

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

Active
dsPIC
16-Bit
40 MIPs
CANbus, I ² C, IrDA, LINbus, SPI, UART/USART
AC'97, Brown-out Detect/Reset, DMA, I ² S, POR, PWM, WDT
85
64KB (64K × 8)
FLASH
-
16K x 8
3V ~ 3.6V
A/D 32x10b/12b
Internal
-40°C ~ 85°C (TA)
Surface Mount
100-TQFP
100-TQFP (12x12)
https://www.e-xfl.com/product-detail/microchip-technology/dspic33fj64gp710at-i-pt

Email: info@E-XFL.COM

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

NOTES:

7.0 INTERRUPT CONTROLLER

- **Note 1:** This data sheet summarizes the features of the dsPIC33FJXXXGPX06A/X08A/ X10A family of devices. However, it is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to Section 6. "Interrupts" (DS70184) in the "dsPIC33F/PIC24H Familv Reference Manual", which is available from the Microchip web site (www.microchip.com).
 - 2: Some registers and associated bits described in this section may not be available on all devices. Refer to **Section 4.0 "Memory Organization"** in this data sheet for device-specific register and bit information.

The dsPIC33FJXXXGPX06A/X08A/X10A interrupt controller reduces the numerous peripheral interrupt request signals to a single interrupt request signal to the dsPIC33FJXXXGPX06A/X08A/X10A CPU. It has the following features:

- Up to eight processor exceptions and software traps
- Seven user-selectable priority levels
- Interrupt Vector Table (IVT) with up to 118 vectors
- A unique vector for each interrupt or exception source
- · Fixed priority within a specified user priority level
- Alternate Interrupt Vector Table (AIVT) for debug support
- · Fixed interrupt entry and return latencies

7.1 Interrupt Vector Table

The Interrupt Vector Table is shown in Figure 7-1. The IVT resides in program memory, starting at location 000004h. The IVT contains 126 vectors consisting of eight non-maskable trap vectors plus up to 118 sources of interrupt. In general, each interrupt source has its own vector. Each interrupt vector contains a 24-bit wide address. The value programmed into each interrupt vector location is the starting address of the associated Interrupt Service Routine (ISR).

Interrupt vectors are prioritized in terms of their natural priority; this priority is linked to their position in the vector table. All other things being equal, lower addresses have a higher natural priority. For example, the interrupt associated with vector 0 will take priority over interrupts at any other vector address.

dsPIC33FJXXXGPX06A/X08A/X10A devices implement up to 67 unique interrupts and five non-maskable traps. These are summarized in Table 7-1 and Table 7-2.

7.1.1 ALTERNATE VECTOR TABLE

The Alternate Interrupt Vector Table (AIVT) is located after the IVT, as shown in Figure 7-1. Access to the AIVT is provided by the ALTIVT control bit (INTCON2<15>). If the ALTIVT bit is set, all interrupt and exception processes use the alternate vectors instead of the default vectors. The alternate vectors are organized in the same manner as the default vectors.

The AIVT supports debugging by providing a means to switch between an application and a support environment without requiring the interrupt vectors to be reprogrammed. This feature also enables switching between applications for evaluation of different software algorithms at run time. If the AIVT is not needed, the AIVT should be programmed with the same addresses used in the IVT.

7.2 Reset Sequence

A device Reset is not a true exception because the interrupt controller is not involved in the Reset process. The dsPIC33FJXXXGPX06A/X08A/X10A device clears its registers in response to a Reset, which forces the PC to zero. The digital signal controller then begins program execution at location 0x000000. The user programs a GOTO instruction at the Reset address which redirects program execution to the appropriate start-up routine.

Note: Any unimplemented or unused vector locations in the IVT and AIVT should be programmed with the address of a default interrupt handler routine that contains a RESET instruction.

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0					
U2TXIE	U2RXIE	INT2IE	T5IE	T4IE	OC4IE	OC3IE	DMA2IE					
bit 15							bit 8					
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0					
IC8IE	IC7IE	AD2IE	INT1IE	CNIE		MI2C1IE	SI2C1IE					
bit 7		1					bit 0					
L												
Legend:												
R = Readable	e bit	W = Writable	bit	U = Unimple	emented bit, read	1 as '0'						
-n = Value at	POR	'1' = Bit is set		'0' = Bit is cl	eared	x = Bit is unkr	nown					
bit 15	U2TXIE: UAF	U2TXIE: UART2 Transmitter Interrupt Enable bit										
	1 = Interrupt request enabled											
	0 = Interrupt	0 = Interrupt request not enabled										
DIT 14	U2RXIE: UAP		nterrupt Enab	Ie dit								
	0 = Interrupt	request enable	u abled									
bit 13	INT2IE: Exter	INT2IE: External Interrupt 2 Enable bit										
	1 = Interrupt i	request enable	d									
	0 = Interrupt	request not ena	abled									
bit 12	T5IE: Timer5	T5IE: Timer5 Interrupt Enable bit										
	1 = Interrupt	request enable	d									
bit 11	0 = Interrupt 1	Interrupt Engl	abled Io bit									
	1 = Interrupt request enabled											
	0 = Interrupt I	request not enable	abled									
bit 10	OC4IE: Output	OC4IE: Output Compare Channel 4 Interrupt Enable bit										
	1 = Interrupt	1 = Interrupt request enabled										
hit Q		ut Compare Ch	ionnel 3 Interr	unt Enable bit	÷							
Dit 9	1 = Interrupt	request enable	d									
	0 = Interrupt request enabled											
bit 8	DMA2IE: DM	A Channel 2 D	ata Transfer (Complete Inter	rrupt Enable bit							
	1 = Interrupt	1 = Interrupt request enabled										
	0 = Interrupt	request not ena	abled									
bit 7	IC8IE: Input (Capture Chann	el 8 Interrupt d	Enable bit								
	1 = Interrupt I 0 = Interrupt I	request enable	u abled									
bit 6	IC7IE: Input (Capture Chann	el 7 Interrupt	Enable bit								
	1 = Interrupt	request enable	d									
	0 = Interrupt	request not ena	abled									
bit 5	AD2IE: ADC2	2 Conversion C	omplete Inter	rupt Enable b	it							
	1 = Interrupt	request enable	d									
hit 4		request not ena	IDIEU Enable bit									
	INTITE: External Interrupt 1 Enable bit											
	0 = Interrupt	request not enable	abled									

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0					
_		U1RXIP<2:0>		—		SPI1IP<2:0>						
bit 15							bit 8					
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0					
_		SPI1EIP<2:0>		—		T3IP<2:0>						
bit 7							bit 0					
Legend:												
R = Readable	bit	W = Writable b	oit	U = Unimple	mented bit, re	ad as '0'						
-n = Value at I	POR	'1' = Bit is set		'0' = Bit is cl	0' = Bit is cleared x = Bit is unkn							
bit 15	Unimplem	ented: Read as '0)' • • •									
bit 14-12	U1RXIP<2	UTRAIP<2:U>: UARTT Receiver Interrupt Priority Dits										
	111 = Inter •	rupt is priority 7 (r	lignest priori	ity interrupt)								
	•											
	•											
	001 = Inter	rupt is priority 1	phlod									
bit 11		anted: Pead as '	ableu v									
bit 10_8	SPI1IP<2:0>: SPI1 Event Interrupt Priority bits											
bit 10-0	111 = Interrupt is priority 7 (highest priority interrupt)											
	•											
	•											
	• $0.01 = \text{Interrupt in priority} 1$											
	001 - Inter	rupt is priority i rupt source is disa	abled									
bit 7	Unimplem	ented: Read as '0)'									
bit 6-4	SPI1EIP<2	::0>: SPI1 Error In	terrupt Prior	ity bits								
	111 = Inter	rupt is priority 7 (h	nighest priori	ity interrupt)								
	•											
	•											
	001 = Inte r	rupt is priority 1										
	000 = Inter	rupt source is disa	abled									
bit 3	Unimplem	ented: Read as '0)'									
bit 2-0	T3IP<2:0>:	: Timer3 Interrupt	Priority bits									
	<pre>111 = Interrupt is priority 7 (highest priority interrupt)</pre>											
	•											
	•											
	001 = Inter	rrupt is priority 1										
	000 = Inter	rupt source is disa	abled									

REGISTER	7-21: IPC6:	INTERRUPT	PRIORITY	CONTROL R	EGISTER 6							
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0					
—		T4IP<2:0>				OC4IP<2:0>						
bit 15							bit 8					
	D 4 4	DAMA	DAMO			DAALO	DAMA					
0-0	R/W-1		R/W-0	0-0	R/W-1	R/W-U	R/W-0					
bit 7		00011 42.04					bit 0					
Legend:												
R = Readabl	le bit	W = Writable I	bit	U = Unimple	mented bit, rea	ad as '0'						
-n = Value at	t POR	'1' = Bit is set		'0' = Bit is cle	eared	x = Bit is unkn	own					
bit 15	Unimpleme	nted: Pead as 'r	،'									
bit 14_12		Unimplemented: Read as "0"										
	111 = Intern	111 = Interrupt is priority 7 (highest priority interrupt)										
	•		ingricot priori	ty monapt)								
	•											
	• 001 - Interr	unt in priority 1										
	001 = Intern	upt is priority 1 upt source is dis:	abled									
bit 11	Unimpleme	nted: Read as '()'									
bit 10-8	OC4IP<2:0>	OC4IP<2:0>: Output Compare Channel 4 Interrupt Priority bits										
	111 = Interru	111 = Interrupt is priority 7 (highest priority interrupt)										
	•	•										
	•											
	• 001 = Interri	upt is priority 1										
	000 = Interru	upt source is disa	abled									
bit 7	Unimpleme	nted: Read as 'd)'									
bit 6-4	OC3IP<2:0>	: Output Compa	re Channel 3	3 Interrupt Prior	rity bits							
	111 = Interru	upt is priority 7 (h	nighest priori	ty interrupt)								
	•											
	•											
	001 = Interro	upt is priority 1										
	000 = Interro	upt source is disa	abled									
bit 3	Unimpleme	nted: Read as '0)'									
bit 2-0	DMA2IP<2:0	DMA2IP<2:0>: DMA Channel 2 Data Transfer Complete Interrupt Priority bits										
	111 = Interru	upt is priority 7 (ł	nighest priori	ty interrupt)								
	•											
	•											
	001 = Intern	upt is priority 1										
	000 = Interru	upt source is disa	abled									

R/W-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
FORCE ⁽¹⁾	—	—	_	—	—	—	—
bit 15	•	•				•	bit 8
U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	IRQSEL6(2)	IRQSEL5(2)	IRQSEL4(2)	IRQSEL3(2)	IRQSEL2 ⁽²⁾	IRQSEL1(2)	IRQSEL0(2)
bit 7							bit 0
Legend:							
R = Readable bit		W = Writable bit		U = Unimpler	nented bit, read	as '0'	
-n = Value at P	OR	'1' = Bit is set		'0' = Bit is cleared		x = Bit is unknown	
bit 15	FORCE: Force	e DMA Transfe	er bit ⁽¹⁾				
	1 = Force a si	ingle DMA tran	sfer (Manual r	mode)			
	0 = Automatic	: DMA transfer	initiation by D	MA request			
bit 14-7	Unimplemen	ted: Read as '	0'				
bit 6-0	IRQSEL<6:0>	-: DMA Periphe	eral IRQ Num	ber Select bits	(2)		
	1111111 = D	MAIRQ127 sel	lected to be C	hannel DMARI	EQ		
	•						
	•						
0000000 = DMAIRQ0 selected to be Channel DMAREQ							

REGISTER 8-2: DMAxREQ: DMA CHANNEL x IRQ SELECT REGISTER

- **Note 1:** The FORCE bit cannot be cleared by the user. The FORCE bit is cleared by hardware when the forced DMA transfer is complete.
 - 2: Please see Table 8-1 for a complete listing of IRQ numbers for all interrupt sources.

10.0 POWER-SAVING FEATURES

- **Note 1:** This data sheet summarizes the features of the dsPIC33FJXXXGPX06A/X08A/ X10A family of devices. However, it is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to Section 9. "Watchdog Timer and Power-Saving Modes" (DS70196) in "dsPIC33F/PIC24H Familv the Reference Manual", which is available the site from Microchip web (www.microchip.com).
 - 2: Some registers and associated bits described in this section may not be available on all devices. Refer to Section 4.0 "Memory Organization" in this data sheet for device-specific register and bit information.

The dsPIC33FJXXXGPX06A/X08A/X10A devices provide the ability to manage power consumption by selectively managing clocking to the CPU and the peripherals. In general, a lower clock frequency and a reduction in the number of circuits being clocked constitutes lower consumed power. dsPIC33FJXXXGPX06A/X08A/X10A devices can manage power consumption in four different ways:

- Clock frequency
- Instruction-based Sleep and Idle modes
- Software-controlled Doze mode
- Selective peripheral control in software

Combinations of these methods can be used to selectively tailor an application's power consumption while still maintaining critical application features, such as timing-sensitive communications.

10.1 Clock Frequency and Clock Switching

dsPIC33FJXXXGPX06A/X08A/X10A devices allow a wide range of clock frequencies to be selected under application control. If the system clock configuration is not locked, users can choose low-power or high-precision oscillators by simply changing the NOSC bits (OSCCON<10:8>). The process of changing a system clock during operation, as well as limitations to the process, are discussed in more detail in **Section 9.0 "Oscillator Configuration"**.

10.2 Instruction-Based Power-Saving Modes

dsPIC33FJXXXGPX06A/X08A/X10A devices have two special power-saving modes that are entered through the execution of a special PWRSAV instruction. Sleep mode stops clock operation and halts all code execution. Idle mode halts the CPU and code execution, but allows peripheral modules to continue operation. The assembly syntax of the PWRSAV instruction is shown in Example 10-1.

Note: SLEEP_MODE and IDLE_MODE are constants defined in the assembler include file for the selected device.

Sleep and Idle modes can be exited as a result of an enabled interrupt, WDT time-out or a device Reset. When the device exits these modes, it is said to "wake-up".

10.2.1 SLEEP MODE

Sleep mode has these features:

- The system clock source is shut down. If an on-chip oscillator is used, it is turned off.
- The device current consumption is reduced to a minimum, provided that no I/O pin is sourcing current
- The Fail-Safe Clock Monitor does not operate during Sleep mode since the system clock source is disabled
- The LPRC clock continues to run in Sleep mode if the WDT is enabled
- The WDT, if enabled, is automatically cleared prior to entering Sleep mode
- Some device features or peripherals may continue to operate in Sleep mode. This includes items such as the input change notification on the I/O ports, or peripherals that use an external clock input. Any peripheral that requires the system clock source for its operation is disabled in Sleep mode.

The device will wake-up from Sleep mode on any of these events:

- Any interrupt source that is individually enabled
- Any form of device Reset
- A WDT time-out

On wake-up from Sleep, the processor restarts with the same clock source that was active when Sleep mode was entered.

EXAMPLE 10-1: PWRSAV INSTRUCTION SYNTAX

PWRSAV#SLEEP_MODE; Put the device into SLEEP modePWRSAV#IDLE_MODE; Put the device into IDLE mode



16.1 SPI Helpful Tips

- 1. In Frame mode, if there is a possibility that the master may not be initialized before the slave:
 - a) If FRMPOL (SPIxCON2<13>) = 1, use a pull-down resistor on SSx.
 - b) If FRMPOL = 0, use a pull-up resistor on \overline{SSx} .

Note:	This	insures	that	the	first	fra	ame
	transr	nission	after	initializ	ation	is	not
	shifte	d or corru	upted.				

- 2. In non-framed 3-wire mode, (i.e., not using SSx from a master):
 - a) If CKP (SPIxCON1<6>) = 1, always place a pull-up resistor on SSx.
 - b) If CKP = <u>0</u>, always place a pull-down resistor on SSx.
- Note: This will insure that during power-up and initialization the master/slave will not lose sync due to an errant SCK transition that would cause the slave to accumulate data shift errors for both transmit and receive appearing as corrupted data.
- FRMEN (SPIxCON2<15>) = 1 and SSEN (SPIxCON1<7>) = 1 are exclusive and invalid. In Frame mode, SCKx is continuous and the Frame sync pulse is active on the SSx pin, which indicates the start of a data frame.
- Note: Not all third-party devices support Frame mode timing. Refer to the SPI electrical characteristics for details.
- In Master mode only, set the SMP bit (SPIxCON1<9>) to a '1' for the fastest SPI data rate possible. The SMP bit can only be set at the same time or after the MSTEN bit (SPIxCON1<5>) is set.
- 5. To avoid invalid slave read data to the master, the user's master software must guarantee enough time for slave software to fill its write buffer before the user application initiates a master write/read cycle. It is always advisable to preload the SPIxBUF transmit register in advance of the next master transaction cycle. SPIxBUF is transferred to the SPI shift register and is empty once the data transmission begins.

16.2 SPI Resources

Many useful resources related to SPI are provided on the main product page of the Microchip web site for the devices listed in this data sheet. This product page, which can be accessed using this link, contains the latest updates and additional information.

Note:	In the event you are not able to access the
	product page using the link above, enter
	this URL in your browser:
	http://www.microchip.com/wwwproducts/
	Devices.aspx?dDocName=en546064

16.2.1 KEY RESOURCES

- Section 18. "Serial Peripheral Interface (SPI)" (DS70206)
- Code Samples
- Application Notes
- · Software Libraries
- Webinars
- All related dsPIC33F/PIC24H Family Reference Manuals Sections
- Development Tools

U-0	U-0	U-0	R-0	R-0	R-0	R-0	R-0			
	_				FILHIT<4:0>	>				
bit 15							bit 8			
U-0	R-1	R-0	R-0	R-0	R-0	R-0	R-0			
				ICODE<6:0>						
bit 7							bit 0			
Legend:										
R = Readable	bit	W = Writable b	oit	U = Unimplemented bit, read as '0'						
-n = Value at F	-n = Value at POR '1' = Bit is set			'0' = Bit is cle	ared	x = Bit is unkn	iown			
bit 15-13	Unimplemen	ted: Read as '0	,							
bit 12-8	FILHIT<4:0>:	Filter Hit Numb	er bits							
2.1.12.0	10000-1111	1 = Reserved								
	01111 = Filte	r 15								
	•									
	•									
	00001 = Filte	r 1								
	00000 = Filter 0									
bit 7	Unimplemen	ted: Read as '0	,							
bit 6-0	ICODE<6:0>: Interrupt Flag Code bits									
	1000101-111	11111 = Reserv	ved							
	1000100 = F	IFO almost full i	nterrupt							
	1000011 = R	eceiver overflov	v interrupt							
	1000010 = W	/ake-up interrup	t							
	1000001 = E									
	0010000-011	111111 = Reserv	ved							
	0001111 = R	B15 buffer Inter	rupt							
	•									
	•									
	• 0001001 = R	B9 buffer interru	tau							
	0001000 = R	B8 buffer interru	upt							
	0000111 = T	RB7 buffer inter	rupt							
	0000110 = T	RB6 buffer inter	rupt							
	0000101 = T	RB5 buffer inter	rupt							
	0000100 = T	RB4 buffer inter	rupt							
	0000011 = 10000010 = 100000000000000000	RB3 buffer inter	rupt							
	0000010 = 10	RB1 huffer inter	runt							
	0000000 = T	RB0 Buffer inter	rupt							

REGISTER 19-3: CIVEC: ECAN™ INTERRUPT CODE REGISTER

REGISTER 19-15: CiBUFPNT4: ECAN™ FILTER 12-15 BUFFER POINTER REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	F15BF	P<3:0>			F14B	P<3:0>	
bit 15							bit 8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	F13BF	><3:0>			F12B	P<3:0>	
bit 7							bit 0
Legend:							
R = Readable	e bit	W = Writable	bit	U = Unimplen	nented bit, rea	d as '0'	
-n = Value at	POR	'1' = Bit is set		·0' = Bit is cle	ared	x = Bit is unkr	nwn
n value at					area		
bit 15-12	F15BP<3:0> 1111 = Filter 1110 = Filter •	: RX Buffer Wri r hits received in r hits received in	tten when Fil າ RX FIFO bu າ RX Buffer 1	ter 15 Hits bits ıffer 4			
	0001 = Filte r 0000 = Filte r	r hits received in r hits received in	n RX Buffer 1 n RX Buffer 0				
bit 11-8	F14BP<3:0> 1111 = Filter 1110 = Filter	RX Buffer Wri hits received in hits received in	tten when Fil າ RX FIFO bu າ RX Buffer 1	ter 14 Hits bits ıffer 4			
	•						
	0001 = Filte r 0000 = Filte r	r hits received in r hits received in	n RX Buffer 1 n RX Buffer 0				
bit 7-4	F13BP<3:0> 1111 = Filter 1110 = Filter •	: RX Buffer Wri r hits received in r hits received in	tten when Fil າ RX FIFO bu າ RX Buffer 1	ter 13 Hits bits ıffer 4			
	• 0001 = Filter 0000 = Filter	r hits received in r hits received in	n RX Buffer 1 n RX Buffer 0				
bit 3-0	F12BP<3:0> 1111 = Filter 1110 = Filter •	: RX Buffer Wri r hits received in r hits received in	tten when Fil າ RX FIFO bu າ RX Buffer 1	ter 12 Hits bits ıffer 4			
	• 0001 = Filter 0000 = Filter	r hits received in r hits received in	n RX Buffer 1 n RX Buffer 0				

REGISTER 20-3: DCICON3: DCI CONTROL REGISTER 3

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—		BCG	i<11:8>	
bit 15							bit 8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
			BCG	<7:0>			
bit 7							bit 0
Legend:							
R = Readable	bit	W = Writable	bit	U = Unimplemented bit, read as '0'			
-n = Value at F	POR	'1' = Bit is set		'0' = Bit is cleared x = Bit is unknown			

bit 15-12 Unimplemented: Read as '0'

bit 11-0 BCG<11:0>: DCI Bit Clock Generator Control bits

NOTES:

NOTES:

48 MPY μrv Mim * Nin . Acc., Xix . Rod., Wy , Wyd. Multiply Wn by Wn to Accumulator 1 1 0. A OB OAB. ASB SAB 49 MPY . Ni Mim * Win . Acc., Xix . Rod., Wy , Wyd. Square Wn to Accumulator 1 1 0. A OB OAB. ASB SAB 49 MPY . Ni Mim * Wn . Acc., Xix . Rod., Wy , Wyd. (Multiply Wm by Wn to Accumulator 1 1 1 0. A OB OAB. ASB SAB 50 MSC Rec _ Xix . Rod., Wy , Wyd. Multiply and Subtract from Accumulator 1 1 1 0. A OB OAB. ASB SAB 51 MUL _ ST _ Wn , No. Ro. Nr.A. (Wnd + 1, Wnd) = signed(Wb) * singend(Wb) 1 1 None 400101 KD , No. No. Ro. Nr.A. (Wnd + 1, Wnd) = signed(Wb) * singend(Wb) 1 1 None 40001 KD , No. No. Ro. Nr.A. (Wnd + 1, Wnd) = signed(Wb) * singend(Wb) 1 1 None 40101 KD , No. Ro. Nr.A. (Wnd + 1, Wnd) = signed(Wb) * unsigned(Wb) 1 1 None 40101 KD , No. Ro. Nr.A. (Wnd + 1, Wnd) = signed(Wb) * unsigned(Wb) 1 1 None 40101 KD , No. Ro. Nr.A. (Wnd + 1, Wnd) = signed(Wb) * unsigned(Wb)	Base Instr #	Assembly Mnemonic	Assembly Syntax		Description	# of Words	# of Cycles	Status Flags Affected
NP NP<	48	MPY	MPY Wm*Wn,A	cc,Wx,Wxd,Wy,Wyd	Multiply Wm by Wn to Accumulator	1	1	OA,OB,OAB, SA,SB,SAB
49 MPY.N MPY.N			MPY Wm*Wm,A	cc,Wx,Wxd,Wy,Wyd	Square Wm to Accumulator	1	1	OA,OB,OAB, SA,SB,SAB
50 MSC KSC Non-AcC, NZ, Kod, Wy, Wyd, YM, YM, YM, YM, YM, YM, YM, YM, YM, YM	49	MPY.N	MPY.N Wm*Wn,Acc,Wx,Wxd,Wy,Wyd		-(Multiply Wm by Wn) to Accumulator	1	1	None
51 MUL MUL, SU MD, SU, MD, MD, MD, MD, MD (Mnd + 1, Wnd) = signed(WD) * signed(WD) 1 1 None MUL, SU MD, SU, MD, MD, MD, MD, MD, MD, MD, MD, MD, MD	50	MSC	MSC	Wm*Wm,Acc,Wx,Wxd,Wy,Wyd , AWB	Multiply and Subtract from Accumulator	1	1	OA,OB,OAB, SA,SB,SAB
MUL. SU Wo. Ms. Mcd. (Mrd + 1, Mrd) = signed(Wb) * unsigned(Wb) 1 1 None MUL. US Kb., Ma, Krad (Wrd + 1, Mrd) = unsigned(Wb) * unsigned(Wb) 1 1 None MUL. US Kb., Ma, Krad (Wrd + 1, Mrd) = unsigned(Wb) * unsigned(Wb) * unsigned(Wb) 1 1 None MUL. US Kb., Hill, E., Wrd (Wrd + 1, Mrd) = unsigned(Wb) * unsigned(Wb) * 1 1 None MUL. US Kb., Hill, E., Wrd (Wrd + 1, Mrd) = unsigned(Wb) * unsigned(Wb) * 1 1 None MUL. US Kb., Hill, E., Wrd (Wrd + 1, Mrd) = unsigned(Wb) * 1 1 None MUL f C Wild, H., Mrd) Wild, H., Wrd) = unsigned(Wb) * 1 1 None MUL, US Kb., Ma, Md Wd = Tis + 1 1 1 C.OC, NOVZ NBG Niggi E.Auge WrEG = i + 1 1 1 C.OC, NOVZ NBG NoP NoP NoOP No Operation 1 1 None So NOP POP POP POP from	51	MUL	MUL.SS	Wb,Ws,Wnd	{Wnd + 1, Wnd} = signed(Wb) * signed(Ws)	1	1	None
Mark Mark (Wnd + 1, Wnd) = unsigned(Wb)*signed(Wb) = 1 1 None MUL.UU Wb, He, Mnd (Wnd + 1, Wnd) = unsigned(Wb)* unsigned(Wb) = 1 1 1 None MUL.UU Wb, #1:15, Wnd (Wnd + 1, Wnd) = unsigned(Wb)* unsigned(Wb)* 1 1 None MUL.UU Wb, #1:15, Wnd (Wnd + 1, Wnd) = unsigned(Wb)* unsigned(Wb)* 1 1 None MUL.UU Wb, #1:15, Wnd (Wnd + 1, Wnd) = unsigned(Wb)* 1 1 None MUL.UU Wb, #1:15, Wnd (Wnd + 1, Wnd) = unsigned(Wb)* 1 1 None MUL.UU Wb, #1:15, Wnd (Wnd + 1, Wnd) = unsigned(Wb)* 1 1 None MUL.UU Wb, #1:15, Wnd (Wnd + 1, Wnd) = unsigned(Wb)* 1 1 None MUL.UU Wb, #1:15, Wnd (Wnd + 1, Wnd) = unsigned(Wb)* 1 1 None MUL.UU Wb, #1:15, Wnd Wol Wal			MUL.SU	Wb,Ws,Wnd	{Wnd + 1, Wnd} = signed(Wb) * unsigned(Ws)	1	1	None
MUL.UU Wb.Ws.Ma. [Wnd+1, Wnd) = unsigned(Wb)* 1 1 None MUL.UU Wb.#lit5, Mnd [Wnd+1, Wnd] = unsigned(Wb)* 1 1 None MUL.UU Wb.#lit5, Mnd [Wnd+1, Wnd] = unsigned(Wb)* 1 1 None MUL f Ws.Miss.Wnd [Wnd+1, Wnd] = unsigned(Wb)* 1 1 None S2 MUL f Mode Masser None None MUL f Acc NegateAccumulator 1 1 1 C.DC.NOVZ NBG MSG w.Md WdEG =f+1 1 1 C.DC.NOVZ NBG NOP NO Poreation 1 1 None S4 POP S0 S No Operation 1 1 None S4 POP POP S Pop fram Top-of-Stack (TOS) 1 1 None S5 POP Wod Pop fram Top-of-Stack (TOS) 1 1 None PU			MUL.US	Wb,Ws,Wnd	{Wnd + 1, Wnd} = unsigned(Wb) * signed(Ws)	1	1	None
NUL. SU NDL. BULLS, Wid. (Wnd + 1, Wnd) = signed(Wb) * unsigned(Wb) * unsigned(Wb) * 1 1 None MUL. UV Wb. #11L5, Wnd (Wnd + 1, Wnd) = unsigned(Wb) * unsigned(Wb) * 1 1 1 None S2 NEG Acc Negate Accumulator 1 1 1 OA GB QAB, SASB,SAB NEG Acc Negate Accumulator 1 1 1 C.DC.N.OVZ NEG f, WEBG f = f + 1 1 1 1 C.DC.N.OVZ S3 NOP NOP No Operation 1 1 1 None 54 POP POP P POP POP POP No Portion Top-of-Stack (TOS) 1 1 None F00. D Wnd Pop from Top-of-Stack (TOS) 1 1 None P09. N Pop from Top-of-Stack (TOS) 1 1 None P09. N Pop from Top-of-Stack (TOS) 1 1 None P09. N Pop Shadow Registers 1			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 £ W3W2 = f * WREG 1 1 None 52 NEG NEG Acc Negate Accumulator 1 1 0A,0B,0AB, SA,8B,3AB NEG £ meg f=f+1 1 1 1 C,DC,N,OVZ NEG £,WEEG WREG=f+1 1 1 1 C,DC,N,OVZ 53 NOP NOP No Operation 1 1 None 54 POP £ Pop ff from Top-of-Stack (TOS) 1 1 None 54 POP £ Pop ff from Top-of-Stack (TOS) to Wdo 1 1 None 55 POS. Pop Stadox Registers 1 1 All 709.8.1 £ Push Wso Push Wso Top-of-Stack (TOS) 1 1 None 705.8.1 PUSH PUSH Wso Push Wso Top-of-Stack (TOS) 1 1 None 705.8.1 PUSH Wso Push Mone/Vso/Vso/Stack (TOS) 1 1 None			MUL.UU	Wb,#lit5,Wnd	{Wnd + 1, Wnd} = unsigned(Wb) * unsigned(lit5)	1	1	None
52 NEG Acc Negate Accumulator 1 1 0A,0B,0AB,SAB NKG £ f=f+1 1 1 C,DC,NO,VZ NKG £,MKEG WREG=f+1 1 1 C,DC,NO,VZ NKG No No No 1 1 C,DC,NO,VZ S3 NOP NOP No Operation 1 1 None 54 POP £ Pop from Top-of-Stack (TOS) 1 1 None 55 POP. Wado Pop from Top-of-Stack (TOS) to Wdo 1 1 All 55 PUSH £ POP.D Wad Pop from Top-of-Stack (TOS) to Wdo 1 1 All 55 PUSH £ Push fito Top-of-Stack (TOS) 1 1 None FUSH £ PUSH fito Top-of-Stack (TOS) 1 1 None FUSH F PUSH Nso Push Wins/Wink +1) to Top-of-Stack (TOS) 1 1 None			MUL	f	W3:W2 = f * WREG	1	1	None
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$	52	NEG	NEG	Acc	Negate Accumulator	1	1	OA,OB,OAB, SA,SB,SAB
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$			NEG	f	$f = \overline{f} + 1$	1	1	C,DC,N,OV,Z
NEG Ws. Wd Wd = Ws + 1 1 1 C.D.C.N.OV.Z 53 NOP NOP No Operation 1 1 None 54 POP POP F Pop from Top-of-Stack (TOS) 1 1 None 54 POP POP Mdo Pop from Top-of-Stack (TOS) 1 1 None 54 POP POP Mdo Pop from Top-of-Stack (TOS) 1 1 None 54 POP Wdo Pop from Top-of-Stack (TOS) 1 1 None 55 PUSH FUSH f Push Mso Push Wso to Top-of-Stack (TOS) 1 1 None 9USH PUSH Mso Push Wso to Top-of-Stack (TOS) 1 1 None 9USH Wso Push Shadow Registers 1 1 None 9USH Mso Computed Call 1 1 None 56 PWRSAV #lit1 Go into Sleep or Idle mode 1 1			NEG	f,WREG	WREG = \overline{f} + 1	1	1	C,DC,N,OV,Z
53 NOP NOP No Operation 1 1 1 None 54 POP POP f Pop ff or Top-of-Stack (TOS) 1 1 None 54 POP POP f Pop for Top-of-Stack (TOS) 1 1 None 54 POP Wdo Pop form Top-of-Stack (TOS) to Wdo 1 1 None 55 POF.D Wnd Pop form Top-of-Stack (TOS) to Wdo 1 1 All 55 PUSH POSH Pop Shadow Registers 1 1 All 56 PUSH. DUSH Msso Push Wso to Top-of-Stack (TOS) 1 1 None 56 PWRSAV PUSH wso Push Wso to Top-of-Stack (TOS) 1 1 None 57 RCALL Expx Push Wso to Top-of-Stack (TOS) 1 1 None 57 RCALL Expx Relative Call 1 1 None 58 REPEAT #Lit1 Go into Ste			NEG	Ws,Wd	$Wd = \overline{Ws} + 1$	1	1	C,DC,N,OV,Z
NOPR No Operation 1 1 None 54 POP f Pop from Top-of-Stack (TOS) 1 1 None 54 POP f Pop from Top-of-Stack (TOS) to Wdo 1 1 None 50 POP. D Wind Pop from Top-of-Stack (TOS) to Wdo 1 1 None 55 POP. J. Wind Pop from Top-of-Stack (TOS) 1 1 All 55 PUSH PUSH f Push viso Push viso top-of-Stack (TOS) 1 1 None 709.8.2 PUSH Wiso Push Wiso top-of-Stack (TOS) 1 1 None 7098.9 PUSH Wiso Push Wiso top-of-Stack (TOS) 1 1 None 7098.0 Push Wiso Push Wins)*/(ms + 1) to Top-of-Stack (TOS) 1 1 None 7098.0 PRESAV PRESAV #Itit1 Go into Steep or Idle mode 1 1 None 56 PWRSAV PRESAV #Itit1 Repeat	53	NOP	NOP		No Operation	1	1	None
54POPfPop from Top-of-Stack (TOS)111NonePDPWdoPop from Top-of-Stack (TOS) to Wdo111NonePOP.DWndWoloPop from Top-of-Stack (TOS) to Wdo112NonePOP.SPop from Top-of-Stack (TOS) to12NoneNoneFUSHPUSHfPush for Top-of-Stack (TOS)111All55PUSHPUSHfPush for Top-of-Stack (TOS)111NonePUSH.JWsoPush Wso to Top-of-Stack (TOS)111NonePUSH.JWsoPush Wso to Top-of-Stack (TOS)111NoneFUSH.JWinsPush Wso to Top-of-Stack (TOS)111None56PWRSAVPWSAV#111Go into Steep or Idle mode11None57RCALLRCALLExprRelative Call12None58REPEATRELLWnComputed Call12None59RESETSoftware device Reset11None60RETFLERETURNReturn from Subroutine13 (2)None61RETURNRETURNReturn from Subroutine13 (2)None62RETURNRETURNReturn from Subroutine110.N.Z64RLCffRotale Left through Carry f11N.Z64 <td></td> <td></td> <td>NOPR</td> <td></td> <td>No Operation</td> <td>1</td> <td>1</td> <td>None</td>			NOPR		No Operation	1	1	None
POP WdoPop from Top-of-Stack (TOS) to Wdo111NonePOP.DWindPop from Top-of-Stack (TOS) to W(nd):W(nd+1)12None55PUSHFOP.SPop Shadow Registers111All55PUSHFUSHfPush fo Top-of-Stack (TOS)111None56PUSH.DWinsPush Wso to Top-of-Stack (TOS)111None57FUSH.SPush Shadow Registers111None57RCALLExprRelative Call12None58REPEATRCALLExprRelative Call12None59RESETROMUED Call12None1None60RETFIERETURNReturn from interrupt13 (2)None61RETURNRETURNReturn from Subroutine11Conz62RETURNRETURNReturn from Subroutine13 (2)None63RLCff = Rotate Left through Carry f11C.N.Z64RLCff = Rotate Left through Carry f11N.Z65RLCff = Rotate Left through Carry f11N.Z64RLCff = Rotate Left through Carry f11N.Z65RLCff = Rotate Left (No Carry) f11N.Z66RLCf <t< td=""><td>54</td><td>POP</td><td>POP</td><td>f</td><td>Pop f from Top-of-Stack (TOS)</td><td>1</td><td>1</td><td>None</td></t<>	54	POP	POP	f	Pop f from Top-of-Stack (TOS)	1	1	None
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$			POP	Wdo	Pop from Top-of-Stack (TOS) to Wdo	1	1	None
POP.SPop Shadow Registers11All55PUSHPUSH fPush fto Top-of-Stack (TOS)11NonePUSH.PUSH fPush WsoPush Wso to Top-of-Stack (TOS)11NonePUSH.DWnoPush Wso to Top-of-Stack (TOS)11NonePUSH.DWnoPush Wso to Top-of-Stack (TOS)12NonePUSH.DWnsPush Shadow Registers11None56PWRSAV#1it1Go into Steep or Idle mode11WDTO,Steep57RCALLExprRelative Call12None58RCALLWnComputed Call12None58REPEAT#lit14Repeat Next Instruction lit14 + 1 times11None59RESETRESETSoftware device Reset11None60RETFIERETFIEReturn from interrupt13 (2)None61RETURNRETURNReturn from Subroutine13 (2)None62RETURNRETURNReturn from Subroutine11C.N.Z64RLCffRotate Left through Carry f11N.Z.64RLNCF.N.WWREGRotate Left Noc Garry f11N.Z.65RCCRCCff=Rotate Left (No Carry) f11N.Z.66RLNCf, WREGWREG = Rotate Left Nongh Carry f11			POP.D	Wnd	Pop from Top-of-Stack (TOS) to W(nd):W(nd + 1)	1	2	None
55PUSHPUSHfPush foTop-of-Stack (TOS)111NonePUSHWsoPush Wso to Top-of-Stack (TOS)111NonePUSH.DWnsPush Wso to Top-of-Stack (TOS)12NonePUSH.SPush Shadow Registers111None56PWRSAVPWRSAV#1it1Go into Sleep or Idle mode111WDTO,Sleep57RCALLExprRelative Call12None58REPEATRCALLExprRelative Call12None59RESETREPEAT#1it14Repeat Next Instruction lit14 + 1 times11None59RESETRETFIEReturn from interrupt13 (2)None61RETUWRETTUW#1it10. WnReturn with literal in Wn13 (2)None62RETURNRETURNREturnf = Rotate Left through Carry f11C,N,Z63RLCf.f.REtGe totate Left through Carry f11NZ64RLNCF.WREGRotate Left (No Carry) f11NZ65RRCRLNCf.KREGWREG = Rotate Left (No Carry) f11NZ64RLNCf.KREGWREG = Rotate Left (No Carry) f11NZ65RRCRRCf.KREGWREG = Rotate Left (No Carry) f11NZ			POP.S		Pop Shadow Registers	1	1	All
PUSHWisoPush WisoPush Wiso to Top-of-Stack (TOS)111NonePUSH.DWinsPush Wins): W(ns + 1) to Top-of-Stack (TOS)12NoneFUSH.SPush Shadow Registers111None56PWRSAVPWRSAV#1it1Go into Sleep or Idle mode11WDTO,Sleep57RCALLExprRelative Call12None58REPEATREPEAT#1it14Repeat Next Instruction lit14 + 1 times11None59RESETREPEATWinRepeat Next Instruction lit14 + 1 times11None59RESETRESETSoftware device Reset11None60RETFIERETFIEReturn from interrupt13 (2)None61RETURRETURNReturn with literal in Wn13 (2)None62RETURNRETURNReturn with literal in Wn13 (2)None63RLCff = Rotate Left through Carry f11C,N,Z64RLNCf, WREGWREG = Rotate Left through Carry f11N,Z64RLNCf, WREGWREG = Rotate Left (No Carry) f11N,Z65RRCff = Rotate Left (No Carry) f11N,Z66RLNCf, WREGWREG = Rotate Left (No Carry) f11N,Z66RLNCf, WREGWREG = Rotate Left (No Carry) f1 <td>55</td> <td>PUSH</td> <td>PUSH</td> <td>f</td> <td>Push f to Top-of-Stack (TOS)</td> <td>1</td> <td>1</td> <td>None</td>	55	PUSH	PUSH	f	Push f to Top-of-Stack (TOS)	1	1	None
$\begin{array}{ c c c c c c c c c c c c c c c c c c c$			PUSH	Wso	Push Wso to Top-of-Stack (TOS)	1	1	None
PUSH.SPush Shadow Registers11None56PWRSAVPWRSAV#lit1Go into Sleep or Idle mode11WDTO,Sleep57RCALLExprRelative Call12None58REPEATREPEAT#lit14Repeat Next Instruction lit14 + 1 times11None58REPEAT#REPEAT#lit14Repeat Next Instruction (Wn) + 1 times11None59RESETRESETSoftware device Reset11None60RETFIERETFIEReturn from interrupt13 (2)None61RETUWRETUW#lit10, WnReturn with literal in Wn13 (2)None62RETURRETURNReturn from Subroutine13 (2)None63RLCfff e Rotate Left through Carry f11C,N,Z64RLCf, WREGWREG = Rotate Left through Carry f11N,Z64RLNCf, WREGWREG = Rotate Left (No Carry) f11N,Z65RRCRRCff e Rotate Left (No Carry) f11N,Z65RRCRRCff e Rotate Right through Carry f11N,Z66RLNCf, WREGWREG = Rotate Left (No Carry) f11N,Z67RLNCf, WREGWREG = Rotate Left (No Carry) f11N,Z68RLNCf, WREGWREG = Rotate Left (No Ca			PUSH.D	Wns	Push W(ns):W(ns + 1) to Top-of-Stack (TOS)	1	2	None
56PWRSAVPWRSAV#lit1Go into Sleep or Idle mode11WDTO,Sleep57RCALLRCALLExprRelative Call12None58REPEATREPEAT#lit14Repeat Next Instruction lit14 + 1 times112None58REPEATREPEAT#lit14Repeat Next Instruction lit14 + 1 times11None59RESETRESETWnRepeat Next Instruction (Wn) + 1 times11None60RETFIERETFIEReturn from interrupt13 (2)None61RETLWRETUW#lit10, WnReturn with literal in Wn13 (2)None62RETURNRETURNReturn from Subroutine13 (2)None63RLCff = Rotate Left through Carry f11C,N,Z64RLNCf, WREGWREG = Rotate Left through Carry f11N,Z65RRCRRCf = Rotate Left (No Carry) f11N,Z65RRCRRCf = Rotate Left (No Carry) f11N,Z66RRCRRCf = Rotate Left (No Carry) f11N,Z65RRCRRCff = Rotate Left (No Carry) f11N,Z65RRCRRCff = Rotate Left (No Carry) f111C,N,Z66RRCRRCf = Rotate Left (No Carry) Ws111N,Z			PUSH.S		Push Shadow Registers	1	1	None
57RCALLRCALLExprRelative Call12None757RCALLWnComputed Call12None758REPEATREPEAT#1114Repeat Next Instruction lit14 + 1 times11None758REPEATREPEAT#1114Repeat Next Instruction lit14 + 1 times11None759RESETRESETWnRepeat Next Instruction (Wn) + 1 times11None759RESETRESETSoftware device Reset11None760RETFIERETFIEReturn from interrupt13 (2)None761RETLW#1110, WnReturn with literal in Wn13 (2)None762RETURNRETURNReturn from Subroutine13 (2)None763RLCffRotate Left through Carry f11C,N,Z764RLNCfgKREGWREG = Rotate Left through Carry f11N,Z764RLNCffRotate Left (No Carry) f11N,Z765RRCRRCff = Rotate Left (No Carry) f11N,Z765RRCRRCff = Rotate Right through Carry f11C,N,Z765RRCRRCff = Rotate Right through Carry f11C,N,Z765RRCRRCff = Rotate Right through Carry f11C,N,Z765<	56	PWRSAV	PWRSAV	#lit1	Go into Sleep or Idle mode	1	1	WDTO,Sleep
RCALLWnComputed Call12None58REPEATREPEAT#lit14Repeat Next Instruction lit14 + 1 times11None59RESETRESETWnRepeat Next Instruction (Wn) + 1 times11None60RETFIERESETSoftware device Reset11None61RETLWRETLW#lit10,WnReturn from interrupt13 (2)None62RETURNRETURNReturn With literal in Wn13 (2)None63RLCffRotate Left through Carry f11C,N,Z64RLCf, WREGWREG = Rotate Left through Carry f11N,Z65RRCRRCff = Rotate Left (No Carry) f11N,Z65RRCKRCff = Rotate Right through Carry f11N,Z65RRCKRCff = Rotate Right through Carry f11C,N,Z66KRCKRCff = Rotate Right through Carry f11N,Z	57	RCALL	RCALL	Expr	Relative Call	1	2	None
58REPEATREPEAT#lit14Repeat Next Instruction lit14 + 1 times11None59RESETRESETWnRepeat Next Instruction (Wn) + 1 times11None60RETFIERESETSoftware device Reset11None61RETLWRETLW#lit10, WnReturn from interrupt13 (2)None61RETURNRETURNRETURNReturn from Subroutine13 (2)None62RETURNRETURNReturn from Subroutine13 (2)None63RLCffRotate Left through Carry f11C,N,Z64RLNCffRotate Left (No Carry) f11N,Z65RRCRRCffRotate Left (No Carry) f11N,Z65RRCRRCffRotate Left (No Carry) f11C,N,Z7RRCffRotate Left (No Carry) f11N,Z65RRCRRCffRotate Left (No Carry) f11N,Z65RRCRRCfMREGWREG = Rotate Right through Carry f11C,N,Z7RRCffRotate Left Right through Carry f11C,N,Z8RRCffRotate Left Right through Carry f11C,N,Z8RRCfgggGGGGG			RCALL	Wn	Computed Call	1	2	None
EXAMPLERepeatWnRepeat Next Instruction (Wn) + 1 times11None59RESETRESETRESETSoftware device Reset11None60RETFIERETFIEReturn from interrupt13 (2)None61RETLWRETLW#1it10,WnReturn with literal in Wn13 (2)None62RETURNRETURNReturn from Subroutine13 (2)None63RLCfff = Rotate Left through Carry f11C,N,Z64RLCf, WREGWREG = Rotate Left through Carry Ms11N,Z64RLNCf, WREGWREG = Rotate Left (No Carry) f11N,Z65RRCRRCff = Rotate Left (No Carry Ms11N,Z65RRCRRCff = Rotate Right through Carry f11C,N,Z66RRCRRCfMREGWREG = Rotate Left (No Carry) f11N,Z67RRCfRRCf = Rotate Left (No Carry) f11N,Z68RRCRRCfF = Rotate Left (No Carry) f11N,Z69RRCRRCfRRCf = Rotate Right through Carry f11C,N,Z60RRCRRCfRRCf = Rotate Right through Carry f11C,N,Z61RRCRRCRRCf = Rotate Right through Carry f11C,N	58	REPEAT	REPEAT	#lit14	Repeat Next Instruction lit14 + 1 times	1	1	None
59RESETRESETSoftware device Reset11None60RETFIERETFIEReturn from interrupt13 (2)None61RETLWRETLW#lit10,WnReturn with literal in Wn13 (2)None62RETURNRETURNReturn from Subroutine13 (2)None63RLCfff = Rotate Left through Carry f11C,N,Z64RLCf,WREGWREG = Rotate Left through Carry fs11C,N,Z64RLNCf,WREGWREG = Rotate Left (No Carry) fs11N,Z65RRCRRCff = Rotate Left (No Carry fs11N,Z65RRCRRCff = Rotate Left (hrough Carry fs11N,Z66RRCRRCff = Rotate Left (ho Carry) fs11N,Z67RRCRRCfRotate Left (No Carry) fs11N,Z68RRCRRCfRotate Left (No Carry) fs11N,Z69RRCRRCfRotate Left (No Carry) fs11N,Z60RRCRRCfRotate Left (No Carry) fs11N,Z61RRCRRCfRotate Left (No Carry) fs11N,Z62RRCRRCfRotate Left (No Carry) fs11C,N,Z63RRCRRCfRotate Right through Carry fs <td></td> <td></td> <td>REPEAT</td> <td>Wn</td> <td>Repeat Next Instruction (Wn) + 1 times</td> <td>1</td> <td>1</td> <td>None</td>			REPEAT	Wn	Repeat Next Instruction (Wn) + 1 times	1	1	None
60RETFIERETFIEReturn from interrupt13 (2)None61RETLWRETLW#1it10,WnReturn with literal in Wn13 (2)None62RETURNRETURNReturn from Subroutine13 (2)None63RLCfff = Rotate Left through Carry f11C,N,Z64RLNCf,WREGWREG = Rotate Left through Carry f11N,Z64RLNCf,WREGWREG = Rotate Left (No Carry) f11N,Z65RRCRRCff = Rotate Left (No Carry f)11N,Z66RRCRRCff = Rotate Left (No Carry) f11N,Z67RRCRRCMREGWREG = Rotate Left (No Carry) f11N,Z68RRCRRCf,WREGWREG = Rotate Left (No Carry) f11N,Z69RRCRRCfRotate Left (No Carry) f11N,Z60RRCRRCfRotate Right through Carry f11C,N,Z61RRCF,WREGWREG = Rotate Right through Carry f11C,N,Z62RRCRRCfReturn for through Carry f11C,N,Z63RRCRRCRRCfRotate Right through Carry f11C,N,Z64RCRRCRRCRRCRRCRRCRRCRRCRRCRRC65RRC	59	RESET	RESET		Software device Reset	1	1	None
61RETLWRETLW#lit10, WnReturn with literal in Wn13 (2)None62RETURNRETURNReturn from Subroutine13 (2)None63RLCfffReturn from Subroutine11C,N,Z63RLCfffReturn from Subroutine11C,N,Z64RLCf, WREGWREGRotate Left through Carry f11C,N,Z64RLNCf, WREGWREGRotate Left (No Carry) f11N,Z64RLNCf, WREGWREG = Rotate Left (No Carry) f11N,Z65RRCRRCff = Rotate Left (No Carry f11N,Z65RRCRRCf, WREGWREG = Rotate Right through Carry f11C,N,Z7RRCf, WREGWREG = Rotate Right through Carry f11C,N,Z7RRCf, WREGWREG = Rotate Right through Carry f11C,N,Z7RRCf, WREGWREG = Rotate Right through Carry f11C,N,Z	60	RETFIE	RETFIE		Return from interrupt	1	3 (2)	None
62 RETURN RETURN Return from Subroutine 1 3 (2) None 63 RLC f f f = Rotate Left through Carry f 1 1 C,N,Z 64 RLC f, WREG WREG = Rotate Left through Carry Ws 1 1 C,N,Z 64 RLNC f, WREG WREG = Rotate Left through Carry Ms 1 1 N,Z 64 RLNC f, WREG WREG = Rotate Left (No Carry) f 1 1 N,Z 64 RLNC f, WREG WREG = Rotate Left (No Carry) f 1 1 N,Z 65 RRC RRC f Rotate Left (No Carry) Ms 1 1 C,N,Z 65 RRC RRC f Rotate Left (No Carry) Ms 1 1 N,Z 65 RRC RRC f Rotate Right through Carry f 1 1 C,N,Z REC f, WREG WREG = Rotate Right through Carry f 1 1 C,N,Z	61	RETLW	RETLW	#lit10,Wn	Return with literal in Wn	1	3 (2)	None
63RLCffRotate Left through Carry f11C,N,ZRLCf, WREGWREG = Rotate Left through Carry f11C,N,ZRLCWs, WdWd = Rotate Left through Carry Ws11C,N,Z64RLNCfff = Rotate Left (No Carry) f11N,ZRLNCf, WREGWREG = Rotate Left (No Carry) f11N,ZRLNCf, WREGWREG = Rotate Left (No Carry) f11N,Z65RRCRRCff = Rotate Right through Carry f11C,N,ZRRCf, WREGWREG = Rotate Right through Carry f11C,N,Z	62	RETURN	RETURN		Return from Subroutine	1	3 (2)	None
RLC f, WREG WREG = Rotate Left through Carry f 1 1 C,N,Z RLC Ws, Wd Wd = Rotate Left through Carry Ws 1 1 C,N,Z 64 RLNC f f R Rotate Left (No Carry) f 1 1 N,Z 64 RLNC f, WREG WREG = Rotate Left (No Carry) f 1 1 N,Z 64 RLNC f, WREG WREG = Rotate Left (No Carry) f 1 1 N,Z 65 RRC RRC f Restant f = Rotate Left (No Carry) Ws 1 1 N,Z 65 RRC RRC f Restant Restant f = Rotate Right through Carry f 1 1 C,N,Z 65 RRC RRC f, WREG WREG = Rotate Right through Carry f 1 1 C,N,Z 8 RRC f, WREG WREG = Rotate Right through Carry f 1 1 C,N,Z 9 RRC f, WREG Wd = Dotate Dight through Carry f 1 1 C,N,Z	63	RLC	RLC	f	f = Rotate Left through Carry f	1	1	C,N,Z
RLC Ws, Wd Wd = Rotate Left through Carry Ws 1 1 C,N,Z 64 RLNC f f Rotate Left (No Carry) f 1 1 N,Z 64 RLNC f mean f = Rotate Left (No Carry) f 1 1 N,Z 64 RLNC f, WREG WREG = Rotate Left (No Carry) f 1 1 N,Z 65 RRC RRC f f = Rotate Right through Carry f 1 1 N,Z 65 RRC RRC f, WREG WREG = Rotate Right through Carry f 1 1 C,N,Z 65 RRC RRC f, WREG Wd = Rotate Right through Carry f 1 1 C,N,Z 65 RRC RRC f, WREG Wd = Rotate Right through Carry f 1 1 C,N,Z 66 RRC RRC f, WREG Wd = Rotate Right through Carry f 1 1 C,N,Z			RLC	1,WREG	WREG = Rotate Left through Carry f	1	1	C,N,Z
04 RLNC I I I N,Z RLNC f, WREG WREG = Rotate Left (No Carry) f 1 1 N,Z RLNC f, WREG WREG = Rotate Left (No Carry) f 1 1 N,Z 65 RRC RRC f Ferror Ferror 1 1 N,Z 65 RRC RRC f, WREG WREG = Rotate Right through Carry f 1 1 C,N,Z RRC f, WREG WREG = Rotate Right through Carry f 1 1 C,N,Z RRC f, WREG Wd = Pototo Pictor Pictor Pictor Carry Wo 1 1 C,N,Z	64	D7.257	RLC Ws,Wd		wu = Rotate Left through Carry Ws	1	1	U,N,Z
RENC F, WREG WREG Rotate Left (No Carry) T T T T N,Z 65 RRC RRC f Rec f Rec f N,Z RRC f, WREG Wd = Rotate Left (No Carry) Ws 1 1 N,Z 65 RRC RRC f Rec f Rotate Right through Carry f 1 1 C,N,Z RRC f, WREG WREG = Rotate Right through Carry f 1 1 C,N,Z	04	RLNC f RLNC f, WREG		I C UDDO		1	1	N,Z
REC REC f f N,Z 65 RRC f f Rec f NREG f RRC f, WREG WREG = Rotate Right through Carry f 1 1 C,N,Z RRC f, WREG WREG = Rotate Right through Carry f 1 1 C,N,Z				I,WREG	WREG = Rotate Left (No Carry) T	1	1	N,∠
$\frac{1}{1} = RC + RC$	6F	DDC	RLNC	ws,Wa	f = Pototo Pight through Correct	1	1	
RKC L, WKEG WKEG = Kolale Kight through Carry Ma I I C,N,Z PPC Ma Wd = Datata Diabt through Carry Ma I I C,N,Z	00	KKC	RRC	L f WDEC	I - Rotate Right through Carry f	1	1	
			AKC	L, WKEG	Wite Brotate Right through Carry We	1	1	

TABLE 23-2: INSTRUCTION SET OVERVIEW (CONTINUED)

TABLE 25-33:SPIX SLAVE MODE (FULL-DUPLEX, CKE = 1, CKP = 1, SMP = 0) TIMING
REQUIREMENTS

AC CHA	ARACTERIS	TICS	$\begin{tabular}{lllllllllllllllllllllllllllllllllll$					
Param No.	Symbol	Characteristic ⁽¹⁾	Min	Тур ⁽²⁾	Max	Units	Conditions	
SP70	TscP	Maximum SCK Input Frequency	_		11	MHz	See Note 3	
SP72	TscF	SCKx Input Fall Time	—	_	_	ns	See parameter DO32 and Note 4	
SP73	TscR	SCKx Input Rise Time	_		_	ns	See parameter DO31 and Note 4	
SP30	TdoF	SDOx Data Output Fall Time	—	_	_	ns	See parameter DO32 and Note 4	
SP31	TdoR	SDOx Data Output Rise Time	_	_	—	ns	See parameter DO31 and Note 4	
SP35	TscH2doV, TscL2doV	SDOx Data Output Valid after SCKx Edge	—	6	20	ns	—	
SP36	TdoV2scH, TdoV2scL	SDOx Data Output Setup to First SCKx Edge	30	—	_	ns	—	
SP40	TdiV2scH, TdiV2scL	Setup Time of SDIx Data Input to SCKx Edge	30	_	—	ns	_	
SP41	TscH2diL, TscL2diL	Hold Time of SDIx Data Input to SCKx Edge	30	_	_	ns	—	
SP50	TssL2scH, TssL2scL	$\overline{SSx} \downarrow$ to SCKx \uparrow or SCKx Input	120	Ι	—	ns	_	
SP51	TssH2doZ	SSx ↑ to SDOx Output High-Impedance ⁽⁴⁾	10	_	50	ns	_	
SP52	TscH2ssH TscL2ssH	SSx after SCKx Edge	1.5 TCY + 40	_		ns	See Note 4	
SP60	TssL2doV	SDOx Data Output Valid after SSx Edge	_	_	50	ns	_	

Note 1: These parameters are characterized, but are not tested in manufacturing.

2: Data in "Typ" column is at 3.3V, 25°C unless otherwise stated.

3: The minimum clock period for SCKx is 91 ns. Therefore, the SCK clock generated by the Master must not violate this specificiation.

4: Assumes 50 pF load on all SPIx pins.

DC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V(unless otherwise stated)Operating temperature $-40^{\circ}C \le TA \le +150^{\circ}C$ for High Temperature					
Parameter No.	Typical	Мах	Units	Conditions				
Power-Down Current (IPD)								
HDC61c	3	5	μA	+150°C	3.3V	Watchdog Timer Current: ΔIWDT ^(2,4)		

Note 1: Base IPD is measured with all peripherals and clocks shut down. All I/Os are configured as inputs and pulled to Vss. WDT, etc., are all switched off, and VREGS (RCON<8>) = 1.

- 2: The ∆ current is the additional current consumed when the module is enabled. This current should be added to the base IPD current.
- 3: These currents are measured on the device containing the most memory in this family.
- 4: These parameters are characterized, but are not tested in manufacturing.

TABLE 26-5: DC CHARACTERISTICS: DOZE CURRENT (IDOZE)

DC CHARA	Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}C \le TA \le +150^{\circ}C$ for High Temperature							
Parameter No.	Typical ⁽¹⁾	Max	Doze Ratio	Units	Conditions			
HDC72a	39	45	1:2	mA				
HDC72f	18	25	1:64	mA	+150°C	3.3V	20 MIPS	
HDC72g	18	25	1:128	mA				

Note 1: Parameters with Doze ratios of 1:2 and 1:64 are characterized, but are not tested in manufacturing.

TABLE 26-14: ADC MODULE SPECIFICATIONS

AC CHARACTERISTICS		Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)Operating temperature $-40^{\circ}C \le TA \le +150^{\circ}C$ for High Temperature							
Param No. Symbol		Characteristic	Min	Тур	Max	Units	Conditions		
Reference Inputs									
HAD08	IREF	Current Drain		250 —	600 50	μA μA	ADC operating, See Note 1 ADC off, See Note 1		

Note 1: These parameters are not characterized or tested in manufacturing.

2: These parameters are characterized, but are not tested in manufacturing.

TABLE 26-15: ADC MODULE SPECIFICATIONS (12-BIT MODE)⁽³⁾

AC CHARACTERISTICS		Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)Operating temperature $-40^{\circ}C \le TA \le +150^{\circ}C$ for High Temperature							
Param No.	Symbol	Characteristic	Min	Тур	Max	Units	Conditions		
ADC Accuracy (12-bit Mode) – Measurements with external VREF+/VREF- ⁽¹⁾									
AD23a	Gerr	Gain Error		5	10	LSb	Vinl = AVss = Vrefl = 0V, AVdd = Vrefh = 3.6V		
AD24a	EOFF	Offset Error		2	5	LSb	Vinl = AVss = Vrefl = 0V, AVdd = Vrefh = 3.6V		
ADC Accuracy (12-bit Mode) – Measurements with internal VREF+/VREF- ⁽¹⁾									
AD23a	Gerr	Gain Error	2	10	20	LSb	VINL = AVSS = 0V, AVDD = 3.6V		
AD24a	Eoff	Offset Error	2	5	10	LSb	VINL = AVSS = 0V, AVDD = 3.6V		
Dynamic Performance (12-bit Mode) ⁽²⁾									
HAD33a FNYQ Input Signal Bandwidth			_		200	kHz			

Note 1: These parameters are characterized, but are tested at 20 ksps only.

2: These parameters are characterized by similarity, but are not tested in manufacturing.

3: Injection currents > | 0 | can affect the ADC results by approximately 4-6 counts.

TABLE 26-16: ADC MODULE SPECIFICATIONS (10-BIT MODE)⁽³⁾

AC CHARACTERISTICS		Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)Operating temperature $-40^{\circ}C \le TA \le +150^{\circ}C$ for High Temperature							
Param No.	Symbol	Characteristic	Min	Тур	Max	Units	Conditions		
ADC Accuracy (12-bit Mode) – Measurements with external VREF+/VREF- ⁽¹⁾									
AD23b	Gerr	Gain Error	—	3	6	LSb	Vinl = AVss = Vrefl = 0V, AVdd = Vrefh = 3.6V		
AD24b	EOFF	Offset Error	—	2	5	LSb	Vinl = AVss = Vrefl = 0V, AVdd = Vrefh = 3.6V		
ADC Accuracy (12-bit Mode) – Measurements with internal VREF+/VREF- ⁽¹⁾									
AD23b	Gerr	Gain Error	—	7	15	LSb	VINL = AVSS = 0V, AVDD = 3.6V		
AD24b	EOFF	Offset Error		3	7	LSb	VINL = AVSS = 0V, AVDD = 3.6V		
Dynamic Performance (10-bit Mode) ⁽²⁾									
HAD33b	Fnyq	Input Signal Bandwidth		_	400	kHz			

Note 1: These parameters are characterized, but are tested at 20 ksps only.

2: These parameters are characterized by similarity, but are not tested in manufacturing.

3: Injection currents > | 0 | can affect the ADC results by approximately 4-6 counts.

APPENDIX A: MIGRATING FROM dsPIC33FJXXXGPX06/X08/X10 DEVICES TO dsPIC33FJXXXGPX06A/X08A/X10A DEVICES

dsPIC33FJXXXGPX06A/X08A/X10A devices were designed to enhance the dsPIC33FJXXXGPX06/X08/ X10 families of devices.

In general, the dsPIC33FJXXXGPX06A/X08A/X10A devices backward-compatible are with dsPIC33FJXXXGPX06/X08/X10 devices; however, manufacturing differences may cause dsPIC33FJXXXGPX06A/X08A/X10A devices to behave differently from dsPIC33FJXXXGPX06/X08/ X10 devices. Therefore, complete system test and recommended characterization is if dsPIC33FJXXXGPX06A/X08A/X10A devices are used to replace dsPIC33FJXXXGPX06/X08/X10 devices.

The following enhancements were introduced:

- Extended temperature support of up to +125°C
- Enhanced Flash module with higher endurance and retention
- New PLL Lock Enable configuration bit
- Added Timer5 trigger for ADC1 and Timer3 trigger for ADC2

NOTES: