

Welcome to E-XFL.COM

#### What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

#### Applications of "<u>Embedded -</u> <u>Microcontrollers</u>"

#### Details

E·XFI

Product Status	Obsolete
Core Processor	PIC
Core Size	16-Bit
Speed	32MHz
Connectivity	I <sup>2</sup> C, IrDA, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	69
Program Memory Size	64KB (22K x 24)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	16K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 3.6V
Data Converters	A/D 16x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	80-TQFP
Supplier Device Package	80-TQFP (12x12)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic24fj64ga108-e-pt

Email: info@E-XFL.COM

Address: Room A, 16/F, Full Win Commercial Centre, 573 Nathan Road, Mongkok, Hong Kong

		Pin Number				
Function	64-Pin TQFP, QFN	80-Pin TQFP	100-Pin TQFP	VO	Buffer	Description
CN0	48	60	74	I	ST	Interrupt-on-Change Inputs.
CN1	47	59	73	I	ST	
CN2	16	20	25	I	ST	
CN3	15	19	24	I	ST	
CN4	14	18	23	I	ST	
CN5	13	17	22	I	ST	
CN6	12	16	21	Ι	ST	
CN7	11	15	20	Ι	ST	
CN8	4	6	10	I	ST	
CN9	5	7	11	I	ST	
CN10	6	8	12	I	ST	
CN11	8	10	14	I	ST	
CN12	30	36	44	I	ST	
CN13	52	66	81	I	ST	
CN14	53	67	82	I	ST	
CN15	54	68	83	I	ST	
CN16	55	69	84	I	ST	
CN17	31	39	49	I	ST	
CN18	32	40	50	I	ST	
CN19	—	65	80	I	ST	
CN20	—	37	47	I	ST	
CN21	—	38	48	I	ST	
CN22	40	50	64	I	ST	
CN23	39	49	63	I	ST	
CN24	17	21	26	I	ST	
CN25	18	22	27	I	ST	
CN26	21	27	32	I	ST	
CN27	22	28	33	I	ST	
CN28	23	29	34	I	ST	
CN29	24	30	35	I	ST	
CN30	27	33	41	I	ST	
CN31	28	34	42	I	ST	
CN32	29	35	43	I	ST	
CN33	—	—	17	I	ST	
CN34	—	—	38	I	ST	
CN35	—	—	58	I	ST	-
CN36	—	—	59	I	ST	-
CN37	—	—	60		ST	
CN38	—	—	61	I	ST	-
CN39	—	—	91	I	ST	
CN40	—	—	92	I	ST	-
CN41	—	23	28	I	ST	
CN42	—	24	29	I	ST	

#### **TABLE 1-4:** PIC24FJ256GA110 FAMILY PINOUT DESCRIPTIONS (CONTINUED)

= TTL input buffer Legend: TTL ANA = Analog level input/output ST = Schmitt Trigger input buffer  $I^2C^{TM} = I^2C/SMBus$  input buffer

		Pin Number				
Function	64-Pin TQFP, QFN	80-Pin TQFP	100-Pin TQFP	I/O	Input Buffer	Description
RPI32	—	_	40	I	ST	Remappable Peripheral (input only).
RPI33	_	13	18	I	ST	
RPI34	_	14	19	I	ST	
RPI35	_	53	67	I	ST	
RPI36	—	52	66	I	ST	
RPI37	48	60	74	I	ST	
RPI38	_	4	6	I	ST	
RPI39	—		7	I	ST	
RPI40	_	5	8	I	ST	
RPI41	_		9	I	ST	
RPI42	_	64	79	I	ST	
RPI43	_	37	47	I	ST	
RPI44	_	44	54	I	ST	
RPI45	35	45	55	I	ST	
RTCC	42	54	68	0	_	Real-Time Clock Alarm/Seconds Pulse Output.
SCL1	37	47	57	I/O	l <sup>2</sup> C	I2C1 Synchronous Serial Clock Input/Output.
SCL2	32	52	58	I/O	l <sup>2</sup> C	I2C2 Synchronous Serial Clock Input/Output.
SCL3	2	2	4	I/O	l <sup>2</sup> C	I2C3 Synchronous Serial Clock Input/Output.
SDA1	36	46	56	I/O	l <sup>2</sup> C	I2C1 Data Input/Output.
SDA2	31	53	59	I/O	l <sup>2</sup> C	I2C2 Data Input/Output.
SDA3	3	3	5	I/O	l <sup>2</sup> C	I2C3 Data Input/Output.
SOSCI	47	59	73	I	ANA	Secondary Oscillator/Timer1 Clock Input.
SOSCO	48	60	74	0	ANA	Secondary Oscillator/Timer1 Clock Output.
T1CK	48	60	74	I	ST	Timer1 Clock.
TCK	27	33	38	Ι	ST	JTAG Test Clock Input.
TDI	28	34	60	I	ST	JTAG Test Data Input.
TDO	24	14	61	0	—	JTAG Test Data Output.
TMS	23	13	17	Ι	ST	JTAG Test Mode Select Input.
VCAP	56	70	85	Р	_	External Filter Capacitor Connection (regulator enabled).
Vdd	10, 26, 38	12, 32, 48	2, 16, 37, 46, 62	Ρ	—	Positive Supply for Peripheral Digital Logic and I/O Pins.
VDDCORE	56	70	85	Р	—	Positive Supply for Microcontroller Core Logic (regulator disabled).
VREF-	15	23	28	Ι	ANA	A/D and Comparator Reference Voltage (low) Input.
VREF+	16	24	29	Ι	ANA	A/D and Comparator Reference Voltage (high) Input.
Vss	9, 25, 41	11, 31, 51	15, 36, 45, 65, 75	Р	_	Ground Reference for Logic and I/O Pins.

#### TABLE 1-4: PIC24FJ256GA110 FAMILY PINOUT DESCRIPTIONS (CONTINUED)

Legend: TTL = TTL input buffer ANA = Analog level input/output ST = Schmitt Trigger input buffer  $I^2C^{TM} = I^2C/SMBus$  input buffer

DS39905E-page 22

### 2.0 GUIDELINES FOR GETTING STARTED WITH 16-BIT MICROCONTROLLERS

#### 2.1 Basic Connection Requirements

Getting started with the PIC24FJ256GA110 family family of 16-bit microcontrollers requires attention to a minimal set of device pin connections before proceeding with development.

The following pins must always be connected:

- All VDD and Vss pins (see Section 2.2 "Power Supply Pins")
- All AVDD and AVss pins, regardless of whether or not the analog device features are used (see Section 2.2 "Power Supply Pins")
- MCLR pin (see Section 2.3 "Master Clear (MCLR) Pin")
- ENVREG/DISVREG and VCAP/VDDCORE pins (PIC24F J devices only) (see Section 2.4 "Voltage Regulator Pins (ENVREG/DISVREG and VCAP/VDDCORE)")

These pins must also be connected if they are being used in the end application:

- PGECx/PGEDx pins used for In-Circuit Serial Programming<sup>™</sup> (ICSP<sup>™</sup>) and debugging purposes (see **Section 2.5 "ICSP Pins"**)
- OSCI and OSCO pins when an external oscillator source is used

(see Section 2.6 "External Oscillator Pins")

Additionally, the following pins may be required:

• VREF+/VREF- pins used when external voltage reference for analog modules is implemented

**Note:** The AVDD and AVss pins must always be connected, regardless of whether any of the analog modules are being used.

The minimum mandatory connections are shown in Figure 2-1.



#### Key (all values are recommendations):

C1 through C6: 0.1 µF, 20V ceramic

C7: 10  $\mu\text{F},\,6.3\text{V}$  or greater, tantalum or ceramic

R1: 10 kΩ

R2: 100Ω to 470Ω

- Note 1: See Section 2.4 "Voltage Regulator Pins (ENVREG/DISVREG and VCAP/VDDCORE)" for explanation of ENVREG/DISVREG pin connections.
  - 2: The example shown is for a PIC24F device with five VDD/VSs and AVDD/AVSs pairs. Other devices may have more or less pairs; adjust the number of decoupling capacitors appropriately.



#### 4.2.2 DATA MEMORY ORGANIZATION AND ALIGNMENT

To maintain backward compatibility with  $PIC^{\circledast}$  devices and improve data space memory usage efficiency, the PIC24F instruction set supports both word and byte operations. As a consequence of byte accessibility, all Effective Address (EA) calculations are internally scaled to step through word-aligned memory. For example, the core recognizes that Post-Modified Register Indirect Addressing mode [Ws++] will result in a value of Ws + 1 for byte operations and Ws + 2 for word operations.

Data byte reads will read the complete word which contains the byte, using the LSb of any EA to determine which byte to select. The selected byte is placed onto the LSB of the data path. That is, data memory and registers are organized as two parallel, byte-wide entities with shared (word) address decode, but separate write lines. Data byte writes only write to the corresponding side of the array or register which matches the byte address.

All word accesses must be aligned to an even address. Misaligned word data fetches are not supported, so care must be taken when mixing byte and word operations or translating from 8-bit MCU code. If a misaligned read or write is attempted, an address error trap will be generated. If the error occurred on a read, the instruction underway is completed; if it occurred on a write, the instruction will be executed but the write will not occur. In either case, a trap is then executed, allowing the system and/or user to examine the machine state prior to execution of the address Fault.

All byte loads into any W register are loaded into the Least Significant Byte. The Most Significant Byte is not modified.

A Sign-Extend (SE) instruction is provided to allow users to translate 8-bit signed data to 16-bit signed values. Alternatively, for 16-bit unsigned data, users can clear the MSB of any W register by executing a Zero-Extend (ZE) instruction on the appropriate address.

Although most instructions are capable of operating on word or byte data sizes, it should be noted that some instructions operate only on words.

#### 4.2.3 NEAR DATA SPACE

The 8-Kbyte area between 0000h and 1FFFh is referred to as the near data space. Locations in this space are directly addressable via a 13-bit absolute address field within all memory direct instructions. The remainder of the data space is indirectly addressable. Additionally, the whole data space is addressable using MOV instructions, which support Memory Direct Addressing with a 16-bit address field.

#### 4.2.4 SFR SPACE

The first 2 Kbytes of the near data space, from 0000h to 07FFh, are primarily occupied with Special Function Registers (SFRs). These are used by the PIC24F core and peripheral modules for controlling the operation of the device.

SFRs are distributed among the modules that they control and are generally grouped together by module. Much of the SFR space contains unused addresses; these are read as '0'. A diagram of the SFR space, showing where SFRs are actually implemented, is shown in Table 4-2. Each implemented area indicates a 32-byte region where at least one address is implemented as an SFR. A complete listing of implemented SFRs, including their addresses, is shown in Tables 4-3 through 4-29.

	SFR Space Address											
	xx00         xx20         xx40         xx60         xx80         xxA0         xxC0         xxE0											
000h		Core		ICN	Interrupts				—			
100h	Tin	ners	(	Capture Compare								
200h	l <sup>2</sup> C™	UART	SPI/UART	SPI/UART SPI/I <sup>2</sup> C S			UART I/O					
300h	A/D	A/D/CTMU	_	—	-	_	—	_	_			
400h	—	—	_	_	_	_			—			
500h	_	_	_	_	-	_	_	_	_			
600h	PMP	RTC/Comp	CRC	_			PPS		—			
700h	—	—	System	NVM/PMD	_	_	—	—	—			

 TABLE 4-2:
 IMPLEMENTED REGIONS OF SFR DATA SPACE

**Legend:** — = No implemented SFRs in this block

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
		IC9IE	OC9IE	SPI3IE	SPF3IE	U4TXIE	U4RXIE
bit 15							bit 8
R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0
U4ERIE	—	MI2C3IE	SI2C3IE	U3TXIE	U3RXIE	U3ERIE	—
bit 7							bit 0
Languate							
Legena:	a hit	VV – Writeble	-it		nonted hit read		
$R = Reauable}{n = Value at}$		'1' = Bit is set	UIL	0' = 0	ared	v – Bitis unkr	
	TOR				aleu		lowin
bit 15-14	Unimplemer	nted: Read as '	)'				
bit 13	IC9IE: Input	Capture Channe	el 9 Interrupt E	nable bit			
	1 = Interrupt	request enabled	3				
	0 = Interrupt	request not ena	bled				
bit 12	OC9IE: Outp	ut Compare Ch	annel 9 Interru	pt Enable bit			
	1 = Interrupt	request enabled	) blod				
bit 11			Enchlo hit				
	1 = Interrunt	request enabled					
	0 = Interrupt	request not ena	bled				
bit 10	SPF3IE: SPI	3 Fault Interrup	Enable bit				
	1 = Interrupt	request enabled	ł				
	0 = Interrupt	request not ena	bled				
bit 9	U4TXIE: UA	RT4 Transmitter	Interrupt Enal	ble bit			
	1 = Interrupt	request enabled	) blad				
hit Q		PT4 Receiver In	bieu	, bit			
DILO	1 = Interrupt	request enabled	iterrupt Enable				
	0 = Interrupt	request not ena	bled				
bit 7	U4ERIE: UA	RT4 Error Interr	upt Enable bit				
	1 = Interrupt	request enabled					
	0 = Interrupt	request not ena	bled				
bit 6	Unimplemer	nted: Read as '0	)'				
bit 5	MI2C3IE: Ma	ster I2C3 Event	t Interrupt Ena	ble bit			
	$\perp = Interrupt$ 0 = Interrupt	request enabled	) bled				
bit 4	SI2C3IE: Sla	ve I2C3 Event I	nterrupt Enabl	e bit			
	1 = Interrupt	request enabled	d	0.510			
	0 = Interrupt	request not ena	bled				
bit 3	U3TXIE: UA	RT3 Transmitter	Interrupt Enal	ble bit			
	1 = Interrupt	request enabled	1 				
	0 = Interrupt	request not ena	bled				
bit 2	1 = Interrupt	R13 Receiver Ir	iterrupt Enable	bit			
	0 = Interrupt	request not ena	bled				
bit 1	U3ERIE: UA	RT3 Error Interr	upt Enable bit				
-	1 = Interrupt	request enabled	1				
	0 = Interrupt	request not ena	bled				
bit 0	Unimplemer	nted: Read as 'o	)'				

#### REGISTER 7-16: IEC5: INTERRUPT ENABLE CONTROL REGISTER 5

REGISTER 7-21:	<b>IPC4: INTERRUPT PRIORITY CONTROL REGISTER 4</b>
----------------	--

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
	CNIP2	CNIP1	CNIP0	_	CMIP2	CMIP1	CMIP0
bit 15					I		bit 8
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
	MI2C1IP2	MI2C1IP1	MI2C1IP0	<u> </u>	SI2C1IP2	SI2C1IP1	SI2C1IP0
bit 7							bit 0
Legend:							
R = Readable	e bit	W = Writable	bit	U = Unimplem	nented bit, read	l as '0'	
-n = Value at	POR	'1' = Bit is set		'0' = Bit is clea	ared	x = Bit is unkr	nown
6:4 <i>4 C</i>		tad. Deed as '	<b>、</b> ,				
		ted: Read as (	) atification later	www.undt Duinewider, bit	-		
DIT 14-12	111 - Intorru	nput Change N	ouncation inter	interrupt)	5		
	•		lighest phonty	interrupt)			
	•						
	•						
	001 = Interru	pt is priority 1 pt source is dis:	abled				
bit 11		ted: Read as '	)'				
bit 10-8	CMIP<2:0>: (	Comparator Inte	errupt Priority b	oits			
	111 = Interru	, pt is priority 7 (ł	nighest priority	interrupt)			
	•						
	•						
	• 001 = Interru	ot is priority 1					
	000 = Interru	pt source is disa	abled				
bit 7	Unimplemen	ted: Read as 'd	)'				
bit 6-4	MI2C1IP<2:0	>: Master I2C1	Event Interrup	t Priority bits			
	111 = Interru	pt is priority 7 (ł	nighest priority	interrupt)			
	•						
	•						
	001 = Interru	pt is priority 1					
	000 = Interru	pt source is disa	abled				
bit 3	Unimplemen	ted: Read as '0	)'				
bit 2-0	SI2C1IP<2:0	Slave I2C1 E	vent Interrupt	Priority bits			
	111 = Interru	pt is priority 7 (ł	nighest priority	interrupt)			
	•						
	•						
	001 = Interru	pt is priority 1					
	000 = Interru	pt source is disa	abled				

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
_	CRCIP2	CRCIP1	CRCIP0	_	U2ERIP2	U2ERIP1	U2ERIP0
bit 15							bit 8
U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
	U1ERIP2	U1ERIP1	U1ERIP0			—	—
bit 7							bit 0
Legend:							
R = Readab	ole bit	W = Writable	bit	U = Unimpler	mented bit, read	d as '0'	
-n = Value a	at POR	'1' = Bit is set		'0' = Bit is cle	ared	x = Bit is unkr	nown
bit 15	Unimplemen	ited: Read as '	0'				
bit 14-12	CRCIP<2:0>	: CRC Generat	or Error Interru	upt Priority bits			
	111 = Interru	pt is priority 7 (	highest priority	(interrupt)			
	•						
	•						
	001 = Interru 000 = Interru	pt is priority 1	abled				
bit 11	Unimplemen	ted: Read as '	0'				
bit 10-8	U2ERIP<2:0:	>: UART2 Error	<sup>-</sup> Interrupt Prio	rity bits			
	111 = Interru	pt is priority 7 (	highest priority	/ interrupt)			
	•						
	•						
	• 001 = Interru	nt is priority 1					
	000 = Interru	pt source is dis	abled				
bit 7	Unimplemen	ited: Read as '	0'				
bit 6-4	U1ERIP<2:0:	>: UART1 Erro	<sup>-</sup> Interrupt Prio	rity bits			
	111 = Interru	pt is priority 7 (	highest priority	/ interrupt)			
	•						
	•						
	• 001 = Interru	nt is priority 1					
	000 = Interru	pt source is dis	abled				
bit 3-0	Unimplemen	ted: Read as '	0'				

#### REGISTER 7-32: IPC16: INTERRUPT PRIORITY CONTROL REGISTER 16

#### 10.4.2 AVAILABLE PERIPHERALS

The peripherals managed by the Peripheral Pin Select are all digital only peripherals. These include general serial communications (UART and SPI), general purpose timer clock inputs, timer related peripherals (input capture and output compare) and external interrupt inputs. Also included are the outputs of the comparator module, since these are discrete digital signals.

Peripheral Pin Select is not available for  $I^2C^{TM}$ , change notification inputs, RTCC alarm outputs or peripherals with analog inputs.

A key difference between pin select and non pin select peripherals is that pin select peripherals are not associated with a default I/O pin. The peripheral must always be assigned to a specific I/O pin before it can be used. In contrast, non pin select peripherals are always available on a default pin, assuming that the peripheral is active and not conflicting with another peripheral.

#### 10.4.2.1 Peripheral Pin Select Function Priority

Pin-selectable peripheral outputs (e.g. OC, UART Transmit) take priority over general purpose digital functions on a pin, such as PMP and port I/O. Specialized digital outputs, such as USB functionality, will take priority over PPS outputs on the same pin. The pin diagrams provided at the beginning of this data sheet list peripheral outputs in order of priority. Refer to them for priority concerns on a particular pin.

Unlike PIC24F devices with fixed peripherals, pin-selectable peripheral inputs never take ownership of a pin. The pin's output buffer is controlled by the TRISx setting or by a fixed peripheral on the pin. If the pin is configured in Digital mode, the PPS input will operate correctly. If an analog function is enabled on the pin, the PPS input will be disabled.

#### 10.4.3 CONTROLLING PERIPHERAL PIN SELECT

Peripheral Pin Select features are controlled through two sets of Special Function Registers: one to map peripheral inputs and one to map outputs. Because they are separately controlled, a particular peripheral's input and output (if the peripheral has both) can be placed on any selectable function pin without constraint.

The association of a peripheral to a peripheral-selectable pin is handled in two different ways, depending on if an input or an output is being mapped.

#### 10.4.3.1 Input Mapping

The inputs of the Peripheral Pin Select options are mapped on the basis of the peripheral; that is, a control register associated with a peripheral dictates the pin it will be mapped to. The RPINRx registers are used to configure peripheral input mapping (see Register 10-1 through Register 10-21). Each register contains two sets of 6-bit fields, with each set associated with one of the pin-selectable peripherals. Programming a given peripheral's bit field with an appropriate 6-bit value maps the RPn pin with that value to that peripheral. For any given device, the valid range of values for any of the bit fields corresponds to the maximum number of Peripheral Pin Select options supported by the device.

#### 10.4.3.2 Output Mapping

In contrast to inputs, the outputs of the Peripheral Pin Select options are mapped on the basis of the pin. In this case, a control register associated with a particular pin dictates the peripheral output to be mapped. The RPORx registers are used to control output mapping. Each register contains two 6-bit fields, with each field being associated with one RPn pin (see Register 10-22 through Register 10-37). The value of the bit field corresponds to one of the peripherals and that peripheral's output is mapped to the pin (see Table 10-3).

Because of the mapping technique, the list of peripherals for output mapping also includes a null value of '000000'. This permits any given pin to remain disconnected from the output of any of the pin-selectable peripherals.

#### 10.4.3.3 Alternate Fixed Pin Mapping

To provide a migration option from earlier high pin count PIC24F devices, PIC24FJ256GA110 family devices implement an additional option for mapping the clock output (SCK) of SPI1. This option permits users to map SCK10UT specifically to the fixed pin function, ASCK1. The SCK1CM bit (ALTRP<0>) controls this mapping; setting the bit maps SCK10UT to ASCK1.

The SCK1CM bit must be set (= 1) before enabling the SPI module. It must remain set while transactions using SPI1 are in progress, in order to prevent transmission errors; when the module is disabled, the bit must be cleared. Additionally, no other RPOUT register should be configured to output the SCK1OUT function while SCK1CM is set.

#### REGISTER 10-9: RPINR10: PERIPHERAL PIN SELECT INPUT REGISTER 10

U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	IC8R5	IC8R4	IC8R3	IC8R2	IC8R1	IC8R0
bit 15							bit 8
U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	IC7R5	IC7R4	IC7R3	IC7R2	IC7R1	IC7R0
bit 7							bit 0
Legend:							
R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'							
-n = Value at POR '1' = Bit is set '0' = Bit is cleared x = Bit is unl			x = Bit is unkr	nown			
							, ,

bit 15-14	Unimplemented: Read as '0'
bit 13-8	IC8R<5:0>: Assign Input Capture 8 (IC8) to Corresponding RPn or RPIn Pin bits
bit 7-6	Unimplemented: Read as '0'
bit 5-0	IC7R<5:0>: Assign Input Capture 7 (IC7) to Corresponding RPn or RPIn Pin bits

#### REGISTER 10-10: RPINR11: PERIPHERAL PIN SELECT INPUT REGISTER 11

U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	OCFBR5	OCFBR4	OCFBR3	OCFBR2	OCFBR1	OCFBR0
bit 15							bit 8

U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
_	—	OCFAR5	OCFAR4	OCFAR3	OCFAR2	OCFAR1	OCFAR0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read	l as '0'
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 15-14 Unimplemented: Read as '0'

bit 13-8 **OCFBR<5:0>:** Assign Output Compare Fault B (OCFB) to Corresponding RPn or RPIn Pin bits

bit 7-6 Unimplemented: Read as '0'

bit 5-0 OCFAR<5:0>: Assign Output Compare Fault A (OCFA) to Corresponding RPn or RPIn Pin bits

#### REGISTER 10-36: RPOR14: PERIPHERAL PIN SELECT OUTPUT REGISTER 14

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	—	RP29R5	RP29R4	RP29R3	RP29R2	RP29R1	RP29R0
bit 15							bit 8

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	RP28R5	RP28R4	RP28R3	RP28R2	RP28R1	RP28R0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read	1 as '0'
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 15-14	Unimplemented: Read as '0'
bit 13-8	RP29R<5:0>: RP29 Output Pin Mapping bits
	Peripheral output number n is assigned to pin, RP29 (see Table 10-3 for peripheral function numbers).
bit 7-6	Unimplemented: Read as '0'
bit 5-0	RP28R<5:0>: RP28 Output Pin Mapping bits
	Peripheral output number n is assigned to pin, RP28 (see Table 10-3 for peripheral function numbers).

#### REGISTER 10-37: RPOR15: PERIPHERAL PIN SELECT OUTPUT REGISTER 15

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	—	RP31R5 <sup>(1)</sup>	RP31R4 <sup>(1)</sup>	RP31R3 <sup>(1)</sup>	RP31R2 <sup>(1)</sup>	RP31R1 <sup>(1)</sup>	RP31R0 <sup>(1)</sup>
bit 15							bit 8

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	RP30R5	RP30R4	RP30R3	RP30R2	RP30R1	RP30R0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read	l as '0'
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 15-14 Unimplemented: Read as '0'

bit 13-8**RP31R<5:0>:** RP31 Output Pin Mapping bits<sup>(1)</sup><br/>Peripheral output number n is assigned to pin, RP31 (see Table 10-3 for peripheral function numbers).bit 7-6**Unimplemented:** Read as '0'

bit 5-0 **RP30R<5:0>:** RP30 Output Pin Mapping bits Peripheral output number n is assigned to pin, RP30 (see Table 10-3 for peripheral function numbers).

Note 1: Unimplemented in 64-pin and 80-pin devices; read as '0'.

#### REGISTER 10-38: ALTRP: ALTERNATE PERIPHERAL PIN MAPPING REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
_	—	—	—	—	—		—
bit 15							bit 8
U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
		—	—	—	—	—	SCK1CM
bit 7			•	•			bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read	d as '0'
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 15-1 Unimplemented: Read as '0'

bit 0

SCK1CM: SCK1 Output Mapping Select bit

1 = SCK1 output function is mapped to ASCK1 pin only

0 = SCK1 output function is mapped according to RPORn registers

To set up the SPI module for the Standard Master mode of operation:

- 1. If using interrupts:
  - a) Clear the SPIxIF bit in the respective IFSx register.
  - b) Set the SPIxIE bit in the respective IECx register.
  - c) Write the SPIxIP bits in the respective IPCx register to set the interrupt priority.
- Write the desired settings to the SPIxCON1 and SPIxCON2 registers with the MSTEN bit (SPIxCON1<5>) = 1.
- 3. Clear the SPIROV bit (SPIxSTAT<6>).
- 4. Enable SPI operation by setting the SPIEN bit (SPIxSTAT<15>).
- Write the data to be transmitted to the SPIxBUF register. Transmission (and reception) will start as soon as data is written to the SPIxBUF register.

To set up the SPI module for the Standard Slave mode of operation:

- 1. Clear the SPIxBUF register.
- 2. If using interrupts:
  - a) Clear the SPIxIF bit in the respective IFSx register.
  - b) Set the SPIxIE bit in the respective IECx register.
  - c) Write the SPIxIP bits in the respective IPCx register to set the interrupt priority.
- Write the desired settings to the SPIxCON1 and SPIxCON2 registers with the MSTEN bit (SPIxCON1<5>) = 0.
- 4. Clear the SMP bit.
- If the CKE bit is set, then the SSEN bit (SPIxCON1<8>) must be set to enable the SSx pin.
- 6. Clear the SPIROV bit (SPIxSTAT<6>).
- 7. Enable SPI operation by setting the SPIEN bit (SPIxSTAT<15>).

#### FIGURE 15-1: SPIX MODULE BLOCK DIAGRAM (STANDARD MODE)



#### REGISTER 16-3: I2CxMSK: I2Cx SLAVE MODE ADDRESS MASK REGISTER

bit 15							bit 8
—	_	—	—	—	—	AMSK9	AMSK8
U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0

| R/W-0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| AMSK7 | AMSK6 | AMSK5 | AMSK4 | AMSK3 | AMSK2 | AMSK1 | AMSK0 |
| bit 7 |       |       |       |       |       |       | bit 0 |

Legend:					
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'			
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown		

bit 15-10 Unimplemented: Read as '0'

bit 9-0

AMSK<9:0>: Mask for Address Bit x Select bits

1 = Enable masking for bit x of incoming message address; bit match not required in this position
 0 = Disable masking for bit x; bit match required in this position

DS39905E-page 192

#### REGISTER 19-10: ALMINSEC: ALARM MINUTES AND SECONDS VALUE REGISTER

U-0	R/W-x						
—	MINTEN2	MINTEN1	MINTEN0	MINONE3	MINONE2	MINONE1	MINONE0
bit 15							bit 8

U-0	R/W-x						
—	SECTEN2	SECTEN1	SECTEN0	SECONE3	SECONE2	SECONE1	SECONE0
bit 7							bit 0

Legend:					
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'			
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown		

bit 15	Unimplemented: Read as '0'
bit 14-12	<b>MINTEN&lt;2:0&gt;:</b> Binary Coded Decimal Value of Minute's Tens Digit bits Contains a value from 0 to 5.
bit 11-8	MINONE<3:0>: Binary Coded Decimal Value of Minute's Ones Digit bits
	Contains a value from 0 to 9.
bit 7	Unimplemented: Read as '0'
bit 6-4	SECTEN<2:0>: Binary Coded Decimal Value of Second's Tens Digit bits
	Contains a value from 0 to 5.
bit 3-0	SECONE<3:0>: Binary Coded Decimal Value of Second's Ones Digit bits
	Contains a value from 0 to 9.

#### 19.2 Calibration

The real-time crystal input can be calibrated using the periodic auto-adjust feature. When properly calibrated, the RTCC can provide an error of less than 3 seconds per month. This is accomplished by finding the number of error clock pulses for one minute and storing the value into the lower half of the RCFGCAL register. The 8-bit signed value loaded into the lower half of RCFGCAL is multiplied by four and will be either added or subtracted from the RTCC timer, once every minute. Refer to the steps below for RTCC calibration:

- 1. Using another timer resource on the device, the user must find the error of the 32.768 kHz crystal.
- 2. Once the error is known, it must be converted to the number of error clock pulses per minute and loaded into the RCFGCAL register.

#### EQUATION 19-1: RTCC CALIBRATION

Error (Clocks per Minute) = (Ideal Frequency<sup>†</sup> – Measured Frequency) \* 60 = Clocks per Minute † Ideal frequency = 32,768 Hz 3. a) If the oscillator is faster then ideal (negative result form Step 2), the RCFGCAL register value needs to be negative. This causes the specified number of clock pulses to be subtracted from the timer counter once every minute.

b) If the oscillator is slower then ideal (positive result from Step 2) the RCFGCAL register value needs to be positive. This causes the specified number of clock pulses to be added from the timer counter once every minute.

 Divide the number of error clocks per minute by 4 to get the correct CAL value and load the RCFGCAL register with the correct value.

(Each 1-bit increment in CAL adds or subtracts 4 pulses.)

Writes to the lower half of the RCFGCAL register should only occur when the timer is turned off, or immediately after the rising edge of the seconds pulse.

**Note:** It is up to the user to include in the error value the initial error of the crystal, drift due to temperature and drift due to crystal aging.

NOTES:

Assembly Mnemonic		Assembly Syntax	Description	# of Words	# of Cycles	Status Flags Affected
GOTO	GOTO	Expr	Go to Address	2	2	None
	GOTO	Wn	Go to Indirect	1	2	None
INC	INC	f	f = f + 1	1	1	C, DC, N, OV, Z
	INC	f,WREG	WREG = f + 1	1	1	C, DC, N, OV, Z
	INC	Ws,Wd	Wd = Ws + 1	1	1	C, DC, N, OV, Z
INC2	INC2	f	f = f + 2	1	1	C, DC, N, OV, Z
	INC2	f,WREG	WREG = f + 2	1	1	C, DC, N, OV, Z
	INC2	Ws,Wd	Wd = Ws + 2	1	1	C, DC, N, OV, Z
IOR	IOR	£	f = f .IOR. WREG	1	1	N, Z
	IOR	f,WREG	WREG = f .IOR. WREG	1	1	N, Z
	IOR	#lit10,Wn	Wd = lit10 .IOR. Wd	1	1	N, Z
	IOR	Wb,Ws,Wd	Wd = Wb .IOR. Ws	1	1	N, Z
	IOR	Wb,#lit5,Wd	Wd = Wb .IOR. lit5	1	1	N, Z
LNK	LNK	#lit14	Link Frame Pointer	1	1	None
LSR	LSR	f	f = Logical Right Shift f	1	1	C, N, OV, Z
	LSR	f,WREG	WREG = Logical Right Shift f	1	1	C, N, OV, Z
	LSR	Ws,Wd	Wd = Logical Right Shift Ws	1	1	C, N, OV, Z
	LSR	Wb,Wns,Wnd	Wnd = Logical Right Shift Wb by Wns	1	1	N, Z
	LSR	Wb,#lit5,Wnd	Wnd = Logical Right Shift Wb by lit5	1	1	N, Z
MOV	MOV	f,Wn	Move f to Wn	1	1	None
	MOV	[Wns+Slit10],Wnd	Move [Wns+Slit10] to Wnd	1	1	None
	MOV	f	Move f to f	1	1	N, Z
	MOV	f,WREG	Move f to WREG	1	1	N, Z
	MOV	#litl6,Wn	Move 16-bit Literal to Wn	1	1	None
	MOV.b	#lit8,Wn	Move 8-bit Literal to Wn	1	1	None
	MOV	Wn,f	Move Wn to f	1	1	None
	MOV	Wns,[Wns+Slit10]	Move Wns to [Wns+Slit10]	1	1	
	MOV	Wso,Wdo	Move Ws to Wd	1	1	None
	MOV	WREG,f	Move WREG to f	1	1	N, Z
	MOV.D	Wns,Wd	Move Double from W(ns):W(ns + 1) to Wd	1	2	None
	MOV.D	Ws,Wnd	Move Double from Ws to W(nd + 1):W(nd)	1	2	None
MUL	MUL.SS	Wb,Ws,Wnd	{Wnd + 1, Wnd} = Signed(Wb) * Signed(Ws)	1	1	None
	MUL.SU	Wb,Ws,Wnd	{Wnd + 1, Wnd} = Signed(Wb) * Unsigned(Ws)	1	1	None
	MUL.US	Wb,Ws,Wnd	{Wnd + 1, Wnd} = Unsigned(Wb) * Signed(Ws)	1	1	None
	MUL.UU	Wb,Ws,Wnd	{Wnd + 1, Wnd} = Unsigned(Wb) * Unsigned(Ws)	1	1	None
	MUL.SU	Wb,#lit5,Wnd	{Wnd + 1, Wnd} = Signed(Wb) * Unsigned(lit5)	1	1	None
	MUL.UU	Wb,#lit5,Wnd	{Wnd + 1, Wnd} = Unsigned(Wb) * Unsigned(lit5)	1	1	None
	MUL	£	W3:W2 = f * WREG	1	1	None
NEG	NEG	f	$f = \overline{f} + 1$	1	1	C, DC, N, OV, Z
	NEG	f,WREG	WREG = $\overline{f}$ + 1	1	1	C, DC, N, OV, Z
	NEG	Ws,Wd	Wd = Ws + 1	1	1	C, DC, N, OV, Z
NOP	NOP		No Operation	1	1	None
	NOPR		No Operation	1	1	None
POP	POP	£	Pop f from Top-of-Stack (TOS)	1	1	None
	POP	Wdo	Pop from Top-of-Stack (TOS) to Wdo	1	1	None
	POP.D	Wnd	Pop from Top-of-Stack (TOS) to W(nd):W(nd + 1)	1	2	None
	POP.S		Pop Shadow Registers	1	1	All
PUSH	PUSH	f	Push f to Top-of-Stack (TOS)	1	1	None
	PUSH	Wso	Push Wso to Top-of-Stack (TOS)	1	1	None
	PUSH.D	Wns	Push W(ns):W(ns + 1) to Top-of-Stack (TOS)	1	2	None
	PUSH.S		Push Shadow Registers	1	1	None

#### TABLE 26-2: INSTRUCTION SET OVERVIEW (CONTINUED)

#### TABLE 26-2: INSTRUCTION SET OVERVIEW (CONTINUED)

PMRSAVPMRSAVP11:1On tho Skep or diff model1IIDUTO, SkepRCALLEXD/CRelative Call12NoneREBURATRisitalRepark Net Instruction (Int 4 + 11 mes111NoneREBURATRisitalRepark Net Instruction (Vm) + 11 mes11NoneRESTRESTSoftware Device Rest13 (2)NoneRESTRETAsitial (NnRelative Vm (Vm) + 11 mes3 (2)NoneRESTRETAsitial (NnRelative Vm (Vm) + 11 mes3 (2)NoneRETARetar Non Suboutine13 (2)NoneRETARetar Non Suboutine13 (2)NoneRETARetar Non Suboutine11C. N. ZRETANo.Retar Non Suboutine11C. N. ZRETANo.Mark Softwart (Nn Samp)11C. N. ZRETANo.Wa Fordat Left Mough Cary f11N. ZRECffractate Right Mough Cary f11N. ZRECf. N. RAWRES = Rotate Right Mough Cary f11C. N. ZRECf. N. RAWRES = Rotate Right Mough Cary f11N. ZRECf. N. RAWRES = Rotate Right Mough Cary f11N. ZRETAw. RAWat State Right Mough Cary f11N. ZRETAw. RAWat State Right Mough Cary f11N. ZRETA <th>Assembly Mnemonic</th> <th></th> <th>Assembly Syntax</th> <th>Description</th> <th># of Words</th> <th># of Cycles</th> <th>Status Flags Affected</th>	Assembly Mnemonic		Assembly Syntax	Description	# of Words	# of Cycles	Status Flags Affected
PRACH BCAL BCAL BCAL BCAL BCAL BCAL BCAL BCAL BCAL BCAL BCAL BCAL 	PWRSAV	PWRSAV	#lit1	Go into Sleep or Idle mode	1	1	WDTO, Sleep
RCLL         Nn         Computed Call         1         1         2         Nome           REPEAT         BLI114         Repeat Next Instruction (Vm) + 1 times         1         1         Nome           REST         REST         Software Device Rest         1         1         Nome           REST         REST         Software Device Rest         1         3(2)         Nome           REST         REL         Relam rim Interrupt         1         3(2)         Nome           RETUR         REL         f         Relam rim Number Device Rest         1         1         0, 0           RETUR         REL         f         Relam rim Subordime         1         1         0, N.Z.           RETUR         RER         Foldate Left Mrough Carry f         1         1         N.Z.           REX         sink         f         Foldate Regit Mrough Carry f         1         1         N.Z.           REX         sink         restart Regit Mrough Carry f         1         1         N.Z.           REX         sink         MREG         Rotate Regit Mrough Carry f         1         1         N.Z.           REX         sink         MREG         Rotate Regit Mrough Carry f	RCALL	RCALL	Expr	Relative Call	1	2	None
REPAR REPART REPART#11140Repeat Nact Instruction (Wn + 1 times)11NoneRESTRESTSoftwar Device Reset111NoneRETTISREST SSoftwar Device Reset13 (2)NoneRETTISRETTSReturn Winn Infering13 (2)NoneRETTUMRETUNReturn Winn Infering13 (2)NoneRETONRETUNReturn Winn Leardin Winn13 (2)NoneREC1C. N.ZOReturn Winn Subroding11C. N.ZORECSingel1Foldate Left Brough Carry Min11C. N.ZORECSingel1Foldate Left Brough Carry Min11N.ZORECSingelFoldate Left Brough Carry Min11C. N.ZORECSingelFoldate Right Brough Carry Min11N.ZORECSingelMelica Right Brough Carry Min11N.ZORECSingelMelica Right Brough Carry Min11N.ZORECSingelMelica Right Brough Carry Min11N.ZORECSingelMelica Right Minugh Carry Min11N.ZORECSingelMelica Right Minugh Carry Min11N.ZORECSingelMelica Right Minugh Carry Min11N.ZORECSingelMinMelica Right Minugh Carry Min11N.ZORECSingelMin <td></td> <td>RCALL</td> <td>Wn</td> <td>Computed Call</td> <td>1</td> <td>2</td> <td>None</td>		RCALL	Wn	Computed Call	1	2	None
No.         Researt No.         Respect Not instruction (Wn) + 1 times         1         1         None           RESTIM         RESTIF         Software Device Reset         1         3 (2)         None           RETTA         RETTA         RETLIN         Return Wn Literal In Vn         1         3 (2)         None           RETTA         RETLIN         Return Wn Literal In Vn         1         3 (2)         None           RETTA         RETLIN         Return Wn Literal In Vn         1         1         C, N.2           REC         \$\bar{L}\$         C         None         C, N.2         C, N.2           REC         \$\bar{L}\$         VMEG = Rotate Left NocQh Carry Ms         1         1         C, N.2           REX         \$\bar{L}\$         VMEG = Rotate Left (No Carry) Ms         1         1         N.2           REX         \$\bar{L}\$         VMEG = Rotate Left (No Carry) Ms         1         1         N.2           REX         \$\bar{L}\$         VMEG = Rotate Right Through Carry Ms         1         1         N.2           REX         \$\bar{L}\$         VMEG = Rotate Right Through Carry Ms         1         1         N.2           REX         \$\bar{L}\$         VMEG = Rotate Right Through Carry Ms </td <td>REPEAT</td> <td>REPEAT</td> <td>#lit14</td> <td>Repeat Next Instruction lit14 + 1 times</td> <td>1</td> <td>1</td> <td>None</td>	REPEAT	REPEAT	#lit14	Repeat Next Instruction lit14 + 1 times	1	1	None
RESETSeRVard Device Reset11NoneRETPISRELIN FILSRelum from Hierup113(2)NoneRETORRELIN #11.0. NnRelum from Subrouline13(2)NoneRECORDRELIN Will Literal in Win13(2)NoneRECORDFindel teat through Carry f113(2)NoneRECORDFindel teat through Carry f111C. N. ZRELINCFindel teat through Carry f11N. ZRELINCFindel teat (No Carry) f11N. ZRELINCFindel teat (No Carry) f11N. ZRENCFindel teat (No Carry) f11N. ZRENCFindel teat (No Carry) f11N. ZRENCFindel teat Right through Carry f11N. ZRENCFindel Right through Carry f		REPEAT	Wn	Repeat Next Instruction (Wn) + 1 times	1	1	None
BETTLERETTLEReturn from interrupt13 (2)NoneBETTLMRETTUMRETURMRETURM13 (2)NoneRETURMRETURMRETURMReturn infor Subractine13 (2)NoneRLCffReturn infor Subractine11C, N, ZRLCffReturn infor Subractine11C, N, ZRLCinform inform in	RESET	RESET		Software Device Reset	1	1	None
BETLWBITLWBeturn with Leen invin.13 (2)NoneRETURNRECUF.M.ROMeturn foro Subrouline11C.N.ZRECUF.M.ROWREG = Rotate Left through Carry f11C.N.ZRECUF.M.ROWREG = Rotate Left through Carry f11N.ZRECUF.M.ROWREG = Rotate Left through Carry f11N.ZRECUF.M.ROWREG = Rotate Left (No Carry) f11N.ZRECUF.M.ROWREG = Rotate Left (No Carry) f11N.ZRECUF.M.ROWREG = Rotate Left (No Carry) f11N.ZRECUF.M.ROWREG = Rotate Right through Carry f11N.ZRECUF.M.ROWREG = Rotate Right through Carry f11N.ZRECUF.M.ROWREG = Rotate Right (No Carry) f11N.ZRECUF.M.ROWREG = FFFh11N.CRECUF.M.ROWREG = FFFh11N.CRECUF.M.ROWREG = Information11N.C	RETFIE	RETFIE		Return from Interrupt	1	3 (2)	None
BETURNReturn from subroundine13 (2)NoneRLCFLf = Rotate Left through Carry f11C, N, ZRLCf, WR80WREG = Rotate Left through Carry f11C, N, ZRLNCf, WR80Wd = Rotate Left through Carry f11N, ZRLNCf, WR80Wd = Rotate Left (No Carry) f11N, ZRLNCf, WR80WGEG = Rotate Left (No Carry) f11N, ZRLNCf, WR80WGEG = Rotate Left (No Carry) f11N, ZRRNCf, WR80WGEG = Rotate Righ through Carry f11N, ZRRNCf, WR80WGEG = Rotate Righ (No Carry) f11N, ZRRNCWa, MaWG = Rotate Righ (No Carry) f11N, ZRRNCWa, MaWGEG = FFFFh11N, RRRNCWa, MaWG = FGEFR11N, NORRNCSETMWR80WG = Rotate Right Wob WR80111RNNSETMWR80WG = GE Shift H11N, NORNNSETMWR80WG = GE Shift Wob WR80111N, NORNNSETMWR80WG = GE Shift Wob WR8011 <td>RETLW</td> <td>RETLW</td> <td>#lit10,Wn</td> <td>Return with Literal in Wn</td> <td>1</td> <td>3 (2)</td> <td>None</td>	RETLW	RETLW	#lit10,Wn	Return with Literal in Wn	1	3 (2)	None
RLCéffRotate Left through Carry ffffCN.ZRLCK, NRGWREG = Rotate Left through Carry M11CN.ZRLNCffRotate Left through Carry M11N.ZRLNCffRotate Left (No Carry M11N.ZRLNCffRotate Left (No Carry M11N.ZRLNCffRotate Left (No Carry M11N.ZRRCffRotate Left (No Carry M11N.ZRRCffRotate Right through Carry M11N.ZRRCffRotate Right through Carry M11N.ZRRCfrRotate Right through Carry M11N.ZRRCfrRotate Right (No Carry M11N.ZRRCfrrRotate Right (No Carry M11N.ZRRCfrrRotate Right (No Carry M11N.ZRRCfrr<	RETURN	RETURN		Return from Subroutine	1	3 (2)	None
NUMBER         F.NKB03         WREG = Rotate Left through Carry Ms         1         1         0.<	RLC	RLC	f	f = Rotate Left through Carry f	1	1	C, N, Z
ILC         Way, Wd         Wd = Rotate Left through Carry Ws         1         1         0.1         0.1         0.1           RLNC         F, WREG         WREG         WREG         1         1         N.Z           RLNC         F, WREG         WREG         Ref Constructure         1         1         N.Z           RRC         E         Facalate Left (No Carry) for         1         1         N.Z           RRC         F, WREG         WREG         Facalate Right through Carry Ms         1         1         C.N.Z           RRC         F, WREG         WREG         Facate Right through Carry Ms         1         1         N.Z.           RRC         F, WREG         WREG         Facate Right (No Carry) for         1         N.Z.         N.Z.           RRC         F, WREG         WREG         FERFER         1         1         N.Z.           SE         WR., Md         Wd = Rotate Right (No Carry) for         1         1         N.Z.           SE         WR., Md         Wd = FEFFFh         1         1         N.D.C.           SETM         WREG         Felder Shift Mb by MrS         1         1         N.Z.           SL         WD., MR., Md		RLC	f,WREG	WREG = Rotate Left through Carry f	1	1	C, N, Z
LINCf = Runcef = Rolate Left (No Carry) f11N.ZRENCF, WREGWREG = Rolate Left (No Carry) f11N.ZRECF, WR.GWREG = Rolate Left (No Carry) f11N.ZRECF, WR.GF = Rolate Right Hough Carry f11N.ZRECF, WR.GWREG = Rolate Right Hough Carry f11C.N.ZRECF, WR.GWREGFolder Right Hough Carry f11N.ZRENCF, WR.GWREG = Rolate Right (No Carry) f11N.ZRENCF, WR.GWREGFolder Right (No Carry) f11N.ZRENCF, WR.GWREGFolder Right (No Carry) f11N.ZSESEF, WR.GWREGFolder Right (No Carry) f11N.ZSESEFFFFh11N.GN.ZSESEFFF11N.GSESEFF11N.ZN.GSESEFF11N.GN.GSESEFFSE11N.GSESEFF <t< td=""><td></td><td>RLC</td><td>Ws,Wd</td><td>Wd = Rotate Left through Carry Ws</td><td>1</td><td>1</td><td>C, N, Z</td></t<>		RLC	Ws,Wd	Wd = Rotate Left through Carry Ws	1	1	C, N, Z
INSC         F, WREG         WREG = Rotate Left (No Carry) f         1         1         N, Z           REG         FLMC         Wa, Wd         Wd = Rotate Left (No Carry) Ws         1         1         N, Z           REG         £, WREG         F. Rotate Right through Carry Ms         1         1         C, N, Z           REC         £, WREG         WWEG = Rotate Right through Carry Ms         1         1         N, Z           RENC         £, MREG         F. Rotate Right (No Carry) Ms         1         1         N, Z           RENC         £, MREG         MWEG = Rotate Right (No Carry) Ms         1         1         N, Z           SET         W.R.WAG         Wd = Rotate Right (No Carry) Ms         1         1         N, Z           SET         W.R.WAG         Wd = Rotate Right (No Carry) Ms         1         1         N, Z           SET         W.R.WAG         WMEG = Explexiteded Ws         1         1         N, Z           SET         WREG         #EFFFh         1         1         None           SET         W.R.WAG         WreG = Left Shift f         1         1         N, Z           SL         #D, NraWAG         Wrd = Left Shift Wb by INS         1         1	RLNC	RLNC	f	f = Rotate Left (No Carry) f	1	1	N, Z
ILNC         Ws, Md         Wd = Rotale Left (No Carry) Ws         1         1         N, Z           RRC         f, WREG         f = Rotale Right through Carry f         1         1         C, N, Z           RRC         w, Wd         WREG = Rotate Right through Carry f         1         1         C, N, Z           RRNC         Wa, Wd         WREG = Rotate Right (No Carry) f         1         1         N, Z           RENC         f, WREG         WREG = Rotate Right (No Carry) f         1         1         N, Z           RENC         f, WREG         WREG = Rotate Right (No Carry) f         1         1         N, Z           SE         ws, Wd         Wd = Rotate Right (No Carry) f         1         1         N, Z           SE         Ws, Wd         Wd = Rotate Right (No Carry) fw         1         1         N, Z           SE         Ws, Wd         Wd = Rotate Right (No Carry) fw         1         1         N, Z           SE         SE         Ws, Wd         Wd = Rotate Right (No Carry) fw         1         1         N, Z           SE         SE         SE         Ws, Wd         WREG = FFFFh         1         1         None           SE         L         f, WREG         He		RLNC	f,WREG	WREG = Rotate Left (No Carry) f	1	1	N, Z
REC $\pm$ f = Rotate Right through Carry f         1         1         C, N, Z           RC $\ell$ , WREG         WREG = Rotate Right through Carry f         1         1         C, N, Z           RRNC $\ell$ $m$ a, wa         Wel = Rotate Right through Carry f         1         1         C, N, Z           RRNC $\ell$ m = Rotate Right through Carry f         1         1         N, Z           RRNC $\ell$ , WREG $\ell$ Rotate Right (No Carry) f         1         1         N, Z           RRNC $\ell$ , WREG $\ell$ Rotate Right (No Carry) Ws         1         1         N, Z           SET $wa$ , $Mad$ Wrd = Sign-Extended Ws         1         1         N, Z           SETM $wa$ $md$ $md$ = Sign-Extended Ws         1         1         N, Z           SETM $wa$ WREG = FFFFh         1         1         None           SL $\ell$ , WREG $md$ Left Shift Ws         1         1         C, N, OV, Z           SL $wa$ , $wad         Wrd = Left Shift Ws by His         1         1         N, Z           SL         wa, wad         Wrd = Left Shift Ws by His         $		RLNC	Ws,Wd	Wd = Rotate Left (No Carry) Ws	1	1	N, Z
RRC         f, WREG         WREG = Rotate Right through Carry f         1         1         C, N, Z           RRC         We, Wd         Wd = Rotate Right through Carry Ws         1         1         N, Z           RRNC         f, MREG         f = Rotate Right (No Carry) f         1         1         N, Z           RRNC         f, MREG         WREG = Rotate Right (No Carry) f         1         1         N, Z           RRNC         m, Md         Wd = Rotate Right (No Carry) f         1         1         N, Z           SE         Ms., Md         Wd = Rotate Right (No Carry) f         1         1         N, Z           SET         SET         Ms.         MG         Wd = Rotate Right (No Carry) f         1         1         N, Z           SET         SET         Ms.         Md         Wd = Rotate Right (No Carry) f         1         1         N, Z           SET         SE         Ms.         Md         Md = Stift f         1         1         N, Z           SET         WREG         Intert Shift f         1         1         None         1         1         N, Z           SET         Ms., Md         Wd = Lef Shift f         1         1         1         N, Z	RRC	RRC	f	f = Rotate Right through Carry f	1	1	C, N, Z
RRC         Ws, Md         Wd = Rotate Right through Carry Ws         1         1         C, N, Z           RRNC         f, WREG         f = Rotate Right (No Carry) f         1         1         N, Z           RRNC         We, Wd         WREG = Rotate Right (No Carry) f         1         1         N, Z           SE         SE         We, Wd         Wd = Rotate Right (No Carry) Ws         1         1         N, Z           SETM         SETM         f         FFFFh         1         1         N, Z           SETM         WREG         FFFFh         1         1         None           SETM         Wa         Was = FFFFh         1         1         None           SETM         Wa         Wes = FFFFh         1         1         C, N, OV, Z           SL         f, WREG         If Left Shift Ms         1         1         C, N, OV, Z           SL         Ms, Md         Wd = Left Shift Ms by Wns         1         1         N, OV, Z           SL         Ms, Mad         Wd = Left Shift Ws         1         1         N, C         C, O, N, OV, Z           SL         Ms, Mad         Wd = Left Shift Ws by Wns         1         1         C, DC, N, OV, Z		RRC	f,WREG	WREG = Rotate Right through Carry f	1	1	C, N, Z
RRNC $f$ $f$ = Rotate Right (No Carry) f         1         1         N, Z           RRNC $f$ , WREG         WREG = Rotate Right (No Carry) f         1         1         N, Z           RENC         Wa, Wd         Wd = Rotate Right (No Carry) Ws         1         1         N, Z           SE         SE         Wa, Wd         Wd = Rotate Right (No Carry) Ws         1         1         N, Z           SETM $f$ FFFFN         1         1         N, Z           SETM $K$ WREG         WREG = FFFFN         1         1         None           SL $f$ f = Left Shift f         1         1         N, OV, Z           SL $f$ , MREG         WREG = Left Shift Wb S         1         1         N, OV, Z           SL $W_{A}$ , Wd         Wd = Left Shift Wb SW Wns         1         1         N, Z           SL $W_{A}$ , Wd         Wnd = Left Shift Wb SW Wns         1         1         N, Z           SUB $f$ $f$ = Left Shift Wb SW Wns         1         1         N, Z           SUB $f$ $f$ = Left Shift Wb SW Wns         1         1         N, Z		RRC	Ws,Wd	Wd = Rotate Right through Carry Ws	1	1	C, N, Z
RENC $f$ , WREG         WREG = Rotate Right (No Carry) M         1         1         N, Z           SE         SE         Wa, Wnd         Waf = Rotate Right (No Carry) Ws         1         1         N, Z           SE         SE W &, Wnd         Wnd = Sign-Extended Ws         1         1         C, N, Z           SETM         É         FFFFh         1         1         None           SETM         WREG         WREG = FFFFh         1         1         None           SL         f         f         Left Shift f         1         1         C, N, OV, Z           SL         w, Wd         WREG         Left Shift Wb         1         1         N, Z           SL         w, Md         Wd = Left Shift Wb         1         1         C, N, OV, Z           SL         wb, Wna, Wnd         Wnd = Left Shift Wb by Uns         1         1         N, Z           SUB         f         f         f         FEEG         1         1         N, Z           SUB         f. WREG         WREG = f - WREG         1         1         C, DC, N, OV, Z           SUB         #Litlo, Wn         Wne = Wn - Hit0         1         1         C, DC, N, OV, Z <td>RRNC</td> <td>RRNC</td> <td>f</td> <td>f = Rotate Right (No Carry) f</td> <td>1</td> <td>1</td> <td>N, Z</td>	RRNC	RRNC	f	f = Rotate Right (No Carry) f	1	1	N, Z
RRNC         Ws.Wd         Wd = Rotate Right (No Carry) Ws         1         1         N,Z           SE         SE         Ws.Wnd         Wnd = Sign-Extended Ws         1         1         C,N,Z           SETM         £ETM         f         f=FFFFh         1         1         1         None           SETM         WRRG         WREG=FFFFh         1         1         1         None           SL         f         f         f=effFh         1         1         1         None           SL         f         f=effFh         1         1         1         None           SL         f.WREG         WREG=Left Shift ff         1         1         0, N,OV,Z           SL         Wb.Wns.Wnd         Wd = Left Shift Wb by Wns         1         1         N,Z           SL         Wb.Ws.Wnd         Wd = Left Shift Wb by Uf5         1         1         N,Z           SUB         f.WREG         f=f-WREG         1         1         0,C,O,N,V,Z           SUB         f.WRES         WREG = f-WREG         1         1         0,C,O,N,V,Z           SUB         f.WREG         f=f-WREG         1         1         0,C,O,C,N,V,Z		RRNC	f,WREG	WREG = Rotate Right (No Carry) f	1	1	N, Z
SE         SE         Ws, Wnd         Wnd = Sign-Extended Ws         1         1         C, N, Z           SETM $f$ f=FFFFh         1         1         None           SETM         WREG         WREG = FFFFh         1         1         None           SL         f         f=Left Shift f         1         1         1         None           SL         f, WREG         WREG = Left Shift f         1         1         1         C, N, OV, Z           SL $f$ , WREG         WREG = Left Shift Ws         1         1         1         C, N, OV, Z           SL         Ws, Wd         Wd = Left Shift Ws         1         1         N, Z           SUB $f$ f=f-WREG         1         1         N, Z           SUB $f$ f=f-WREG         1         1         C, DC, N, OV, Z           SUB $f$ f=f-WREG         1         1         C, DC, N, OV, Z           SUB $f$ JWR Wd         Wd = Wb - Ws         1         1         C, DC, N, OV, Z           SUB $f$ JWR Wd         Wd = Wb - Ws         1         1         C, DC, N, OV, Z		RRNC	Ws,Wd	Wd = Rotate Right (No Carry) Ws	1	1	N, Z
SETM         SETM         f         f = FFFFh         1         1         None           SETM         WREG         WREG         WREG = FFFFh         1         1         None           SL         f         f=left Shift f         1         1         None           SL         f, WREG         WREG = Left Shift f         1         1         C, N, OV, Z           SL         f, WREG         WREG = Left Shift fWb by Wns         1         1         C, N, OV, Z           SL         Ws, Md         Wd = Left Shift Wb by Wns         1         1         N, OV, Z           SL         Wb, Mins, Wnd         Wnd = Left Shift Wb by Wns         1         1         N, Z           SUB         f         f=f         FHEG         1         1         C, DC, N, OV, Z           SUB         f, WREG         WREG = f - WREG         1         1         C, DC, N, OV, Z           SUB         f, WREG         WREG = f - WREG         1         1         C, DC, N, OV, Z           SUB         f, WREG         WREG = f - WREG         1         1         C, DC, N, OV, Z           SUB         f, WLEG         WD + Wn - III10         1         1         C, DC, N, OV, Z	SE	SE	Ws,Wnd	Wnd = Sign-Extended Ws	1	1	C, N, Z
SETM         WREG         WREG = FFFFh         1         1         None           SETM         Ws         Ws = FFFFh         1         1         1         None           SL         f         f         Left Shift f         1         1         1         0, N, OV, Z           SL         f, WREG         WREG = Left Shift f         1         1         0, N, OV, Z           SL         wb, Wd         Wd = Left Shift Wb by Wns         1         1         0, N, OV, Z           SL         Wb, Mins, Wnd         Wnd = Left Shift Wb by Wns         1         1         N, Z           SL         Wb, #lit5, wnd         Wnd = Left Shift Wb by Uns         1         1         0, D, OV, Z           SUB         f         f=f-WREG         1         1         N, Z           SUB         f.WREG         Wb, #lit5, wd         Wd = Wb Ws         1         1         0, D, N, V, Z           SUB         f.ULL, Wn         Wn = Wn – III10         1         1         0, D, N, V, Z           SUB         f.ULL, Wn         Wd = Wb – Ws         1         1         0, D, N, V, Z           SUB         f.WREG         f=f - WREG – (C)         1         1         0, D, N, V, Z	SETM	SETM	f	f = FFFFh	1	1	None
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$		SETM	WREG	WREG = FFFFh	1	1	None
$ SL \qquad SL \qquad f \qquad f = Left Shift f \qquad 1 \qquad 1 \qquad C, N, OV, Z \\ \hline SL \qquad f, WREG \qquad WREG = Left Shift f \qquad 1 \qquad 1 \qquad C, N, OV, Z \\ \hline SL \qquad Ws, Wd \qquad Wd = Left Shift Ws \qquad 1 \qquad 1 \qquad C, N, OV, Z \\ \hline SL \qquad Wb, Wns, Wnd \qquad Wnd = Left Shift Ws \qquad 1 \qquad 1 \qquad N, Z \\ \hline SL \qquad Wb, Wns, Wnd \qquad Wnd = Left Shift Wb by Wns \qquad 1 \qquad 1 \qquad N, Z \\ \hline SU \qquad SU \qquad Wb, #lit5, Wnd \qquad Wnd = Left Shift Wb by Wns \qquad 1 \qquad 1 \qquad N, Z \\ \hline SUB \qquad SUB \qquad f \qquad f = f = WREG \qquad 1 \qquad 1 \qquad C, DC, N, OV, Z \\ \hline SUB \qquad f, WREG = f = WREG = f = WREG \qquad 1 \qquad 1 \qquad C, DC, N, OV, Z \\ \hline SUB \qquad #lit10, Wn \qquad Wn = Wn = Wn = Wn = Wn = Wn = Wn =$		SETM	Ws	Ws = FFFFh	1	1	None
$ \begin{array}{ c c c c c c c c } \hline SL & f, WEG & WREG = Left Shift f & 1 & 1 & C, N, OV, Z \\ \hline SL & Ws, Wd & Wd = Left Shift Ws & 1 & 1 & C, N, OV, Z \\ \hline SL & Wb, Wns, Wnd & Wnd = Left Shift Wb by Wns & 1 & 1 & N, Z \\ \hline SL & Wb, Hilts, Wnd & Wnd = Left Shift Wb by Uns & 1 & 1 & N, Z \\ \hline SL & Wb, Hilts, Wnd & Wnd = Left Shift Wb by Uns & 1 & 1 & N, Z \\ \hline SUB & SUB & f & f = f - WREG & 1 & 1 & C, DC, N, OV, Z \\ \hline SUB & f, WEG & WREG = f - WREG & 1 & 1 & C, DC, N, OV, Z \\ \hline SUB & Wb, Ws, Wd & Wn = Wn - Hi10 & 1 & 1 & C, DC, N, OV, Z \\ \hline SUB & Wb, Ws, Wd & Wd = Wb - Ws & 1 & 1 & C, DC, N, OV, Z \\ \hline SUB & Wb, Hilts, Wd & Wd = Wb - Ws & 1 & 1 & C, DC, N, OV, Z \\ \hline SUB & Mb, Hilts, Wd & Wd = Wb - Hif5 & 1 & 1 & C, DC, N, OV, Z \\ \hline SUBB & f, WREG & WREG = f - WREG - (\overline{C}) & 1 & 1 & C, DC, N, OV, Z \\ \hline SUBB & f, WREG & WREG = f - WREG - (\overline{C}) & 1 & 1 & C, DC, N, OV, Z \\ \hline SUBB & f, WREG & WREG = f - WREG - (\overline{C}) & 1 & 1 & C, DC, N, OV, Z \\ \hline SUBB & f, WREG & WREG = f - WREG - (\overline{C}) & 1 & 1 & C, DC, N, OV, Z \\ \hline SUBB & f, WREG & WREG = f - WREG - (\overline{C}) & 1 & 1 & C, DC, N, OV, Z \\ \hline SUBB & f, WREG & WREG = f - WREG - (\overline{C}) & 1 & 1 & C, DC, N, OV, Z \\ \hline SUBB & f, WREG & WREG = f - WREG - (\overline{C}) & 1 & 1 & C, DC, N, OV, Z \\ \hline SUBB & f, WREG & WREG = f - WREG - 1 & 1 & C, DC, N, OV, Z \\ \hline SUBB & Wb, Ws, Md & Wd = Wb - Hist - (\overline{C}) & 1 & 1 & C, DC, N, OV, Z \\ \hline SUBB & Wb, Ws, Md & Wd = Wb - Hist - (\overline{C}) & 1 & 1 & C, DC, N, OV, Z \\ \hline SUBR & f, WREG & WREG = WREG - f & 1 & 1 & C, DC, N, OV, Z \\ \hline SUBR & Mb, Ws, Md & Wd = Ws - Wb & 1 & 1 & C, DC, N, OV, Z \\ \hline SUBR & Wb, Ws, Md & Wd = Ws - Wb & (\overline{C}) & 1 & 1 & C, DC, N, OV, Z \\ \hline SUBR & Mb, Ws, Hilts, Wd & Wd = Ws - Wb & (\overline{C}) & 1 & 1 & C, DC, N, OV, Z \\ \hline SUBR & Mb, Hilts, Wd & Wd = Ws - Wb - (\overline{C}) & 1 & 1 & C, DC, N, OV, Z \\ \hline SUBR & Mb, Hilts, Wd & Wd = Ws - Wb - (\overline{C}) & 1 & 1 & C, DC, N, OV, Z \\ \hline SUBR & Mb, Hilts, Md & Wd = Ws - Wb - (\overline{C}) & 1 & 1 & C, DC, N, OV, Z \\ \hline SWBR & Mb, Hilts, Md & Wd = Ws - Wb - (\overline{C}) & 1 & 1 & C, DC, N, OV, Z \\ \hline SWBR & Mb, Hilts, Md & Wd = W$	SL	SL	f	f = Left Shift f	1	1	C, N, OV, Z
SL         Ws, Wd         Wd = Left Shift Ws         1         1         1         C, N, OV, Z           SL         Wb, Wns, Wnd         Wnd = Left Shift Wb by Wns         1         1         N, Z           SUB         SL         Wb, #lit5, Wnd         Wnd = Left Shift Wb by It5         1         1         N, Z           SUB         SUB         f         f=f-WREG         1         1         C, DC, N, OV, Z           SUB         f, WREG         WREG = f-WREG         1         1         C, DC, N, OV, Z           SUB         #lit10, Wn         Wn = Wn - lit10         1         1         C, DC, N, OV, Z           SUB         #lit10, Wn         Wd = Wb - Ws         1         1         C, DC, N, OV, Z           SUB         #lit10, Wn         Wd = Wb - Ws         1         1         C, DC, N, OV, Z           SUBB         f         f=f-WREG-(C)         1         1         C, DC, N, OV, Z           SUBB         f.WREG         WREG = f-WREG-(C)         1         1         C, DC, N, OV, Z           SUBB         f.WREG         Wd = Wb - IIt5         1         1         C, DC, N, OV, Z           SUBB         f.WREG         Wd = Wb - Ws - (C)         1         1		SL	f,WREG	WREG = Left Shift f	1	1	C, N, OV, Z
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$		SL	Ws,Wd	Wd = Left Shift Ws	1	1	C, N, OV, Z
SL         Wb, #lit5, Wnd         Wnd = Left Shift Wb by lit5         1         1         N, Z           SUB         f         f         f=f-WREG         1         1         C, DC, N, OV, Z           SUB         f, WREG         WREG = f-WREG         1         1         C, DC, N, OV, Z           SUB         #lit10, Wn         Wn = Wn - lit10         1         1         C, DC, N, OV, Z           SUB         Wb, Ws, Wd         Wd = Wb - Ws         1         1         C, DC, N, OV, Z           SUB         Wb, #lit5, Wd         Wd = Wb - Ws         1         1         C, DC, N, OV, Z           SUBB         f.         f=f-WREG-(C)         1         1         C, DC, N, OV, Z           SUBB         f., WREG         MREG = f-WREG - (C)         1         1         C, DC, N, OV, Z           SUBB         f., WREG         WREG = f-WREG - (C)         1         1         C, DC, N, OV, Z           SUBB         f., WREG         WREG = f-WREG - (C)         1         1         C, DC, N, OV, Z           SUBB         #lit10, Wn         Wn = Wn - lit10 - (C)         1         1         C, DC, N, OV, Z           SUBB         Wb, Wa, Wd         Wd = Wb - Hit5 - (C)         1         1		SL	Wb,Wns,Wnd	Wnd = Left Shift Wb by Wns	1	1	N, Z
$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$		SL	Wb,#lit5,Wnd	Wnd = Left Shift Wb by lit5	1	1	N, Z
$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$	SUB	SUB	f	f = f – WREG	1	1	C, DC, N, OV, Z
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$		SUB	f,WREG	WREG = f – WREG	1	1	C, DC, N, OV, Z
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$		SUB	#lit10,Wn	Wn = Wn – lit10	1	1	C, DC, N, OV, Z
SUB         Wb, #lit5, Wd         Wd = Wb - lit5         1         1         C, DC, N, OV, Z           SUBB         f         f=f-WREG-(C̄)         1         1         C, DC, N, OV, Z           SUBB         f, WREG         WREG = f-WREG-(C̄)         1         1         C, DC, N, OV, Z           SUBB         flit10, Wn         Wn = Wn - lit10-(C̄)         1         1         C, DC, N, OV, Z           SUBB         #lit10, Wn         Wn = Wn - lit10-(C̄)         1         1         C, DC, N, OV, Z           SUBB         #lit10, Wn         Wn = Wn - lit10-(C̄)         1         1         C, DC, N, OV, Z           SUBB         wb, Ws, Wd         Wd = Wb - Ws - (C̄)         1         1         C, DC, N, OV, Z           SUBB         wb, #lit5, Wd         Wd = Wb - Ws - (C̄)         1         1         C, DC, N, OV, Z           SUBR         f, WREG         WREG = WREG - f         1         1         C, DC, N, OV, Z           SUBR         f, Ws, Wd         Wd = Ws - Wb         1         1         C, DC, N, OV, Z           SUBR         f, WREG         WREG = WREG - f         1         1         C, DC, N, OV, Z           SUBR         Wb, #lit5, Wd         Wd = lit5 - Wb         1         1		SUB	Wb,Ws,Wd	Wd = Wb – Ws	1	1	C. DC. N. OV. Z
SUBB         f         f = f - WREG - ( $\overline{C}$ )         1         1         C, DC, N, OV, Z           SUBB         f, WREG         WREG = f - WREG - ( $\overline{C}$ )         1         1         C, DC, N, OV, Z           SUBB         #1it10, Wn         Wn = Wn - lit10 - ( $\overline{C}$ )         1         1         C, DC, N, OV, Z           SUBB         #1it10, Wn         Wn = Wn - lit10 - ( $\overline{C}$ )         1         1         C, DC, N, OV, Z           SUBB         Wb, Ws, Wd         Wd = Wb - Ws - ( $\overline{C}$ )         1         1         C, DC, N, OV, Z           SUBB         Wb, Ws, Wd         Wd = Wb - Ws - ( $\overline{C}$ )         1         1         C, DC, N, OV, Z           SUBB         Wb, #1it5, Wd         Wd = Wb - Ws - ( $\overline{C}$ )         1         1         C, DC, N, OV, Z           SUBR         f, WREG         WREG = WREG - f         1         1         C, DC, N, OV, Z           SUBR         f, WREG         WREG = WREG - f         1         1         C, DC, N, OV, Z           SUBR         wb, #1it5, Wd         Wd = Ws - Wb         1         1         C, DC, N, OV, Z           SUBR         f, WREG         f = WREG - f - ( $\overline{C}$ )         1         1         C, DC, N, OV, Z           SUBR         f, WREG         WREG = WREG - f -		SUB	Wb,#lit5,Wd	Wd = Wb – lit5	1	1	C. DC. N. OV. Z
SUBB         I <td>SUBB</td> <td>SUBB</td> <td>f</td> <td><math>f = f - WREG - (\overline{C})</math></td> <td>1</td> <td>1</td> <td>C DC N OV Z</td>	SUBB	SUBB	f	$f = f - WREG - (\overline{C})$	1	1	C DC N OV Z
$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$		STIBB	f WPFC	WREG = $f - WREG - (\overline{C})$	1	1	C DC N OV Z
SUBB         #ITCID, WI         WII = WII = IITO - (C)         I         I         C, DC, N, OY, Z           SUBB         Wb, Ws, Wd         Wd = Wb - Ws - (C)         1         1         C, DC, N, OY, Z           SUBB         Wb, #lit5, Wd         Wd = Wb - lit5 - (C)         1         1         C, DC, N, OY, Z           SUBR         SUBR         f         f         F         WREG - f         1         1         C, DC, N, OY, Z           SUBR         f, WREG         WREG = WREG - f         1         1         C, DC, N, OY, Z           SUBR         f, WREG         Wd = Ws - Wb         1         1         C, DC, N, OY, Z           SUBR         Wb, Ws, Wd         Wd = Ws - Wb         1         1         C, DC, N, OY, Z           SUBR         Wb, Ws, Wd         Wd = Ws - Wb         1         1         C, DC, N, OY, Z           SUBR         Wb, #lit5, Wd         Wd = It5 - Wb         1         1         C, DC, N, OY, Z           SUBR         f, WREG         WREG = WREG - f - (C)         1         1         C, DC, N, OY, Z           SUBR         f, WREG         Wd = Ws - Wb - (C)         1         1         C, DC, N, OY, Z           SUBR         Wb, #lit5, Wd         Wd = Ws - Wb -		GUDD			1	1	C, DC, N, OV, Z
SUBB         Wb, Ws, Wd         Wd = Wb - Ws - (C)         1         1         C, DC, N, OV, Z           SUBB         Wb, #lit5, Wd         Wd = Wb - lit5 - (C)         1         1         C, DC, N, OV, Z           SUBR         SUBR         f         f         f = WREG - f         1         1         C, DC, N, OV, Z           SUBR         f, WREG         WREG = WREG - f         1         1         C, DC, N, OV, Z           SUBR         f, WB, Ws, Wd         Wd = Ws - Wb         1         1         C, DC, N, OV, Z           SUBR         wb, Ws, Wd         Wd = Ws - Wb         1         1         C, DC, N, OV, Z           SUBR         Wb, Ws, Wd         Wd = Ws - Wb         1         1         C, DC, N, OV, Z           SUBR         Wb, #lit5, Wd         Wd = Ws - Wb         1         1         C, DC, N, OV, Z           SUBBR         f         MREG         MREG = Cf - (C)         1         1         C, DC, N, OV, Z           SUBBR         f, WREG         WREG = WREG - f - (C)         1         1         C, DC, N, OV, Z           SUBBR         f, WB, Ws, Wd         Wd = Ws - Wb - (C)         1         1         C, DC, N, OV, Z           SUBBR         Wb, #lit5, Wd         Wd = lit5		SUBB	#11010, Wn	$W_{1} = V_{1} - H_{1} - H_{1} - (C)$	1	1	C, DC, N, OV, Z
SUBB         Wb, #lit5, Wd         Wd = Wb - lit5 - (C)         1         1         C, DC, N, OV, Z           SUBR         f         f         f = WREG - f         1         1         C, DC, N, OV, Z           SUBR         f, WREG         WREG = WREG - f         1         1         C, DC, N, OV, Z           SUBR         f, Wb, Ws, Wd         Wd = Ws - Wb         1         1         C, DC, N, OV, Z           SUBR         Wb, #lit5, Wd         Wd = Ws - Wb         1         1         C, DC, N, OV, Z           SUBR         Wb, #lit5, Wd         Wd = Ws - Wb         1         1         C, DC, N, OV, Z           SUBR         Wb, #lit5, Wd         Wd = Iit5 - Wb         1         1         C, DC, N, OV, Z           SUBBR         f         WREG         WREG = MREG - f - (C)         1         1         C, DC, N, OV, Z           SUBBR         f, WREG         WREG = WREG - f - (C)         1         1         C, DC, N, OV, Z           SUBBR         f, WREG         Wb, Ws, Wd         Wd = Ws - Wb - (C)         1         1         C, DC, N, OV, Z           SUBR         Wb, #lit5, Wd         Wd = Ws - Wb - (C)         1         1         C, DC, N, OV, Z           SWAP         Wn         Wn =		SUBB	Wb,Ws,Wd	VVd = VVb - VVS - (C)	1	1	C, DC, N, OV, Z
$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$		SUBB	Wb,#lit5,Wd	Wd = Wb - lit5 - (C)	1	1	C, DC, N, OV, Z
$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$	SUBR	SUBR	f	f = WREG – f	1	1	C, DC, N, OV, Z
$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$		SUBR	f,WREG	WREG = WREG – f	1	1	C, DC, N, OV, Z
SUBR         Wb, #lit5, Wd         Wd = lit5 - Wb         1         1         C, DC, N, OV, Z           SUBBR         £         f         f = WREG - f - (C̄)         1         1         C, DC, N, OV, Z           SUBBR         f, WREG         WREG = WREG - f - (C̄)         1         1         C, DC, N, OV, Z           SUBBR         f, WREG         WREG = WREG - f - (C̄)         1         1         C, DC, N, OV, Z           SUBBR         Wb, Ws, Wd         Wd = Ws - Wb - (C̄)         1         1         C, DC, N, OV, Z           SUBBR         Wb, #lit5, Wd         Wd = lit5 - Wb - (C̄)         1         1         C, DC, N, OV, Z           SWAP         SWAP.b         Wn         Wn = Nibble Swap Wn         1         1         None		SUBR	Wb,Ws,Wd	Wd = Ws – Wb	1	1	C, DC, N, OV, Z
SUBBR         f         f = WREG - f - (C̄)         1         1         C, DC, N, OV, Z           SUBBR         f, WREG         WREG = WREG - f - (C̄)         1         1         C, DC, N, OV, Z           SUBBR         f, WREG         WREG = WREG - f - (C̄)         1         1         C, DC, N, OV, Z           SUBBR         Wb, Ws, Wd         Wd = Ws - Wb - (C̄)         1         1         C, DC, N, OV, Z           SUBBR         Wb, #lit5, Wd         Wd = lit5 - Wb - (C̄)         1         1         C, DC, N, OV, Z           SWAP         SWAP.b         Wn         Wn = Nibble Swap Wn         1         1         None           SWAP         Wn         Wn = Byte Swap Wn         1         1         None		SUBR	Wb,#lit5,Wd	Wd = lit5 – Wb	1	1	C, DC, N, OV, Z
SUBBR         f, WREG         WREG = WREG - f - (C̄)         1         1         C, DC, N, OV, Z           SUBBR         Wb, Ws, Wd         Wd = Ws - Wb - (C̄)         1         1         C, DC, N, OV, Z           SUBBR         Wb, Hlit5, Wd         Wd = Ws - Wb - (C̄)         1         1         C, DC, N, OV, Z           SWAP         Wb, #lit5, Wd         Wd = lit5 - Wb - (C̄)         1         1         C, DC, N, OV, Z           SWAP         Wn         Wn = Nibble Swap Wn         1         1         None           SWAP         Wn         Wn = Byte Swap Wn         1         1         None	SUBBR	SUBBR	f	$f = WREG - f - (\overline{C})$	1	1	C, DC, N, OV, Z
SUBBR         Wb, Ws, Wd         Wd = Ws - Wb - (C̄)         1         1         C, DC, N, OV, Z           SUBBR         Wb, #lit5, Wd         Wd = lit5 - Wb - (C̄)         1         1         C, DC, N, OV, Z           SWAP         SWAP.b         Wn         Wn = Nibble Swap Wn         1         1         None           SWAP         Wn         Wn = Byte Swap Wn         1         1         None		SUBBR	f,WREG	WREG = WREG $- f - (\overline{C})$	1	1	C, DC, N, OV, Z
SUBBR         Wb,#lit5,Wd         Wd = lit5 - Wb - (C̄)         1         1         C, DC, N, OV, Z           SWAP         SWAP.b         Wn         Wn = Nibble Swap Wn         1         1         None           SWAP         Wn         Wn = Byte Swap Wn         1         1         None		SUBBR	Wb,Ws,Wd	$Wd = Ws - Wb - (\overline{C})$	1	1	C, DC, N, OV, Z
SWAP         SWAP.b         Wn         Wn = Nibble Swap Wn         1         1         None           SWAP         Wn         Wn = Byte Swap Wn         1         1         None		SUBBR	Wb,#lit5,Wd	$Wd = lit5 - Wb - (\overline{C})$	1	1	C, DC, N, OV, Z
SWAP         Wn         Wn = Byte Swap Wn         1         1         None	SWAP	SWAP.b	Wn	Wn = Nibble Swap Wn	1	1	None
		SWAP	Wn	Wn = Byte Swap Wn	1	1	None

#### FIGURE 28-8: BAUD RATE GENERATOR OUTPUT TIMING



#### FIGURE 28-9: START BIT EDGE DETECTION



#### TABLE 28-22: AC SPECIFICATIONS

Symbol	Characteristics	Min	Тур	Max	Units
TLW	BCLKx High Time	20	Tcy/2		ns
THW	BCLKx Low Time	20	(TCY * BRGx) + TCY/2	_	ns
TBLD	BCLKx Falling Edge Delay from UxTX	-50	—	50	ns
Твно	BCLKx Rising Edge Delay from UxTX	Tcy/2 – 50	—	Tcy/2 + 50	ns
Twak	Min. Low on UxRX Line to Cause Wake-up	—	1	_	μs
Тстѕ	Min. Low on UxCTS Line to Start Transmission	Тсү	—	—	ns
TSETUP	Start bit Falling Edge to System Clock Rising Edge Setup Time	3	—	—	ns
TSTDELAY	Maximum Delay in the Detection of the Start bit Falling Edge	—	—	TCY + TSETUP	ns



#### FIGURE 28-23: PARALLEL MASTER PORT WRITE TIMING DIAGRAM

TABLE 28-36:	PARALLEL	MASTER	PORT WRITE	TIMING	REQUIREMENTS

AC CHARACTERISTICS			Standard Operating Conditions: 2.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}C \le TA \le +85^{\circ}C$ for Industrial				
Param. No	Symbol	Characteristics <sup>(1)</sup>	Min	Тур	Мах	Units	Conditions
PM11		PMWR Pulse Width	_	0.5 TCY	—	ns	
PM12		Data Out Valid before PMWR or PMENB goes Inactive (data setup time)	—	0.75 TCY	—	ns	
PM13		PMWR or PMEMB Invalid to Data Out Invalid (data hold time)	—	0.25 TCY	—	ns	
PM16		PMCSx Pulse Width	Tcy – 5	_	—	ns	

**Note 1:** Wait states disabled for all cases.