W) + (f)
Status Affected	C, DC, Z
OP-Code	00 0111 dfff ffff
Description	Add the contents of the W register with register ‘f’. If ‘d’ is 0, the result is stored in the W register. If ‘d’ is 1, the result is stored back in register ‘f’.
Cycle	1
Example	ADDWF FSR, 0 1> B : W = 0x18, FSR = 0x2E ,C=?, DC=?,Z=? A : W = 0X46, FSR = 0x2E ,C=0,DC=1,Z=0 2> B : W = 0x18, FSR = 0XF7 ,C=?,DC=?,Z=? A : W = 0x0F, FSR = 0XF7 C=1,DC=0,Z=0 3> B : W = 0x18, FSR = 0XE8 ,C=?,DC=?,Z=? A : W = 0x00, FSR = 0XE8 ,C=1,DC=1,Z=1

ANDLW	Logical AND Literal "k" with W
Syntax	ANDLW k
Operands	k : 00h ~ FFh
Operation	(W) ← (W) ‘AND’ k
Status Affected	Z
OP-Code	01 1011 kkkk kkkk
Description	The contents of W register are AND’ed with the eight-bit literal 'k'. The result is placed in the W register.
Cycle	1
Example	ANDLW 0x5F 1> B : W = 0xA3 , Z=? A : W = 0x03 , Z=0 2> B : W = 0xA0 , Z=? A : W = 0x00 , Z=1

ANDWF	AND W with “f”	
Syntax	ANDWF f [,d]	
Operands	f : 00h ~ 7Fh d : 0, 1	
Operation	(Destination) ← (W) ‘AND’ (f)	
Status Affected	Z	
OP-Code	00 0101 dfff ffff	
Description	AND the W register with register ‘f’. If ‘d’ is 0, the result is stored in the W register. If ‘d’ is 1, the result is stored back in register ‘f’.	
Cycle	1	
Example	ANDWF FSR, 1	1>B : W = 0x17, FSR = 0xC2,Z=? A : W = 0x17, FSR = 0x02,Z=0 2>B : W = 0x15, FSR = 0xC2,Z=? A : W = 0x17, FSR = 0x00,Z=1

BCF	Clear "b" bit of "f"	
Syntax	BCF f [,b]	
Operands	f : 00h ~ 3Fh b : 0 ~ 7	
Operation	(f.b) ← 0	
Status Affected	-	
OP-Code	01 000b bbff ffff	
Description	Bit ‘b’ in register ‘f’ is cleared.	
Cycle	1	
Example	BCF FLAG_REG, 7	B : FLAG_REG = 0xC7 A : FLAG_REG = 0x47

BSF	Set "b" bit of "f"	
Syntax	BSF f [,b]	
Operands	f : 00h ~ 3Fh b : 0 ~ 7	
Operation	(f.b) ← 1	
Status Affected	-	
OP-Code	01 001b bbff ffff	
Description	Bit ‘b’ in register ‘f’ is set.	
Cycle	1	
Example	BSF FLAG_REG, 7	B : FLAG_REG = 0x0A A : FLAG_REG = 0x8A

BTFSC	Test “b” bit of “f”, skip if clear(0)	
Syntax	BTFSC f [,b]	
Operands	f : 00h ~ 3Fh b : 0 ~ 7	
Operation	Skip next instruction if (f.b) = 0	
Status Affected	-	
OP-Code	01 010b bbff ffff	
Description	If bit ‘b’ in register ‘f’ is ‘1’, then the next instruction is executed. If bit ‘b’ in register ‘f’ is ‘0’, then the next instruction is discarded, and a NOP is executed instead, making this a 2 cycle instruction.	
Cycle	1 or 2	
Example	LABEL1: BTFSC FLAG, 1 LABEL2: GOTO LABEL4 LABEL3: ...	B : PC = LABEL1 A : if FLAG.1 = 0, PC = LABEL3 if FLAG.1 = 1, PC = LABEL2

BTFSS	Test "b" bit of "f", skip if set(1)
Syntax	BTFSS f [,b]
Operands	f : 00h ~ 3Fh b : 0 ~ 7
Operation	Skip next instruction if (f.b) = 1
Status Affected	-
OP-Code	01 011b bbff ffff
Description	If bit 'b' in register 'f' is '0', then the next instruction is executed. If bit 'b' in register 'f' is '1', then the next instruction is discarded, and a NOP is executed instead, making this a 2 cycle instruction.
Cycle	
Example	LABEL1: BTFSS FLAG, 1 — B : PC = LABEL1 LABEL2: GOTO LABEL4 — A : if FLAG.1 = 0, PC = LABEL2 LABEL3: ... — if FLAG.1 = 1, PC = LABEL3

CALL	Call subroutine "k"
Syntax	CALL k
Operands	K : 00h ~ FFFh
Operation	Operation: TOS ← (PC)+ 1, PC.11~0 ← k
Status Affected	-
OP-Code	10 kkkk kkkk kkkk
Description	Call Subroutine. First, return address (PC+1) is pushed onto the stack. The 12-bit immediate address is loaded into PC bits <11:0>. CALL is a two-cycle instruction.
Cycle	2
Example	LABEL1: CALL SUB1 — B : PC = LABEL1 LABEL2: ... — A : PC = SUB1, TOS = LABEL2 ... SUB1: ...

CLRF	Clear "f"
Syntax	CLRF f
Operands	f : 00h ~ 7Fh
Operation	(f) ← 00h, Z ← 1
Status Affected	Z
OP-Code	00 0001 1fff ffff
Description	The contents of register 'f' are cleared and the Z bit is set.
Cycle	1
Example	CLRF FLAG_REG B : FLAG_REG = 0x5A,Z=? A : FLAG_REG = 0x00,Z = 1

CLRW	Clear W
Syntax	CLRW
Operands	-
Operation	(W) ← 00h, Z ← 1
Status Affected	Z
OP-Code	00 0001 0100 0000
Description	W register is cleared and Zero bit (Z) is set.
Cycle	1
Example	CLRW B : W = 0x5A,Z=? A : W = 0x00,Z = 1

CLRWDT — Clear Watchdog Timer

Syntax	CLRWDT
Operands	-
Operation	WDTE ← 00h
Status Affected	TO,PD
OP-Code	00 0000 0000 0100 (Instruction Set A) 01 1110 0000 0100 (Instruction Set B)
Description	CLRWDT instruction resets the Watchdog Timer.
Cycle	1
Example	CLRWDT B : WDT counter = ?,TO=?,PD=? A : WDT counter = 0x00,TO=0,PD=0

COMF — Complement "f"

Syntax	COMF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) ← ($\bar{f}$)
Status Affected	Z
OP-Code	00 1001 dfff ffff
Description	The contents of register 'f' are complemented. If 'd' is 0, the result is stored in W. If 'd' is 1, the result is stored back in register 'f'.
Cycle	1
Example	COMF REG1,0 B : REG1 = 0x13,Z=? A : REG1 = 0x13, W = 0xEC,Z=0

DECF — Decrement "f"

Syntax	DECF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) ← (f) - 1
Status Affected	Z
OP-Code	00 0011 dfff ffff
Description	Decrement register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.
Cycle	1
Example	DECF CNT, 1 B : CNT = 0x01, Z = 0 A : CNT = 0x00, Z = 1

DECFSZ — Decrement "f", Skip if 0

Syntax	DECFSZ f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) ← (f) - 1, skip next instruction if result is 0
Status Affected	-
OP-Code	00 1011 dfff ffff
Description	The contents of register 'f' are decremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'. If the result is 1, the next instruction is executed. If the result is 0, then a NOP is executed instead, making it a 2 cycle instruction.
Cycle	1 or 2
Example	LABEL1 DECFSZ CNT, 1 LABEL2 GOTO LABEL4 LABEL3 .... B : PC = LABEL1 A : CNT = CNT – 1 if CNT=0, PC = LABEL3 if CNT≠0, PC = LABEL2

GOTO — Unconditional Branch

Syntax	GOTO k
Operands	k : 00h ~ FFFh
Operation	PC.11~0 ← k
Status Affected	-
OP-Code	11 kkkk kkkk kkkk
Description	GOTO is an unconditional branch. The 12-bit immediate value is loaded into PC bits <11:0>. GOTO is a two-cycle instruction.
Cycle	2
Example	LABEL1: GOTO LABEL2 — B : PC = LABEL1 .... — A : PC = LABEL2 LABEL2: ....

INCF — Increment “f”

Syntax	INCF f [,d]
Operands	f : 00h ~ 7Fh
Operation	(destination) ← (f) + 1
Status Affected	Z
OP-Code	00 1010 dfff ffff
Description	The contents of register ‘f’ are incremented. If ‘d’ is 0, the result is placed in the W register. If ‘d’ is 1, the result is placed back in register ‘f’.
Cycle	1
Example	INCF CNT, 1 — B : CNT = 0xFF, Z = 0 A : CNT = 0x00, Z = 1

INCFSZ — Increment “f”, Skip if 0

Syntax	INCFSZ f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) ← (f) + 1, skip next instruction if result is 0
Status Affected	-
OP-Code	00 1111 dfff ffff
Description	The contents of register ‘f’ are incremented. If ‘d’ is 0, the result is placed in the W register. If ‘d’ is 1, the result is placed back in register ‘f’. If the result is 1, the next instruction is executed. If the result is 0, a NOP is executed instead, making it a 2 cycle instruction.
Cycle	1 or 2
Example	LABEL1: INCFSZ CNT, 1 — B : PC = LABEL1 LABEL2: GOTO LABEL4 — A : CNT = CNT + 1 LABEL3: ... — if CNT=0, PC = LABEL3 if CNT≠0, PC = LABEL2

IORLW — Inclusive OR Literal with W

Syntax	IORLW k
Operands	k : 00h ~ FFh
Operation	(W) ← (W) OR k
Status Affected	Z
OP-Code	01 1010 kkkk kkkk
Description	The contents of the W register are OR’ed with the eight-bit literal 'k'. The result is placed in the W register.
Cycle	1
Example	IORLW 0x35 — B : W = 0x9A , Z = ? A : W = 0xBF, Z = 0

IORWF — Inclusive OR W with "f"

Syntax	IORWF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) ← (W) OR (f)
Status Affected	Z
OP-Code	00 0100 dfff ffff
Description	Inclusive OR the W register with register 'f'. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.
Cycle	1
Example	IORWF RESULT, 0 B : RESULT = 0x13, W = 0x91, Z = ? A : RESULT = 0x13, W = 0x93, Z = 0

MOVFW — Move "f" to W

Syntax	MOVFW f
Operands	f : 00h ~ 7Fh
Operation	(W) ← (f)
Status Affected	-
OP-Code	00 1000 0fff ffff
Description	The contents of register f are moved to W register.
Cycle	1
Example	MOVFW FSR B : FSR = 0xC2, W = ? A : FSR = 0xC2, W = 0xC2

MOVLW — Move Literal to W

Syntax	MOVLW k
Operands	k : 00h ~ FFh
Operation	(W) ← k
Status Affected	-
OP-Code	01 1001 kkkk kkkk
Description	The eight-bit literal 'k' is loaded into W register.
Cycle	1
Example	MOVLW 0x5A B : W = ? A : W = 0x5A

MOVRW — Move "r" to W

Syntax	MOVRW r
Operands	r : 20h ~ FFh
Operation	(W) ← (r)
Status Affected	-
OP-Code	01 1111 rrrr rrrr (Instruction Set B)
Description	Move data from register 'r' to W register.
Cycle	1
Example	MOVRW REG1 B : REG1 = 0x4F, W = ? A : REG1 = 0x4F, W = 0x4F

MOVWF	Move W to “f”	
Syntax	MOVWF f	
Operands	f : 00h ~ 7Fh	
Operation	(f) ← (W)	
Status Affected	-	
OP-Code	00 0000 1fff ffff	
Description	Move data from W register to register ‘f’.	
Cycle	1	
Example	MOVWF REG1	B : REG1 = 0xFF, W = 0x4F A : REG1 = 0x4F, W = 0x4F

MOVWR	Move W to “r”	
Syntax	MOVWR r	
Operands	r : 00h ~ FFh	
Operation	(r) ← (W)	
Status Affected	-	
OP-Code	00 0000 00rr rrrr (Instruction Set A) 01 1110 rrrr rrrr (Instruction Set B)	
Description	Move data from W register to register ‘r’.	
Cycle	1	
Example	MOVWR REG1	B : REG1 = 0xFF, W = 0x4F A : REG1 = 0x4F, W = 0x4F

NOP	No Operation	
Syntax	NOP	
Operands	-	
Operation	No Operation	
Status Affected	-	
OP-Code	00 0000 0000 0000	
Description	No Operation	
Cycle	1	
Example	NOP	-

RET	Return from Subroutine	
Syntax	RET	
Operands	-	
Operation	PC ← TOS	
Status Affected	-	
OP-Code	00 0000 0100 0000	
Description	Return from subroutine. The stack is POPed and the Top-of-Stack (TOS) is loaded into the program counter (PC). This is a two-cycle instruction.	
Cycle	2	
Example	LABEL: RET LABEL1: ...	B : PC=LABEL,TOS=LABEL2 A : PC = LABEL2

RETI	Return from Interrupt
Syntax	RETI
Operands	-
Operation	PC ← TOS, GIE ← 1
Status Affected	-
OP-Code	00 0000 0110 0000
Description	Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in to the program counter (PC). Interrupts are enabled. This is a two-cycle

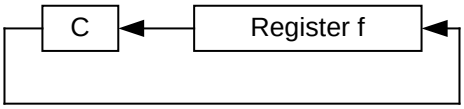
	instruction.	
Cycle	2	
Example	LABEL: RETI	B : PC=LABEL,TOS=LABEL2,GIE=0
	LABEL1: ...	A : PC = LABEL2, GIE = 1

RETLW Return with Literal in W

Syntax	RETLW k
Operands	k : 00h ~ FFh
Operation	PC ← TOS, (W) ← k
Status Affected	-
OP-Code	01 1000 kkkk kkkk
Description	The W register is loaded with the 8-bit literal ‘k’. The stack is POPed and the Top-of-Stack (TOS) is loaded into the program counter (PC). This is a two-cycle instruction.
Cycle	2

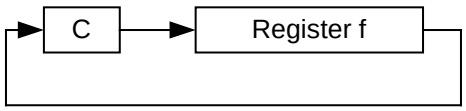
Example	LABEL: RETLW 08H	B : PC=LABEL,TOS=LABEL2,W = 41H
	LABEL1: ...	A : PC=LABEL2,W=08H

RLF Rotate Left f through Carry

Syntax	RLF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	
Status Affected	C
OP-Code	00 1101 dfff ffff
Description	The contents of register ‘f’ are rotated one bit to the left through the Carry Flag. If ‘d’ is 0, the result is placed in the W register. If ‘d’ is 1, the result is stored back in register ‘f’.
Cycle	1

Example	RLF REG1,0	B : REG1 = 1110 0110,W=02H, C = 0
		A : REG1 = 1110 0110
		W = 1100 1100, C = 1

RRF Rotate Right “f” through Carry

Syntax	RRF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	
Status Affected	C
OP-Code	00 1100 dfff ffff
Description	The contents of register ‘f’ are rotated one bit to the right through the Carry Flag. If ‘d’ is 0, the result is placed in the W register. If ‘d’ is 1, the result is placed back in register ‘f’.
Cycle	1

Example	RRF REG1,1	B : REG1 = 1110 0111,W=02H, C = 0
		A : REG1 = 0111 0011
		W = 02H, C = 1

SLEEP — Go into standby mode, Clock oscillation stops

Syntax	SLEEP
Operands	-
Operation	-
Status Affected	TO,PD
OP-Code	00 0000 0000 0011 (Instruction Set A) 01 1110 0000 0011 (Instruction Set B)
Description	Go into SLEEP mode with the oscillator stopped.
Cycle	1
Example	SLEEP B : CPU RUN,TO=?,PD=? A : CPU STOP,TO=0,PD=1

SUBWF — Subtract W from "f"

Syntax	SUBWF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(W) ← (f) – (W)
Status Affected	C, DC, Z
OP-Code	00 0010 dfff ffff
Description	Subtract (2's complement method) W register from register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.
Cycle	1
Example	SUBWF REG1,1 1>B : REG1 = 03H, W = 02H, C = ?, DC=?,Z = ? A : REG1 = 01H, W = 02H, C = 1, DC=1,Z = 0 2>B : REG1 =02H, W =02H, C = ?, DC=?,Z = ? A : REG1 = 00H, W = 02H, C = 1, DC=1,Z = 1 3>B : REG1 = 01H, W =02H, C = ?, DC=?,Z = ? A : REG1 = FFh, W = 02H, C = 0, DC=0,Z = 0

SWAPF — Swap Nibbles in "f"

Syntax	SWAPF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination,7~4) ← (f.3~0), (destination.3~0) ← (f.7~4)
Status Affected	-
OP-Code	00 1110 dfff ffff
Description	The upper and lower nibbles of register 'f' are exchanged. If 'd' is 0, the result is placed in W register. If 'd' is 1, the result is placed in register 'f'.
Cycle	1
Example	SWAPF REG, 0 B : REG1 = 0xA5, W = 0xFF A : REG1 = 0xA5, W = 0x5A

TBLWL Return DPTR low byte to W

Syntax	TBLWL
Operands	-
Operation	(W) ← ROM[DPTR] low byte content, Where DPTR={DPH[max:8],DPL[7:0]}
Status Affected	-
OP-Code	00 0000 0101 0000
Description	The W register is loaded with low byte of ROM[DPTR]. This is a two-cycle instruction.
Cycle	2
Example	

```
:
MOVLW   12H
MOVWF   DPH             ; Where DPH is F-plane register
MOVLW   34H
MOVWF   DPL             ; Where DPL is F-plane register
TBLWL                   ; W=0x86
TBLWH                   ; W=0x19


ORG 1234H
MOVLW   86H             ; Instruction code is 0x1986
ADDLW   97H             ; Instruction code is 0x1D97
```

TBLWH Return DPTR high byte to W

Syntax	TBLWH
Operands	-
Operation	(W) ← ROM[DPTR] high byte content, Where DPTR={DPH[max:8],DPL[7:0]}
Status Affected	-
OP-Code	00 0000 0101 1000
Description	The W register is loaded with high byte of ROM[DPTR]. This is a two-cycle instruction.
Cycle	2
Example	

```
:
MOVLW   12H
MOVWF   DPH             ; Where DPH is F-plane register
MOVLW   34H
MOVWF   DPL             ; Where DPL is F-plane register
TBLWL                   ; W=0x86
TBLWH                   ; W=0x19


ORG 1234H
MOVLW   86H             ; Instruction code is 0x1986
ADDLW   97H             ; Instruction code is 0x1D97
```

TESTZ	Test if “f” is zero	
Syntax	TESTZ f	
Operands	f : 00h ~ 7Fh	
Operation	Set Z flag if (f) is 0	
Status Affected	Z	
OP-Code	00 1000 1fff ffff	
Description	If the content of register ‘f’ is 0, Zero flag is set to 1.	
Cycle	1	
Example	TESTZ REG1	B : REG1 = 0, Z = ? A : REG1 = 0, Z = 1

XORLW	Exclusive OR Literal with W	
Syntax	XORLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← (W) XOR k	
Status Affected	Z	
OP-Code	01 1111 kkkk kkkk (Instruction Set A) 01 1101 kkkk kkkk (Instruction Set B)	
Description	The contents of the W register are XOR’ed with the eight-bit literal ‘k’. The result is placed in the W register.	
Cycle	1	
Example	XORLW 0xAF	B : W = 0xB5 , Z=? A : W = 0x1A , Z=0

XORWF	Exclusive OR W with “f”	
Syntax	XORWF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	(destination) ← (W) XOR (f)	
Status Affected	Z	
OP-Code	00 0110 dfff ffff	
Description	Exclusive OR the contents of the W register with register ‘f’. If ‘d’ is 0, the result is stored in the W register. If ‘d’ is 1, the result is stored back in register ‘f’.	
Cycle	1	
Example	XORWF REG 1	B : REG = 0xAF, W = 0xB5 , Z=? A : REG = 0x1A, W = 0xB5 , Z=0