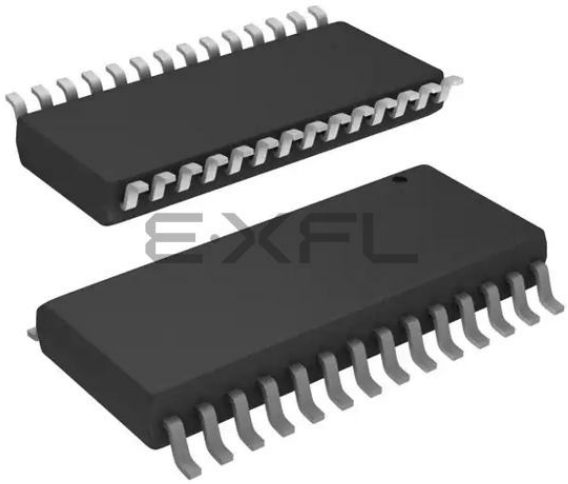


Welcome to E-XFL.COM

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 "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	64MHz
Connectivity	CANbus, I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	25
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 24x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SOIC (0.295", 7.50mm Width)
Supplier Device Package	28-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f25k83-e-so

FIGURE 4-4: DATA MEMORY MAP FOR PIC18(L)F25/26K83 DEVICES

Bank	BSR<5:0>	addr<7:0>	PIC18(L)F25K83	PIC18(L)F26K83	Address addr<13:0>
Bank 0	00 0000	00h FFh	Access RAM	Access RAM	0000h 005Fh
			GPR	GPR	0060h 00FFh
Bank 1	00 0001	00h FFh	GPR	GPR	0100h • • • 03FFh
Bank 2	00 0010	00h FFh			
Bank 3	00 0011	00h FFh			
Banks 4 to 7	00 0100 — 00 0111	00h • • FFh	GPR	GPR	0400h • • 07FFh
Banks 8 to 15	00 1000 — 00 1111	00h • • FFh	Unimplemented	GPR	0800h • • 0FFFh
Banks 16 to 31	01 0000 — 01 1111	00h • • FFh		Unimplemented	1000h • • 1FFFh
Banks 32 to 53	10 0000 — 11 0111	00h • • • FFh			2000 • • • 37FF
Bank 54	d54	00h • FFh	CAN Test	CAN Test	3600h • 36FFh
Bank 55	d55	00h • FFh	CAN SFR	CAN SFR	3700h • 37FFh
Banks 56 to 62	11 1000 — 11 1110	00h • • FFh	SFR	SFR	3800h • • 3EFFh
Bank 63	11 1111	00h FFh	SFR	SFR	3F00h 3F5Fh 3F60h 3FFFh

Note 1: Depends on the number of SFRs. Refer subsequent SFR tables.

4.5.5 STATUS REGISTER

The STATUS register, shown in Register 4-2, contains the arithmetic status of the ALU. As with any other SFR, it can be the operand for any instruction.

If the STATUS register is the destination for an instruction that affects the Z, DC, C, OV or N bits, the results of the instruction are not written; instead, the STATUS register is updated according to the instruction performed. Therefore, the result of an instruction with the STATUS register as its destination may be different than intended. As an example, `CLRF STATUS` will set the Z bit and leave the remaining Status bits unchanged (‘`0uuu u1uu`’).

It is recommended that only `BCF`, `BSF`, `SWAPF`, `MOVFF`, `MOVWF` and `MOVFFL` instructions are used to alter the STATUS register, because these instructions do not affect the Z, C, DC, OV or N bits in the STATUS register.

For other instructions that do not affect Status bits, see the instruction set summaries in **Section 42.2 “Extended Instruction Set”** and Table 42-3.

Note:	The C and DC bits operate as the borrow and digit borrow bits, respectively, in subtraction.

4.5.6 CALL SHADOW REGISTER

When `CALL`, `CALLW`, `RCALL` instructions are used, the WREG, BSR and STATUS are automatically saved in hardware and can be accessed using the WREG_CSHAD, BSR_CSHAD and STATUS_CSHAD registers.

4.8.3 MAPPING THE ACCESS BANK IN INDEXED LITERAL OFFSET MODE

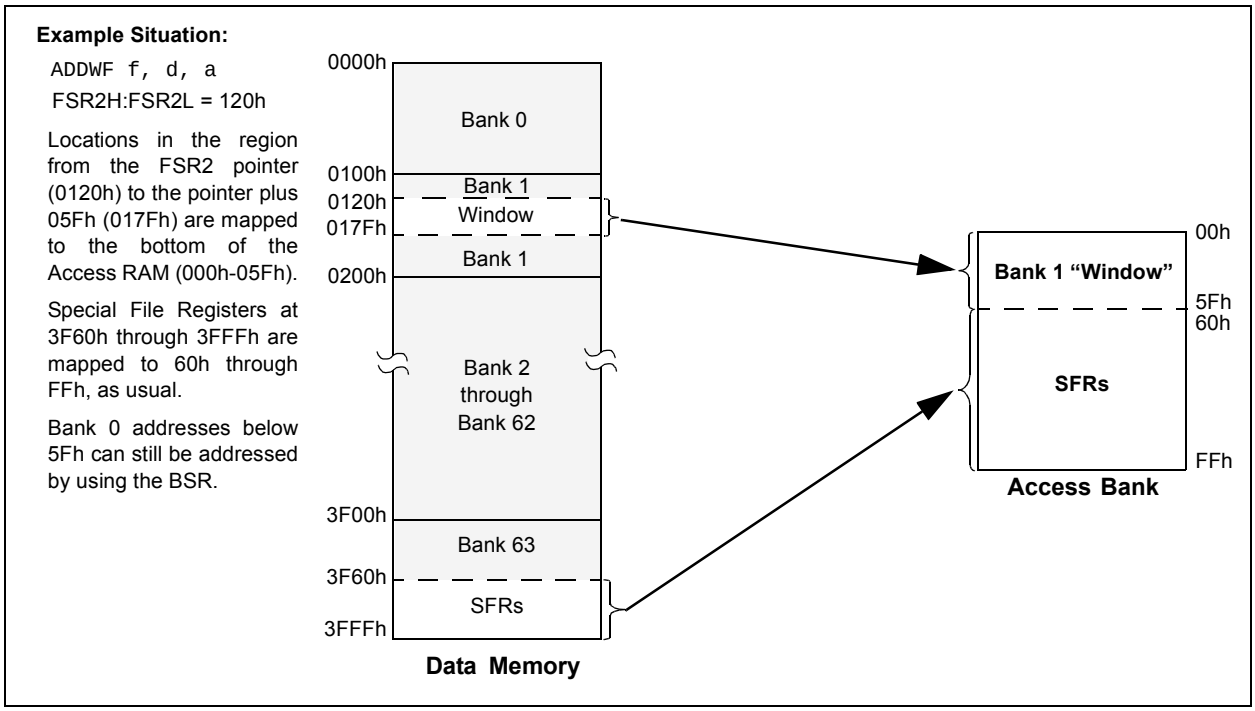
The use of Indexed Literal Offset Addressing mode effectively changes how the first 96 locations of Access RAM (00h to 5Fh) are mapped. Rather than containing just the contents of the bottom section of Bank 0, this mode maps the contents from a user defined “window” that can be located anywhere in the data memory space. The value of FSR2 establishes the lower boundary of the addresses mapped into the window, while the upper boundary is defined by FSR2 plus 95 (5Fh). Addresses in the Access RAM above 5Fh are mapped as previously described (see **Section 4.5.4 “Access Bank”**). An example of Access Bank remapping in this addressing mode is shown in Figure 4-8.

Remapping of the Access Bank applies *only* to operations using the Indexed Literal Offset mode. Operations that use the BSR (Access RAM bit is ‘1’) will continue to use direct addressing as before.

4.9 PIC18 Instruction Execution and the Extended Instruction Set

Enabling the extended instruction set adds eight additional commands to the existing PIC18 instruction set. These instructions are executed as described in **Section 42.2 “Extended Instruction Set”**.

FIGURE 4-8: REMAPPING THE ACCESS BANK WITH INDEXED LITERAL OFFSET ADDRESSING



6.4 Low-Power Brown-out Reset (LPBOR)

The Low-Power Brown-out Reset (LPBOR) provides an additional BOR circuit for low power operation. Refer to Figure 6-2 to see how the BOR interacts with other modules.

The LPBOR is used to monitor the external VDD pin. When too low of a voltage is detected, the device is held in Reset.

6.4.1 ENABLING LPBOR

The LPBOR is controlled by the LPBOREN bit of Configuration Word 2L. When the device is erased, the LPBOR module defaults to disabled.

6.4.1.1 LPBOR Module Output

The output of the LPBOR module is a signal indicating whether or not a Reset is to be asserted. This signal is OR'd together with the Reset signal of the BOR module to provide the generic BOR signal, which goes to the PCON0 register and to the power control block.

6.5 MCLR

The MCLR is an optional external input that can reset the device. The MCLR function is controlled by the MCLRE bit of Configuration Words and the LVP bit of Configuration Words (Table 6-2). The RMCLR bit in the PCON0 register will be set to '0' if a MCLR Reset has occurred.

TABLE 6-2: MCLR CONFIGURATION

MCLRE	LVP	MCLR
x	1	Enabled
1	0	Enabled
0	0	Disabled

6.5.1 MCLR ENABLED

When MCLR is enabled and the pin is held low, the device is held in Reset. The MCLR pin is connected to VDD through an internal weak pull-up.

The device has a noise filter in the MCLR Reset path. The filter will detect and ignore small pulses.

Note:	An internal Reset event (RESET instruction, BOR, WWDT, POR stack), does not drive the MCLR pin low.

6.5.2 MCLR DISABLED

When MCLR is disabled, the MCLR pin becomes input-only and pin functions such as internal weak pull-ups are under software control. See **Section 16.2 "I/O Priorities"** for more information.

6.6 Windowed Watchdog Timer (WWDT) Reset

The Windowed Watchdog Timer generates a Reset if the firmware does not issue a CLRWDT instruction within the time-out period or window set. The TO and PD bits in the STATUS register and the RWDT bit in the PCON0 register are changed to indicate a WWDT Reset. The WDTWV bit in the PCON0 register indicates if the WDT Reset has occurred due to a time out or a window violation. See **Section 11.0 "Windowed Watchdog Timer (WWDT)"** for more information.

6.7 RESET Instruction

A RESET instruction will cause a device Reset. The RI bit in the PCON0 register will be set to '0'. See Table 6-3 for default conditions after a RESET instruction has occurred.

6.8 Stack Overflow/Underflow Reset

The device can reset when the Stack Overflows or Underflows. The STKOVF or STKUNF bits of the PCON0 register indicate the Reset condition. These Resets are enabled by setting the STVREN bit in Configuration Words. See **Section 4.2.5 "Return Address Stack"** for more information.

6.9 Programming Mode Exit

Upon exit of Programming mode, the device will behave as if a POR has just occurred.

6.10 Power-up Timer (PWRT)

The Power-up Timer provides a selected time-out duration on POR or Brown-out Reset.

The device is held in Reset as long as PWRT is active. The PWRT delay allows additional time for the VDD to rise to an acceptable level. The Power-up Timer is selected by setting the PWRTS<1:0> Configuration bits, appropriately.

The Power-up Timer starts after the release of the POR and BOR/LPBOR if enabled, as shown in Figure 6-1.

8.1 Clock Source

The input to the reference clock output can be selected using the CLKRCLK register.

8.1.1 CLOCK SYNCHRONIZATION

Once the reference clock enable (EN) is set, the module is ensured to be glitch-free at start-up.

When the reference clock output is disabled, the output signal will be disabled immediately.

Clock dividers and clock duty cycles can be changed while the module is enabled, but glitches may occur on the output. To avoid possible glitches, clock dividers and clock duty cycles should be changed only when the CLKREN is clear.

8.2 Programmable Clock Divider

The module takes the clock input and divides it based on the value of the DIV<2:0> bits of the CLKRCON register (Register 8-1).

The following configurations can be made based on the DIV<2:0> bits:

- Base FOSC value
- FOSC divided by 2
- FOSC divided by 4
- FOSC divided by 8
- FOSC divided by 16
- FOSC divided by 32
- FOSC divided by 64
- FOSC divided by 128

The clock divider values can be changed while the module is enabled; however, in order to prevent glitches on the output, the DIV<2:0> bits should only be changed when the module is disabled (EN = 0).

8.3 Selectable Duty Cycle

The DC<1:0> bits of the CLKRCON register can be used to modify the duty cycle of the output clock. A duty cycle of 25%, 50%, or 75% can be selected for all clock rates, with the exception of the undivided base FOSC value.

The duty cycle can be changed while the module is enabled; however, in order to prevent glitches on the output, the DC<1:0> bits should only be changed when the module is disabled (EN = 0).

Note:	The DC1 bit is reset to '1'. This makes the default duty cycle 50% and not 0%.

8.4 Operation in Sleep Mode

The reference clock output module clock is based on the system clock. When the device goes to Sleep, the module outputs will remain in their current state. This will have a direct effect on peripherals using the reference clock output as an input signal. No change should occur in the module from entering or exiting from Sleep.

9.3.2 NATURAL ORDER (HARDWARE) PRIORITY

When more than one interrupt with the same user specified priority level are requested, the priority conflict is resolved by using a method called “Natural Order Priority”. Natural order priority is a fixed priority scheme that is based on the Interrupt Vector Table. Table 9-2 shows the natural order priority and the interrupt vector number assigned for each source.

TABLE 9-2: INTERRUPT VECTOR PRIORITY TABLE

Vector Number	Interrupt Source
0	Software Interrupt
1	HLVD
2	OSF
3	CSW
4	NVM
5	SCAN
6	CRC
7	IOC
8	INT0
9	ZCD
10	AD
11	ADT
12	C1
13	SMT1
14	SMT1PRA
15	SMT1PWA
16	DMA1SCNT
17	DMA1DCNT
18	DMA1OR
19	DMA1A
20	SPI1RX
21	SPI1TX
22	SPI1
23	I2C1RX
24	I2C1TX
25	I2C1
26	I2C1E
27	U1RX
28	U1TX
29	U1E
30	U1
31	TMR0
32	TMR1
33	TMR1G
34	TMR2
35	CCP1
36	NCO
37	CWG1
38	CLC1
39	INT1
40	RXB0IF/FIFOIF
41	RXB1IF/RXBnIF

Vector Number	Interrupt Source
42	TXB0IF
43	TXB1IF
44	TXB2IF/TXBnIF
45	ERRIF
46	WAKIF
47	IRXIF
48	C2
49	SMT2
50	SMT2PRA
51	SMT2PWA
52	DMA2SCNT
53	DMA2DCNT
54	DMA2OR
55	DMA2A
56	I2C2RX
57	I2C2TX
58	I2C2
59	I2C2E
60	U2RX
61	U2TX
62	U2E
63	U2
64	TMR3
65	TMR3G
66	TMR4
67	CCP2
68	CWG2
69	CLC2
70	INT2
71	TMR5
72	TMR5G
73	TMR6
74	CCP3
75	CWG3
76	CLC3
77	CCP4
78	CLC4
79	—
80	—
81	—

The natural order priority scheme has vector interrupt 0 as the highest priority and vector interrupt 81 as the lowest priority.

For example, when two concurrently occurring interrupt sources that are both designated high priority using the IPRx register will be resolved using the natural order priority (i.e., the interrupt with a lower corresponding vector number will preempt the interrupt with the higher vector number).

The ability for the user to assign every interrupt source to high or low priority levels means that the user program can give an interrupt with a low natural order priority a higher overall priority level.

9.4 Interrupt Operation

All pending interrupts are indicated by the flag bit being equal to a ‘1’ in the PIRx register. All pending interrupts are resolved using the priority scheme explained in **Section 9.3 “Interrupt Priority”**.

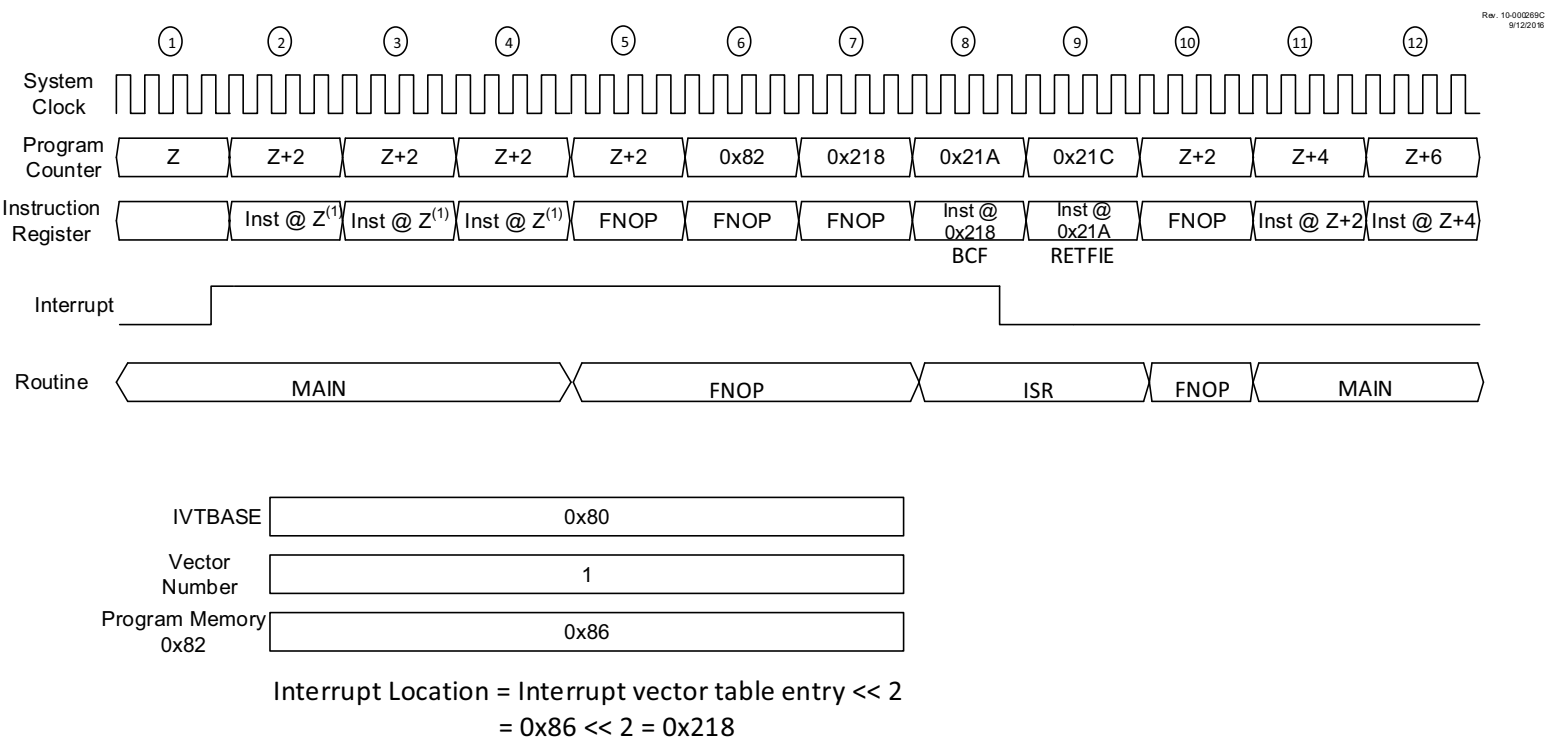
Once the interrupt source to be serviced is resolved, the program execution vectors to the resolved interrupt vector addresses, as explained in **Section 9.2 “Interrupt Vector Table (IVT)”**. The vector number is also stored in the WREG register. Most of the flag bits are required to be cleared by the application software, but in some cases, device hardware clears the interrupt automatically. Some flag bits are read-only in the PIRx registers, these flags are a summary of the source interrupts and the corresponding interrupt flags of the source must be cleared.

A valid interrupt can be either a high or low priority interrupt when in main routine or a high priority interrupt when in low priority Interrupt Service Routine. Depending on order of interrupt requests received and their relative timing, the CPU will be in the state of execution indicated by the STAT bits of the INTCON1 register (Register 9-2).

The State machine shown in Figure 9-1 and the subsequent sections detail the execution of interrupts when received in different orders.

Note:	The state of GIEH/L is not changed by the hardware when servicing an interrupt. The internal state machine is used to keep track of execution states. These bits can be manipulated in the user code resulting in transferring execution to the main routine and ignoring existing interrupts.

FIGURE 9-9: INTERRUPT TIMING DIAGRAM – THREE CYCLE INSTRUCTION



Note 1: Instruction @ Z is a three-cycle instruction.

REGISTER 9-19: PIE6: PERIPHERAL INTERRUPT ENABLE REGISTER 6

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
DMA2AIE	DMA2ORIE	DMA2DCNTIE	DMA2SCNTIE	SMT2PWAIE	SMT2PRAIE	SMT2IE	C2IE
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

bit 7 **DMA2AIE:** DMA Abort Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 6 **DMA2ORIE:** DMA2 Overrun Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 5 **DMA2DCNTIE:** DMA2 Destination Count Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 4 **DMA2SCNTIE:** DMA2 Source Count Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 3 **SMT2PWAIE:** SMT2 Pulse-Width Acquisition Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 2 **SMT2PRAIE:** SMT2 Period Acquisition Receive Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 1 **SMT2IE:** SMT2 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 0 **C2IE:** C2 Interrupt Enable bit
1 = Enabled
0 = Disabled

13.0 NONVOLATILE MEMORY (NVM) CONTROL

Nonvolatile Memory (NVM) is separated into two types: Program Flash Memory (PFM) and Data EEPROM Memory.

PFM, Data EEPROM, User IDs and Configuration bits can all be accessed using the REG<1:0> bits of the NVMCON1 register.

The write time is controlled by an on-chip timer. The write/erase voltages are generated by an on-chip charge pump rated to operate over the operating voltage range of the device.

NVM can be protected in two ways, by either code protection or write protection. Code protection ($\overline{CP}$ and $\overline{CPD}$ bits in Configuration Word 5L) disables access, reading and writing to both PFM and Data EEPROM Memory via external device programmers. Code protection does not affect the self-write and erase functionality. Code protection can only be reset by a device programmer performing a Bulk Erase to the device, clearing all nonvolatile memory, Configuration bits and User IDs.

Write protection prohibits self-write and erase to a portion or all of the PFM, as defined by the WRT bits of Configuration Word 4H. Write protection does not affect a device programmer's ability to read, write or erase the device.

TABLE 13-1: NVM ORGANIZATION AND ACCESS INFORMATION

Memory	PC<20:0> ICSP™ Addr<21:0> TBLPTR<21:0> NVMADDR<9:0>	Execution	User Access		
		CPU Execution	REG	TABLAT	NVMDAT
Program Flash Memory (PFM)	00 0000h ••• 01 FFFFh	Read	10	Read/ Write(1)	—(3)
User IDs(2)	20 0000h ••• 20 000Fh	No Access	x1	Read/ Write	—(3)
Reserved	20 0010h 2F FFFFh	No Access	—(3)		
Configuration	30 0000h ••• 30 0009h	No Access	x1	Read/ Write(1)	—(3)
Reserved	30 000Ah 30 FFFFh	No Access	—(3)		
User Data Memory (Data EEPROM)	31 0000h ••• 31 03FFh	No Access	00	—(3)	Read/ Write(1)
Reserved	31 0400h 3E FFFFh	No Access	—(3)		
Device Information Area (DIA)	3F 0000h ••• 3F 003Fh	No Access	x1	Read	—(3)
Reserved	3F 0040h 3F FF09h	No Access	—(3)		
Device Configuration Information (DCI)	3F FF00h ••• 3F FF09h	No Access	x1	Read	—(3)
Reserved	3F FF0Ah 3F FFFBh	No Access	—(3)		
Revision ID/ Device ID	3F FFFCh ••• 3F FFFFh	No Access	x1	Read	—(3)

Note 1: Subject to Memory Write Protection settings.
2: User IDs are eight words ONLY. There is no code protection, table read protection or write protection implemented for this region.
3: Reads as '0', writes clear the WR bit and WRERR bit is set.

25.6.4 HIGH AND LOW MEASURE MODE

This mode measures the high and low pulse time of the SMTSIGx relative to the SMT clock. It begins incrementing the SMTxTMR on a rising edge on the SMTSIGx input, then updates the SMTxCPW register with the value and resets the SMTxTMR on a falling edge, starting to increment again. Upon observing another rising edge, it updates the SMTxCPR register with its current value and once again resets the SMTxTMR value and begins incrementing again. See Figure 25-8 and Figure 25-9.

26.0 COMPLEMENTARY WAVEFORM GENERATOR (CWG) MODULE

The Complementary Waveform Generator (CWG) produces half-bridge, full-bridge, and steering of PWM waveforms. It is backwards compatible with previous CCP functions. The PIC18(L)F25/26K83 family has three instances of the CWG module.

Each of the CWG modules has the following features:

- Six operating modes:
  - Synchronous Steering mode
  - Asynchronous Steering mode
  - Full-Bridge mode, Forward
  - Full-Bridge mode, Reverse
  - Half-Bridge mode
  - Push-Pull mode
- Output polarity control
- Output steering
- Independent 6-bit rising and falling event dead-band timers
  - Clocked dead band
  - Independent rising and falling dead-band enables
- Auto-shutdown control with:
  - Selectable shutdown sources
  - Auto-restart option
  - Auto-shutdown pin override control

26.1 Fundamental Operation

The CWG generates two output waveforms from the selected input source.

The off-to-on transition of each output can be delayed from the on-to-off transition of the other output, thereby, creating a time delay immediately where neither output is driven. This is referred to as dead time and is covered in **Section 26.6 “Dead-Band Control”**.

It may be necessary to guard against the possibility of circuit faults or a feedback event arriving too late or not at all. In this case, the active drive must be terminated before the Fault condition causes damage. This is referred to as auto-shutdown and is covered in **Section 26.10 “Auto-Shutdown”**.

26.2 Operating Modes

The CWG module can operate in six different modes, as specified by the MODE<2:0> bits of the CWGxCON0 register:

- Half-Bridge mode
- Push-Pull mode
- Asynchronous Steering mode
- Synchronous Steering mode
- Full-Bridge mode, Forward
- Full-Bridge mode, Reverse

All modes accept a single pulse data input, and provide up to four outputs as described in the following sections.

All modes include auto-shutdown control as described in **Section 26.10 “Auto-Shutdown”**.

Note:	Except as noted for Full-bridge mode (**Section 26.2.3 “Full-Bridge Modes”**), mode changes should only be performed while EN = 0 (Register 26-1).

26.2.1 HALF-BRIDGE MODE

In Half-Bridge mode, two output signals are generated as true and inverted versions of the input as illustrated in Figure 26-2. A non-overlap (dead-band) time is inserted between the two outputs as described in **Section 26.6 “Dead-Band Control”**. The output steering feature cannot be used in this mode. A basic block diagram of this mode is shown in Figure 26-1.

The unused outputs CWGxC and CWGxD drive similar signals as CWGxA and CWGxB, with polarity independently controlled by the POLC and POLD bits of the CWGxCON1 register, respectively.

27.0 CONFIGURABLE LOGIC CELL (CLC)

The Configurable Logic Cell (CLCx) module provides programmable logic that operates outside the speed limitations of software execution. The logic cell takes up to 32 input signals and, through the use of configurable gates, reduces the 32 inputs to four logic lines that drive one of eight selectable single-output logic functions.

Input sources are a combination of the following:

- I/O pins
- Internal clocks
- Peripherals
- Register bits

The output can be directed internally to peripherals and to an output pin.

There are four CLC modules available on this device - CLC1, CLC2, CLC3 and CLC4.

Note:	The CLC1, CLC2, CLC3 and CLC4 are four separate module instances of the same CLC module design. Throughout this section, the lower case 'x' in register names is a generic reference to the CLC number (which should be substituted with 1, 2, 3, or 4 during code development). For example, the control register is generically described in this chapter as CLCxCON, but the actual device registers are CLC1CON, CLC2CON, CLC3CON and CLC4CON.

Refer to Figure 27-1 for a simplified diagram showing signal flow through the CLCx.

Possible configurations include:

- Combinatorial Logic
  - AND
  - NAND
  - AND-OR
  - AND-OR-INVERT
  - OR-XOR
  - OR-XNOR
- Latches
  - S-R
  - Clocked D with Set and Reset
  - Transparent D with Set and Reset

30.11 Register Definitions: Modulation Control

Long bit name prefixes for the Modulation peripheral is shown below. Refer to **Section 1.3.2.2 "Long Bit Names"** for more information.

Peripheral	Bit Name Prefix
MD1	MD1

REGISTER 30-1: MD1CON0: MODULATION CONTROL REGISTER 0

R/W-0/0	U-0	R-0/0	R/W-0/0	U-0	U-0	U-0	R/W-0/0
EN	—	OUT	OPOL	—	—	—	BIT
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

bit 7 **EN:** Modulator Module Enable bit
- 1 = Modulator module is enabled and mixing input signals
- 0 = Modulator module is disabled and has no output

bit 6 **Unimplemented:** Read as '0'

bit 5 **OUT:** Modulator Output bit
Displays the current output value of the Modulator module.(1)

bit 4 **OPOL:** Modulator Output Polarity Select bit
- 1 = Modulator output signal is inverted; idle high output
- 0 = Modulator output signal is not inverted; idle low output

bit 3-1 **Unimplemented:** Read as '0'

bit 0 **BIT:** Allows software to manually set modulation source input to module(2)
- 1 = Modulator selects Carrier High
- 0 = Modulator selects Carrier Low

Note 1: The modulated output frequency can be greater and asynchronous from the clock that updates this register bit, the bit value may not be valid for higher speed modulator or carrier signals.

2: BIT bit must be selected as the modulation source in the MD1SRC register for this operation.

bit 3 **MDR:** Master Data Request *(Master pause)*
1 = Master state mechine pauses until data is read/written to proceed (SCL is output held low)
0 = Master clocking of data is enabled.

MMA = 1 & RXBF = 1
pause_for_rx - Set by hardware on 7th falling SCL edge
- User must read from I2CRXB to release SCL
MMA = 1 & TXBE = 1 & I2CCNT!= 0
pause_for_tx - Set by hardware on 8th falling SCL edge
- User must write to I2CTXB to release SCL
ADB = 1
- I2CCNT is ignored for the high and low address in 10-bit mode
pause_for_restart - Set by hardware on 9th falling SCL edge
RSEN = 1 & MMA = 1 && I2CCNT = 0 || ACKSTAT = 1
- User must set START or write to I2CTXB to release SCL and shift Restart onto bus

bit 2-0 **MODE<2:0>:** I^2C Mode Select bits
111 = I^2C Muti-Master mode (SMBus 2.0 Host), **(5)**
Works as both mode<2:0> = 001 and mode<2:0> = 100
110 = I^2C Muti-Master mode (SMBus 2.0 Host), **(5)**
Works as both mode<2:0> = 000 and mode<2:0> = 100
101 = I^2C Master mode, 10-bit address
100 = I^2C Master mode, 7-bit address
011 = I^2C Slave mode, one 10-bit address with masking
010 = I^2C Slave mode, two 10-bit address
001 = I^2C Slave mode, two 7-bit address with masking
000 = I^2C Slave mode, four 7-bit address

Note 1: SDA and SCL pins must be configured for open-drain with internal or external pull-up
2: SDA and SCL pins must be selected as both input and output in PPS
3: CSTR can be set by more than one hardware source, all sources must be addressed by user software before the SCL line is released. CSTR is a module Status bit, and does not show the true bus state.
4: SMA is set on the same SCL edge as CSTR for a matching received address
5: In this mode, ADRIE should be set, this allows an interrupt to clear the BCLIF condition and allow the ACK of matching address.
6: In 10-bit Slave mode, when ADB = 1, CSTR will set when the high address has not been read out of I2CxRXB before the low address is shifted in.

REGISTER 33-7: I2CxSTAT1: I²C STATUS REGISTER 1

R/W/HS-0	U-0	R-1	U-0	R/W/HS-0	R/S-0/0	U-0	R-0
TXWE(2)	—	TXBE(1, 3)	—	RXRE(2)	CLRBF	—	RXBF(1,3)
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	HS = Hardware set HC = Hardware clear

bit 7 **TXWE:** Transmit Write Error Status bit (2)
1 = A new byte of data was written to I2CTXB when it was full (Must be cleared by software)
0 = No transmit write error

bit 6 **Unimplemented:** Read as '0'

bit 5 **TXBE:** Transmit Buffer Empty Status bit
1 = I2CTXB is empty (Cleared by writing the I2CTXB register)
0 = I2CTXB is full

bit 4 **Unimplemented:** Read as '0'

bit 3 **RXRE:** Receive Read Error Status bit
1 = A byte of data was read from I2CxRXB when it was empty. (Must be cleared by software)
0 = No receive overflow

bit 2 **CLRBF:** Clear Buffer bit
Setting this bit clears/empties the receive and transmit buffers, causing reset of RXBF and TXBE.
Setting this bit clears the RXIF and TXIF interrupt flags.
This bit is set-only special function, and always reads '0'

bit 1 **Unimplemented:** Read as '0'

bit 0 **RXBF:** Receive Buffer Full Status bit
1 = I2CRXB has received new data (Cleared by reading the I2CRXB register)
0 = I2CRXB is empty

Note 1: The bits are held in Reset when I2CEN = 0.
2: Will cause NACK to be sent for slave address and master/slave data read bytes.
3: Used as triggers for DMA operation.

REGISTER 34-44: RXMnEIDL: RECEIVE ACCEPTANCE MASK 'n' EXTENDED IDENTIFIER MASK REGISTERS, LOW BYTE [0 ≤ n ≤ 1]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
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 7-0 **EID<7:0>:** Extended Identifier Mask bits

REGISTER 34-45: RXFCONn: RECEIVE FILTER CONTROL REGISTER 'n' [0 ≤ n ≤ 1](1)

RXFCON0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	RXF7EN	RXF6EN	RXF5EN	RXF4EN	RXF3EN	RXF2EN	RXF1EN	RXF0EN

RXFCON1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	RXF15EN	RXF14EN	RXF13EN	RXF12EN	RXF11EN	RXF10EN	RXF9EN	RXF8EN
	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 7-0 **RXF<7:0>EN:** Receive Filter n Enable bits
0 = Filter is disabled
1 = Filter is enabled

Note 1: This register is available in Mode 1 and 2 only.

Note: Register 34-46 through Register 34-51 are writable in Configuration mode only.

37.1.5 INTERRUPTS

The ADC module allows for the ability to generate an interrupt upon completion of an Analog-to-Digital conversion. The ADC Interrupt Flag is the ADIF bit in the PIR1 register. The ADC Interrupt Enable is the ADIE bit in the PIE1 register. The ADIF bit must be cleared in software.

Note 1:	The ADIF bit is set at the completion of every conversion, regardless of whether or not the ADC interrupt is enabled.
2:	The ADC operates during Sleep only when the FRC oscillator is selected.

This interrupt can be generated while the device is operating or while in Sleep. If the device is in Sleep, the interrupt will wake up the device. Upon waking from Sleep, the next instruction following the SLEEP instruction is always executed. If the user is attempting to wake up from Sleep and resume in-line code execution, the ADIE bit of the PIEx register and the GIE bits of the INTCON0 register must both be set. If all these bits are set, the execution will switch to the Interrupt Service Routine.

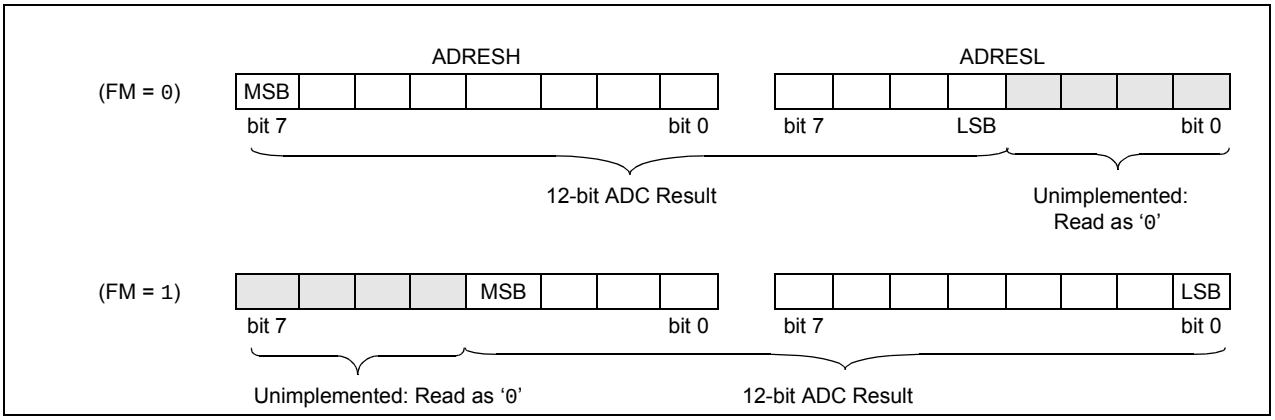
37.1.6 RESULT FORMATTING

The 12-bit ADC conversion result can be supplied in two formats, left justified or right justified. The FM bits of the ADCON0 register controls the output format.

Figure 37-3 shows the two output formats.

Writes to the ADRES register pair are always right justified regardless of the selected format mode. Therefore, data read after writing to ADRES when ADFRM0 = 0 will be shifted left four places.

FIGURE 37-3: 12-BIT ADC CONVERSION RESULT FORMAT



IORWF **Inclusive OR W with f**

Syntax: IORWF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (W) .OR. (f) → dest

Status Affected: N, Z

Encoding:

0001	00da	ffff	ffff

Description: Inclusive OR W with register 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default).
If 'a' is '0', the Access Bank is selected.
If 'a' is '1', the BSR is used to select the GPR bank.
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 42.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: IORWF RESULT, 0, 1

Before Instruction
RESULT = 13h
W = 91h
After Instruction
RESULT = 13h
W = 93h

LFSR **Load FSR**

Syntax: LFSR f, k

Operands: $0 \le f \le 2$
$0 \le k \le 16383$

Operation: k → FSRf

Status Affected: None

Encoding:

1110 1111	1110 0000	$00k_{13}k$ k_7kkk	kkkk kkkk

Description: The 14-bit literal 'k' is loaded into the File Select Register pointed to by 'f'.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k' MSB	Process Data	Write literal 'k' MSB to FSRfH
Decode	Read literal 'k' LSB	Process Data	Write literal 'k' to FSRfL

Example: LFSR 2, 3ABh

After Instruction
FSR2H = 03h
FSR2L = ABh

RLNCF **Rotate Left f (No Carry)**

Syntax: RLNCF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

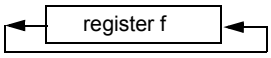
Operation: (f<n>) → dest<n + 1>,
(f<7>) → dest<0>

Status Affected: N, Z

Encoding:

0100	01da	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the left. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f' (default).
If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 42.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: RLNCF REG, 1, 0

Before Instruction
REG = 1010 1011
After Instruction
REG = 0101 0111

RRCF **Rotate Right f through Carry**

Syntax: RRCF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

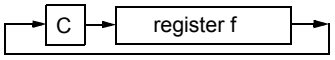
Operation: (f<n>) → dest<n – 1>,
(f<0>) → C,
(C) → dest<7>

Status Affected: C, N, Z

Encoding:

0011	00da	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the right through the CARRY flag. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default).
If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 42.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: RRCF REG, 0, 0

Before Instruction
REG = 1110 0110
C = 0
After Instruction
REG = 1110 0110
W = 0111 0011
C = 0

TABLE 42-3: EXTENSIONS TO THE PIC18 INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word				Status Affected
				MSb			LSb	
ADDULNK	k	Add FSR2 with (k) & return	2	1110	1000	11kk	kkkk	None
MOVSF	z_s, f_d	Move z_s (source) to 1st word	2	1110	1011	0zzz	zzzz	None
		f_d (destination) 2nd word	2	1111	ffff	ffff	ffff	
MOVSFL	z_s, f_d	Opcode 1st word		0000	0000	0000	0010	None
		Move z_s (source) to 2nd word	3	1111	xxxz	zzzz	zzff	
		f_d (full destination) 3rd word		1111	ffff	ffff	ffff	
MOVSS	z_s, z_d	Move z_s (source) to 1st word		1110	1011	1zzz	zzzz	None
		z_d (destination) 2nd word	2	1111	xxxx	xzzz	zzzz	
PUSHL	k	Push literal to POSTDEC2	1	1110	1010	kkkk	kkkk	None
SUBULNK	k	Subtract (k) from FSR2 & return	2	1110	1001	11kk	kkkk	None

Note 1: If Program Counter (PC) is modified or a conditional test is true, the instruction requires an additional cycle. The extra cycle is executed as a NOP.

2: Some instructions are multi word instructions. The second/third words of these instructions will be decoded as a NOP, unless the first word of the instruction retrieves the information embedded in these 16-bits. This ensures that all program memory locations have a valid instruction.

3: Only available when extended instruction set is enabled.

4: f_s and f_d do not cover the full memory range. Two MSBs of bank selection are forced to ‘b00 to limit the range of these instructions to lower 4k addressing space.