Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

PIC12(L)F1571/2

8-Pin MCU with High-Precision 16-Bit PWMs

Description:

PIC12(L)F1571/2 microcontrollers combine the capabilities of 16-bit PWMs with Analog to suit a variety of applications. These devices deliver three 16-bit PWMs with independent timers for applications where high resolution is needed, such as LED lighting, stepper motors, power supplies and other general purpose applications. The core independent peripherals (16-bit PWMs, Complementary Waveform Generator), Enhanced Universal Synchronous Asynchronous Receiver Transceiver (EUSART) and Analog (ADCs, Comparator and DAC) enable closed-loop feedback and communication for use in multiple market segments. The EUSART peripheral enables the communication for applications such as LIN.

Core Features:

- C Compiler Optimized RISC Architecture
- Only 49 Instructions
- Operating Speed:
  - DC – 32 MHz clock input
  - 125 ns minimum instruction cycle
- Interrupt Capability
- 16-Level Deep Hardware Stack
- Two 8-Bit Timers
- One 16-Bit Timer
- Three Additional 16-Bit Timers available using the 16-Bit PWMs
- Power-on Reset (POR)
- Power-up Timer (PWRT)
- Low-Power Brown-out Reset (LPBOR)
- Programmable Watchdog Timer (WDT) up to 256s
- Programmable Code Protection

Memory:

- Up to 3.5 Kbytes Flash Program Memory
- Up to 256 Bytes Data SRAM Memory
- Direct, Indirect and Relative Addressing modes
- High-Endurance Flash Data Memory (HEF)
  - 128 bytes if nonvolatile data storage
  - 100k erase/write cycles

Operating Characteristics:

- Operating Voltage Range:
  - 1.8V to 3.6V (PIC12LF1571/2)
  - 2.3V to 5.5V (PIC12F1571/2)
- Temperature Range:
  - Industrial: -40°C to +85°C
  - Extended: -40°C to +125°C
- Internal Voltage Reference module
- In-Circuit Serial Programming™ (ICSP™) via Two Pins

eXtreme Low-Power (XLP) Features:

- Sleep mode: 20 nA @ 1.8V, Typical
- Watchdog Timer: 260 nA @ 1.8V, Typical
- Operating Current:
  - 30 μA/MHz @ 1.8V, typical

Digital Peripherals:

- 16-Bit PWM:
  - Three 16-bit PWMs with independent timers
  - Multiple Output modes (Edge-Aligned, Center-Aligned, Set and Toggle on Register Match)
  - User settings for phase, duty cycle, period, offset and polarity
  - 16-bit timer capability
  - Interrupts generated based on timer matches with Offset, Duty Cycle, Period and Phase registers
- Complementary Waveform Generator (CWG):
  - Rising and falling edge dead-band control
  - Multiple signal sources
- Enhanced Universal Synchronous Asynchronous Receiver Transceiver (EUSART):
  - Supports LIN applications

Device I/O Port Features:

- Six I/Os
- Individually Selectable Weak Pull-ups
- Interrupt-On-Change Pins Option with Edge-Selectable Option

Analog Peripherals:

- 10-Bit Analog-to-Digital Converter (ADC):
  - Up to four external channels
  - Conversion available during Sleep
- Comparator:
  - Low-Power/High-Speed modes
  - Fixed Voltage Reference at (non)inverting input(s)
  - Comparator outputs externally accessible
  - Synchronization with Timer1 clock source
  - Software hysteresis enable
- 5-Bit Digital-to-Analog Converter (DAC):
  - 5-bit resolution, rail-to-rail
  - Positive reference selection
  - Unbuffered I/O pin output
  - Internal connections to ADCs and comparators
- Voltage Reference:
  - Fixed voltage reference with 1.024V, 2.048V and 4.096V output levels

Clocking Structure:

- Precision Internal Oscillator:
  - Factory calibrated ±1%, typical
  - Software-selectable clock speeds from 31 kHz to 32 MHz
- External Oscillator Block with:
  - Resonator modes up to 20 MHz
  - Two External Clock modes up to 32 MHz
- Fail-Safe Clock Monitor
- Digital Oscillator Input Available

PIC12(L)F1571/2 FAMILY TYPES

Device	Data Sheet Index	Program Memory Flash (K words)	Data SRAM (bytes)	High-Endurance Flash (bytes)	I/O Pins	8-Bit/16-Bit Timers	Comparators	16-Bit PWM	10-Bit ADC (ch)	5-Bit DAC	CWG	EUSART	Debug(1)	XLP
PIC12(L)F1571	**A**	1	128	128	6	2/4(2)	1	3	4	1	1	0	I	Y
PIC12(L)F1572	**A**	2	256	128	6	2/4(2)	1	3	4	1	1	1	I	Y

Note 1: I – Debugging integrated on chip.

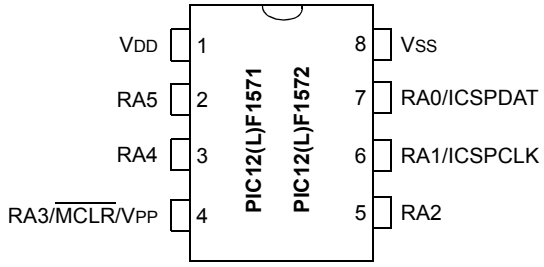
2: Three additional 16-bit timers available when not using the 16-bit PWM outputs.

Data Sheet Index: (Unshaded devices are described in this document.)

A DS40001723 PIC12(L)F1571/2 Data Sheet, 8-Pin Flash, 8-Bit MCU with High-Precision 16-Bit PWM.

PIN DIAGRAMS

Pin Diagram – 8-Pin PDIP, SOIC, DFN, MSOP, UDFN



Note: See Table 1 for location of all peripheral functions.

TABLE 1: 8-PIN ALLOCATION TABLE (PIC12(L)F1571/2)

I/O	8-Pin PDIP/SOIC/MSOP/DFN/UDFN	ADC	Reference	Comparator	Timers	PWM	EUSART(2)	CWG	Interrupt	Pull-up	Basic
RA0	7	AN0	DAC1OUT	C1IN+	—	PWM2	TX(2) CK(2)	CWG1B	IOC	Y	ICSPDAT ICDDAT
RA1	6	AN1	VREF+	C1IN0-	—	PWM1	RX(2) DT(2)	—	IOC	Y	ICSPCLK ICDCLK
RA2	5	AN2	—	C1OUT	T0CKI	PWM3	—	$\overline{\text{CWG1FLT}}$ CWG1A	IOC INT	Y	—
RA3	4	—	—	—	T1G(1)	—	—	—	IOC	Y	$\overline{\text{MCLR}}$ VPP
RA4	3	AN3	—	C1IN1-	T1G	PWM2(1)	TX(1,2) CK(1,2)	CWG1B(1)	IOC	Y	CLKOUT
RA5	2	—	—	—	T1CKI	PWM1(1)	RX(1,2) DT(1,2)	CWG1A(1)	IOC	Y	CLKIN
VDD	1	—	—	—	—	—	—	—	—	—	VDD
Vss	8	—	—	—	—	—	—	—	—	—	Vss

Note 1: Alternate pin function selected with the APFCON (Register 11-1) register.
2: PIC12(L)F1572 only.

Table of Contents

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@microchip.com**. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000000A is version A of document DS30000000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Web site; **http://www.microchip.com**
- Your local Microchip sales office (see last page)

When contacting a sales office, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our web site at **www.microchip.com** to receive the most current information on all of our products.

NOTES:

1.0 DEVICE OVERVIEW

The PIC12(L)F1571/2 devices are described within this data sheet. The block diagram of these devices is shown in Figure 1-1, the available peripherals are shown in Table 1-1 and the pinout descriptions are shown in Table 1-2.

TABLE 1-1: DEVICE PERIPHERAL SUMMARY

Peripheral		PIC12(L)F1571	PIC12(L)F1572
Analog-to-Digital Converter (ADC)		●	●
Complementary Wave Generator (CWG)		●	●
Digital-to-Analog Converter (DAC)		●	●
Enhanced Universal Synchronous/Asynchronous Receiver/Transmitter (EUSART)			●
Fixed Voltage Reference (FVR)		●	●
Temperature Indicator		●	●
Comparators			
	C1	●	●
PWM Modules			
	PWM1	●	●
	PWM2	●	●
	PWM3	●	●
Timers			
	Timer0	●	●
	Timer1	●	●
	Timer2	●	●

1.1 Register and Bit Naming Conventions

1.1.1 REGISTER NAMES

When there are multiple instances of the same peripheral in a device, the peripheral control registers will be depicted as the concatenation of a peripheral identifier, peripheral instance and control identifier. The control registers section will show just one instance of all the register names with an 'x' in the place of the peripheral instance number. This naming convention may also be applied to peripherals when there is only one instance of that peripheral in the device to maintain compatibility with other devices in the family that contain more than one.

1.1.2 BIT NAMES

There are two variants for bit names:

- Short name: Bit function abbreviation
- Long name: Peripheral abbreviation + short name

1.1.2.1 Short Bit Names

Short bit names are an abbreviation for the bit function. For example, some peripherals are enabled with the EN bit. The bit names shown in the registers are the short name variant.

Short bit names are useful when accessing bits in C programs. The general format for accessing bits by the short name is *RegisterName*bits.*ShortName*. For example, the enable bit, EN, in the COG1CON0 register can be set in C programs with the instruction, `COG1CON0bits.EN = 1`.

Short names are generally not useful in assembly programs because the same name may be used by different peripherals in different bit positions. When this occurs, during the include file generation, all instances of that short bit name are appended with an underscore, plus the name of the register in which the bit resides, to avoid naming contentions.

1.1.2.2 Long Bit Names

Long bit names are constructed by adding a peripheral abbreviation prefix to the short name. The prefix is unique to the peripheral, thereby making every long bit name unique. The long bit name for the COG1 enable bit is the COG1 prefix, G1, appended with the enable bit short name, EN, resulting in the unique bit name G1EN.

Long bit names are useful in both C and assembly programs. For example, in C, the COG1CON0 enable bit can be set with the `G1EN = 1` instruction. In assembly, this bit can be set with the `BSF COG1CON0,G1EN` instruction.

1.1.2.3 Bit Fields

Bit fields are two or more adjacent bits in the same register. Bit fields adhere only to the short bit naming convention. For example, the three Least Significant bits of the COG1CON0 register contain the mode control bits. The short name for this field is MD. There is no long bit name variant. Bit field access is only possible in C programs. The following example demonstrates a C program instruction for setting the COG1 to the Push-Pull mode:

```
COG1CON0bits.MD = 0x5;
```

Individual bits in a bit field can also be accessed with long and short bit names. Each bit is the field name appended with the number of the bit position within the field. For example, the Most Significant mode bit has the short bit name, MD2, and the long bit name is G1MD2. The following two examples demonstrate assembly program sequences for setting the COG1 to Push-Pull mode:

Example 1:

```
MOVLW  ~(1<<G1MD1)
ANDWF  COG1CON0,F
MOVLW  1<<G1MD2 | 1<<G1MD0
IORWF  COG1CON0,F
```

Example 2:

```
BSF    COG1CON0,G1MD2
BCF    COG1CON0,G1MD1
BSF    COG1CON0,G1MD0
```

1.1.3 REGISTER AND BIT NAMING EXCEPTIONS

1.1.3.1 Status, Interrupt and Mirror Bits

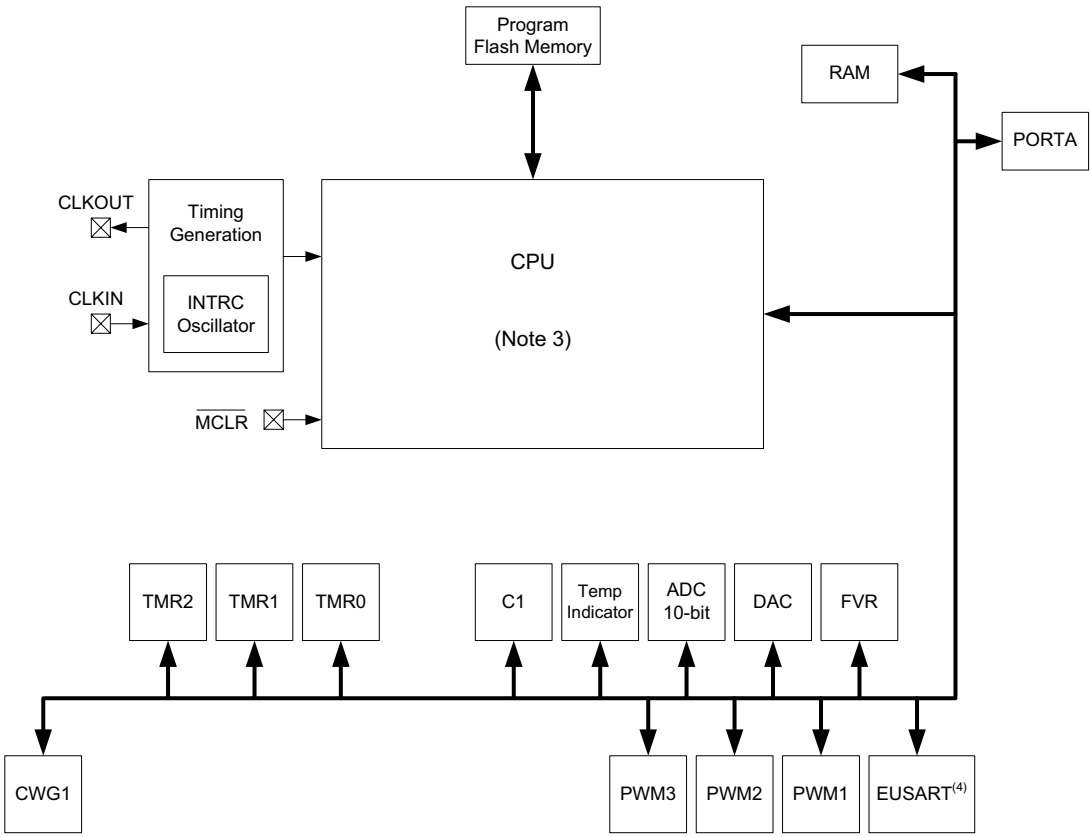
Status, interrupt enables, interrupt flags and mirror bits are contained in registers that span more than one peripheral. In these cases, the bit name shown is unique so there is no prefix or short name variant.

1.1.3.2 Legacy Peripherals

There are some peripherals that do not strictly adhere to these naming conventions. Peripherals that have existed for many years and are present in almost every device are the exceptions. These exceptions were necessary to limit the adverse impact of the new conventions on legacy code. Peripherals that do adhere to the new convention will include a table in the registers section indicating the long name prefix for each peripheral instance. Peripherals that fall into the exception category will not have this table. These peripherals include, but are not limited to, the following:

- EUSART
- MSSP

FIGURE 1-1: PIC12(L)F1571/2 BLOCK DIAGRAM



2: See Table 1-1 for peripherals available on specific devices.
3: See Figure 2-1.
4: PIC12(L)F1572 only.

TABLE 1-2: PIC12(L)F1571/2 PINOUT DESCRIPTION

Name	Function	Input Type	Output Type	Description
RA0/AN0/C1IN+/DACOUT/ TX(2)/CK(2)/CWG1B/PWM2/ ICSPDAT/ICDDAT	RA0	(3)	(4)	General purpose I/O.
	AN0			ADC channel input.
	C1IN+			Comparator positive input.
	DACOUT			Digital-to-Analog Converter output.
	TX			USART asynchronous transmit.
	CK			USART synchronous clock.
	CWG1B			CWG complementary output.
	PWM2			PWM output.
	ICSPDAT			ICSP™ data I/O.
	ICDDAT			In-circuit debug data.
RA1/AN1/VREF+/C1IN0-/RX(2)/ DT(2)/PWM1/ICSPCLK/ICDCLK	RA1	(3)	(4)	General purpose I/O.
	AN1			ADC channel input.
	VREF+			ADC Voltage Reference input.
	C1IN0-			Comparator negative input.
	RX			USART asynchronous input.
	DT			USART synchronous data.
	PWM1			PWM output.
	ICSPCLK			ICSP programming clock.
	ICDCLK			In-circuit debug clock.
RA2/AN2/C1OUT/T0CKI/ $\overline{\text{CWG1FLT}}$/CWG1A/PWM3/INT	RA2	(3)	(4)	General purpose I/O.
	AN2			ADC channel input.
	C1OUT			Comparator output.
	T0CKI			Timer0 clock input.
	CWG1FLT			Complementary Waveform Generator Fault input.
	CWG1A			CWG complementary output.
	PWM3			PWM output.
	INT			External interrupt.
RA3/VPP/T1G(1)/$\overline{\text{MCLR}}$	RA3	(3)	(4)	General purpose input with IOC and WPU.
	VPP			Programming voltage.
	T1G			Timer1 gate input.
	$\overline{\text{MCLR}}$			Master Clear with internal pull-up.
RA4/AN3/C1IN1-/T1G/TX(1,2)/ CK(1,2)/CWG1B(1)/PWM2(1)/ CLKOUT	RA4	(3)	(4)	General purpose I/O.
	AN3			ADC channel input.
	C1IN1-			Comparator negative input.
	T1G			Timer1 gate input.
	TX			USART asynchronous transmit.
	CK			USART synchronous clock.
	CWG1B			CWG complementary output.
	PWM2			PWM output.
	CLKOUT			FOSC/4 output.

Legend: AN = Analog input or output CMOS = CMOS compatible input or output OD = Open-Drain
TTL = TTL compatible input ST = Schmitt Trigger input with CMOS levels I^2C = Schmitt Trigger input with I^2C levels
HV = High Voltage XTAL = Crystal

Note 1: Alternate pin function selected with the APFCON (Register 11-1) register.
2: PIC12(L)F1572 only.
3: Input type is selected by the port.
4: Output type is selected by the port.

TABLE 1-2: PIC12(L)F1571/2 PINOUT DESCRIPTION (CONTINUED)

Name	Function	Input Type	Output Type	Description
RA5/T1CKI/RX(1,2)/DT(1,2)/ CWG1A(1)/PWM1(1)/CLKIN	RA5	(3)	(4)	General purpose I/O.
	T1CKI			Timer1 clock input.
	RX			USART asynchronous input.
	DT			USART synchronous data.
	CWG1A			CWG complementary output.
	PWM1			PWM output.
	CLKIN			External Clock input (EC mode).
VDD	VDD	Power	—	Positive supply.
VSS	VSS	Power	—	Ground reference.

Legend: AN = Analog input or output CMOS = CMOS compatible input or output OD = Open-Drain
TTL = TTL compatible input ST = Schmitt Trigger input with CMOS levels I^2C = Schmitt Trigger input with I^2C levels
HV = High Voltage XTAL = Crystal

Note 1: Alternate pin function selected with the APFCON (Register 11-1) register.
2: PIC12(L)F1572 only.
3: Input type is selected by the port.
4: Output type is selected by the port.

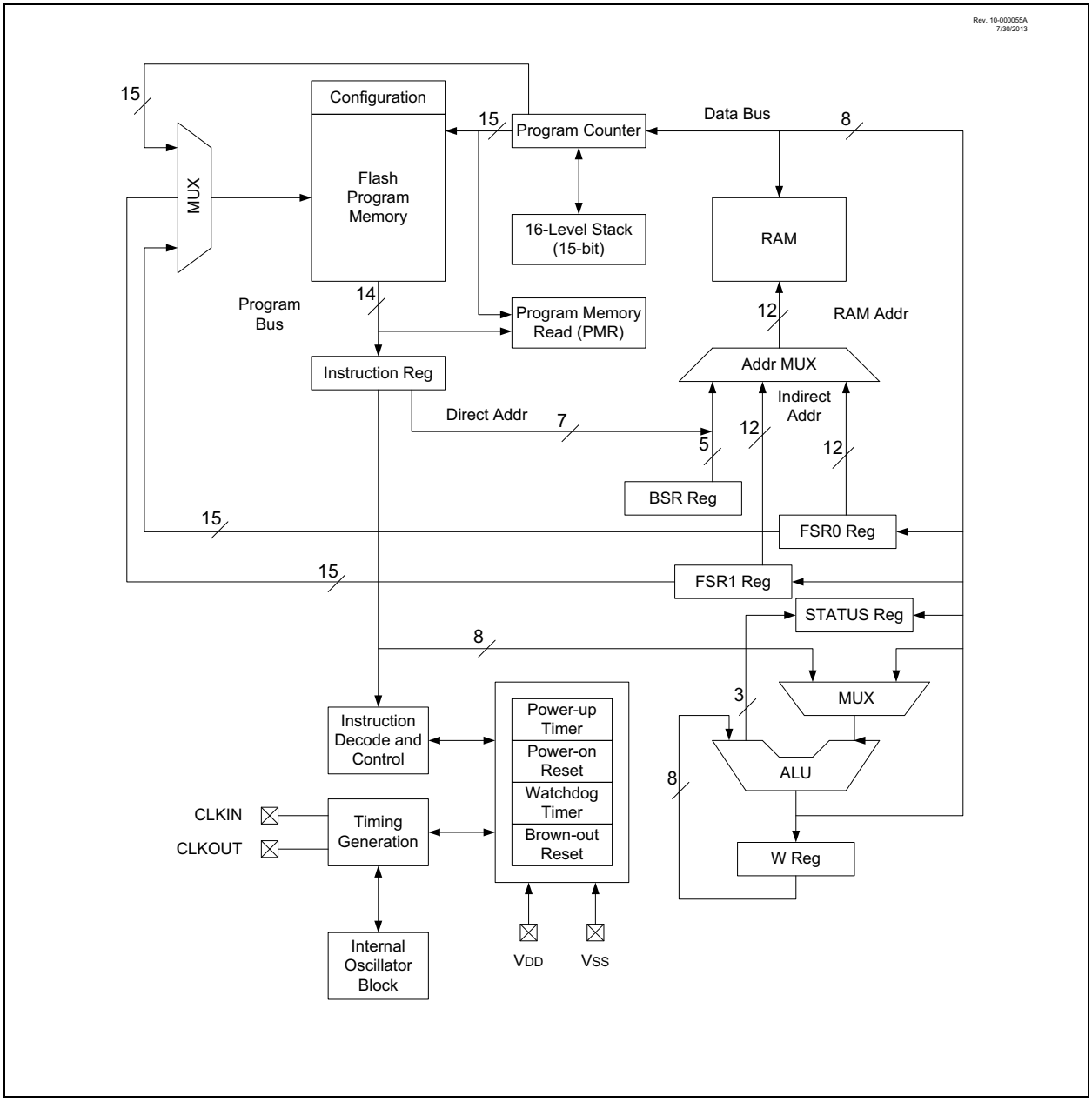
NOTES:

2.0 ENHANCED MID-RANGE CPU

This family of devices contains an enhanced mid-range 8-bit CPU core. The CPU has 49 instructions. Interrupt capability includes automatic context saving. The hardware stack is 16 levels deep and has Overflow and Underflow Reset capability. Direct, Indirect and Relative Addressing modes are available. Two File Select Registers (FSRs) provide the ability to read program and data memory.

- Automatic Interrupt Context Saving
- 16-Level Stack with Overflow and Underflow
- File Select Registers
- Instruction Set

FIGURE 2-1: CORE BLOCK DIAGRAM



2.1 Automatic Interrupt Context Saving

During interrupts, certain registers are automatically saved in shadow registers and restored when returning from the interrupt. This saves stack space and user code. See **Section 7.5 "Automatic Context Saving"**, for more information.

2.2 16-Level Stack with Overflow and Underflow

These devices have a hardware stack memory, 15 bits wide and 16 words deep. A Stack Overflow or Underflow will set the appropriate bit (STKOVF or STKUNF) in the PCON register, and if enabled, will cause a Software Reset. See **Section 3.5 "Stack"** for more details.

2.3 File Select Registers

There are two 16-bit File Select Registers (FSR). FSRs can access all file registers and program memory, which allows one Data Pointer for all memory. When an FSR points to program memory, there is one additional instruction cycle in instructions using INDF to allow the data to be fetched. General purpose memory can now also be addressed linearly, providing the ability to access contiguous data larger than 80 bytes. There are also new instructions to support the FSRs. See **Section 3.6 "Indirect Addressing"** for more details.

2.4 Instruction Set

There are 49 instructions for the enhanced mid-range CPU to support the features of the CPU. See **Section 25.0 "Instruction Set Summary"** for more details.

3.0 MEMORY ORGANIZATION

These devices contain the following types of memory:

- Program Memory:
  - Configuration Words
  - Device ID
  - User ID
  - Flash Program Memory
- Data Memory:
  - Core Registers
  - Special Function Registers
  - General Purpose RAM
  - Common RAM

The following features are associated with access and control of program memory and data memory:

- PCL and PCLATH
- Stack
- Indirect Addressing

3.1 Program Memory Organization

The enhanced mid-range core has a 15-bit Program Counter (PC) capable of addressing a 32K x 14 program memory space. Table 3-1 shows the memory sizes implemented. Accessing a location above these boundaries will cause a wraparound within the implemented memory space. The Reset vector is at 0000h and the interrupt vector is at 0004h (see Figure 3-1).

3.2 High-Endurance Flash

This device has a 128-byte section of high-endurance Program Flash Memory (PFM) in lieu of data EEPROM. This area is especially well-suited for non-volatile data storage that is expected to be updated frequently over the life of the end product. See **Section 10.2 "Flash Program Memory Overview"** for more information on writing data to PFM. See **Section 3.2.1.2 "Indirect Read with FSR"** for more information about using the FSR registers to read byte data stored in PFM.

TABLE 3-1: DEVICE SIZES AND ADDRESSES

Device	Program Memory Space (Words)	Last Program Memory Address	High-Endurance Flash Memory Address Range(1)
PIC12(L)F1571	1,024	03FFh	0380h-03FFh
PIC12(L)F1572	2,048	07FFh	0780h-07FFh

Note 1: High-endurance Flash applies to the low byte of each address in the range.

FIGURE 3-1: PROGRAM MEMORY MAP AND STACK FOR PIC12(L)F1571

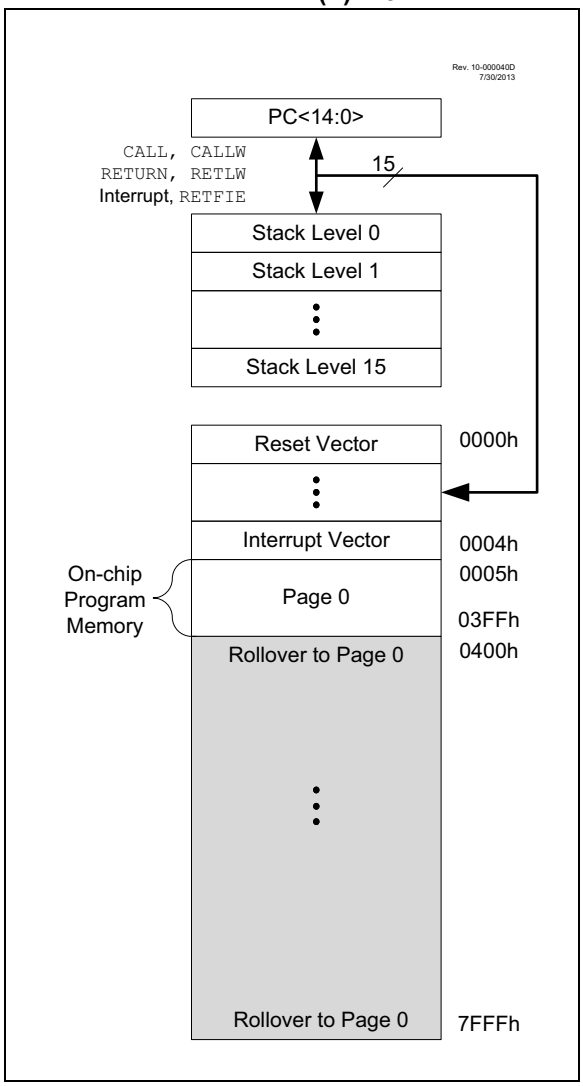
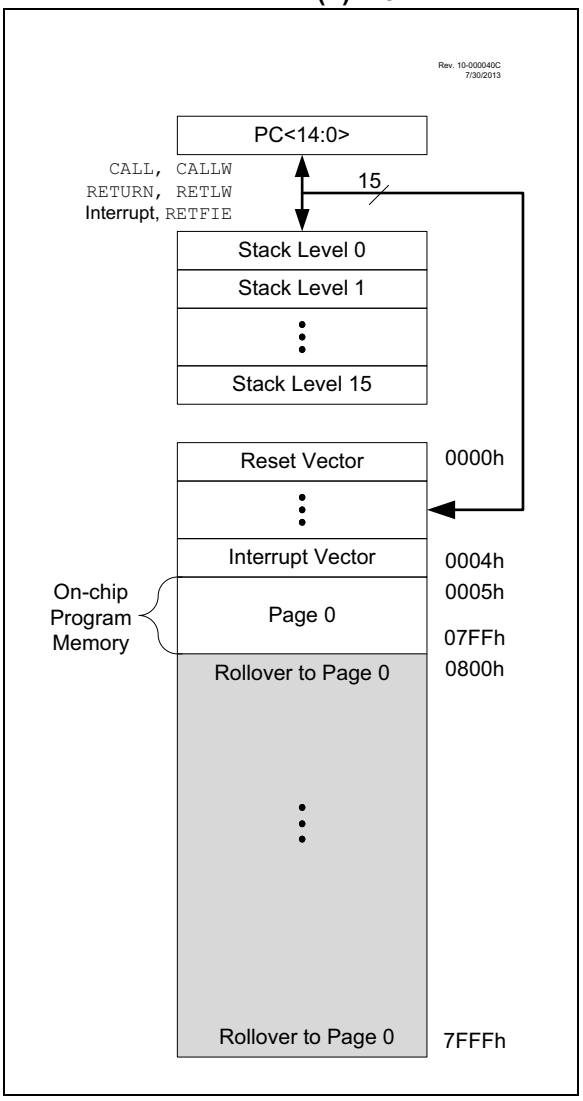


FIGURE 3-2: PROGRAM MEMORY MAP AND STACK FOR PIC12(L)F1572



3.2.1 READING PROGRAM MEMORY AS DATA

There are two methods of accessing constants in program memory. The first method is to use tables of RETLW instructions. The second method is to set an FSR to point to the program memory.

3.2.1.1 RETLW Instruction

The RETLW instruction can be used to provide access to tables of constants. The recommended way to create such a table is shown in Example 3-1.

EXAMPLE 3-1: RETLW INSTRUCTION

```
constants
    BRW                 ;Add Index in W to
                        ;program counter to
                        ;select data
    RETLW DATA0         ;Index0 data
    RETLW DATA1         ;Index1 data
    RETLW DATA2
    RETLW DATA3


my_function
    ;… LOTS OF CODE…
    MOVLW       DATA_INDEX
    call constants
    ;… THE CONSTANT IS IN W
```

The BRW instruction makes this type of table very simple to implement. If your code must remain portable with previous generations of microcontrollers, then the BRW instruction is not available, so the older table read method must be used.

3.2.1.2 Indirect Read with FSR

The program memory can be accessed as data by setting bit 7 of the FSRnH register and reading the matching INDFn register. The MOVIW instruction will place the lower eight bits of the addressed word in the W register. Writes to the program memory cannot be performed via the INDFn registers. Instructions that access the program memory via the FSR require one extra instruction cycle to complete. Example 3-2 demonstrates accessing the program memory via an FSR.

The HIGH operator will set bit<7> if a label points to a location in program memory.

EXAMPLE 3-2: ACCESSING PROGRAM MEMORY VIA FSR

```
constants
  DW      DATA0         ;First constant
  DW      DATA1         ;Second constant
  DW      DATA2
  DW      DATA3
my_function
  ;… LOTS OF CODE…
  MOVLW   DATA_INDEX
  ADDLW   LOW constants
  MOVWF   FSR1L
  MOVLW   HIGH constants ;MSb is set
                          automatically
  MOVWF   FSR1H
  BTFSC   STATUS,C      ;carry from ADDLW?
  INCF    FSR1H,f       ;yes
  MOVIW   0[FSR1]
;THE PROGRAM MEMORY IS IN W
```

3.3 Data Memory Organization

The data memory is partitioned in 32 memory banks with 128 bytes in a bank. Each bank consists of (Figure 3-3):

- 12 Core Registers
- 20 Special Function Registers (SFR)
- Up to 80 bytes of General Purpose RAM (GPR)
- 16 bytes of Common RAM

The active bank is selected by writing the bank number into the Bank Select Register (BSR). Unimplemented memory will read as '0'. All data memory can be accessed either directly (via instructions that use the file registers) or indirectly via the two File Select Registers (FSR). See **Section 3.6 "Indirect Addressing"** for more information.

Data memory uses a 12-bit address. The upper five bits of the address define the bank address and the lower seven bits select the registers/RAM in that bank.

3.3.1 CORE REGISTERS

The core registers contain the registers that directly affect the basic operation. The core registers occupy the first 12 addresses of every data memory bank (addresses: x00h/x08h through x0Bh/x8Bh). These registers are listed below in Table 3-2. For detailed information, see Table 3-9.

TABLE 3-2: CORE REGISTERS

Addresses	BANKx
x00h or x80h	INDF0
x01h or x81h	INDF1
x02h or x82h	PCL
x03h or x83h	STATUS
x04h or x84h	FSR0L
x05h or x85h	FSR0H
x06h or x86h	FSR1L
x07h or x87h	FSR1H
x08h or x88h	BSR
x09h or x89h	WREG
x0Ah or x8Ah	PCLATH
x0Bh or x8Bh	INTCON

3.3.1.1 STATUS Register

The STATUS register, shown in Register 3-1, contains:

- The arithmetic status of the ALU
- The Reset status

The STATUS register can be the destination for any instruction, like any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the $\overline{TO}$ and $\overline{PD}$ bits are not writable. Therefore, the result of an instruction with the STATUS register as destination may be different than intended.

For example, `CLRF STATUS` will clear the upper three bits and set the Z bit. This leaves the STATUS register as '`000u u1uu`' (where `u` = unchanged).

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF` and `MOVWF` instructions are used to alter the STATUS register, because these instructions do not affect any Status bits. For other instructions not affecting any Status bits, refer to **Section 25.0 "Instruction Set Summary"**).

Note 1: The C and DC bits operate as $\overline{Borrow}$ and $\overline{Digit\ Borrow}$ out bits, respectively, in subtraction.

REGISTER 3-1: STATUS: STATUS REGISTER

U-0	U-0	U-0	R-1/q	R-1/q	R/W-0/u	R/W-0/u	R/W-0/u
—	—	—	$\overline{TO}$	$\overline{PD}$	Z	DC(1)	C(1)
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	q = Value depends on condition

bit 7-5 **Unimplemented:** Read as '0'

bit 4 **$\overline{TO}$:** Time-out bit
- 1 = After power-up, `CLRWDT` instruction or `SLEEP` instruction
- 0 = A WDT time-out occurred

bit 3 **$\overline{PD}$:** Power-Down bit
- 1 = After power-down or by the `CLRWDT` instruction
- 0 = By execution of the `SLEEP` instruction

bit 2 **Z:** Zero bit
- 1 = The result of an arithmetic or logic operation is zero
- 0 = The result of an arithmetic or logic operation is not zero

bit 1 **DC:** Digit Carry/$\overline{Digit\ Borrow}$ bit (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)(1)
- 1 = A carry-out from the 4th low-order bit of the result occurred
- 0 = No carry-out from the 4th low-order bit of the result

bit 0 **C:** Carry/$\overline{Borrow}$ bit(1) (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)(1)
- 1 = A carry-out from the Most Significant bit of the result occurred
- 0 = No carry-out from the Most Significant bit of the result occurred

Note 1: For $\overline{Borrow}$, the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For rotate (`RRF`, `RLF`) instructions, this bit is loaded with either the high-order or low-order bit of the source register.

3.3.2 SPECIAL FUNCTION REGISTER

The Special Function Registers are registers used by the application to control the desired operation of peripheral functions in the device. The Special Function Registers occupy the 20 bytes after the core registers of every data memory bank (addresses: x0Ch/x8Ch through x1Fh/x9Fh). The registers associated with the operation of the peripherals are described in the appropriate peripheral chapter of this data sheet.

3.3.3 GENERAL PURPOSE RAM

There are up to 80 bytes of GPR in each data memory bank. The Special Function Registers occupy the 20 bytes after the core registers of every data memory bank (addresses: x0Ch/x8Ch through x1Fh/x9Fh).

3.3.3.1 Linear Access to GPR

The general purpose RAM can be accessed in a non-banked method via the FSRs. This can simplify access to large memory structures. See **Section 3.6.2 "Linear Data Memory"** for more information.

3.3.4 COMMON RAM

There are 16 bytes of common RAM accessible from all banks.

3.3.5 DEVICE MEMORY MAPS

The memory maps for PIC12(L)F1571/2 are as shown in Table 3-3 through Table 3-8.

FIGURE 3-3: BANKED MEMORY PARTITIONING

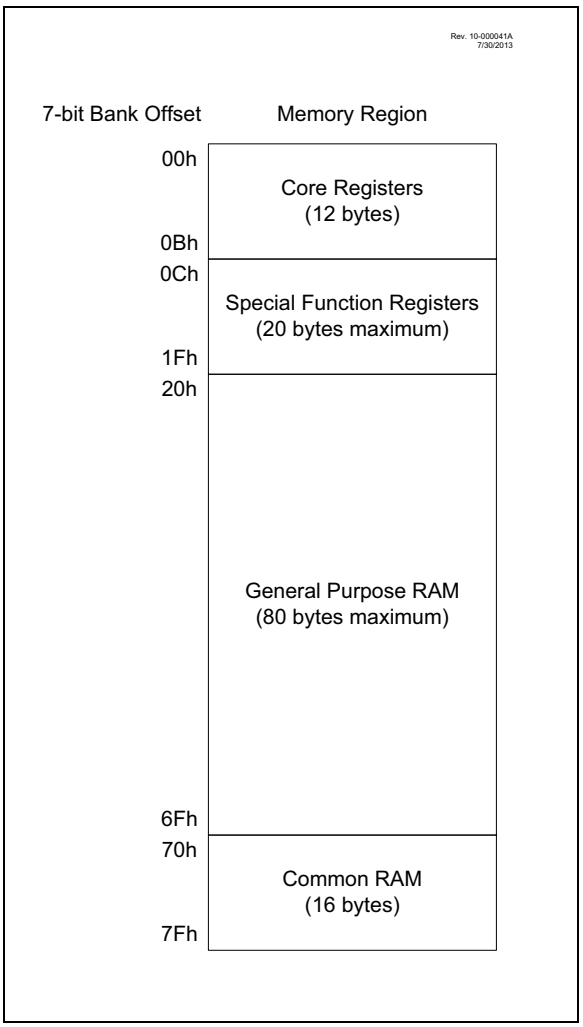


TABLE 3-3: PIC12(L)F1571 MEMORY MAP, BANK 0-7

	BANK 0		BANK 1		BANK 2		BANK 3		BANK 4		BANK 5		BANK 6		BANK 7
000h	Core Registers (Table 3-2)	080h	Core Registers (Table 3-2)	100h	Core Registers (Table 3-2)	180h	Core Registers (Table 3-2)	200h	Core Registers (Table 3-2)	280h	Core Registers (Table 3-2)	300h	Core Registers (Table 3-2)	380h	Core Registers (Table 3-2)
00Bh		08Bh		10Bh		18Bh		20Bh		28Bh		30Bh		38Bh	
00Ch	PORTA	08Ch	TRISA	10Ch	LATA	18Ch	ANSELA	20Ch	WPUA	28Ch	ODCONA	30Ch	SLRCONA	38Ch	INLVLA
00Dh	—	08Dh	—	10Dh	—	18Dh	—	20Dh	—	28Dh	—	30Dh	—	38Dh	—
00Eh	—	08Eh	—	10Eh	—	18Eh	—	20Eh	—	28Eh	—	30Eh	—	38Eh	—
00Fh	—	08Fh	—	10Fh	—	18Fh	—	20Fh	—	28Fh	—	30Fh	—	38Fh	—
010h	—	090h	—	110h	—	190h	—	210h	—	290h	—	310h	—	390h	—
011h	PIR1	091h	PIE1	111h	CM1CON0	191h	PMADRL	211h	—	291h	—	311h	—	391h	IOCAP
012h	PIR2	092h	PIE2	112h	CM1CON1	192h	PMADRH	212h	—	292h	—	312h	—	392h	IOCAN
013h	PIR3	093h	PIE3	113h	—	193h	PMDATL	213h	—	293h	—	313h	—	393h	IOCAF
014h	—	094h	—	114h	—	194h	PMDATH	214h	—	294h	—	314h	—	394h	—
015h	TMR0	095h	OPTION_REG	115h	CMOUT	195h	PMCON1	215h	—	295h	—	315h	—	395h	—
016h	TMR1L	096h	PCON	116h	BORCON	196h	PMCON2	216h	—	296h	—	316h	—	396h	—
017h	TMR1H	097h	WDTCON	117h	FVRCON	197h	VREGCON(1)	217h	—	297h	—	317h	—	397h	—
018h	T1CON	098h	OSCTUN E	118h	DACxCON0	198h	—	218h	—	298h	—	318h	—	398h	—
019h	T1GCON	099h	OSCCON	119h	DACxCON1	199h	—	219h	—	299h	—	319h	—	399h	—
01Ah	TMR2	09Ah	OSCSTAT	11Ah	—	19Ah	—	21Ah	—	29Ah	—	31Ah	—	39Ah	—
01Bh	PR2	09Bh	ADRESL	11Bh	—	19Bh	—	21Bh	—	29Bh	—	31Bh	—	39Bh	—
01Ch	T2CON	09Ch	ADRESH	11Ch	—	19Ch	—	21Ch	—	29Ch	—	31Ch	—	39Ch	—
01Dh	—	09Dh	ADCON0	11Dh	APFCON	19Dh	—	21Dh	—	29Dh	—	31Dh	—	39Dh	—
01Eh	—	09Eh	ADCON1	11Eh	—	19Eh	—	21Eh	—	29Eh	—	31Eh	—	39Eh	—
01Fh	—	09Fh	ADCON2	11Fh	—	19Fh	—	21Fh	—	29Fh	—	31Fh	—	39Fh	—
020h	General Purpose Register 80 Bytes	0A0h	General Purpose Register 48 Bytes	120h	Unimplemented Read as '0'	1A0h	Unimplemented Read as '0'	220h	Unimplemented Read as '0'	2A0h	Unimplemented Read as '0'	320h	Unimplemented Read as '0'	3A0h	Unimplemented Read as '0'
06Fh		0BFh		16Fh		1EFh		26Fh		2EFh		36Fh		3EFh	
		0C0h	Unimplemented Read as '0'												
		0EFh													
070h	Common RAM	0F0h	Common RAM (Accesses 70h-7Fh)	170h	Common RAM (Accesses 70h-7Fh)	1F0h	Common RAM (Accesses 70h-7Fh)	270h	Common RAM (Accesses 70h-7Fh)	2F0h	Common RAM (Accesses 70h-7Fh)	370h	Common RAM (Accesses 70h-7Fh)	3F0h	Common RAM (Accesses 70h-7Fh)
07Fh		0FFh		17Fh		1FFh		27Fh		2FFh		37Fh		3FFh	

Legend: ☐ = Unimplemented data memory locations, read as '0'.

Note 1: PIC12F1571 only.

TABLE 3-4: PIC12(L)F1572 MEMORY MAP, BANK 0-7

BANK 0		BANK 1		BANK 2		BANK 3		BANK 4		BANK 5		BANK 6		BANK 7	
000h	Core Registers (Table 3-2)	080h	Core Registers (Table 3-2)	100h	Core Registers (Table 3-2)	180h	Core Registers (Table 3-2)	200h	Core Registers (Table 3-2)	280h	Core Registers (Table 3-2)	300h	Core Registers (Table 3-2)	380h	Core Registers (Table 3-2)
00Bh		08Bh		10Bh		18Bh		20Bh		28Bh		30Bh		38Bh	
00Ch	PORTA	08Ch	TRISA	10Ch	LATA	18Ch	ANSELA	20Ch	WPUA	28Ch	ODCONA	30Ch	SLRCONA	38Ch	INLVLA
00Dh	—	08Dh	—	10Dh	—	18Dh	—	20Dh	—	28Dh	—	30Dh	—	38Dh	—
00Eh	—	08Eh	—	10Eh	—	18Eh	—	20Eh	—	28Eh	—	30Eh	—	38Eh	—
00Fh	—	08Fh	—	10Fh	—	18Fh	—	20Fh	—	28Fh	—	30Fh	—	38Fh	—
010h	—	090h	—	110h	—	190h	—	210h	—	290h	—	310h	—	390h	—
011h	PIR1	091h	PIE1	111h	CM1CON0	191h	PMADRL	211h	—	291h	—	311h	—	391h	IOCAP
012h	PIR2	092h	PIE2	112h	CM1CON1	192h	PMADRH	212h	—	292h	—	312h	—	392h	IOCAN
013h	PIR3	093h	PIE3	113h	—	193h	PMDATL	213h	—	293h	—	313h	—	393h	IOCAF
014h	—	094h	—	114h	—	194h	PMDATH	214h	—	294h	—	314h	—	394h	—
015h	TMR0	095h	OPTION_REG	115h	CMOUT	195h	PMCON1	215h	—	295h	—	315h	—	395h	—
016h	TMR1L	096h	PCON	116h	BORCON	196h	PMCON2	216h	—	296h	—	316h	—	396h	—
017h	TMR1H	097h	WDTCON	117h	FVRCON	197h	VREGCON(1)	217h	—	297h	—	317h	—	397h	—
018h	T1CON	098h	OSCTUNE	118h	DAC1CON0	198h	—	218h	—	298h	—	318h	—	398h	—
019h	T1GCON	099h	OSCCON	119h	DAC1CON1	199h	RCREG	219h	—	299h	—	319h	—	399h	—
01Ah	TMR2	09Ah	OSCSTAT	11Ah	—	19Ah	TXREG	21Ah	—	29Ah	—	31Ah	—	39Ah	—
01Bh	PR2	09Bh	ADRESL	11Bh	—	19Bh	SPBRG	21Bh	—	29Bh	—	31Bh	—	39Bh	—
01Ch	T2CON	09Ch	ADRESH	11Ch	—	19Ch	SPBRGH	21Ch	—	29Ch	—	31Ch	—	39Ch	—
01Dh	—	09Dh	ADCON0	11Dh	APFCON	19Dh	RCSTA	21Dh	—	29Dh	—	31Dh	—	39Dh	—
01Eh	—	09Eh	ADCON1	11Eh	—	19Eh	TXSTA	21Eh	—	29Eh	—	31Eh	—	39Eh	—
01Fh	—	09Fh	ADCON2	11Fh	—	19Fh	BAUDCON	21Fh	—	29Fh	—	31Fh	—	39Fh	—
020h	General Purpose Register 80 Bytes	0A0h	General Purpose Register 80 Bytes	120h	General Purpose Register 80 Bytes	1A0h	Unimplemented Read as '0'	220h	Unimplemented Read as '0'	2A0h	Unimplemented Read as '0'	320h	Unimplemented Read as '0'	3A0h	Unimplemented Read as '0'
06Fh		0EFh		16Fh		1EFh		26Fh		2EFh		36Fh		3EFh	
070h	Common RAM	0F0h	Accesses 70h-7Fh	170h	Accesses 70h-7Fh	1F0h	Accesses 70h-7Fh	270h	Accesses 70h-7Fh	2F0h	Accesses 70h-7Fh	370h	Accesses 70h-7Fh	3F0h	Accesses 70h-7Fh
07Fh		0FFh		17Fh		1FFh		27Fh		2FFh		37Fh		3FFh	

Legend: ☐ = Unimplemented data memory locations, read as '0'.

Note 1: PIC12F1572 only.

TABLE 3-5: PIC12(L)F1571/2 MEMORY MAP, BANK 8-23

BANK 8	BANK 9	BANK 10	BANK 11	BANK 12	BANK 13	BANK 14	BANK 15
400h Core Registers (Table 3-2) 40Bh	480h Core Registers (Table 3-2) 48Bh	500h Core Registers (Table 3-2) 50Bh	580h Core Registers (Table 3-2) 58Bh	600h Core Registers (Table 3-2) 60Bh	680h Core Registers (Table 3-2) 68Bh	700h Core Registers (Table 3-2) 70Bh	780h Core Registers (Table 3-2) 78Bh
40Ch —	48Ch —	50Ch —	58Ch —	60Ch —	68Ch —	70Ch —	78Ch —
40Dh —	48Dh —	50Dh —	58Dh —	60Dh —	68Dh —	70Dh —	78Dh —
40Eh —	48Eh —	50Eh —	58Eh —	60Eh —	68Eh —	70Eh —	78Eh —
40Fh —	48Fh —	50Fh —	58Fh —	60Fh —	68Fh —	70Fh —	78Fh —
410h —	490h —	510h —	590h —	610h —	690h —	710h —	790h —
411h —	491h —	511h —	591h —	611h —	691h CWG1DBR	711h —	791h —
412h —	492h —	512h —	592h —	612h —	692h CWG1DBF	712h —	792h —
413h —	493h —	513h —	593h —	613h —	693h CWG1CON0	713h —	793h —
414h —	494h —	514h —	594h —	614h —	694h CWG1CON1	714h —	794h —
415h —	495h —	515h —	595h —	615h —	695h CWG1CON2	715h —	795h —
416h —	496h —	516h —	596h —	616h —	696h —	716h —	796h —
417h —	497h —	517h —	597h —	617h —	697h —	717h —	797h —
418h —	498h —	518h —	598h —	618h —	698h —	718h —	798h —
419h —	499h —	519h —	599h —	619h —	699h —	719h —	799h —
41Ah —	49Ah —	51Ah —	59Ah —	61Ah —	69Ah —	71Ah —	79Ah —
41Bh —	49Bh —	51Bh —	59Bh —	61Bh —	69Bh —	71Bh —	79Bh —
41Ch —	49Ch —	51Ch —	59Ch —	61Ch —	69Ch —	71Ch —	79Ch —
41Dh —	49Dh —	51Dh —	59Dh —	61Dh —	69Dh —	71Dh —	79Dh —
41Eh —	49Eh —	51Eh —	59Eh —	61Eh —	69Eh —	71Eh —	79Eh —
41Fh —	49Fh —	51Fh —	59Fh —	61Fh —	69Fh —	71Fh —	79Fh —
420h Unimplemented Read as '0' 46Fh	4A0h Unimplemented Read as '0' 4EFh	520h Unimplemented Read as '0' 56Fh	5A0h Unimplemented Read as '0' 5EFh	620h Unimplemented Read as '0' 66Fh	6A0h Unimplemented Read as '0' 6EFh	720h Unimplemented Read as '0' 76Fh	7A0h Unimplemented Read as '0' 7EFh
470h Accesses 70h-7Fh 47Fh	4F0h Accesses 70h-7Fh 4FFh	570h Accesses 70h-7Fh 57Fh	5F0h Accesses 70h-7Fh 5FFh	670h Accesses 70h-7Fh 67Fh	6F0h Accesses 70h-7Fh 6FFh	770h Accesses 70h-7Fh 77Fh	7F0h Accesses 70h-7Fh 7FFh

BANK 16	BANK 17	BANK 18	BANK 19	BANK 20	BANK 21	BANK 22	BANK 23
800h Core Registers (Table 3-2) 80Bh	880h Core Registers (Table 3-2) 88Bh	900h Core Registers (Table 3-2) 90Bh	980h Core Registers (Table 3-2) 98Bh	A00h Core Registers (Table 3-2) A0Bh	A80h Core Registers (Table 3-2) A8Bh	B00h Core Registers (Table 3-2) B0Bh	B80h Core Registers (Table 3-2) B8Bh
80Ch Unimplemented Read as '0' 86Fh	88Ch Unimplemented Read as '0' 8EFh	90Ch Unimplemented Read as '0' 96Fh	98Ch Unimplemented Read as '0' 9EFh	A0Ch Unimplemented Read as '0' A6Fh	A8Ch Unimplemented Read as '0' AEFh	B0Ch Unimplemented Read as '0' B6Fh	B8Ch Unimplemented Read as '0' BEFh
870h Accesses 70h-7Fh 87Fh	8F0h Accesses 70h-7Fh 8FFh	970h Accesses 70h-7Fh 97Fh	9F0h Accesses 70h-7Fh 9FFh	A70h Accesses 70h-7Fh A7Fh	AF0h Accesses 70h-7Fh AFFh	B70h Accesses 70h-7Fh B7Fh	BF0h Accesses 70h-7Fh BFFh

Legend: ☐ = Unimplemented data memory locations, read as '0'.

TABLE 3-6: PIC12(L)F1571/2 MEMORY MAP, BANK 24-31

BANK 24		BANK 25		BANK 26		BANK 27		BANK 28		BANK 29		BANK 30		BANK 31	
C00h	Core Registers (Table 3-2)	C80h	Core Registers (Table 3-2)	D00h	Core Registers (Table 3-2)	D80h	Core Registers (Table 3-2)	E00h	Core Registers (Table 3-2)	E80h	Core Registers (Table 3-2)	F00h	Core Registers (Table 3-2)	F80h	Core Registers (Table 3-2)
C0Bh		C8Bh		D0Bh		D8Bh		E0Bh		E8Bh		F0Bh		F8Bh	
C0Ch	—	C8Ch	—	D0Ch	—	D8Ch	See Table 3-7 for Register Mapping Details	E0Ch	—	E8Ch	—	F0Ch	—	F8Ch	See Table 3-7 for Register Mapping Details
C0Dh	—	C8Dh	—	D0Dh	—			E0Dh	—	E8Dh	—	F0Dh	—		
C0Eh	—	C8Eh	—	D0Eh	—			E0Eh	—	E8Eh	—	F0Eh	—		
C0Fh	—	C8Fh	—	D0Fh	—			E0Fh	—	E8Fh	—	F0Fh	—		
C10h	—	C90h	—	D10h	—			E10h	—	E90h	—	F10h	—		
C11h	—	C91h	—	D11h	—			E11h	—	E91h	—	F11h	—		
C12h	—	C92h	—	D12h	—			E12h	—	E92h	—	F12h	—		
C13h	—	C93h	—	D13h	—			E13h	—	E93h	—	F13h	—		
C14h	—	C94h	—	D14h	—			E14h	—	E94h	—	F14h	—		
C15h	—	C95h	—	D15h	—			E15h	—	E95h	—	F15h	—		
C16h	—	C96h	—	D16h	—			E16h	—	E96h	—	F16h	—		
C17h	—	C97h	—	D17h	—			E17h	—	E97h	—	F17h	—		
C18h	—	C98h	—	D18h	—			E18h	—	E98h	—	F18h	—		
C19h	—	C99h	—	D19h	—			E19h	—	E99h	—	F19h	—		
C1Ah	—	C9Ah	—	D1Ah	—			E1Ah	—	E9Ah	—	F1Ah	—		
C1Bh	—	C9Bh	—	D1Bh	—			E1Bh	—	E9Bh	—	F1Bh	—		
C1Ch	—	C9Ch	—	D1Ch	—			E1Ch	—	E9Ch	—	F1Ch	—		
C1Dh	—	C9Dh	—	D1Dh	—			E1Dh	—	E9Dh	—	F1Dh	—		
C1Eh	—	C9Eh	—	D1Eh	—			E1Eh	—	E9Eh	—	F1Eh	—		
C1Fh	—	C9Fh	—	D1Fh	—			E1Fh	—	E9Fh	—	F1Fh	—		
C20h	Unimplemented Read as '0'	CA0h	Unimplemented Read as '0'	D20h	Unimplemented Read as '0'			E20h	Unimplemented Read as '0'	EA0h	Unimplemented Read as '0'	F20h	Unimplemented Read as '0'		
C6Fh		CEFh		D6Fh		DEFh		E6Fh		EEFh		F6Fh		FEFh	
C70h	Accesses 70h-7Fh	CF0h	Accesses 70h-7Fh	D70h	Accesses 70h-7Fh	DF0h	Accesses 70h-7Fh	E70h	Accesses 70h-7Fh	EF0h	Accesses 70h-7Fh	F70h	Accesses 70h-7Fh	FF0h	Accesses 70h-7Fh
CFFh		CFFh		D7Fh		DFFh		E7Fh		EFFh		F7Fh		FFFh	

Legend: ☐ = Unimplemented data memory locations, read as '0'.

TABLE 3-7: PIC12(L)F1571/2 MEMORY MAP, BANK 27

	Bank 31
D8Ch	—
D8Dh	—
D8Eh	PWMEN
D8Fh	PWMLD
D90h	PWMOUT
D91h	PWM1PHL
D92h	PWM1PHH
D93h	PWM1DCL
D94h	PWM1DCH
D95h	PWM1PRL
D96h	PWM1PRH
D97h	PWM1OFL
D98h	PWM1OFH
D99h	PWM1TMRL
D9Ah	PWM1TMRH
D9Bh	PWM1CON
D9Ch	PWM1INTE
D9Dh	PWM1INTF
D9Eh	PWM1CLKCON
D9Fh	PWM1LDCON
DA0h	PWM1OFCON
DA1h	PWM2PHL
DA2h	PWM2PHH
DA3h	PWM2DCL
DA4h	PWM2DCH
DA5h	PWM2PRL
DA6h	PWM2PRH
DA7h	PWM2OFL
DA8h	PWM2OFH
DA9h	PWM2TMRL
DAAh	PWM2TMRH
DABh	PWM2CON
DACh	PWM2INTE
DADh	PWM2INTF
DAEh	PWM2CLKCON
DAFh	PWM2LDCON
DB0h	PWM2OFCON
DB1h	PWM3PHL
DB2h	PWM3PHH
DB3h	PWM3DCL
DB4h	PWM3DCH
DB5h	PWM2PRL
DB6h	PWM3PRH
DB7h	PWM3OFL
DB8h	PWM3OFH
DB9h	PWM3TMRL
DBAh	PWM3TMRH
DBBh	PWM3CON
DBCh	PWM3INTE
DBDh	PWM3INTF
DBEh	PWM3CLKCON
DBFh	PWM3LDCON
DC0h	PWM3OFCON
DC1h – DEFh	—

Legend: = Unimplemented data memory locations, read as '0'.

TABLE 3-8: PIC12(L)F1571/2 MEMORY MAP, BANK 31

	Bank 31
F8Ch – FE3h	Unimplemented Read as '0'
FE4h	STATUS_SHAD
FE5h	WREG_SHAD
FE6h	BSR_SHAD
FE7h	PCLATH_SHAD
FE8h	FSR0L_SHAD
FE9h	FSR0H_SHAD
FEAh	FSR1L_SHAD
FEBh	FSR1H_SHAD
FECh	—
FEDh	STKPTR
FEEh	TOSL
FEFh	TOSH

Legend: = Unimplemented data memory locations, read as '0'.