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

8-Pin Flash Microcontrollers with XLP Technology

High-Performance RISC CPU

- Only 49 Instructions to Learn:
  - All single-cycle instructions except branches
- Operating Speed:
  - DC – 32 MHz oscillator/clock input
  - DC – 125 ns instruction cycle
- Interrupt Capability with Automatic Context Saving
- 16-Level Deep Hardware Stack with Optional Overflow/Underflow Reset
- Direct, Indirect and Relative Addressing modes:
  - Two full 16-bit File Select Registers (FSRs)
  - FSRs can read program and data memory

Flexible Oscillator Structure

- Precision 32 MHz Internal Oscillator Block:
  - Factory calibrated to ± 1%, typical
  - Software selectable frequencies range of 31 kHz to 32 MHz
- 31 kHz Low-Power Internal Oscillator
- Four Crystal modes up to 32 MHz
- Three External Clock modes up to 32 MHz
- 4X Phase Lock Loop (PLL)
- Fail-Safe Clock Monitor:
  - Allows for safe shutdown if peripheral clock stops
- Two-Speed Oscillator Start-up
- Reference Clock module:
  - Programmable clock output frequency and duty-cycle

Special Microcontroller Features

- Operating Voltage Range:
  - 2.3V-5.5V (PIC12F1840)
  - 1.8V-3.6V (PIC12LF1840)
- Self-Reprogrammable under Software Control
- Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Programmable Brown-out Reset (BOR)
- Extended Watchdog Timer (WDT)
- In-Circuit Serial Programming™ (ICSP™) via Two Pins
- In-Circuit Debug (ICD) via Two Pins
- Enhanced Low-Voltage Programming (LVP)
- Programmable Code Protection
- Power-Saving Sleep mode

Extreme Low-Power Management with PIC12LF1840 XLP

- Sleep mode: 20 nA @ 1.8V, typical
- Watchdog Timer: 500 nA @ 1.8V, typical
- Timer1 Oscillator: 300 nA @ 32 kHz, 1.8V, typical
- Operating Current: 30 μA/MHz @ 1.8V, typical

Analog Features

- Analog-to-Digital Converter (ADC) module:
  - 10-bit resolution, 4 channels
  - Conversion available during Sleep
- Analog Comparator module:
  - One rail-to-rail analog comparator
  - Power mode control
  - Software controllable hysteresis
- Voltage Reference module:
  - Fixed Voltage Reference (FVR) with 1.024V, 2.048V and 4.096V output levels
  - 5-bit rail-to-rail resistive DAC with positive and negative reference selection

Peripheral Highlights

- 5 I/O Pins and 1 Input-Only Pin:
  - High current sink/source 25 mA/25 mA
  - Programmable weak pull-ups
  - Programmable interrupt-on-change pins
- Timer0: 8-Bit Timer/Counter with 8-Bit Prescaler
- Enhanced Timer1:
  - 16-bit timer/counter with prescaler
  - External Gate Input mode
  - Dedicated, low-power 32 kHz oscillator driver
- Timer2: 8-Bit Timer/Counter with 8-Bit Period Register, Prescaler and Postscaler
- Enhanced CCP (ECCP) module:
  - Software selectable time bases
  - Auto-shutdown and auto-restart
  - PWM steering
- Master Synchronous Serial Port (MSSP) with SPI and I^{2}C™ with:
  - 7-bit address masking
  - SMBus/PMBus™ compatibility
- Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) module:
  - RS-232, RS-485 and LIN compatible
  - Auto-Baud Detect
- Capacitive Sensing (CPS) module (mTouch™):
  - 4 input channels

Peripheral Features (Continued)

- Data Signal Modulator module:
  - Selectable modulator and carrier sources
- SR Latch:
  - Multiple Set/Reset input options
  - Emulates 555 Timer applications

PIC12(L)F1822/1840/PIC16(L)F182X/1847 Family Types

Device	Data Sheet Index	Program Memory Flash (words)	Data EEPROM (bytes)	Data SRAM (bytes)	I/O's(2)	10-bit ADC (ch)	CapSense (ch)	Comparators	Timers (8/16-bit)	EUSART	MSSP (I^2C^{TM}/SPI)	ECCP (Full-Bridge) ECCP (Half-Bridge) CCP	SR Latch	Debug(1)	XLP
PIC12(L)F1822	(1)	2K	256	128	6	4	4	1	2/1	1	1	0/1/0	Y	I/H	Y
PIC12(L)F1840	(2)	4K	256	256	6	4	4	1	2/1	1	1	0/1/0	Y	I/H	Y
PIC16(L)F1823	(1)	2K	256	128	12	8	8	2	2/1	1	1	1/0/0	Y	I/H	Y
PIC16(L)F1824	(3)	4K	256	256	12	8	8	2	4/1	1	1	1/1/2	Y	I/H	Y
PIC16(L)F1825	(4)	8K	256	1024	12	8	8	2	4/1	1	1	1/1/2	Y	I/H	Y
PIC16(L)F1826	(5)	2K	256	256	16	12	12	2	2/1	1	1	1/0/0	Y	I/H	Y
PIC16(L)F1827	(5)	4K	256	384	16	12	12	2	4/1	1	2	1/1/2	Y	I/H	Y
PIC16(L)F1828	(3)	4K	256	256	18	12	12	2	4/1	1	1	1/1/2	Y	I/H	Y
PIC16(L)F1829	(4)	8K	256	1024	18	12	12	2	4/1	1	2	1/1/2	Y	I/H	Y
PIC16(L)F1847	(6)	8K	256	1024	16	12	12	2	4/1	1	2	1/1/2	Y	I/H	Y

Note 1: I - Debugging, Integrated on Chip; H - Debugging, available using Debug Header.
2: One pin is input-only.

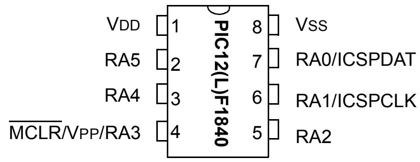
Data Sheet Index: (Unshaded devices are described in this document.)

1: DS41413 PIC12(L)F1822/PIC16(L)F1823 Data Sheet, 8/14-Pin Flash Microcontrollers.
2: DS41441 PIC12(L)F1840 Data Sheet, 8-Pin Flash Microcontrollers.
3: DS41419 PIC16(L)F1824/1828 Data Sheet, 28/40/44-Pin Flash Microcontrollers.
4: DS41440 PIC16(L)F1825/1829 Data Sheet, 14/20-Pin Flash Microcontrollers.
5: DS41391 PIC16(L)F1826/1827 Data Sheet, 18/20/28-Pin Flash Microcontrollers.
6: DS41453 PIC16(L)F1847 Data Sheet, 18/20/28-Pin Flash Microcontrollers.

Note: For other small form-factor package availability and marking information, please visit **http://www.microchip.com/packaging** or contact your local sales office.

FIGURE 1: 8-PIN DIAGRAM FOR PIC12(L)F1840

PDIP, SOIC, DFN, UDFN



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

TABLE 1: 8-PIN ALLOCATION TABLE (PIC12(L)F1840)

I/O	8-Pin PDIP/SOIC/DFN/UDFN	ADC	Reference	Cap Sense	Comparator	SR Latch	Timers	ECCP	EUSART	MSSP	Interrupt	Modulator	Pull-up	Basic
RA0	7	AN0	DACOUT	CPS0	C1IN+	—	—	P1B	TX CK	SDO $\overline{SS}$(1)	IOC	MDOUT	Y	ICSPDAT ICDDAT
RA1	6	AN1	VREF	CPS1	C1IN0-	SRI	—	—	RX DT	SCL SCK	IOC	MDMIN	Y	ICSPCLK ICPCLK
RA2	5	AN2	—	CPS2	C1OUT	SRQ	T0CKI	CCP1 P1A FLT0	—	SDA SDI	INT/ IOC	MDCIN1	Y	—
RA3	4	—	—	—	—	—	T1G(1)	—	—	$\overline{SS}$	IOC	—	Y	$\overline{MCLR}$ VPP
RA4	3	AN3	—	CPS3	C1IN1-	—	T1G T1OSO	P1B(1)	TX(1) CK(1)	SDO(1)	IOC	MDCIN2	Y	OSC2 CLKOUT CLKR
RA5	2	—	—	—	—	SRNQ	T1CKI T1OSI	CCP1(1) P1A(1)	RX(1) DT(1)	—	IOC	—	Y	OSC1 CLKIN
VDD	1	—	—	—	—	—	—	—	—	—	—	—	—	VDD
VSS	8	—	—	—	—	—	—	—	—	—	—	—	—	VSS

Note 1: Alternate pin function selected with the APFCON (Register 12-1) register.

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.

1.0 DEVICE OVERVIEW

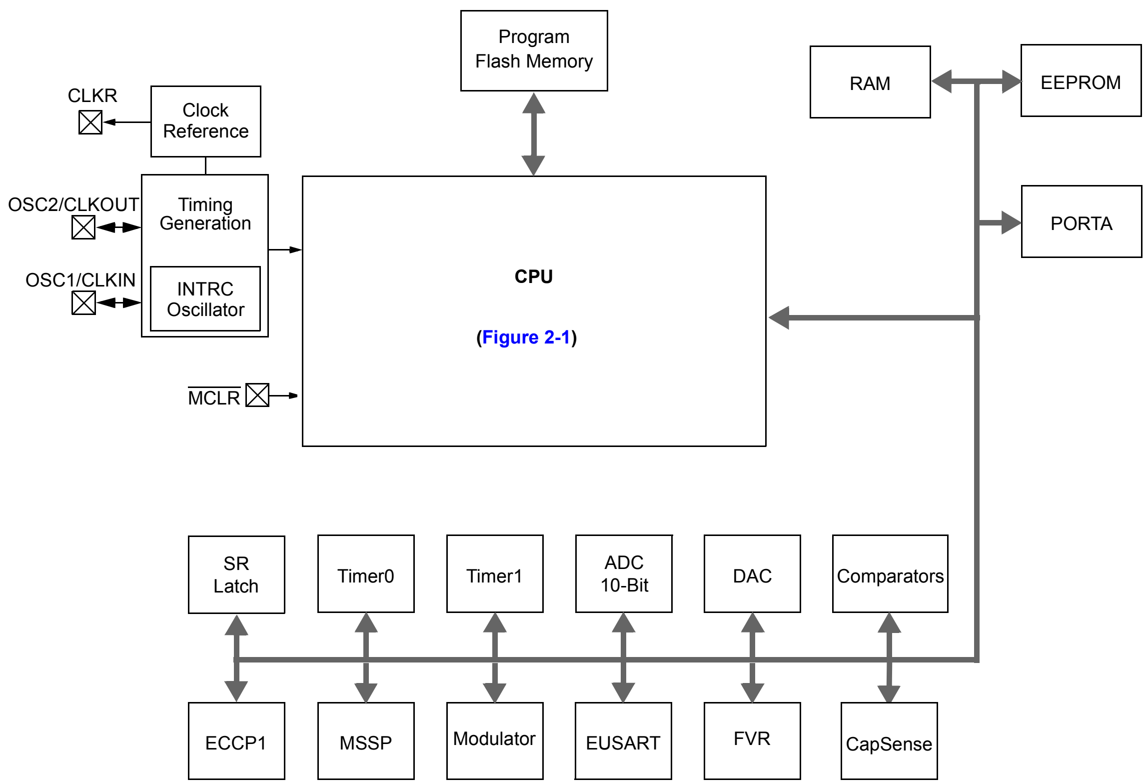
The PIC12(L)F1840 are described within this data sheet. They are available in 8-pin packages. Figure 1-1 shows a block diagram of the PIC12(L)F1840 devices. Table 1-2 shows the pinout descriptions.

Reference Table 1-1 for peripherals available per device.

TABLE 1-1: DEVICE PERIPHERAL SUMMARY

Peripheral		PIC12(L)F1840
ADC		●
Capacitive Sensing (CPS) Module		●
Data EEPROM		●
Digital-to-Analog Converter (DAC)		●
Digital Signal Modulator (DSM)		●
EUSART		●
Fixed Voltage Reference (FVR)		●
SR Latch		●
Capture/Compare/PWM Modules		
	ECCP1	●
Comparators		
	C1	●
Master Synchronous Serial Ports		
	MSSP	●
Timers		
	Timer0	●
	Timer1	●
	Timer2	●

FIGURE 1-1: PIC12(L)F1840 BLOCK DIAGRAM



Note 1: See applicable chapters for more information on peripherals.

2: See Table 1-1 for peripherals available on specific devices.

TABLE 1-2: PIC12(L)F1840 PINOUT DESCRIPTION

Name	Function	Input Type	Output Type	Description
RA0/AN0/CPS0/C1IN+/DACOUT/TX/CK/SDO/$\overline{\text{SS}}$(1)/P1B/MDOUT/ICSPDAT/ICDDAT	RA0	TTL	CMOS	General purpose I/O.
	AN0	AN	—	ADC Channel 0 input.
	CPS0	AN	—	Capacitive sensing input 0.
	C1IN+	AN	—	Comparator C1 positive input.
	DACOUT	—	AN	Digital-to-Analog Converter output.
	TX	—	CMOS	USART asynchronous transmit.
	CK	ST	CMOS	USART synchronous clock.
	SDO	—	CMOS	SPI data output.
	$\overline{\text{SS}}$	ST	—	Slave Select input.
	P1B	—	CMOS	PWM output.
	MDOUT	—	CMOS	Modulator output.
	ICSPDAT	ST	CMOS	ICSP™ Data I/O.
RA1/AN1/CPS1/VREF/C1IN0-/SRI/RX/DT/SCL/SCK/MDMIN/ICSPCLK/ICDCLK	RA1	TTL	CMOS	General purpose I/O.
	AN1	AN	—	ADC Channel 1 input.
	CPS1	AN	—	Capacitive sensing input 1.
	VREF	AN	—	ADC and DAC Positive Voltage Reference input.
	C1IN0-	AN	—	Comparator C1 negative input.
	SRI	ST	—	SR Latch input.
	RX	ST	—	USART asynchronous input.
	DT	ST	CMOS	USART synchronous data.
	SCL	I^2C™	OD	I^2C™ clock.
	SCK	ST	CMOS	SPI clock.
	MDMIN	ST	—	Modulator source input.
	ICSPCLK	ST	—	Serial Programming Clock.
RA2/AN2/CPS2/C1OUT/SRQ/T0CKI/CCP1/P1A/FLT0/SDA/SDI/INT/MDCIN1	RA2	ST	CMOS	General purpose I/O.
	AN2	AN	—	ADC Channel 2 input.
	CPS2	AN	—	Capacitive sensing input 2.
	C1OUT	—	CMOS	Comparator C1 output.
	SRQ	—	CMOS	SR Latch non-inverting output.
	T0CKI	ST	—	Timer0 clock input.
	CCP1	ST	CMOS	Capture/Compare/PWM 1.
	P1A	—	CMOS	PWM output.
	FLT0	ST	—	ECCP Auto-Shutdown Fault input.
	SDA	I^2C™	OD	I^2C™ data input/output.
	SDI	CMOS	—	SPI data input.
	INT	ST	—	External interrupt.
	MDCIN1	ST	—	Modulator Carrier Input 1.
RA3/$\overline{\text{SS}}$/T1G(1)/VPP/$\overline{\text{MCLR}}$	RA3	TTL	—	General purpose input.
	$\overline{\text{SS}}$	ST	—	Slave Select input.
	T1G	ST	—	Timer1 Gate input.
	VPP	HV	—	Programming voltage.
	$\overline{\text{MCLR}}$	ST	—	Master Clear with internal pull-up.

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 12-1) register.

TABLE 1-2: PIC12(L)F1840 PINOUT DESCRIPTION (CONTINUED)

Name	Function	Input Type	Output Type	Description
RA4/AN3/CPS3/OSC2/ CLKOUT/T1OSO/C1IN1-/CLKR/ SDO(1)/CK(1)/TX(1)/P1B(1)/ T1G/MDCIN2	RA4	TTL	CMOS	General purpose I/O.
	AN3	AN	—	ADC Channel 3 input.
	CPS3	AN	—	Capacitive sensing input 3.
	OSC2	—	XTAL	Crystal/Resonator (LP, XT, HS modes).
	CLKOUT	—	CMOS	FOSC/4 output.
	T1OSO	XTAL	XTAL	Timer1 oscillator connection.
	C1IN1-	AN	—	Comparator C1 negative input.
	CLKR	—	CMOS	Clock Reference output.
	SDO	—	CMOS	SPI data output.
	CK	ST	CMOS	USART synchronous clock.
	TX	—	CMOS	USART asynchronous transmit.
	P1B	—	CMOS	PWM output.
	T1G	ST	—	Timer1 Gate input.
	MDCIN2	ST	—	Modulator Carrier Input 2.
RA5/CLKIN/OSC1/T1OSI/ T1CKI/SRNQ/P1A(1)/CCP1(1)/ DT(1)/RX(1)	RA5	TTL	CMOS	General purpose I/O.
	CLKIN	CMOS	—	External clock input (EC mode).
	OSC1	XTAL	—	Crystal/Resonator (LP, XT, HS modes).
	T1OSI	XTAL	XTAL	Timer1 oscillator connection.
	T1CKI	ST	—	Timer1 clock input.
	SRNQ	—	CMOS	SR Latch inverting output.
	P1A	—	CMOS	PWM output.
	CCP1	ST	CMOS	Capture/Compare/PWM 1.
	DT	ST	CMOS	USART synchronous data.
	RX	ST	—	USART asynchronous input.
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

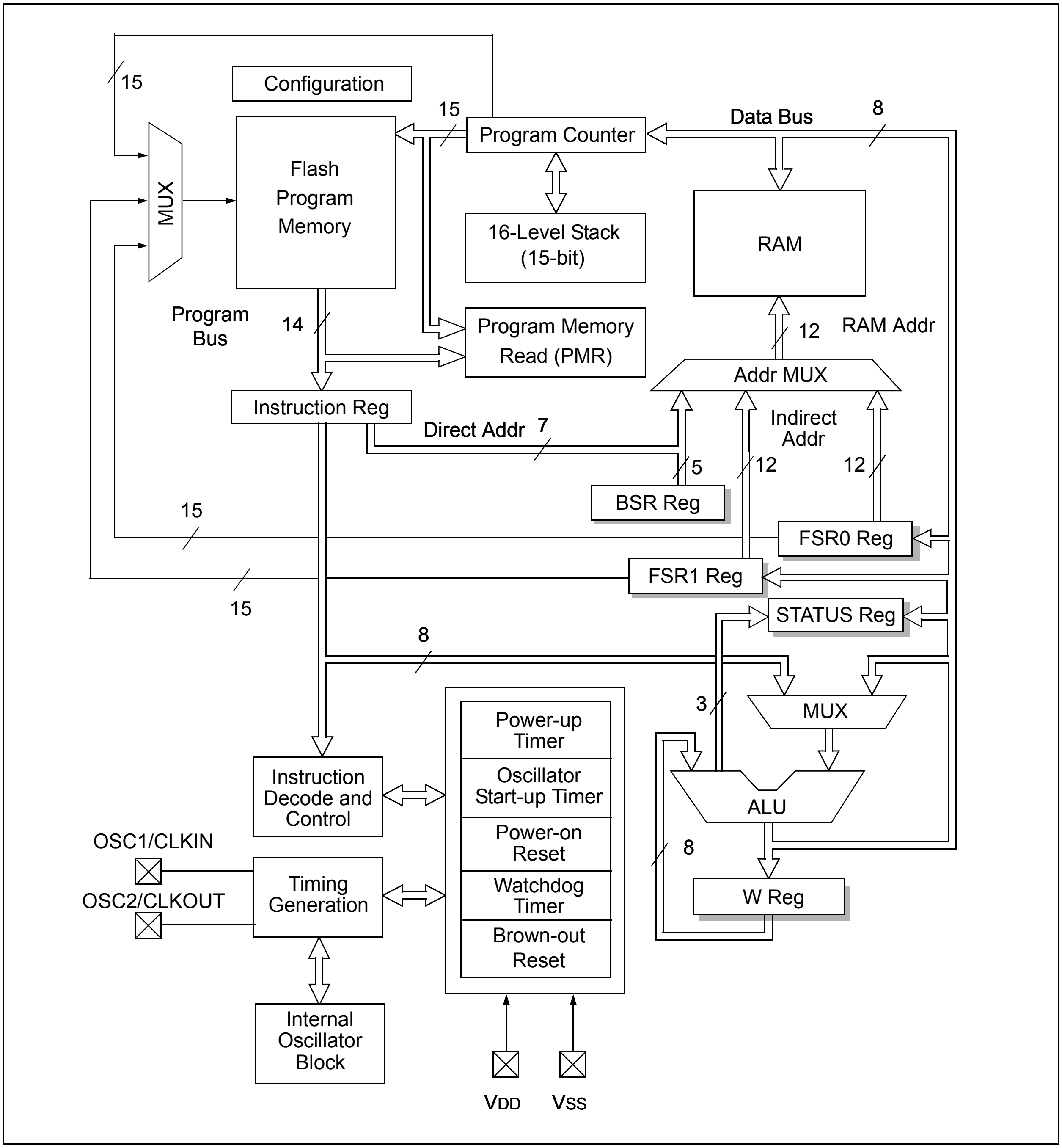
Note 1: Alternate pin function selected with the APFCON (Register 12-1) register.

2.0 ENHANCED MID-RANGE CPU

This family of devices contain 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 8.5 "Automatic Context Saving"**, for more information.

2.2 16-level Stack with Overflow and Underflow

These devices have an external 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 29.0 "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
- Data EEPROM memory[(1)]

Note 1: The Data EEPROM Memory and the method to access Flash memory through the EECON registers is described in **Section 11.0 "Data EEPROM and Flash Program Memory Control"**.

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

- PCL and PCLATH
- Stack
- Indirect Addressing

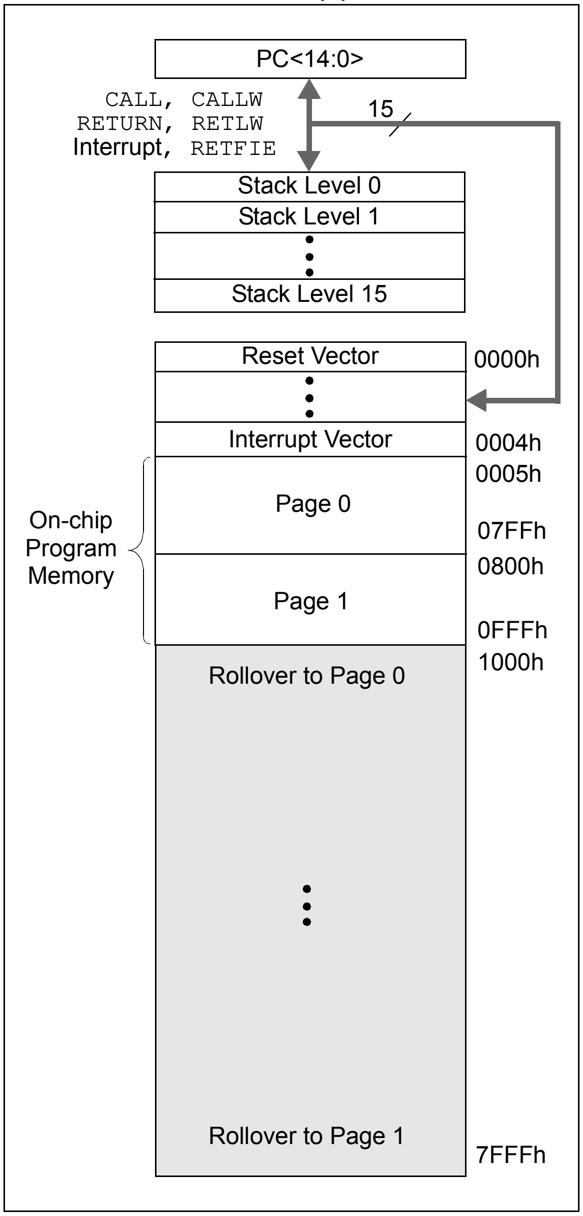
3.1 Program Memory Organization

The enhanced mid-range core has a 15-bit program counter capable of addressing a 32K x 14 program memory space. Table 3-1 shows the memory sizes implemented for the PIC12(L)F1840 family. Accessing a location above these boundaries will cause a wrap-around within the implemented memory space. The Reset vector is at 0000h and the interrupt vector is at 0004h (see Figure 3-1).

TABLE 3-1: DEVICE SIZES AND ADDRESSES

Device	Program Memory Space (Words)	Last Program Memory Address
PIC12(L)F1840	4, 096	0FFFh

FIGURE 3-1: PROGRAM MEMORY MAP AND STACK FOR PIC12(L)F1840



3.1.1 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.

3.1.1.1 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.

3.1.1.2 Indirect Read with FSR

The program memory can be accessed as data by setting bit 7 of the FSRxH register and reading the matching INDFx 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 INDF 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 directive will set bit<7> if a label points to a location in program memory.

EXAMPLE 3-2: ACCESSING PROGRAM MEMORY VIA FSR

```
constants
    RETLW DATA0         ;Index0 data
    RETLW DATA1         ;Index1 data
    RETLW DATA2
    RETLW DATA3
my_function
    ;… LOTS OF CODE…
    MOVLW   LOW constants
    MOVWF   FSR1L
    MOVLW   HIGH constants
    MOVWF   FSR1H
    MOVIW 0[FSR1]
;THE PROGRAM MEMORY IS IN W
```

3.2 Data Memory Organization

The data memory is partitioned in 32 memory banks with 128 bytes in a bank. Each bank consists of (Figure 3-2):

- 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.

3.2.1 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 3-5.

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

3.2.1.1 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 $\overline{\text{TO}}$ and $\overline{\text{PD}}$ 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 29.0 "Instruction Set Summary"**).

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

3.3 Register Definitions: Status

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

bit 3 **$\overline{\text{PD}}$:** Power-Down bit
`1` = After power-up 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/$\overline{\text{Digit Borrow}}$ 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/$\overline{\text{Borrow}}$ 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 $\overline{\text{Borrow}}$, the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand.

3.3.1 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.

3.3.2 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).

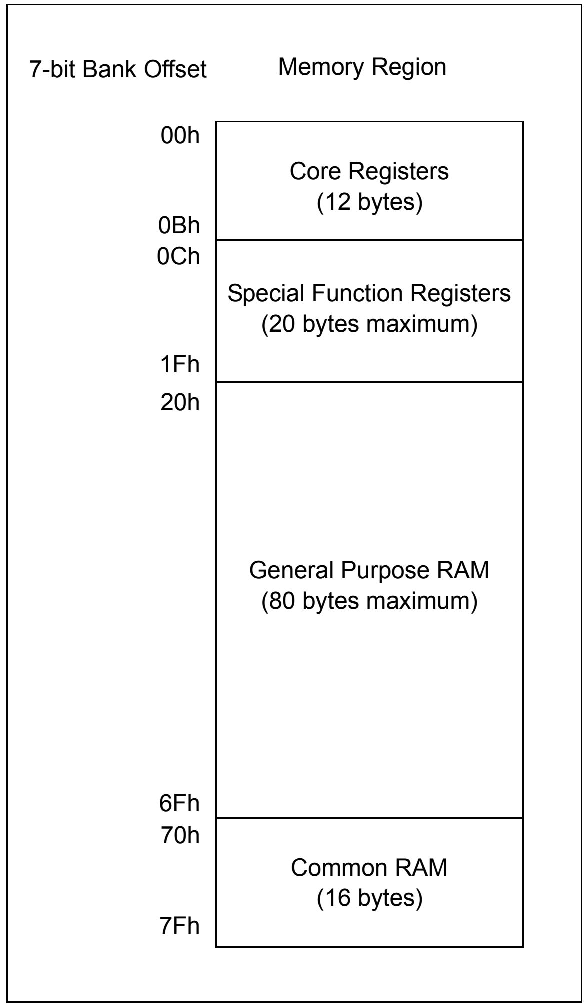
3.3.2.1 Linear Access to GPR

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.

3.3.3 COMMON RAM

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

FIGURE 3-2: BANKED MEMORY PARTITIONING



3.3.4 DEVICE MEMORY MAPS

The memory maps for the device family are as shown in Table 3-3.

TABLE 3-3: PIC12(L)F1840 MEMORY MAP, BANKS 0-7

	BANK 0		BANK 1		BANK 2		BANK 3		BANK 4		BANK 5		BANK 6		BANK 7
000h–00Bh	Core Registers (Table 3-2)	080h–08Bh	Core Registers (Table 3-2)	100h–10Bh	Core Registers (Table 3-2)	180h–18Bh	Core Registers (Table 3-2)	200h–20Bh	Core Registers (Table 3-2)	280h–28Bh	Core Registers (Table 3-2)	300h–30Bh	Core Registers (Table 3-2)	380h–38Bh	Core Registers (Table 3-2)
00Ch	PORTA	08Ch	TRISA	10Ch	LATA	18Ch	ANSELA	20Ch	WPUA	28Ch	—	30Ch	—	38Ch	—
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	EEADRL	211h	SSPBUF	291h	CCPR1L	311h	—	391h	IOCAP
012h	PIR2	092h	PIE2	112h	CM1CON1	192h	EEADRH	212h	SSPADD	292h	CCPR1H	312h	—	392h	IOCAN
013h	—	093h	—	113h	—	193h	EEDATL	213h	SSPMASK	293h	CCP1CON	313h	—	393h	IOCAF
014h	—	094h	—	114h	—	194h	EEDATH	214h	SSPSTAT	294h	PWM1CON	314h	—	394h	—
015h	TMR0	095h	OPTION_REG	115h	CMOUT	195h	EECON1	215h	SSP1CON1	295h	CCP1AS	315h	—	395h	—
016h	TMR1L	096h	PCON	116h	BORCON	196h	EECON2	216h	SSP1CON2	296h	PSTR1CON	316h	—	396h	—
017h	TMR1H	097h	WDTCON	117h	FVRCON	197h	VREGCON(1)	217h	SSP1CON3	297h	—	317h	—	397h	—
018h	T1CON	098h	OSCTUNE	118h	DACCON0	198h	—	218h	—	298h	—	318h	—	398h	—
019h	T1GCON	099h	OSCCON	119h	DACCON1	199h	RCREG	219h	—	299h	—	319h	—	399h	—
01Ah	TMR2	09Ah	OSCSTAT	11Ah	SRCON0	19Ah	TXREG	21Ah	—	29Ah	—	31Ah	—	39Ah	CLKRCON
01Bh	PR2	09Bh	ADRESL	11Bh	SRCON1	19Bh	SPBRGL	21Bh	—	29Bh	—	31Bh	—	39Bh	—
01Ch	T2CON	09Ch	ADRESH	11Ch	—	19Ch	SPBRGH	21Ch	—	29Ch	—	31Ch	—	39Ch	MDCON
01Dh	—	09Dh	ADCON0	11Dh	APFCON	19Dh	RCSTA	21Dh	—	29Dh	—	31Dh	—	39Dh	MDSRC
01Eh	CPSCON0	09Eh	ADCON1	11Eh	—	19Eh	TXSTA	21Eh	—	29Eh	—	31Eh	—	39Eh	MDCARL
01Fh	CPSCON1	09Fh	—	11Fh	—	19Fh	BAUDCON	21Fh	—	29Fh	—	31Fh	—	39Fh	MDCARH
020h–06Fh	General Purpose Register 80 Bytes	0A0h–0EFh	General Purpose Register 80 Bytes	120h–16Fh	General Purpose Register 80 Bytes	1A0h–1EFh	Unimplemented Read as '0'	220h–26Fh	Unimplemented Read as '0'	2A0h–2EFh	Unimplemented Read as '0'	320h–36Fh	Unimplemented Read as '0'	3A0h–3EFh	Unimplemented Read as '0'
070h–07Fh	Common RAM	0F0h–0FFh	Accesses 70h – 7Fh	170h–17Fh	Accesses 70h – 7Fh	1F0h–1FFh	Accesses 70h – 7Fh	270h–27Fh	Accesses 70h – 7Fh	2F0h–2FFh	Accesses 70h – 7Fh	370h–37Fh	Accesses 70h – 7Fh	3F0h–3FFh	Accesses 70h – 7Fh

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

Note 1: Available only on PIC12F1840.

TABLE 3-3: PIC12(L)F1840 MEMORY MAP (CONTINUED)

Bank	Core Registers (Table 3-2)	Unimplemented Read as ‘0’	Common RAM (Accesses 70h – 7Fh)
BANK 8	400h – 40Bh	40Ch – 46Fh	470h – 47Fh
BANK 9	480h – 48Bh	48Ch – 4EFh	4F0h – 4FFh
BANK 10	500h – 50Bh	50Ch – 56Fh	570h – 57Fh
BANK 11	580h – 58Bh	58Ch – 5EFh	5F0h – 5FFh
BANK 12	600h – 60Bh	60Ch – 66Fh	670h – 67Fh
BANK 13	680h – 68Bh	68Ch – 6EFh	6F0h – 6FFh
BANK 14	700h – 70Bh	70Ch – 76Fh	770h – 77Fh
BANK 15	780h – 78Bh	78Ch – 7EFh	7F0h – 7FFh
BANK 16	800h – 80Bh	80Ch – 86Fh	870h – 87Fh
BANK 17	880h – 88Bh	88Ch – 8EFh	8F0h – 8FFh
BANK 18	900h – 90Bh	90Ch – 96Fh	970h – 97Fh
BANK 19	980h – 98Bh	98Ch – 9EFh	9F0h – 9FFh
BANK 20	A00h – A0Bh	A0Ch – A6Fh	A70h – A7Fh
BANK 21	A80h – A8Bh	A8Ch – AEFh	AF0h – AFFh
BANK 22	B00h – B0Bh	B0Ch – B6Fh	B70h – B7Fh
BANK 23	B80h – B8Bh	B8Ch – BEFh	BF0h – BFFh
BANK 24	C00h – C0Bh	C0Ch – C6Fh	C70h – C7Fh
BANK 25	C80h – C8Bh	C8Ch – CEFh	CF0h – CFFh
BANK 26	D00h – D0Bh	D0Ch – D6Fh	D70h – D7Fh
BANK 27	D80h – D8Bh	D8Ch – DEFh	DF0h – DFFh
BANK 28	E00h – E0Bh	E0Ch – E6Fh	E70h – E7Fh
BANK 29	E80h – E8Bh	E8Ch – EEFh	EF0h – EFFh
BANK 30	F00h – F0Bh	F0Ch – F6Fh	F70h – F7Fh

Legend: ░ = Unimplemented data memory locations, read as ‘0’

TABLE 3-4: PIC12(L)F1840 MEMORY MAP, BANK 31

Address	Bank 31
FA0h–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'.

3.3.5 CORE FUNCTION REGISTERS SUMMARY

The Core Function registers listed in Table 3-5 can be addressed from any Bank.

TABLE 3-5: CORE FUNCTION REGISTERS SUMMARY

Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
Bank 0-31											
x00h or x80h	INDF0	Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register)								xxxx xxxx	uuuu uuuu
x01h or x81h	INDF1	Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register)								xxxx xxxx	uuuu uuuu
x02h or x82h	PCL	Program Counter (PC) Least Significant Byte								0000 0000	0000 0000
x03h or x83h	STATUS	—	—	—	$\overline{TO}$	$\overline{PD}$	Z	DC	C	---1 1000	---q quuu
x04h or x84h	FSR0L	Indirect Data Memory Address 0 Low Pointer								0000 0000	uuuu uuuu
x05h or x85h	FSR0H	Indirect Data Memory Address 0 High Pointer								0000 0000	0000 0000
x06h or x86h	FSR1L	Indirect Data Memory Address 1 Low Pointer								0000 0000	uuuu uuuu
x07h or x87h	FSR1H	Indirect Data Memory Address 1 High Pointer								0000 0000	0000 0000
x08h or x88h	BSR	—	—	—	BSR4	BSR3	BSR2	BSR1	BSR0	---0 0000	---0 0000
x09h or x89h	WREG	Working Register								0000 0000	uuuu uuuu
x0Ah or x8Ah	PCLATH	—	Write Buffer for the upper 7 bits of the Program Counter							-000 0000	-000 0000
x0Bh or x8Bh	INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	0000 0000	0000 0000

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

TABLE 3-6: SPECIAL FUNCTION REGISTER SUMMARY

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
Bank 0											
00Ch	PORTA	—	—	RA5	RA4	RA3	RA2	RA1	RA0	--xx xxxx	--xx xxxx
00Dh to 010h	—	Unimplemented								—	—
011h	PIR1	TMR1GIF	ADIF	RCIF	TXIF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
012h	PIR2	OSFIF	—	C1IF	EEIF	BCL1IF	—	—	—	0-00 0---	0-00 0---
013h	—	Unimplemented								—	—
014h	—	Unimplemented								—	—
015h	TMR0	Timer0 Module Register								xxxx xxxx	uuuu uuuu
016h	TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
017h	TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
018h	T1CON	TMR1CS1	TMR1CS0	T1CKPS<1:0>		T1OSCEN	T1SYNC	—	TMR1ON	0000 00-0	uuuu uu-u
019h	T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/DONE	T1GVAL	T1GSS<1:0>		0000 0x00	uuuu uxuu
01Ah	TMR2	Timer2 Module Register								0000 0000	0000 0000
01Bh	PR2	Timer2 Period Register								1111 1111	1111 1111
01Ch	T2CON	—	T2OUTPS<3:0>				TMR2ON	T2CKPS<1:0>		-000 0000	-000 0000
01Dh	—	Unimplemented								—	—
01Eh	CPSCON0	CPSON	CPSRM	—	—	CPSRNG<1:0>		CPSOUT	T0XCS	00-- 0000	00-- 0000
01Fh	CPSCON1	—	—	—	—	—	—	CPSCH<1:0>		---- --00	---- --00
Bank 1											
08Ch	TRISA	—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	--11 1111	--11 1111
08Dh to 090h	—	Unimplemented								—	—
091h	PIE1	TMR1GIE	ADIE	RCIE	TXIE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
092h	PIE2	OSFIE	—	C1IE	EEIE	BCL1IE	—	—	—	0-00 0---	0-00 0---
093h	—	Unimplemented								—	—
094h	—	Unimplemented								—	—
095h	OPTION_REG	WPUEN	INTEDG	TMR0CS	TMR0SE	PSA	PS<2:0>			1111 1111	1111 1111
096h	PCON	STKOVF	STKUNF	—	—	RMCLR	RI	POR	BOR	00-- 11qq	qq-- qquu
097h	WDTCON	—	—	WDTPS<4:0>					SWDTEN	--01 0110	--01 0110
098h	OSCTUNE	—	—	TUN<5:0>						--00 0000	--00 0000
099h	OSCCON	SPLLEN	IRCF<3:0>				—	SCS<1:0>		0011 1-00	0011 1-00
09Ah	OSCSTAT	T1OSCR	PLLR	OSTS	HFIOFR	HFIOFL	MFIOFR	LFIOFR	HFIOFS	10q0 0q00	qqqq qq0q
09Bh	ADRESL	ADC Result Register Low								xxxx xxxx	uuuu uuuu
09Ch	ADRESH	ADC Result Register High								xxxx xxxx	uuuu uuuu
09Dh	ADCON0	—	CHS<4:0>					GO/DONE	ADON	-000 0000	-000 0000
09Eh	ADCON1	ADFM	ADCS<2:0>			—	—	ADPREF<1:0>		0000 --00	0000 --00
09Fh	—	Unimplemented								—	—

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, r = reserved.
Shaded locations are unimplemented, read as ‘0’.

Note 1: These registers can be addressed from any bank.
2: PIC12F1840 only.
3: Unimplemented, read as ‘1’.

TABLE 3-6: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
Bank 2											
10Ch	LATA	—	—	LATA5	LATA4	—	LATA2	LATA1	LATA0	`--xx -xxx`	`--uu -uuu`
10Dh to 110h	—	Unimplemented								—	—
111h	CM1CON0	C1ON	C1OUT	C1OE	C1POL	—	C1SP	C1HYS	C1SYNC	`0000 -100`	`0000 -100`
112h	CM1CON1	C1INTP	C1INTN	C1PCH<1:0>		—	—	—	C1NCH	`0000 ---0`	`0000 ---0`
113h	—	Unimplemented								—	—
114h	—	Unimplemented								—	—
115h	CMOUT	—	—	—	—	—	—	—	MC1OUT	`---- ---0`	`---- ---0`
116h	BORCON	SBOREN	BORFS	—	—	—	—	—	BORRDY	`10-- ---q`	`uu-- ---u`
117h	FVRCON	FVREN	FVRRDY	TSEN	TSRNG	CDAFVR<1:0>		ADFVR<1:0>		`0q00 0000`	`0q00 0000`
118h	DACCON0	DACEN	DACLPS	DACOE	—	DACPSS<1:0>		—	—	`000- 00--`	`000- 00--`
119h	DACCON1	—	—	—	DACR<4:0>					`---0 0000`	`---0 0000`
11Ah	SRCON0	SRLEN	SRCLK<2:0>			SRQEN	SRNQEN	SRPS	SRPR	`0000 0000`	`0000 0000`
11Bh	SRCON1	SRSPE	SRSCKE	Reserved	SRSC1E	SRRPE	SRRCKE	Reserved	SRRC1E	`0000 0000`	`0000 0000`
11Ch	—	Unimplemented								—	—
11Dh	APFCON	RXDTSEL	SDOSEL	SSSEL	---	T1GSEL	TXCKSEL	P1BSEL	CCP1SEL	`000- 0000`	`000- 0000`
11Eh	—	Unimplemented								—	—
11Fh	—	Unimplemented								—	—
Bank 3											
18Ch	ANSELA	—	—	—	ANSA4	—	ANSA2	ANSA1	ANSA0	`---1 -111`	`---1 -111`
18Dh to 190h	—	Unimplemented								—	—
191h	EEADRL	EEPROM/Program Memory Address Register Low Byte								`0000 0000`	`0000 0000`
192h	EEADRH	—[3]	EEPROM / Program Memory Address Register High Byte							`1000 0000`	`1000 0000`
193h	EEDATL	EEPROM/Program Memory Read Data Register Low Byte								`xxxx xxxx`	`uuuu uuuu`
194h	EEDATH	—	—	EEPROM / Program Memory Read Data Register High Byte						`--xx xxxx`	`--uu uuuu`
195h	EECON1	EEPGD	CFGS	LWLO	FREE	WRERR	WREN	WR	RD	`0000 x000`	`0000 q000`
196h	EECON2	EEPROM control register 2								`0000 0000`	`0000 0000`
197h	VREGCON[2]	—	—	—	—	—	—	VREGPM	Reserved	`---- --01`	`---- --01`
198h	—	Unimplemented								—	—
199h	RCREG	USART Receive Data Register								`0000 0000`	`0000 0000`
19Ah	TXREG	USART Transmit Data Register								`0000 0000`	`0000 0000`
19Bh	SPBRGL	Baud Rate Generator Data Register Low								`0000 0000`	`0000 0000`
19Ch	SPBRGH	Baud Rate Generator Data Register High								`0000 0000`	`0000 0000`
19Dh	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	`0000 000x`	`0000 000x`
19Eh	TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	`0000 0010`	`0000 0010`
19Fh	BAUDCON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	`01-0 0-00`	`01-0 0-00`

Legend: `x` = unknown, `u` = unchanged, `q` = value depends on condition, - = unimplemented, `r` = reserved.
Shaded locations are unimplemented, read as ‘`0`’.

Note 1: These registers can be addressed from any bank.
2: PIC12F1840 only.
3: Unimplemented, read as ‘`1`’.

TABLE 3-6: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
Bank 4											
20Ch	WPUA	—	—	WPUA5	WPUA4	WPUA3	WPUA2	WPUA1	WPUA0	`--11 1111`	`--11 1111`
20Dh to 210h	—	Unimplemented								—	—
211h	SSP1BUF	Synchronous Serial Port Receive Buffer/Transmit Register								`xxxx xxxx`	`uuuu uuuu`
212h	SSP1ADD	ADD<7:0>								`0000 0000`	`0000 0000`
213h	SSP1MSK	MSK<7:0>								`1111 1111`	`1111 1111`
214h	SSP1STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	`0000 0000`	`0000 0000`
215h	SSP1CON1	WCOL	SSP1OV	SSP1EN	CKP	SSP1M<3:0>				`0000 0000`	`0000 0000`
216h	SSP1CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	`0000 0000`	`0000 0000`
217h	SSP1CON3	ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN	`0000 0000`	`0000 0000`
218h to 21Fh	—	Unimplemented								—	—
Bank 5											
28Ch to 290h	—	Unimplemented								—	—
291h	CCPR1L	Capture/Compare/PWM Register 1 (LSB)								`xxxx xxxx`	`uuuu uuuu`
292h	CCPR1H	Capture/Compare/PWM Register 1 (MSB)								`xxxx xxxx`	`uuuu uuuu`
293h	CCP1CON	P1M<1:0>		DC1B<1:0>		CCP1M<3:0>				`0000 0000`	`0000 0000`
294h	PWM1CON	P1RSEN	P1DC<6:0>							`0000 0000`	`0000 0000`
295h	CCP1AS	CCP1ASE	CCP1AS<2:0>			PSS1AC<1:0>		PSS1BD<1:0>		`0000 0000`	`0000 0000`
296h	PSTR1CON	—	—	—	STR1SYNC	Reserved	Reserved	STR1B	STR1A	`---0 rr01`	`---0 rr01`
297h to 29Fh	—	Unimplemented								—	—
Bank 6											
30Ch to 31Fh	—	Unimplemented								—	—
Bank 7											
38Ch to 390h	—	Unimplemented								—	—
391h	IOCAP	—	—	IOCAP5	IOCAP4	IOCAP3	IOCAP2	IOCAP1	IOCAP0	`--00 0000`	`--00 0000`
392h	IOCAN	—	—	IOCAN5	IOCAN4	IOCAN3	IOCAN2	IOCAN1	IOCAN0	`--00 0000`	`--00 0000`
393h	IOCAF	—	—	IOCAF5	IOCAF4	IOCAF3	IOCAF2	IOCAF1	IOCAF0	`--00 0000`	`--00 0000`
394h to 399h	—	Unimplemented								—	—
39Ah	CLKRCON	CLKREN	CLKROE	CLKRSLR	CLKRDC<1:0>		CLKRDIV<2:0>			`0011 0000`	`0011 0000`
39Bh	—	Unimplemented								—	—
39Ch	MDCON	MDEN	MDOE	MDSLR	MDOPOL	MDOUT	—	—	MDBIT	`0010 ---0`	`0010 ---0`
39Dh	MDSRC	MDMSODIS	—	—	—	MDMS<3:0>				`x--- xxxx`	`u--- uuuu`
39Eh	MDCARL	MDCLODIS	MDCLPOL	MDCLSYNC	—	MDCL<3:0>				`xxx- xxxx`	`uuu- uuuu`
39Fh	MDCARH	MDCHODIS	MDCHPOL	MDCHSYNC	—	MDCH<3:0>				`xxx- xxxx`	`uuu- uuuu`

Legend: `x` = unknown, `u` = unchanged, `q` = value depends on condition, - = unimplemented, `r` = reserved.
Shaded locations are unimplemented, read as '`0`'.

Note 1: These registers can be addressed from any bank.
2: PIC12F1840 only.
3: Unimplemented, read as '`1`'.

TABLE 3-6: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
Banks 8-30											
x0Ch/ x8Ch — x1Fh/ x9Fh	—	Unimplemented								—	—
Bank 31											
F8Ch — FE3h	—	Unimplemented								—	—
FE4h	STATUS_ SHAD	—	—	—	—	—	Z_SHAD	DC_SHAD	C_SHAD	`---- -xxx`	`---- -uuu`
FE5h	WREG_ SHAD	Working Register Shadow								`0000 0000`	`uuuu uuuu`
FE6h	BSR_ SHAD	—	—	—	Bank Select Register Shadow					`---x xxxx`	`---u uuuu`
FE7h	PCLATH_ SHAD	—	Program Counter Latch High Register Shadow							`-xxx xxxx`	`uuuu uuuu`
FE8h	FSR0L_ SHAD	Indirect Data Memory Address 0 Low Pointer Shadow								`xxxx xxxx`	`uuuu uuuu`
FE9h	FSR0H_ SHAD	Indirect Data Memory Address 0 High Pointer Shadow								`xxxx xxxx`	`uuuu uuuu`
FEAh	FSR1L_ SHAD	Indirect Data Memory Address 1 Low Pointer Shadow								`xxxx xxxx`	`uuuu uuuu`
FEBh	FSR1H_ SHAD	Indirect Data Memory Address 1 High Pointer Shadow								`xxxx xxxx`	`uuuu uuuu`
FECh	—	Unimplemented								—	—
FEDh	STKPTR	—	—	—	Current Stack Pointer					`---1 1111`	`---1 1111`
FEEh	TOSL	Top-of-Stack Low byte								`xxxx xxxx`	`uuuu uuuu`
FEFh	TOSH	—	Top-of-Stack High byte							`-xxx xxxx`	`-uuu uuuu`

Legend: `x` = unknown, `u` = unchanged, `q` = value depends on condition, - = unimplemented, `r` = reserved. Shaded locations are unimplemented, read as ‘`0`’.

Note 1: These registers can be addressed from any bank.

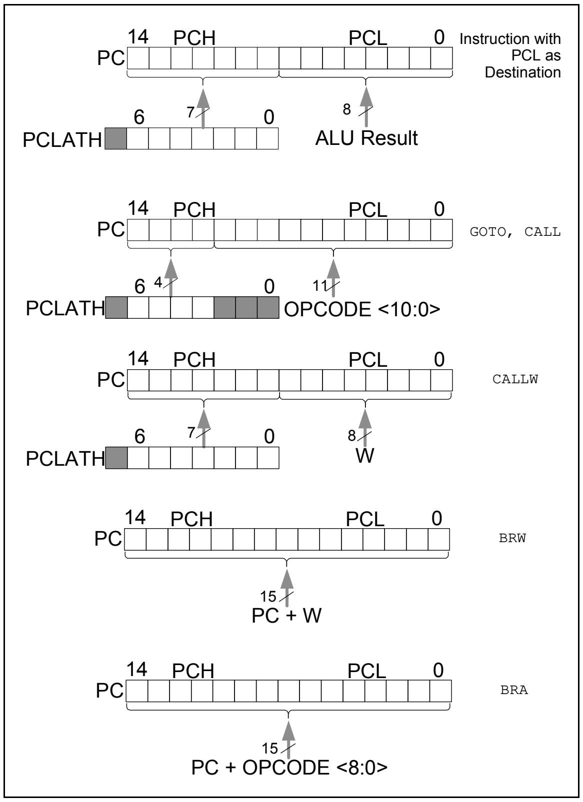
2: PIC12F1840 only.

3: Unimplemented, read as ‘`1`’.

3.4 PCL and PCLATH

The Program Counter (PC) is 15 bits wide. The low byte comes from the PCL register, which is a readable and writable register. The high byte (PC<14:8>) is not directly readable or writable and comes from PCLATH. On any Reset, the PC is cleared. Figure 3-3 shows the five situations for the loading of the PC.

FIGURE 3-3: LOADING OF PC IN DIFFERENT SITUATIONS



3.4.1 MODIFYING PCL

Executing any instruction with the PCL register as the destination simultaneously causes the Program Counter PC<14:8> bits (PCH) to be replaced by the contents of the PCLATH register. This allows the entire contents of the program counter to be changed by writing the desired upper seven bits to the PCLATH register. When the lower eight bits are written to the PCL register, all 15 bits of the program counter will change to the values contained in the PCLATH register and those being written to the PCL register.

3.4.2 COMPUTED GOTO

A computed `GOTO` is accomplished by adding an offset to the program counter (`ADDWF PCL`). When performing a table read using a computed `GOTO` method, care should be exercised if the table location crosses a PCL memory boundary (each 256-byte block). Refer to Application Note AN556, *"Implementing a Table Read"* (DS00556).

3.4.3 COMPUTED FUNCTION CALLS

A computed function `CALL` allows programs to maintain tables of functions and provide another way to execute state machines or look-up tables. When performing a table read using a computed function `CALL`, care should be exercised if the table location crosses a PCL memory boundary (each 256-byte block).

If using the `CALL` instruction, the PCH<2:0> and PCL registers are loaded with the operand of the `CALL` instruction. PCH<6:3> is loaded with PCLATH<6:3>.

The `CALLW` instruction enables computed calls by combining PCLATH and W to form the destination address. A computed `CALLW` is accomplished by loading the W register with the desired address and executing `CALLW`. The PCL register is loaded with the value of W and PCH is loaded with PCLATH.

3.4.4 BRANCHING

The branching instructions add an offset to the PC. This allows relocatable code and code that crosses page boundaries. There are two forms of branching, `BRW` and `BRA`. The PC will have incremented to fetch the next instruction in both cases. When using either branching instruction, a PCL memory boundary may be crossed.

If using `BRW`, load the W register with the desired unsigned address and execute `BRW`. The entire PC will be loaded with the address PC + 1 + W.

If using `BRA`, the entire PC will be loaded with PC + 1 +, the signed value of the operand of the `BRA` instruction.