

# Scenix SX instruction set notes

by Loren Blaney

This is the Scenix instruction set grouped by function. These mnemonics follow the Microchip standard rather than the Parallax standard used by Scenix.

ADDWF	f, d	C, DC, Z			
SUBWF	f, d	C, DC, Z			
ANDWF	f, d	Z	ANDLW	k	Z
IORWF	f, d	Z	IORLW	k	Z
XORWF	f, d	Z	XORLW	k	Z
MOVWF	f	-	MOVLW	k	-
MOVF	f, d	Z			
CLRF	f	Z	CLRW	-	Z
COMF	f, d	Z			
DECf	f, d	Z	INCF	f, d	Z
DECFSZ	f, d	-	INCFSZ	f, d	-
RLF	f, d	C	RRF	f, d	C
SWAPF	f, d	-			
BCF	f, b	-	BSF	f, b	-
BTFSC	f, b	-	BTFSS	f, b	-
GOTO	n	- (512)	* PAGE	n	PA2: PA0 (8)
CALL	k	- (256)			
* RET	-	-	RETLW	k	-
* RETP	-	PA2: PA0			
* RETI	-	All but T0, PD	* RETIW	-	All but T0, PD
* MOVWM	-	-	* MODE	n	- (16)
* MOVWV	-	-			
* IREAD	-	-			
* BANK	n	- (8)			
CLRWDI	-	T0, PD			
SLEEP	-	T0, PD			
OPTION	-	-			
TRIS	f	-			
NOP	-	-			

\* = new instruction not in PIC16C5x

## There is some confusion as to what the new instructions added by Scenix actually do. Here are some details.

MODE n - The four literal bits of "n" are loaded into the MODE register.

Status affected: None

Encoding: 0000 0101 mmm

Words: 1 Cycles: 1 (for both PIC16C5x compatible mode and turbo mode)

Example: MODE 0Eh ; After instruction, MODE register contains 1110

MOVWV - Copies the four bits in the MODE register into W. The high four bits of W are zeroed.

MOVWM - Copies the four low bits of W into the MODE register.

IREAD - Used to read 12-bit instruction memory. The contents of address (MODE:W) is copied into MODE:W. In other words: First store the high four bits of the address to be read into the MODE register and the low eight bits of the address into the W register. After the IREAD instruction is executed, the high four bits of the addressed location are in MODE and the low eight bits are in W.

The IREAD instruction has the interesting property that it takes the same amount of time to execute regardless of the TURBO\_ bit in FUSE (DEVICE TURBO). It takes 4 cycles in turbo mode and 1 cycle in compatible mode.

- RETI - Return from interrupt. The PC, W STATUS and FSR registers are restored to the values they had when the interrupt occurred. Shadow registers are used to hold these values; the subroutine return address stack is not used. Thus subroutines can be called to a depth of eight levels while interrupts are enabled.
- RETIW - Same as RETI but adds the contents of W to RTCC without affecting the prescaler. This provides a periodic interrupt that can be adjusted by 256 steps (minus the time it takes to execute the interrupt routine).
- RET - Same as RETLW but the W register is not affected. (BEWARE: MPASM without warning, assembles RETURN as RETLW 0.)
- RETP - Same as RET but the return address bits 11, 10 & 9 (on the stack) are written to the page-select bits PA2, PA1 & PA0 in the STATUS register. Thus the page-select bits are properly set to the page being returned to.
- PAGE n - The three literal bits of "n" are loaded into the page-select bits PA2:PA0 in the STATUS register. This normally prepares a GOTO or CALL across a page boundary. PA2 is only used in chips with 4K of instruction memory (SX48BD and SX52BD).
- BANK n - The three literal bits of "n" are loaded into the bank-select bits (the high three bits) in the FSR register. In PIC16C54 compatible mode, only one bank is enabled and the three high bits in FSR are always set. (INCF SZ FSR still skips when FSR is initially 0FFh even though the result is 0E0h.)
- PAGE and BANK cause an unusual operation when preceded by a skip instruction such as BTFSS and INCF SZ. Both the PAGE or BANK instruction and the following instruction are skipped, and the operation takes 3 cycles. If the following instruction happens to be another PAGE or BANK then it is also skipped along with the next instruction, and the operation takes 4 cycles. Any number of PAGE and BANK instructions can be skipped in this manner. Interrupts don't interfere with this operation.
- SUBWF - If the CF bit in FUSEX is 0 (DEVICE CARRYX), the complement of C is subtracted (borrowed) from the result. The intent of the CF bit is to make multiprecision adds and subtracts easier, as shown here:

```

;FF:= FF + GG with correct carry out (CF_ is 0)
  MOVF  G, W      ;add low bytes
  BCF   STATUS, CF ;clear carry
  ADDWF F
  MOVF  G+1, W    ;add high bytes and carry
  ADDWF F+1

;FF:= FF - GG with correct carry out (CF_ is 0)
  MOVF  G, W      ;subtract low bytes
  BSF   STATUS, CF ;set carry SUBWF F ; F - G
  MOVF  G+1, W    ;subtract high bytes SUBWF F+1

```

The following routines accomplish the same thing (although with an extra instruction), and they're compatible with the PIC16C5x:

```

;FF:= FF + GG with correct carry out (CF_ is 1)
  MOVF  G, W      ;add low bytes
  ADDWF F
  MOVF  G+1, W    ;get ready to add high bytes
  BTFSC STATUS, CF ;skip if there was no carry into high byte
  INCF  SZG+1, W   ;else add in carry; if result is 0 then high
  ADDWF F+1       ; byte doesn't change and CF is still set

;FF:= FF - GG with correct carry out (CF_ is 1)
  MOVF  G, W      ;subtract low bytes
  SUBWF F
  MOVF  G+1, W    ;get ready to subtract high bytes
  BTFSS STATUS, CF ;skip if there was no borrow from high byte
  INCF  SZG+1, W   ;else increase amount to subtract by 1; if it's
  SUBWF F+1       ; 0 then high byte doesn't change, nor does CF

```

Since the carry-out is correct in all four of these routines, they can easily be extended for more bytes of precision.

The PIC16C5x can do a double precision add in five instructions if there is a location set aside to hold a zero. Unfortunately, this can't be extended to more than two bytes of precision.

```

;FF:= FF + GG with correct carry out (CF_ is 1)
  MOVF  G, W      ;add low bytes
  ADDWF F
  RLF   ZERO, W   ;shift in carry forming either 01h or 00h
  ADDWF G+1, W    ;add carry and high bytes
  ADDWF F+1

```

# Microchip PIC16C5x instruction set summary with Scenix extensions

Mnemonic	Operands	Description	Cycles /Turbo	Status Affected	12-Bit Opcode		Notes
					MSB	LSB	
ADDWF	f, d	Add W and f	1	C, DC, Z	0001	11df ffff	1, 2, 4
ANDWF	f, d	AND W with f	1	Z	0001	01df ffff	2, 4
CLRF	f	Clear f	1	Z	0000	011f ffff	4
CLRW	-	Clear W	1	Z	0000	0100 0000	
COMF	f, d	Complement f	1	Z	0010	01df ffff	
DECF	f, d	Decrement f	1	Z	0000	11df ffff	2, 4
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	None	0010	11df ffff	2, 4
INCF	f, d	Increment f	1	Z	0010	10df ffff	2, 4
INCFSZ	f, d	Increment f, Skip if 0	1(2)	None	0011	11df ffff	2, 4
IORWF	f, d	Inclusive OR W with f	1	Z	0001	00df ffff	2, 4
MOVF	f, d	Move f	1	Z	0010	00df ffff	2, 4
MOVWF	f	Move W to f	1	None	0000	001f ffff	1, 4
NOP	-	No Operation	1	None	0000	0000 0000	
RLF	f, d	Rotate left f through Carry	1	C	0011	01df ffff	2, 4
RRF	f, d	Rotate right f through Carry	1	C	0011	00df ffff	2, 4
SUBWF	f, d	Subtract W from f	1	C, DC, Z	0000	10df ffff	1, 2, 4
SWAPF	f, d	Swap nibbles of f	1	None	0011	10df ffff	2, 4
XORWF	f, d	Exclusive OR W with f	1	Z	0001	10df ffff	2, 4

## Bit-oriented file register operations

BCF	f, b	Bit Clear f	1	None	0100	bbbf ffff	2, 4
BSF	f, b	Bit Set f	1	None	0101	bbbf ffff	2, 4
BTFSC	f, b	Bit Test f, Skip if Clear	1(2)	None	0110	bbbf ffff	
BTFSS	f, b	Bit Test f, Skip if Set	1(2)	None	0111	bbbf ffff	

## Literal and control operations

ANDLW	k	AND Literal with W	1	Z	1110	kkkk kkkk	
CALL	k	Call subroutine	2/3	None	1001	kkkk kkkk	1
CLRWDT	k	Clear Watchdog Timer. Pre=0, TO=1, PD=1	1	TO, PD	0000	0000 0100	
GOTO	k	Unconditional branch	2/3	None	101k	kkkk kkkk	
IORLW	k	Inclusive OR Literal with W	1	Z	1101	kkkk kkkk	
MOVLW	k	Move Literal to W	1	None	1100	kkkk kkkk	
OPTION	k	Load OPTION register with W	1	None	0000	0000 0010	
RETLW	k	Return (fr subr) move Literal to W	2/3	None	1000	kkkk kkkk	
SLEEP	-	Power down. WDT=0, Pre=0, TO=1, PD=0	1	TO, PD	0000	0000 0011	
TRIS	f	Move W into Port Control Register	1	None	0000	0000 0fff	3
XORLW	k	Exclusive OR Literal to W	1	Z	1111	kkkk kkkk	

## Scenix extensions

Mnemonic		Operands	Description	Cycles	Status	12-Bit Opcode		Notes
				/Turbo	Affected	MSB	LSB	
BANK	n		Write n into FSR7, FSR6 and FSR5	1	None	0000	0001 1nnn	5
IREAD	-		Read word at (MDE:W) into MDE:W	1/4	None	0000	0100 0001	
MDE	n		Write n into MDE register	1	None	0000	0101 nnnn	
MOVW	-		Mvve MDE bits to W (High nibble=0)	1	None	0000	0100 0010	
MOVW	-		Mvve W to MDE register	1	None	0000	0100 0011	
PAGE	n		Write n into PA2, PA1 and PA0	1	PA2, PA1, PA0	0000	0001 0nnn	5
RET	-		Return without affecting W	2/3	None	0000	0000 1100	
RETI	-		Return from interrupt. Pop PC, W, STATUS, FSR	2/3	All but TO, PD	0000	0000 1110	
RETIW	-		RETI and add W to RTCC	2/3	All but TO, PD	0000	0000 1111	
RETP	-		Return, W unaffected. Pop PA2, PA1, PA0	2/3	PA2, PA1, PA0	0000	0000 1101	

### Notes

1. The 9th bit of the program counter will be forced to a '0' by any instruction that writes to the PC except GOTO.
2. When an I/O register is modified as a function of itself (e.g. MOVF PORTB, 1; Test), the value used will be the value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, then the data used will be '0'.
3. The instruction TRIS f, where f = 5, 6 or 7 causes the contents of the W register to be written to the port control register selected by the MDE register. A '1' forces the pin to a high impedance state and disables the output buffers.
4. If this instruction is executed on the RTCC register (and, where applicable, d=1), the prescaler will be cleared if it is assigned to RTCC rather than the WDT.
5. If this instruction is immediately preceded by a skip instruction and the next instruction (such as BTFSS or DECFSZ, and the condition is true) then both this instruction and the next instruction will be skipped and the operation will consume 3 cycles.
6. Abbreviations:
  - b = bit 0..7 in f
  - C = carry status bit
  - d = destination: W or f (d=0 for W, d=1 for f)
  - DC = digit carry status bit
  - FSR = file select register
  - f = file register (RAM)
  - k = 8-bit constant (literal) (256)
  - n = constant
  - PAn = page select status bits (n = 0, 1, 2)
  - PD = power-down (sleep-mode) status bit (true low)
  - TO = watchdog timer time-out status bit (true low)
  - W = working register (accumulator)
  - Z = zero status bit