dsPIC30F

dsPIC™ High-Performance 16-bit Digital Signal Controller Family Overview

High Performance CPU:

- C-compiler optimized instruction set architecture
- 94 Base instructions
  - Flexible addressing modes
- Linear program memory addressing up to 4M x 24-bit
- Linear data memory up to 64K bytes
- Up to 144K bytes on-chip FLASH program memory
  - 48K single word instructions (initially)
- Up to 8K bytes on-chip data RAM
- Up to 4K bytes EEPROM
- Two 40-bit wide accumulators with optional saturation logic
- 16 x 16-bit working register array
- Up to 30 MIPs operation:
  - DC - 120 MHz clock input
  - 4 MHz - 10 MHz osc./clock input with PLL active (4X, 8X, 16X)
- 24-bit wide instructions, 16-bit wide data path
- Dual Address Generation Units enabling dual data fetch for DSP operations
- Up to 32 interrupt sources
- 15 Exception Vectors (8 interrupts & 7 Traps)
  - Programmable Priority levels for 8 interrupts
  - 3 cycle fixed latency; 1 “fast” at 1 cycle
- 16 x 16 Single Cycle Hardware Fractional/Integer Multiplier
- Single Cycle Multiply-Accumulate (MAC) operation
- 40 stage Barrel Shifter

Peripheral Features:

- High current sink/source I/O pins 25 mA/25 mA
- Multiple external interrupt pins
- Timer module:
  - Five 16-bit timers/counters
  - 4 of the timers may be optionally configured as two 32-bit timer/counter
- 32 kHz real-time clock support on Timer1
- Capture Input functions (16-bit, up to 8 pins)
- Compare / PWM outputs functions (up to 8 pins)
  - 16-bit, max resolution 33.3 ns (TCY)
  - Dual Compare mode available
- Motor control PWM module
- Quadrature encoder module
- Data Converter Interface (DCI), supports common audio CODEC protocols
  - Including I^2S, AC’97
- 3-wire SPI™ modules (Supports all 4 SPI modes)
- I^2C™ module (supports full multi - master / slave mode and 7-bit/10-bit addressing)
- Addressable UART modules: Supports Interrupt on Address bit and Wake-up on Start Bit Detection
- CAN Bus modules
- As many as 54 programmable digital I/O pins
  - Some with interrupt on change

Advanced Analog Features:

- 10-Bit Analog-to-Digital Converters (A/D) with:
  - 16 input channels, typically
  - 500 ksps conversion rate
  - Conversion available during sleep
- 12-Bit Analog-to-Digital Converters (A/D) with:
  - 16 input channels, typically
  - 100 ksps conversion rate
  - Conversion available during sleep
- Programmable Low Voltage detection (LVD)
  - Supports interrupt on low voltage detection
- Programmable Brown-out Reset generation

Special Microcontroller Features:

- Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Fail safe clock monitor operation
- Programmable code protection
- Selectable Power Management modes
  - SLEEP mode, IDLE mode, SLOWDOWN mode

- Selectable oscillator options, including:
  - 4X/8X/16X Phase Lock Loop (of primary oscillator)
  - Secondary Oscillator (32 kHz) clock input (Timer1)
  - High speed internal RC oscillator
- In-Circuit Serial Programming™ (ICSP™) via 3 pins and power/ground

CMOS Technology:

- Low-power, high-speed FLASH technology
- Fully static design
- Wide operating voltage range (2.5V to 5.5V)
- Industrial and extended temperature ranges
- Low power consumption

Packaging:

- 100-pin TQFP
- 64-pin TQFP
- 40-pin DIP, 44-pin TQFP
- 28-pin DIP (300 mil.), 28-pin SSOP

1.0 CPU CORE ARCHITECTURAL DESCRIPTION

The dsPIC30F Digital Signal Controller is a modified Harvard Architecture core with a 16-bit datapath and a 24-bit wide instruction memory. The dsPIC30F core seamlessly integrates the superior control attributes of a 16-bit MCU and the computation power of a DSP.

The dsPIC30F instruction set adds many enhancements to the previous PICMicro Microcontroller (MCU) instruction sets, while maintaining an easy migration path from these PICMicro MCU platforms.

1.1 Core Overview

The core has a 24-bit instruction word, with a variable length opcode field. The PC (program counter) is 23 bits wide (with the LS-bit always clear, see Figure 1-3 and Table 1-1), addressing up to 4M long words (24 bits). An PIC18C-like instruction prefetch mechanism is used to help maintain throughput. Deeper levels of pipelining have been intentionally avoided to maintain good real-time performance. Unconditional overhead free program loop constructs are supported using the DO and REPEAT instructions, both of which are interruptable at any point.

The working register array is comprised of 16 x 16-bit registers, each of which can act as data, address or offset registers. One working register (W15) operates as the software stack pointer for interrupts and calls.

The data space is 32K words of word or byte addressable space, which is split into two blocks referred to as X and Y data memory. Each block has its own independent Address Generation Unit (AGU). Most instructions operate solely through the X memory AGU which will make it appear as one linear space encompassing all data space (X and Y). The MAC class of DSP instructions will operate through both the X and Y AGUs, splitting the data address space into two parts (see Section 1.2.1). The X and Y data space boundary is arbitrary and defined through the address decode of each memory array.

The upper 32K bytes of data space memory can optionally be mapped into program space at any 16K program word boundary defined by the 8-bit Data Space Program PAGE (DSPPAG) register. This lets any instruction access program space as if it were data space (other than the additional access cycle it consumes), plus it allows external RAM hooked onto the external program space bus to be mapped into data space, effectively providing an external data space path.

Overhead free circular buffers (modulo addressing) are supported in both X and Y address spaces. They are intended to remove the loop overhead for DSP algorithms, but X modulo addressing can be universally applied using any instructions.

The X AGU also supports bit reverse addressing to greatly simplify input or output data reordering for radix-2 FFT algorithms.

The Instruction Set Architecture (ISA) has been significantly enhanced beyond that of the PIC18C, but maintains an acceptable level of backward compatibility. All PIC18C instructions and addressing modes are supported either directly or through simple macros. Many of the ISA enhancements have been driven by compiler efficiency needs (see Section 1.1.1).

The core supports inherent (no operand), relative, literal, memory direct and 3 groups of addressing modes (MODE1, MODE2 and MODE3) for register direct and register indirect modes. There are 11 addressing modes in total, plus some special varients for DSP instruction. Instructions are associated with predefined addressing modes depending upon their functional requirements. Please refer to the Instruction Set Description document [DS70026n_C] for more details.

For most instructions, the core is capable of executing a data (or program data) memory read, a working register (data) read, a data memory write and a program (instruction) memory read per instruction cycle. As a result, 3 operand instructions can be supported, allowing A+B=C operations to be executed in a single cycle.

A DSP engine has been included to significantly enhance the core arithmetic capability and throughput. It features a high speed 16-bit by 16-bit multiplier, a 40-bit ALU, two 40-bit saturating accumulators and a 40-bit bi-directional barrel shifter. The barrel shifter is capable of shifting a 40-bit value up to 15 bits right or up to 16 bits left in a single cycle. The DSP instructions operate seamlessly with all other instructions and have been designed for optimal real-time performance. The MAC class of instructions can concurrently fetch two data operands from memory while multiplying two W registers and accumulating the results. This requires that the data space be split for these instructions and linear for all others. This is achieved in a transparent and flexible manner through dedicating certain working registers to each address space for the MAC class of instructions.

The core features a sophisticated interrupt structure with 15 individually prioritized vectors. The interrupts and exceptions consist of reset, 7 traps and 8 interrupts. Up to 32 interrupt sources are supported. One interrupt level may be selected (typically the highest one) to execute as a fast (1 cycle entry, 1 cycle exit) interrupt. This function is actually an extension of the logic required to allow a REPEAT instruction loop to be interrupted, which can significantly reduce latency in some application.

A block diagram of the core is shown in Figure 1-1.

1.1.1 COMPILER DRIVEN ENHANCEMENTS

In addition to DSP performance requirements, the core architecture was strongly influenced by recommendations which would lead to a more efficient (code size and speed) C compiler.

1. For most instructions, the core is capable of executing a data (or program data) memory read, a working register (data) read, a data memory write and a program (instruction) memory read per instruction cycle. As a result, 3 operand instructions can be supported, allowing A+B=C operations to be executed in a single cycle.
2. Instruction addressing modes are extremely flexible to meet compiler needs.
3. The working register array is comprised of 16 x 16-bit registers, each of which can act as data, address or offset registers. One working register (W15) operates as the software stack pointer for interrupts and calls.
4. Linear indirect access of all data space is possible, plus the memory direct address range has been extended to 8K bytes. This, together with the addition of 16-bit direct address LOAD and STORE instructions, has provided a contiguous linear addressing space.
5. Linear indirect access of 32K word (64K byte) pages within program space is possible using any working register via new table read and write instructions.
6. Part of data space can be mapped into program space, allowing constant data to be accessed as if it were in data space.

1.1.2 INSTRUCTION FETCH MECHANISM

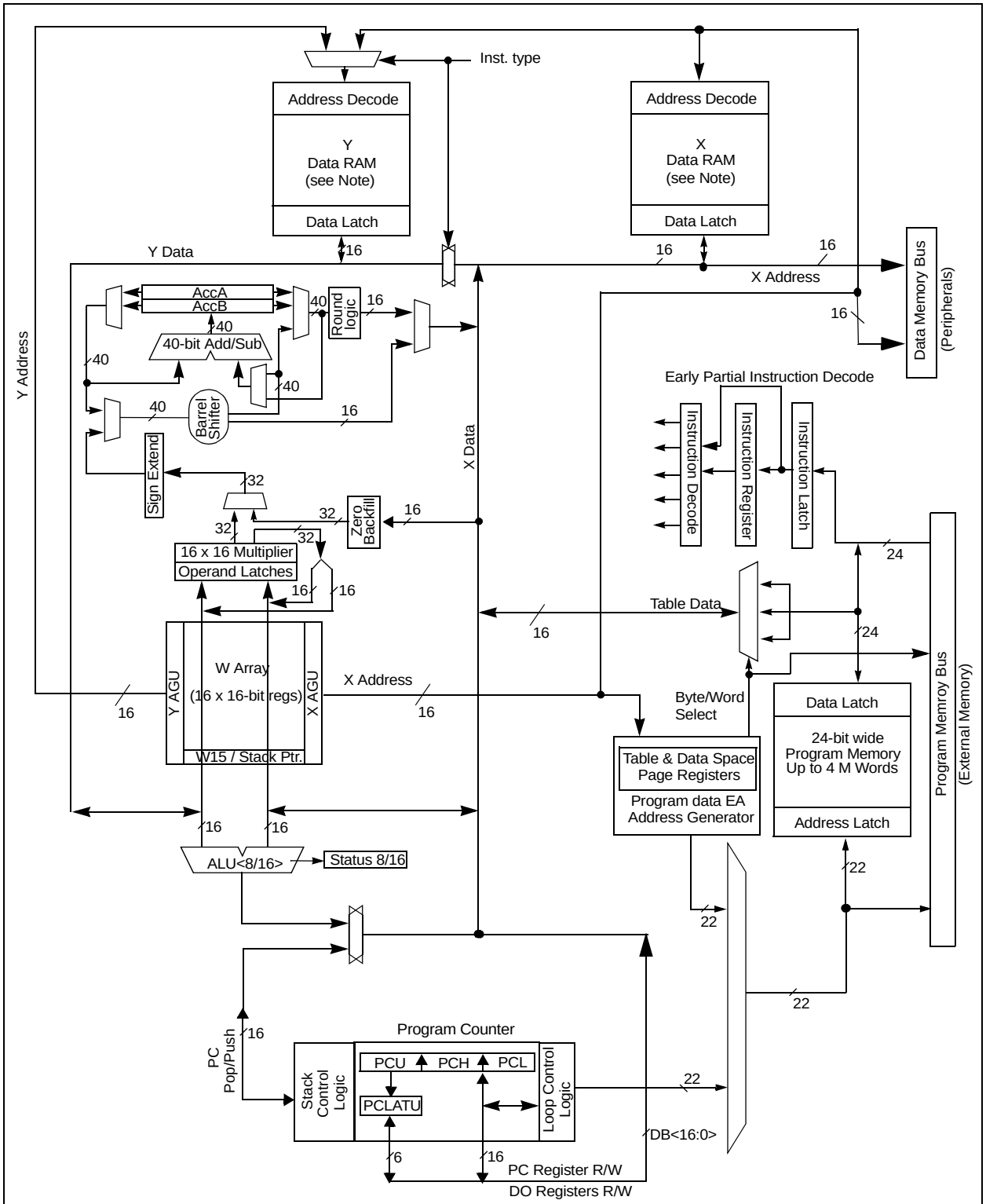
A one-stage pre-fetching mechanism accesses each instruction a cycle ahead to maximize available execution time. Most instructions execute in a single cycle. Exceptions are:

1. Flow control instructions (such as program Branches, Calls, Returns) take 2 cycles since the IR (instruction register) and pre-fetch buffer must be flushed and refilled.
2. Instructions where one operand is to be fetched from program space (using any method). These operations consume 2 cycles (with the notable exception of the MAC class of DSP instructions executed within a REPEAT loop which executes in 1 cycle).

Most instructions access data as required during instruction execution. Instructions which utilize the multiplier array must have data available at the beginning of the instruction cycle. Consequently, this data must be prefetched, usually by the preceding instruction, resulting in a simple out of order data processing model.

A programmer model diagram is shown in Figure 1-2.

FIGURE 1-1: CPU CORE BLOCK DIAGRAM



Note: The RAM is logically separated in two sections for DSP operations to allow fetching of two variables in one cycle. However, for all MCU operations, the RAM appears as one contiguous space.

FIGURE 1-2: PROGRAMMER MODEL DIAGRAM

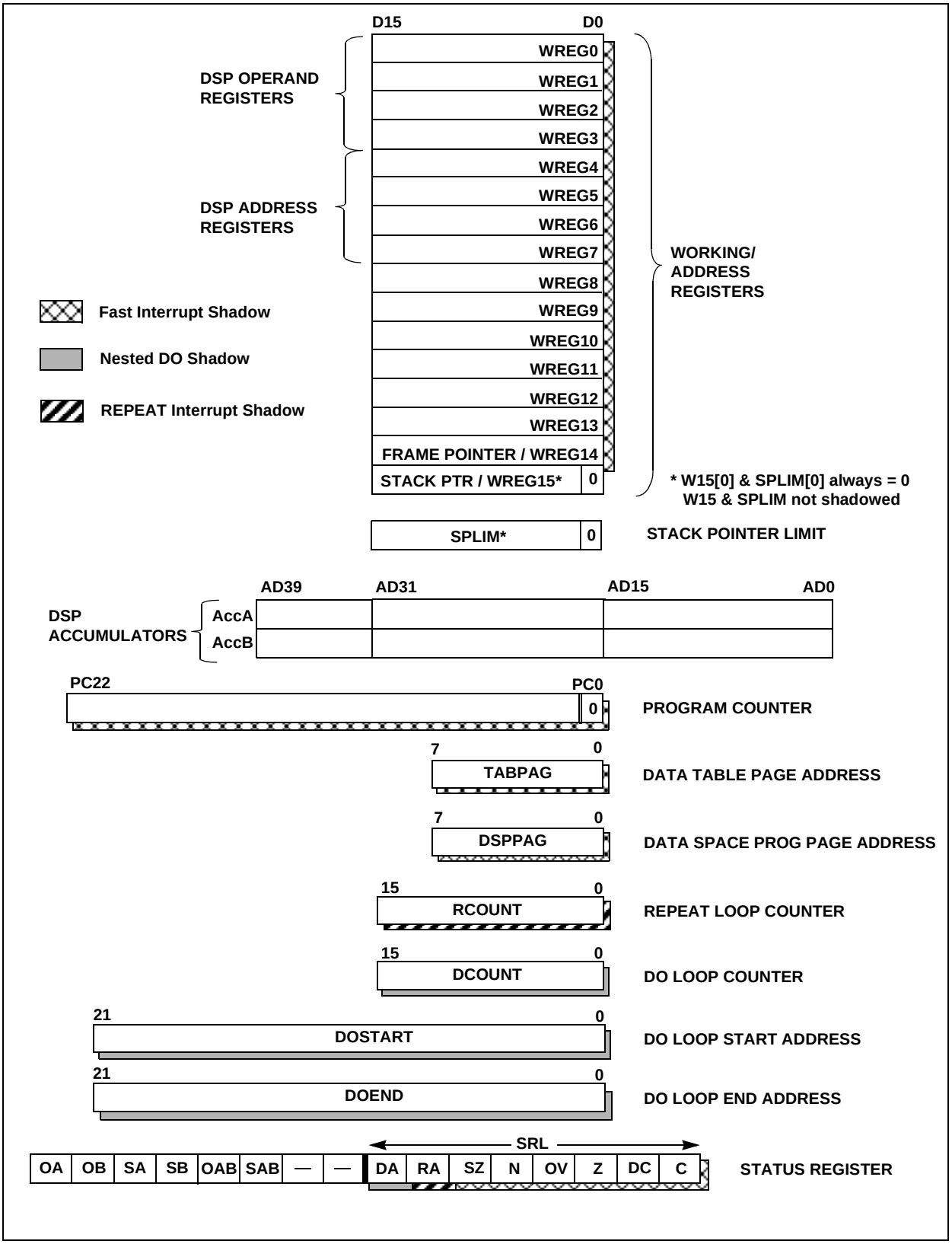
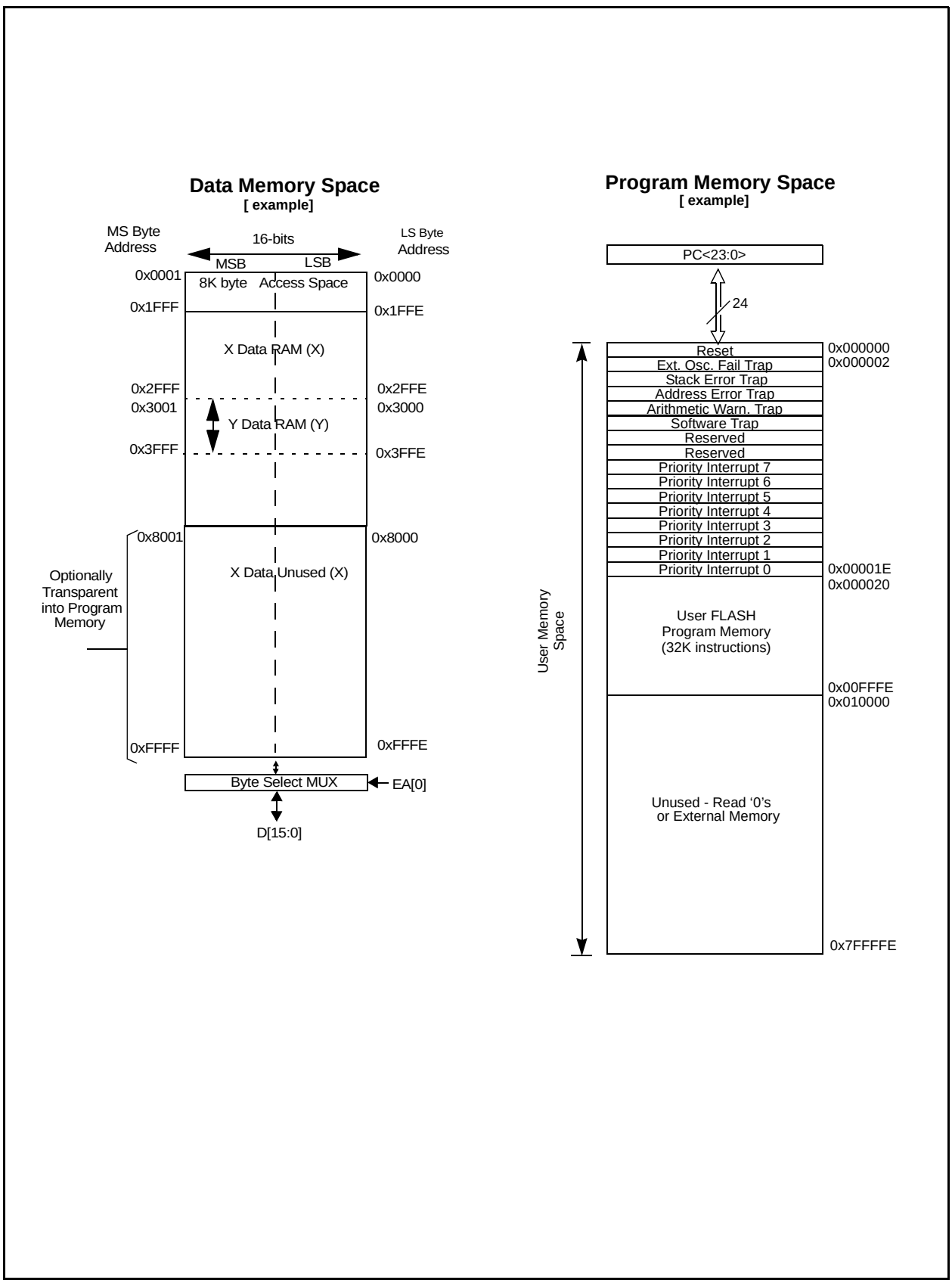


FIGURE 1-3: MEMORY MAP DIAGRAM



1.2 Data Address Space

The core features one program space and two data spaces. The data spaces can be considered either separately (for some DSP instructions) or together as one linear address range (for MCU instructions). The data spaces are accessed using two Address Generation Units (AGUs) and separate data paths.

1.2.1 DATA SPACES

The X AGU is used by all instructions and supports all addressing modes. It also supports modulo and bit reversed addressing for any instruction (subject to addressing mode restrictions). The X data path is the return data path for all single data space access instructions.

The Y AGU and data path are used in concert with the X AGU by the MAC class of instructions to provide two concurrent data read paths. No writes occur across the Y-bus. This class of instructions dedicate two W register pointers, W6 and W7, to always operate through the Y AGU and address Y data space independently from X data space. Note that during accumulator write to Data Space, the data address space is considered combined X and Y, so the write will occur across the X-bus. Consequently, it can be to any address irrespective of where the EA is directed.

The Y AGU only supports post modification addressing modes associated with the MAC class of instructions. It also supports modulo addressing for automated circular buffers. Of course, all other instructions can access the Y data address space through the X AGU when it is regarded as part of the composite linear space.

The boundary between the X and Y data spaces is arbitrary and is defined by the memory address decode only (the CPU has no knowledge of the physical location of X or Y memory). The boundary is not user programable, but may change from variant to varient. Obviously, to present a linear data space to the MCU instructions, the address spaces of X and Y data spaces must be contiguous, but this is not an architectural necessity.

All effective addresses (EA) are 16 bits wide and point to bytes within the data space to facilitate backward compatibility with the PIC18C. Consequently, the data space address range is 64K bytes or 32K words.

1.2.2 DATA SPACE WIDTH

The core data width is 16 bits. All internal registers and data space memory are organized as 16 bits wide (some CPU registers are not 16 bits wide). Data space memory is organized in byte addressable, 16-bit wide blocks.

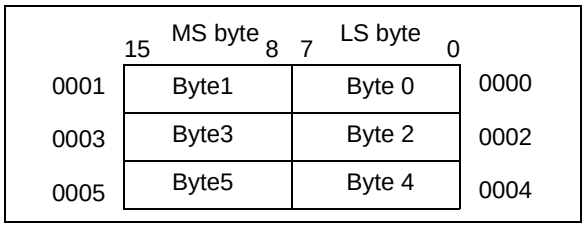
1.2.3 DATA ALIGNMENT

To help maintain PIC18C backward compatibility and improve data space memory usage efficiency, the DSC Core supports both word and byte operations, by way of an instruction attribute. Data is aligned in data memory and registers as words, but all data space EAs resolve to bytes (see Figure 1-4). Data byte reads will read the complete word which contains the byte, using the LS-bit of any EA to determine which byte to select. The selected byte is placed onto the LS-byte of the X data path (no byte accesses are possible from the Y data path as the MAC class of instruction can only fetch words). That is, data memory and registers are organized as two parallel byte wide entities with shared (word) address decode but separate write lines. Data byte writes will only write to the corresponding side of the array or register which matches the byte address. For word accesses, the LS-bit of the EA is ignored (don't care).

As a consequence of this byte accessibility, all effective address calculations (including those generated by the DSP operations which are restricted to word size) are automatically scaled to step through word aligned memory. For example, the core recognizes that post modified register indirect addressing mode, [Ws]+=1, will result in a value of Ws+1 for byte operations and Ws+2 for word operations.

All word accesses must be aligned (to an even address). Misaligned word data fetches are not supported, so care must therefore be taken when mixing byte and word operations or translating from PIC18C code. Should a misaligned read or write be attempted, an address fault trap will be forced.

FIGURE 1-4: DATA ALIGNMENT



All byte loads into any W register are loaded into the LS-byte. The MS-byte is not modified.

Note: Byte operations use the 16-bit ALU and can produce results in excess of 8 bits. However, to maintain PIC18C backwards compatibility, the ALU result from all byte operations is written back as a byte (i.e., MS byte not modified), and the status register is updated based only upon the state of the LS-byte of the result.

A sign extend (SE) instruction is provided to allow users to translate 8-bit signed data to16-bit signed values. Alternatively, for 16-bit unsigned data, users can clear the MS-byte of any W register though executing a CLR.b instruction on the appropriate address. (All CPU core registers are memory mapped into data space).

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

1.3 Program Address Space

The program address space is 4M long words. It is addressable by a 22-bit value from either the PC, table instruction EA or data space EA when program space is mapped into data space as defined by Table 1-1. Note that the program space address is incremented by two between successive program words in order to provide compatibility with data space addressing. Consequently, the LS-bit of the program space address is always 0, resulting in 22 bits of address. Program space data accesses use the LS-bit of the program space address as a byte select (same as data space)

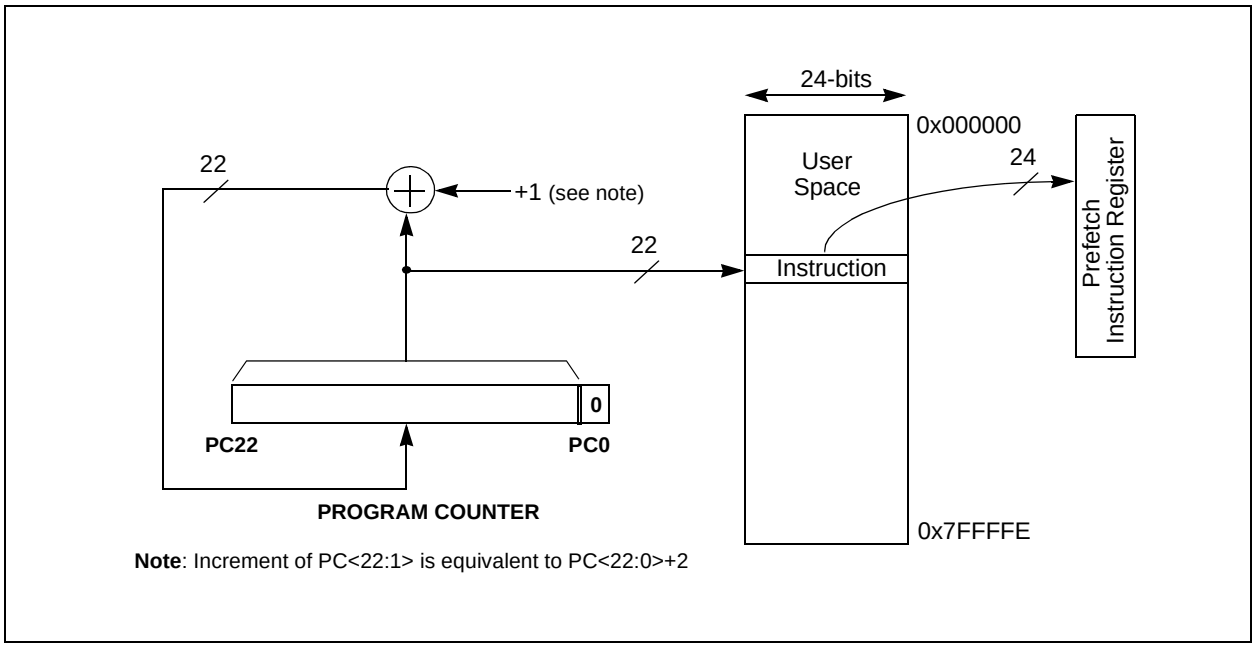
TABLE 1-1: PROGRAM SPACE ADDRESS CONSTRUCTION

Access Type	Program Space Address			
	[22:16]	[15]	[14:1]	[0]
Instruction Access	PC[22:1]			0
TBLRD/TBLWT	TABPAG[6:0]	Data EA [15:0]		
DS Window into PS	DSPPAG[7:0]		Data EA [14:0]	

The program memory width is 24 bits (long word). To support data storage and FLASH programming, the array must support both word wide access from bits 0-15 and byte wide access from bits 16-23.

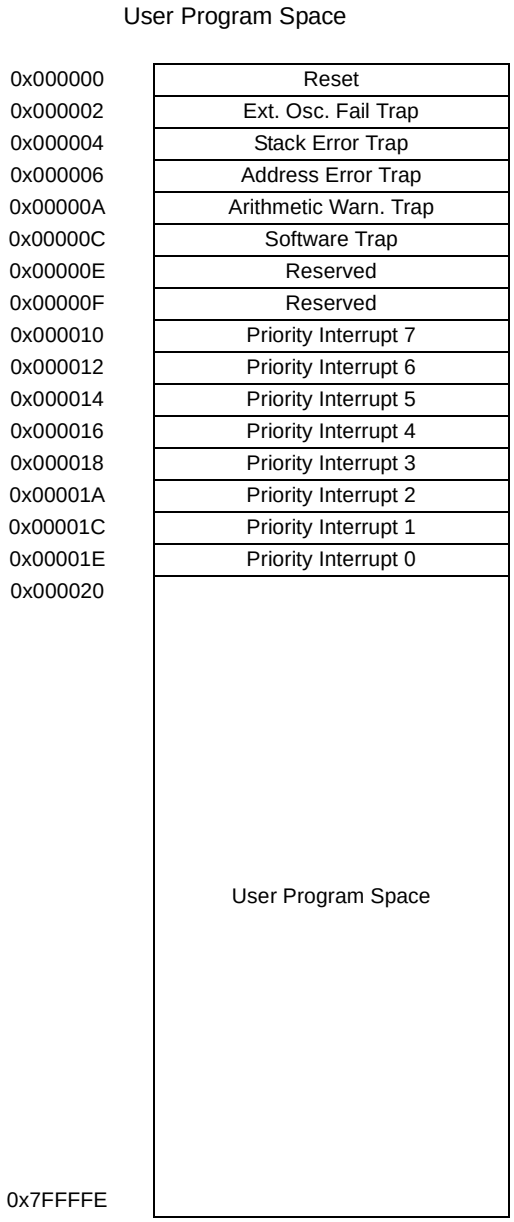
See Figure 1-5 and Figure 1-6 for program space addressing conventions.

FIGURE 1-5: INSTRUCTION FETCH EXAMPLE



Note: Increment of PC<22:1> is equivalent to PC<22:0>+2

FIGURE 1-6: PROGRAM SPACE MEMORY MAP



1.4 DSP Engine

The DSP engine is a block of hardware which is fed data from the W register array, but contains its own specialized result registers. It is controlled from the same single issue instruction decoder that directs the MCU ALU. In addition, all operand effective addresses are generated in the W register array. Some DSP instructions (e.g., ED and EDAC instructions) utilize both the DSP engine and the MCU ALU resources concurrently. The DSP engine consists of a high speed 16-bit x 16-bit multiplier, a barrel shifter and a 40-bit adder/subtractor with two target registers, round and saturation logic.

Data input to the DSP engine is derived from:

1. Directly the W array (registers W0, W1, W2 or W3) for the MAC class of instructions (MAC, MSA, MPY, MPYN, SQR, SQRAC, CLRAC and MOVSAC) and MCU multiply instructions.
2. The X-bus for all other DSP instructions
3. The X-bus for all MCU instructions which use the barrel shifter

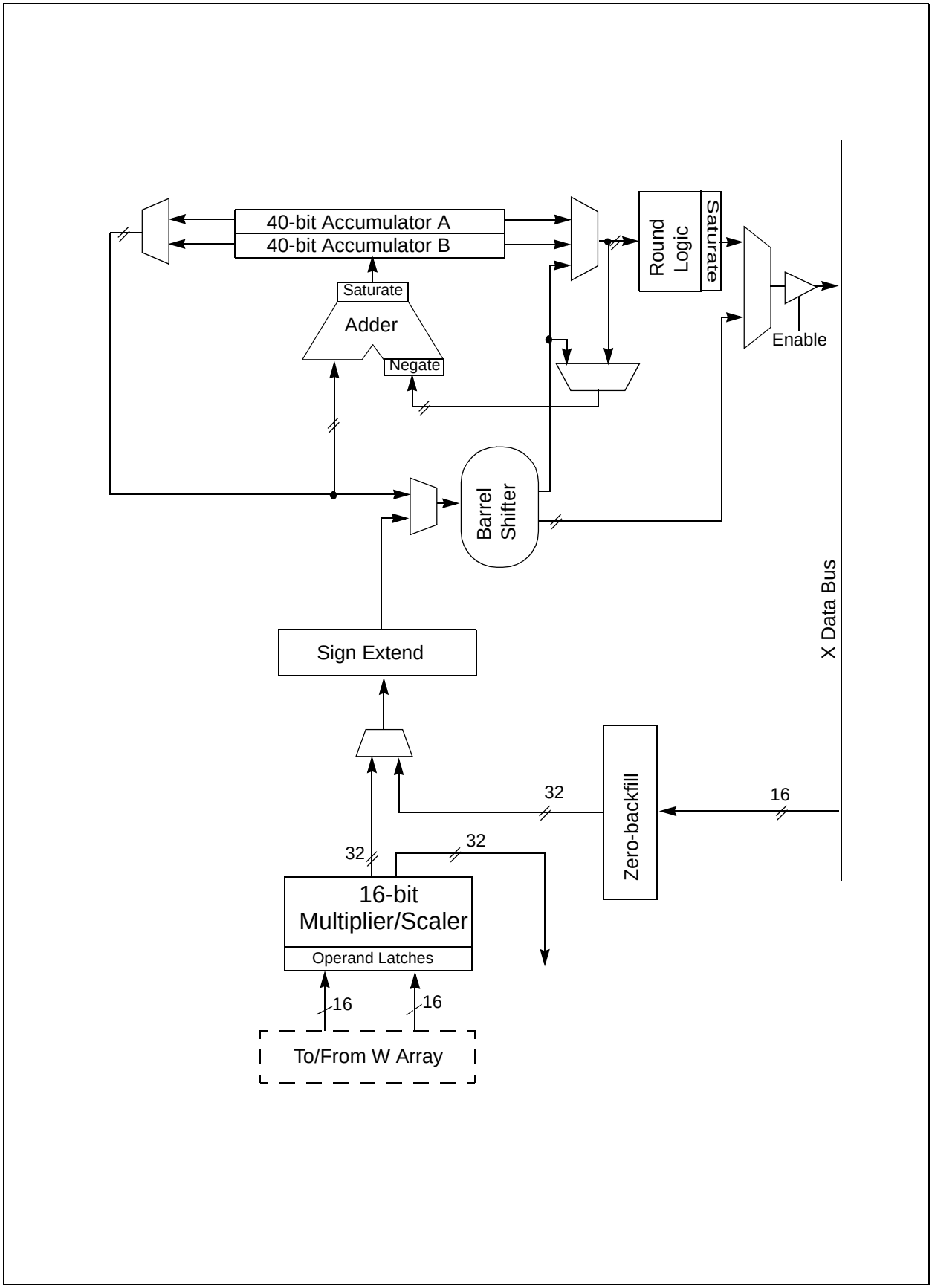
Data output from the DSP engine is written to:

1. The target accumulator, as defined by the DSP instruction being executed
2. The X-bus for MAC, MSA, CLRAC and MOVSAC accumulator writes where the EA is derived from W9 only (MPY, MPYN, SQR and SQRAC do not offer an accumulator write option)
3. The X-bus for all MCU instructions which use the barrel shifter
4. The W array for some MCU multiply instructions.

The DSP engine also has the capability to perform inherent accumulator to accumulator operations which require no additional data. These instructions are ADDAB, SUBAB and NEGAC.

A block diagram of the DSP engine is shown in Figure 1-7.

FIGURE 1-7: DSP ENGINE BLOCK DIAGRAM



2.0 DEVELOPMENT TOOLS SUPPORT

Microchip is offering a comprehensive package of development tools and libraries to support the dsPIC architecture. In addition, the company is partnering with many third party tools manufacturers for additional dsPIC support. The Microchip tools will include:

- MPLAB® 6.00 Integrated Development Environment (IDE)
- dsPIC Language Suite including MPLAB C30 C Compiler, Assembler, Linker and Librarian
- MPLAB SIM Software Simulator
- MPLAB ICE 4000 In-Circuit Emulator
- MPLAB ICD 2 In-Circuit Debugger
- PRO MATE® II Universal Device Programmer
- PICSTART® Plus Development Programmer

2.1 MPLAB V6.00 Integrated Development Environment Software

The MPLAB Integrated Development Environment is available at no cost. The IDE gives users the flexibility to edit, compile and emulate all from a single user interface. Engineers can design and develop code for the dsPIC in the same design environment that they have used for PICmicro microcontrollers.

The MPLAB IDE is a 32-bit Windows® based application. It provides many advanced features for the critical engineer in a modern, easy to use interface. MPLAB integrates:

- Full featured, color coded text editor
- Easy to use project manager with visual display
- Source level debugging
- Enhanced source level debugging for ‘C’ (Structures, automatic variables, and so on)
- Customizable toolbar and key mapping
- Dynamic status bar displays processor condition at a glance
- Context sensitive, interactive on-line help
- Integrated MPLAB SIM instruction simulator
- User interface for PRO MATE II and PICSTART Plus device programmers (sold separately)
- User interface for MPLAB ICE 4000 In-Circuit Emulator (sold separately)
- User interface for MPLAB ICD 2 In-Circuit Debugger

The MPLAB IDE allows the engineer to:

- Edit your source files in either assembly or ‘C’
- One-touch compile and download to dsPIC program memory on emulator or simulator. Updates all project information.
- Debug using:
  - Source files
  - Machine code
  - Mixed-mode source and machine code

The ability to use the MPLAB IDE with multiple development and debugging targets allows users to easily switch from the cost-effective simulator to a full-featured emulator with minimal retraining.

2.2 dsPIC Language Suite

The Microchip Technology MPLAB C30 C compiler is a fully ANSI compliant product with standard libraries for the dsPIC architecture. It is highly optimizing and takes advantage of many dsPIC architecture specific features to provide efficient software code generation.

MPLAB C30 also provides extensions that allow for excellent support of the hardware such as interrupts and peripherals. It is fully integrated with the MPLAB IDE for high level, source debugging.

- 16-bit native data types
- Efficient use of register-based, 3-operand instructions
- Complex addressing modes
- Efficient multi-bit shift operations
- Efficient signed/unsigned comparisons

MPLAB C30 comes complete with its own in assembler, linker and librarian. These allow the user to write mixed-mode C and assembly programs and link the resulting object files into a single executable file. The compiler is sold separately. The assembler, linker and librarian are available for free with MPLAB.

2.3 MPLAB SIM Software Simulator

The MPLAB SIM software simulator allows code development in a PC-hosted environment by simulating the dsPIC on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a file, or user-defined key press, to any of the pins.

The execution can be performed in single step, execute until break, or trace mode.

The MPLAB SIM simulator fully supports symbolic debugging using the MPLAB C30 compiler and assembler. The software simulator offers the flexibility to develop and debug code outside of the laboratory environment, making it an excellent multi-project software development tool.

2.4 MPLAB ICE 4000 In-Circuit Emulator

The MPLAB ICE 4000 In-Circuit Emulator is intended to provide the product development engineer with a complete hardware design tool for the dsPIC. Software control of the emulator is provided by MPLAB, allowing editing, building, downloading and source debugging from a single environment.

The MPLAB ICE 4000 is a full-featured emulator system with enhanced trace, trigger and data monitoring features. Interchangeable processor modules allow the system to be easily reconfigured for emulation of different processors.

The MPLAB ICE 4000 supports the extended, high end PICmicro microcontrollers, the 18CXXX and 18FXXX devices, as well as the dsPIC family of digital signal controllers. The modular architecture of the MPLAB ICE in-circuit emulator allows expansion to support new devices.

The MPLAB ICE in-circuit emulator system has been designed as a real-time emulation system, with advanced features that are generally found on more expensive development tools.

- Full-speed emulation, up to 50MHz bus speed, or 200MHz external clock speed
- Low-voltage emulation down to 1.8 volts
- Configured with 2Mb program emulation memory, additional modular memory up to 16Mb
- 32K x 136-bit wide Trace Memory
- Unlimited software breakpoints
- Complex break, trace and trigger logic
- Multi-level trigger up to 4 levels
- Filter trigger functions to trace specific event
- 16-bit Pass counter for triggering on sequential events
- 16-bit Delay counter
- 48-bit time stamp
- Stopwatch feature
- Time between events
- Statistical performance analysis
- Code coverage analysis
- USB and parallel printer port PC connection

2.5 MPLAB ICD 2 In-Circuit Debugger

Microchip's In-Circuit Debugger, MPLAB ICD, is a powerful, low cost, run-time development tool. This tool is based on the PICmicro and dsPIC FLASH devices

The MPLAB ICD utilizes the in-circuit debugging capability built into the various devices. This feature, along with Microchip's In-Circuit Serial Programming™ protocol, offers cost-effective in-circuit debugging from the graphical user interface of MPLAB. This enables a designer to develop and debug source code by watching variables, single-stepping and setting break points. Running at full speed enables testing hardware in real-time.

- Full speed operation to the range of the device
- Serial or USB PC connector
- Serial interface externally powered
- USB powered from PC interface
- Low-noise power (VPP and VDD) for use with analog and other noise sensitive applications
- Operation down to 2.0v
- Can be used as an ICD or in-expensive serial programmer
- Modular application connector as MPLAB-ICD
- Limited number of breakpoints
- "Smart watch" variable windows
- Some chip resources required (RAM, program memory and 2 pins)

2.6 PRO MATE II Universal Device Programmer

The PRO MATE II universal device programmer is a full-featured programmer, capable of operating in stand-alone mode, as well as PC-hosted mode. The PRO MATE II device programmer is CE compliant.

The PRO MATE II device programmer has programmable VDD and VPP supplies, which allow it to verify programmed memory at VDD min and VDD max for maximum reliability when programming requiring this capability. It has an LCD display for instructions and error messages, keys to enter commands. Interchangeable socket modules all package types.

In stand-alone mode, the PRO MATE II device programmer can read, verify, or program PICmicro devices. It can also set code protection in this mode.

- Runs under MPLAB
- Field upgradable firmware
- DOS Command Line interface for production
- Host, Safe, and “Stand Alone” operation
- Automatic downloading of object file
- SQTP serialization adds unique serial number to each device programmed
- In-Circuit Serial Programming Kit (sold separately)
- Interchangeable socket modules supports all package options (sold separately)

3.0 EXCEPTION PROCESSING

The dsPIC has 15 exception sources plus RESET, which are arbitrated based on a priority scheme.

Exceptions are either RESET, fixed priority non-maskable traps or user programmable priority interrupts. The exception priority table is shown in Figure 3-1. The interrupts are enabled, prioritized, and controlled using centralized special function registers.

All interrupt sources can be user assigned to one of 8 priority levels, 0 through 7. Each level is associated with an interrupt vector as shown in Figure 3-1. Level 6 and 0 represent the highest and lowest maskable priorities respectively. Level 7 interrupts are non-maskable and are handled slightly differently from all other interrupts.

Certain interrupts have specialized control bits for features like edge or level triggered interrupts, interrupt on change, etc. Control of these features remains within the peripheral module which generates the interrupt.

3.1 Interrupt Priority

The Interrupt Priority bits for each individual interrupt are located in bits within the Interrupt Priority Control registers (INTCON). These bits define the priority level assigned to a particular interrupt.

Multiple interrupts can be assigned the same priority. Once in the Interrupt Service Routine (ISR) for a particular priority level, the interrupt can be determined by polling the interrupt flag bits.

Each interrupt priority has a corresponding interrupt vector. When an interrupt is serviced, the PC is loaded with the interrupt vector that corresponds to the priority level of that interrupt. There are 8 different interrupt vectors (see Figure 3-1).

3.1.1 CPU PRIORITY REGISTER

The CPU Priority register is used to indicate the current priority of all pending interrupts and traps.

The initial (reset) state of the CPU Priority register is 0xFFFF. It contains one bit for each interrupt level 0 through 7 and trap priority level 8 through 14. It also contains one bit for RESET. When an interrupt or trap is being serviced, the priority status bit associated with the interrupt is cleared in this register. This register can also be used to disable higher priority interrupts than the one currently being serviced.

The Global Interrupt Enable (GIE) bit will disable all interrupt levels 0 through 6 when clear. Exceptions greater than priority level 6 are non-maskable so they cannot be disabled in software.

3.2 Exception Sequence

All interrupt event flags are sampled simultaneously and at a specific CPU clock phase. A pending interrupt indicated by the flag bit being equal to a '1' will cause the interrupt to occur. When an interrupt is latched, the interrupt priority bits associated with a pending flag are sampled in the next clock cycle before entering the ISR.

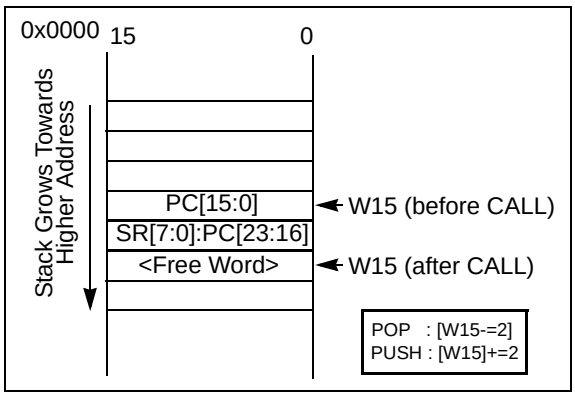
The sampling sequence is needed to determine if more than one interrupt flag, with different priorities, have been simultaneously latched. Each of the interrupt status bits is arbitrated simultaneously.

Each of the pending interrupts has an associated priority. The status of the pending interrupt is presented on one bit of an 8-bit Interrupt Request (IRQ) bus corresponding to one of 8 priorities. Each bit on the request bus indicates to the CPU that at least one interrupt of priority 'n' is present.

If the IRQ bits sampled indicate a priority lower than or equal to the current CPU priority, then no interrupt sequence will occur. When all higher (priority) status bits are set as a result of the termination of their respective ISR's, then the ISR of the pending status bit will be serviced.

When an interrupt is serviced, the return address is pushed onto the stack together with the least significant byte of the Status Register (SR) as shown in Figure 3-1. Working Register 15 is used as the implied stack pointer.

FIGURE 3-1: INTERRUPT STACK FRAME



If interrupt nesting is disabled, subsequent interrupts of priority level 0 through 6 are prevented from causing a further exception sequence. However, interrupts continue to be arbitrated and the CPU Priority register continues to be updated to reflect all subsequent interrupts which become pending. Individual interrupt flag bits within the IFS (Interrupt Flag Status) register(s) are set regardless of the status of the GIE and the individual Interrupt Priority bits. The GIE bit is also cleared on RESET. Note that traps and priority 7 interrupts are not disabled by the GIE bit and are always enabled.

If interrupt nesting is enabled, subsequent interrupts will be arbitrated and will clear the CPU Priority register bits accordingly. Should any of these be of a higher priority than that currently being serviced, an interrupt at that level will be initiated.

Note: Traps and priority 7 interrupts are always nestable.

3.2.1 INTERRUPT/TRAP/RESET VECTORS

Interrupt, trap and reset vectors are automatically loaded into the PC when servicing an interrupt, trap or following a RESET. The vectors are contained in locations 0x000000 through 0x00001F of program memory. These locations contain 24-bit addresses, and in order to preserve robustness, an address error trap will take place should the PC attempt to fetch any of these words during normal execution. This prevents execution of random data.

TABLE 3-1: EXCEPTION VECTOR TABLE

Reset Vector
Ext. Oscillator Fail Trap Vector
Stack Error Trap Vector
Address Error Trap Vector
Arithmetic Warning Trap Vector
Software Trap Vector
Reserved Vector
Reserved Vector
Priority 7 Interrupt Vector
Priority 6 Interrupt Vector
Priority 5 Interrupt Vector
Priority 4 Interrupt Vector
Priority 3 Interrupt Vector
Priority 2 Interrupt Vector
Priority 1 Interrupt Vector
Priority 0 Interrupt Vector

Traps can be considered as non-maskable, nestable interrupts which adhere to a predefined priority as shown in Table 3-1. They are intended to provide the user a means to correct erroneous operation during debug and when operating within the application. The software traps also provide a means to emulate new or unsupported instructions.

Note: If the user does not intend to take corrective action in the event of a trap error condition, these vectors must be loaded with the reset vector address.

Note that many of these trap conditions can only be detected when they happen. Consequently, the questionable instruction is allowed to complete prior to trap exception processing. If the user chooses to recover from the error, the result of the erroneous action which caused the trap may therefore have to be corrected.

3.2.1.1 RESET Sources

In addition to external and power-on resets, there are three sources of error conditions which will 'trap' to the reset vector. The possibility of recovery from these conditions is remote, so a hardware reset is the most robust course of action.

- Watchdog Time-out:
  The windowed watchdog has been reset too early or has timed out, indicating that the processor is no longer executing the correct flow of code.
- Illegal Instruction Trap:
  The dsPIC 8-bit opcode field must be fully decoded. Attempted execution of any unused slots will result in an illegal instruction trap. Note that a fetch of an illegal instruction will not result in an illegal instruction trap if that instruction is flushed prior to execution due to a flow change.
- Brown-out Detect:
  A momentary dip in the power supply to the device has been detected which may result in malfunction.

3.2.1.2 TRAP Sources

The following traps are provided with increasing priority. However, as all traps are nestable, priority has little effect.

- Software trap:
  Execution of a TRAP opcode will cause an interrupt.
- Arithmetic Error trap:
  The Arithmetic Error trap will execute under the following three circumstances. It is assumed that the DSP engine configuration will be consistent within an application, so polling flags to determine the error condition should not be necessary.

1. Should an attempt be made to divide by zero, the divide operation will be aborted on a cycle boundary and the trap taken.
2. If enabled, an Arithmetic Error trap will be taken when an arithmetic operation on either accumulator A or B causes an overflow from bit 31 and the accumulator guard bits are unutilized.
3. If enabled, an Arithmetic Error trap will be taken when an arithmetic operation on either accumulator A or B causes a catastrophic overflow from bit 39 and all saturation is disabled.

- Address Error Trap:
  This trap will be initiated when any of the following circumstances occurs:

1. A misaligned data word fetch is attempted
2. A data fetch from unimplemented data address space is attempted
3. A program fetch from unimplemented user program address space is attempted
4. A program fetch from vector address space is attempted
5. A read (for address) of an uninitialized W register is attempted

- Stack Error Trap
  This trap will be initiated under the following conditions:

1. The stack pointer is loaded with a value which is greater than the (user programmable) limit value written into the SPLIM register (stack overflow).
2. The stack pointer is loaded with a value which is less than 0x0200 (simple stack underflow).

- Oscillator Fail Trap:
  This trap will be initiated should the external oscillator fail and operation become reliant on an internal RC backup.

It is conceivable that multiple traps can become active within the same cycle (e.g., a misaligned word stack write to an overflowed address). In such a case, the fixed priority shown in Figure 3-1 will be implemented which may require the user to check if other traps are pending in order to completely correct the fault.

4.0 RESETS

The dsPIC30FXXX differentiates between various kinds of RESET:

a) Power-on Reset (POR)
b) $\overline{\text{MCLR}}$ Reset during normal operation
c) $\overline{\text{MCLR}}$ Reset during SLEEP
d) Watchdog Timer (WDT) Reset (during normal operation)
e) Programmable Brown-out Reset (PBOR)
f) RESET Instruction

Most registers are unaffected by a RESET. Their status is unknown on POR and unchanged by all other RESETs. The other registers are forced to a "RESET" state on Power-on Reset, $\overline{\text{MCLR}}$, WDT Reset, Brown-out Reset, $\overline{\text{MCLR}}$ Reset during SLEEP and by the RESET instruction.

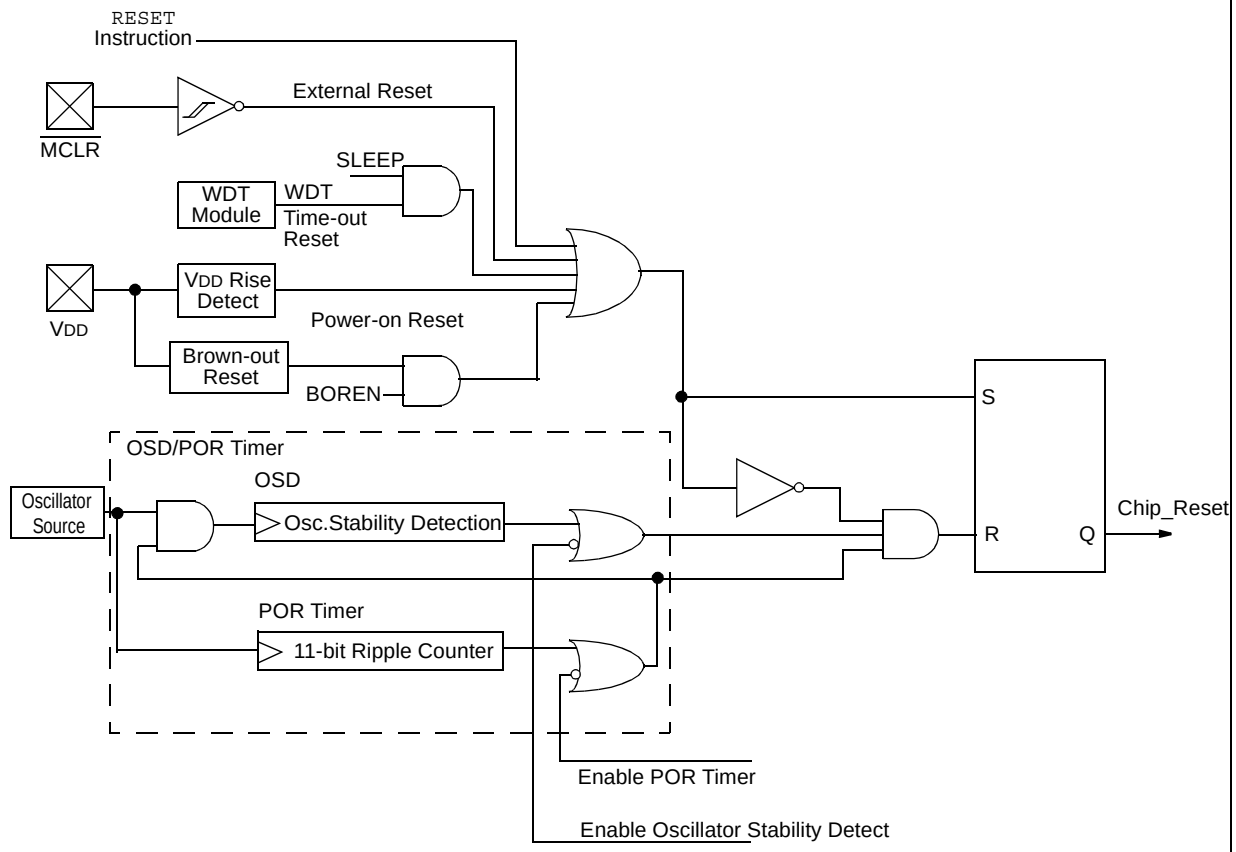
Most registers are not affected by a WDT wake-up, since this is viewed as the resumption of normal operation. Status bits from the Reset Condition register, are set or cleared differently in different RESET situations. These bits are used in software to determine the nature of the RESET.

A simplified block diagram of the on-chip RESET circuit is shown in Figure 4-1.

The dsPIC30F devices have a $\overline{\text{MCLR}}$ noise filter in the $\overline{\text{MCLR}}$ Reset path. The filter will detect and ignore small pulses.

A WDT Reset does not drive $\overline{\text{MCLR}}$ pin low.

FIGURE 4-1: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT



Note 1: This is a separate oscillator from the RC oscillator of the CLKI pin.

5.0 LOW VOLTAGE DETECT

In many applications, the ability to determine if the device voltage (V_{DD}) is below a specified voltage level is a desirable feature. A window of operation for the application can be created, where the application software can do "housekeeping tasks" before the device voltage exits the valid operating range. This can be done using the Low Voltage Detect (LVD) module.

This module contains software programmable circuitry, where a device voltage trip point can be specified (internal reference voltage). When the voltage of the device becomes lower than the specified point, an interrupt flag is set. If the LVD interrupt is enabled, the program execution will take a level 7 (interrupt) exception and take appropriate action.

The Low Voltage Detect circuitry is completely under software control. This allows the circuitry to be "turned off" by the software, which minimizes the current consumption for the device.

6.0 DSC PERIPHERALS

The Digital Signal Controller (DSC) family of 16-bit MCU devices will provide the integrated functionality of many peripheral functions. The initial library of functions that will be utilized (one or more) on the DSC devices are as follows:

- 10-bit high speed A/D converter
- 12-bit high resolution A/D converter
- General Purpose 16-bit timers
- Watchdog timer module
- Motor Control PWM module
- Quadrature Encoder module
- Input Capture module
- Output Compare/ PWM module
- Serial Peripheral Interface (SPI™) module
- UART module
- I^2C™ module
- Controller Area Network (CAN) module
- I/O pins

6.1 A/D Modules

There will be 2 versions of A/D converters available for dsPIC30F family of devices. There is a 10-bit high speed A/D module and a 12-bit high resolution A/D module.

6.1.1 10-BIT A/D FEATURES

- 10-bit resolution
- Uni-polar differential Inputs
- Up to 16 input channels
- Selectable reference inputs
- ±1 LSB max DNL
- ±2 LSB max INL
- Up to four on-chip sample and hold amplifiers
- Single supply operation: 2.7V - 5.5V
- 500KSPS sampling rate at 5V
- Ability to convert while the device sleeps
- Low power CMOS technology
- 5nA typical standby current, 2µA max
- 2.5 mA typical active current at 5V

6.1.2 12-BIT A/D FEATURES

- 12-bit resolution
- Uni-polar differential Inputs
- Up to 16 input channels
- Selectable reference inputs
- ±1 LSB max DNL
- ±2 LSB max INL
- Up to four on-chip sample and hold amplifiers
- Single supply operation: 2.7V - 5.5V
- 100KSPS sampling rate at 2.7V
- Ability to convert while the device sleeps
- Low power CMOS technology
- 5nA typical standby current, 2µA max
- 2.5 mA typical active current at 5V

6.1.3 APPLICATIONS

- DC Brushless Motor control
- SR motor control
- AC Induction Motor Control
- Remote Sensors
- Sensor Interface
- Process Control
- Data Acquisition

6.1.4 DESCRIPTION

The A/D modules provide up to 16 analog inputs with both single ended and differential inputs. These modules offer on-board sample and hold circuitry.

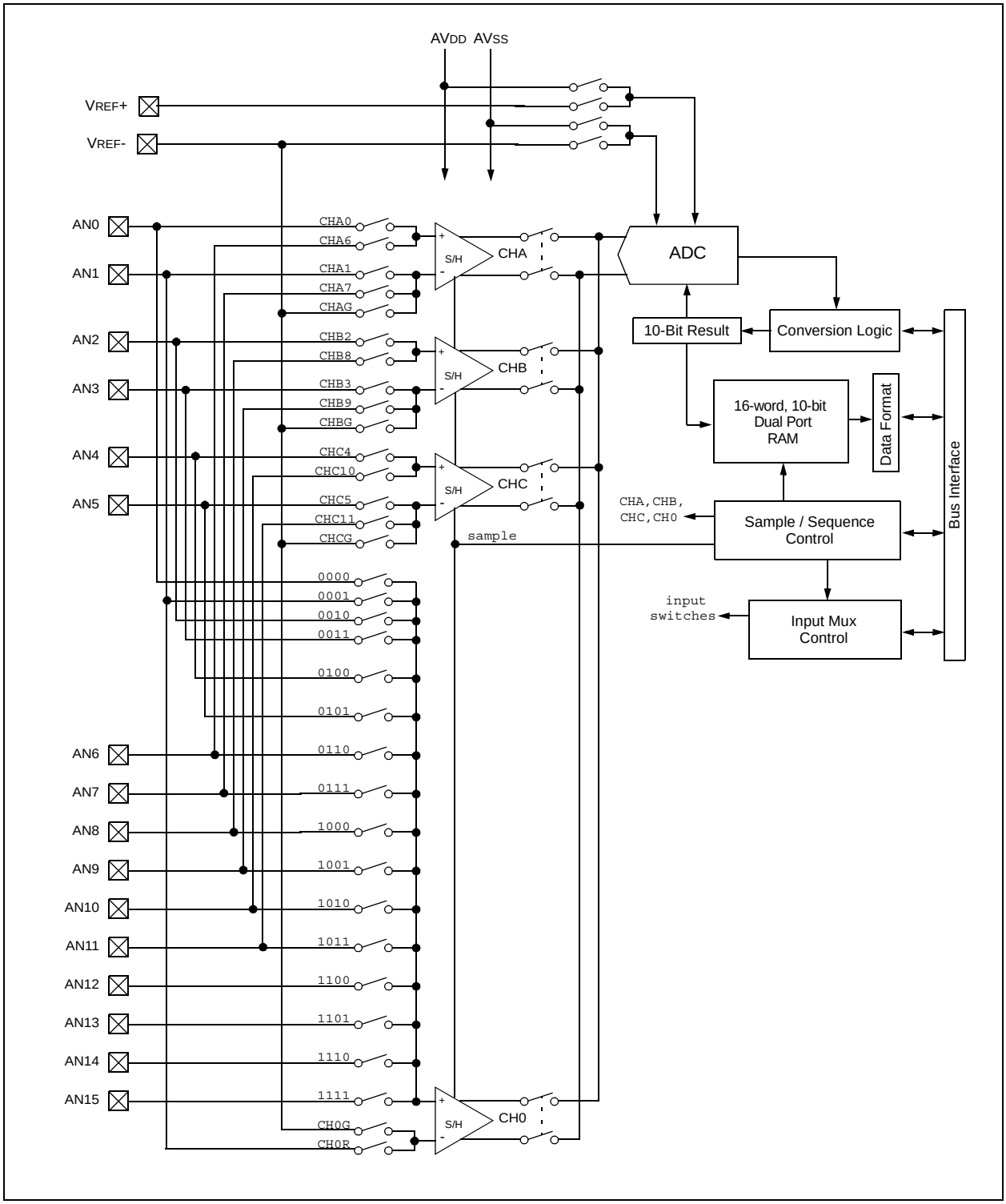
To minimize control loop errors due to finite update times (conversion plus computations), a high speed low latency ADC is required.

In addition, several hardware features have been added to the peripheral interface to improve real-time performance in a typical DSP based application.

3. Result alignment options
4. Automated sampling
5. Dual Port data buffer
6. External conversion start control

The block diagram of the A/D module is shown in Figure 6-1.

FIGURE 6-1: FUNCTIONAL BLOCK DIAGRAM



6.2 General Purpose Timer

The General Purpose (GP) Timer module provides the time base elements for Input Capture, Output Compare/PWM and can be configured for a Real-time clock operation as well as various timer/counter modes.

Figure 6-2 and Figure 6-3 depict the simplified block diagrams of the two GP Timer Modules.

The GP timer module consists of one 16-bit timer and one 32-bit timer, (which can be configured as two 16-bit timers), with selectable operating modes. These timers will be utilized by other dsPIC peripheral modules such as:

- Input Capture
- Output Compare
- Real-Time Clock

FIGURE 6-2: 16-BIT TIMER MODULE SIMPLIFIED BLOCK DIAGRAM

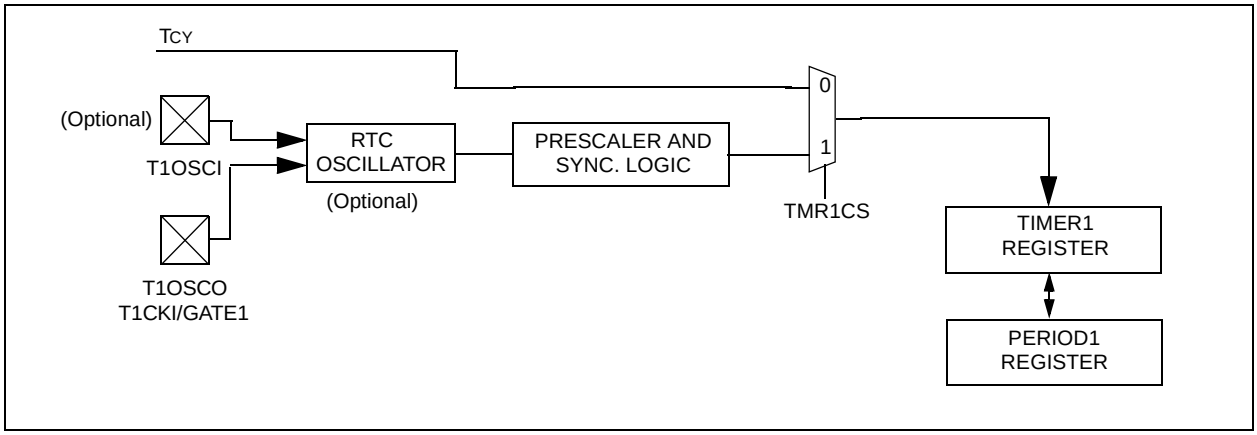
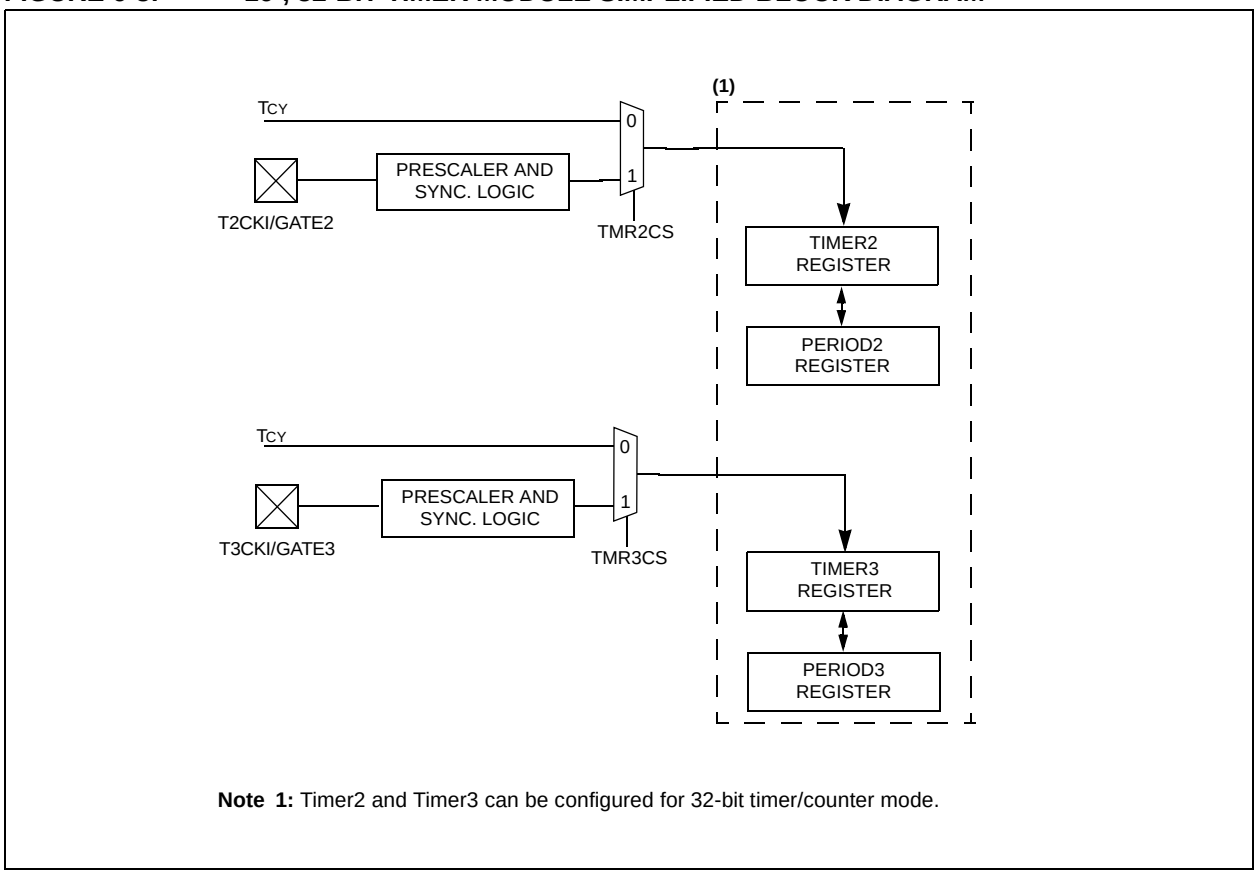


FIGURE 6-3: 16-, 32-BIT TIMER MODULE SIMPLIFIED BLOCK DIAGRAM



Note 1: Timer2 and Timer3 can be configured for 32-bit timer/counter mode.

6.3 Watchdog Timer Module

This is the description of Watchdog Timer (WDT) for the dsPIC30F family.

6.3.1 OVERVIEW

The primary function of the Watchdog Timer (WDT) is to reset the processor in the event of a software malfunction. The WDT is a free running timer which runs off an on-chip RC oscillator, requiring no external component. Therefore, the WDT timer will continue to operate even if the main processor clock (e.g., the crystal oscillator) fails.

6.3.2 ENABLING AND DISABLING THE WDT

The Watchdog timer can be “Enabled” or “Disabled” only through a configuration bit (WDTEN) in the Configuration Register.

WDTEN=1 enables the Watchdog timer. The enabling is done when programming the device. By default, after chip-erase, WDTEN bit =1. Any device programmer capable of programming dsPIC devices (such as Microchip’s PRO MATE® II and PICSTART® Plus programmers) allows programming of this and other configuration bits to the desired state.

If enabled, the WDT will increment until it overflows or “times out”. A WDT time-out will force a device reset (except during SLEEP). To prevent a WDT time-out, the user must clear the Watchdog timer using a CLRWDT instruction.

If a WDT times out during SLEEP, the device will wake up. The status bit will be cleared (“0”) to indicate a Wake-up resulting from WDT time-out

6.4 Motor Control PWM Module

This module simplifies the task of generating multiple, synchronized pulse width modulated (PWM) outputs. In particular, the following power and motion control applications are supported by the PWM module:

- Three-Phase AC Induction Motor
- Switched Reluctance (SR) Motor
- Brushless DC (BLDC) Motor
- Uninterruptable Power Supply (UPS)

6.4.1 FEATURES OVERVIEW

The PWM module has the following features:

- Up to 8 PWM I/O pins with 4 duty cycle generators
- Up to 16-bit resolution
- ‘On-the-Fly’ PWM frequency changes
- Edge and center aligned output modes
- Single-pulse generation mode
- Interrupt support for asymmetrical updates in center-aligned mode.
- Output override control for electrically commutated motor (ECM) operation
- ‘Special Event’ comparator for scheduling other peripheral events

A simplified block diagram of the PWM module is shown in Figure 6-4.

This module contains 4 duty cycle generators, numbered 1 through 4. The module has 8 PWM output pins, numbered 0 through 7. The eight I/O pins are grouped into odd numbered/even numbered pairs. For complementary loads, the even PWM pins must always be the complement of the corresponding odd I/O pin to prevent damage to the power transistor devices. Consequently, the signals on the even numbered I/O pins have certain limitations when the module is in the complementary operating mode.

6.4.2 PWM TIMEBASE

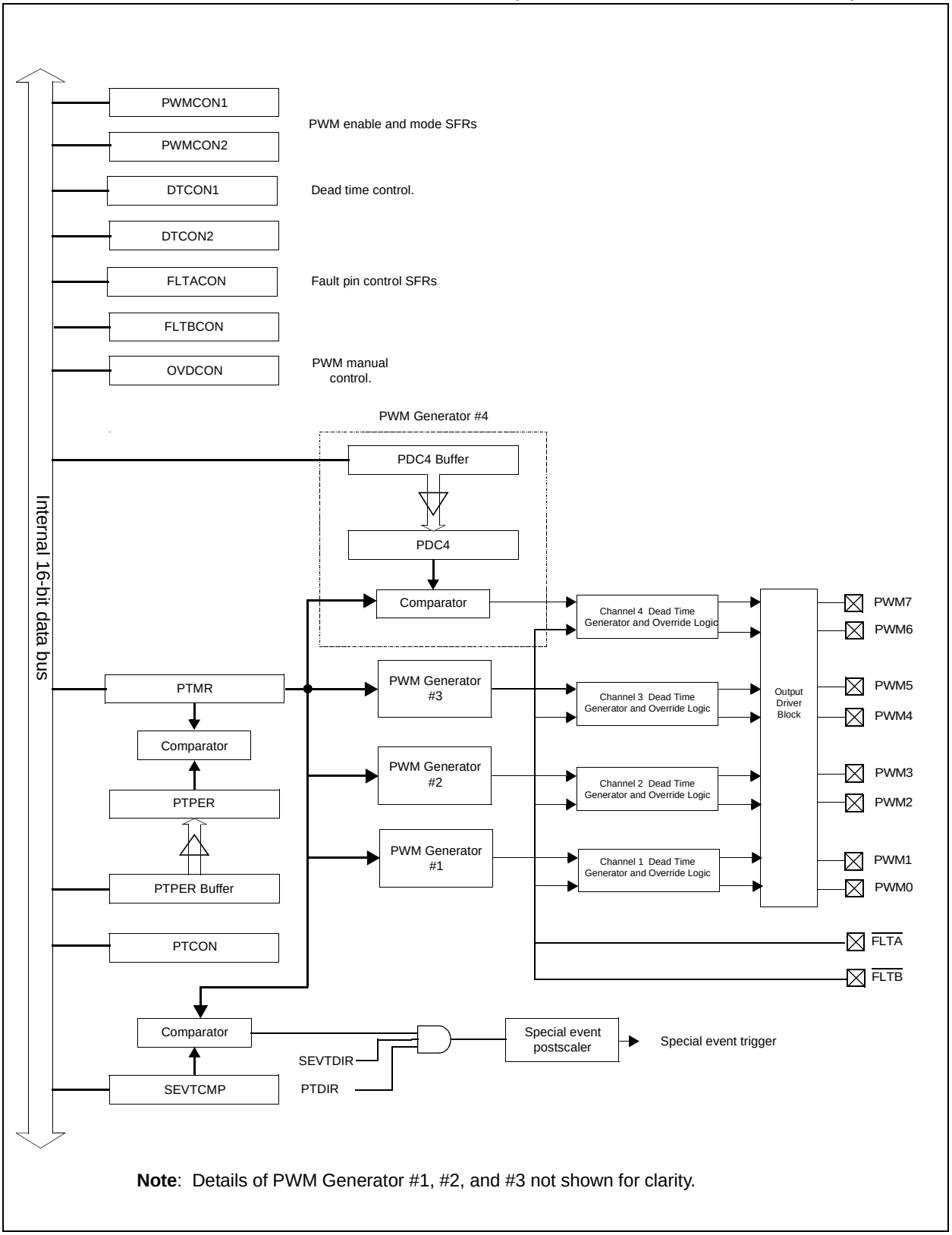
The PWM timebase is provided by a 16-bit timer with a prescaler and postscaler. The PWM timebase is configured via a special function register (SFR).

The PWM timebase can be configured for four different modes of operation:

- Free running mode
- Single-shot mode
- Continuous up/down count mode
- Continuous up/down count mode with interrupts for double-updates.

These four modes are selected by the PTMOD1:PTMOD0 bits in the PTCON SFR. The up/down counting modes support center-aligned PWM generation. The single-shot mode allows the PWM module to support pulse control of certain electronically commutated motors (ECMs).

FIGURE 6-4: PWM MODULE BLOCK DIAGRAM (FULL MODULE IMPLEMENTATION)



Note: Details of PWM Generator #1, #2, and #3 not shown for clarity.

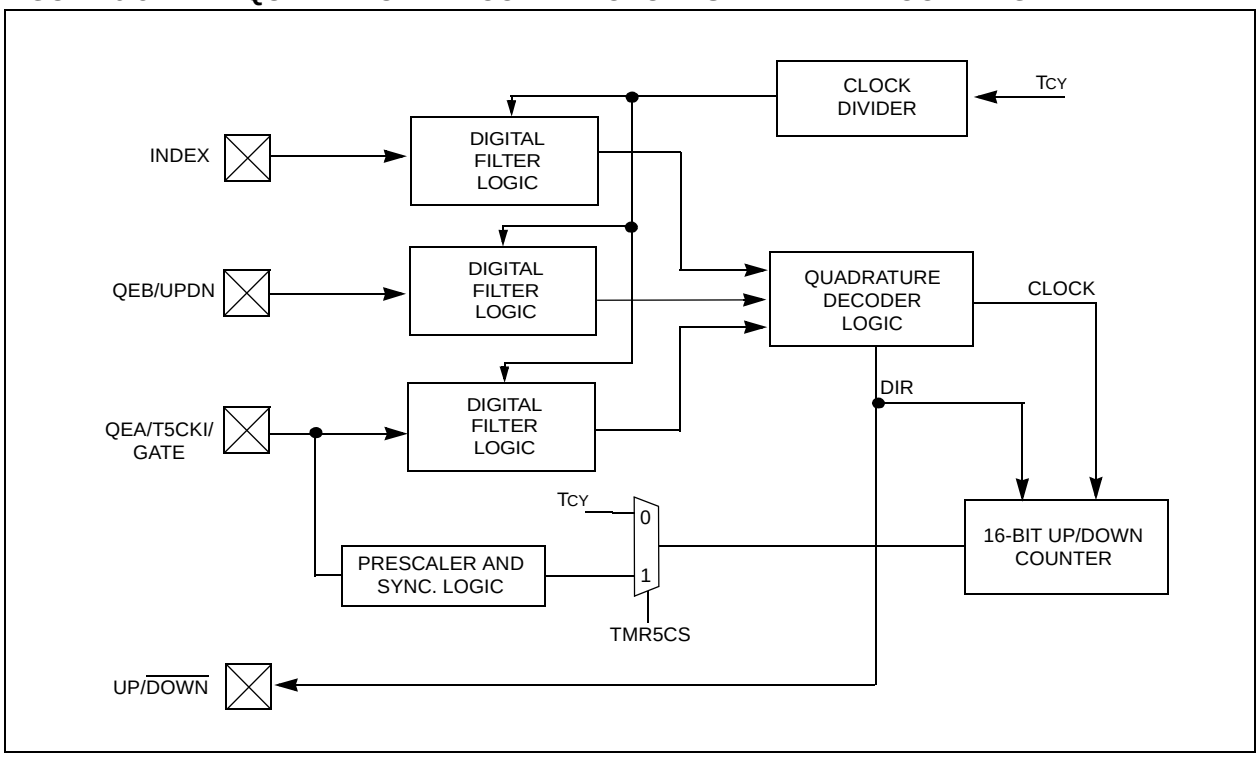
6.5 QEI Module

The Quadrature Encoder Interface (QEI) module is described below.

The module provides the Interface to incremental encoders for obtaining motor positioning data. Incremental encoders are very useful and specific to motor control applications.

Figure 6-5 depicts a simplified block diagram of the QEI Module.

FIGURE 6-5: QUADRATURE ENCODER MODULE SIMPLIFIED BLOCK DIAGRAM



6.5.1 OVERVIEW

The Quadrature Encoder Interface (QEI) is a key feature requirement for several motor control applications, such as Switched Reluctance (SR) motor and AC Induction Motor (ACIM). The operational features of the QEI are, but not limited to:

- Three input channels for two phase signals and index pulse
- 16-bit up/down position counter
- Count direction status
- Position measurement (x2 and x4) mode
- Programmable digital noise filters on inputs
- Alternate 16-bit timer/counter mode
- Quadrature Encoder Interface interrupts

6.6 Input Capture Module

This is a description of the Input Capture module and associated operational modes. Input Capture modules are useful in applications requiring Frequency (Period) and Pulse measurement.

6.6.1 OVERVIEW

Input capture is useful for such modes as:

- Frequency/Period/Pulse Measurements
- Additional sources of external interrupts

Table 6-1 presents the timer resource allocation for the input capture module.

TABLE 6-1: SUGGESTED TIMER RESOURCE

Functional Mode	Timer Resource
Input Capture	Timer 2 and Timer 3

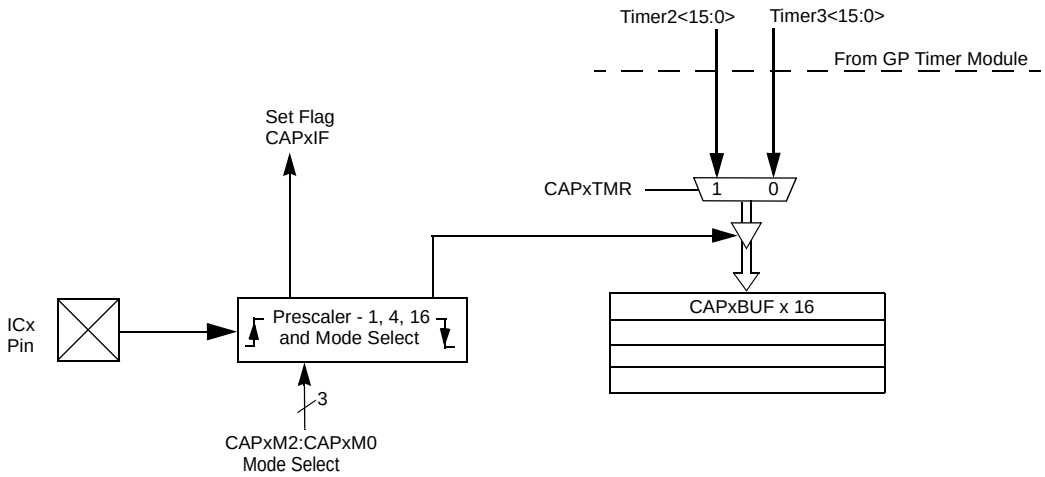
6.6.2 INPUT CAPTURE MODULE OPERATIONS

The Input Capture module consists of four input capture channels. The key operational features are:

- Simple capture event mode
- Timer2 and Timer3 mode selection
- Input Capture during sleep mode
- Interrupt on input capture event

These operating modes are determined by setting the appropriate control and configuration bits. Figure 6-6 depicts the Input Capture mode block diagram.

FIGURE 6-6: INPUT CAPTURE MODE BLOCK DIAGRAM



Note 1: Where 'x' is shown reference is made to the registers or bits associated to the respective input capture channels 1 through N.

6.6.3 SIMPLE CAPTURE EVENT MODE

The simple capture events are as follows:

- Capture every falling edge
- Capture every rising edge
- Capture every 4th rising edge
- Capture every 16th rising edge

These simple input capture modes are configured by setting the appropriate control and configuration bits.

6.7 Output Compare/ PWM module

This is a description of the Output Compare module and associated operational modes. The Output Compare module features are quite useful in applications requiring operational modes such as:

- Generation of variable width output pulses
- Power Factor Correction
- Simple PWM Operation

The following section provides a basic description of the Output Compare/PWM module. Table 6-2 presents the timer resource allocation.

TABLE 6-2: OUTPUT COMPARE SUGGESTED TIMER RESOURCE

Functional Mode	Timer Resource
Output Compare 1 - N	Timer 2 or Timer 3

6.7.1 OUTPUT COMPARE MODULARITY

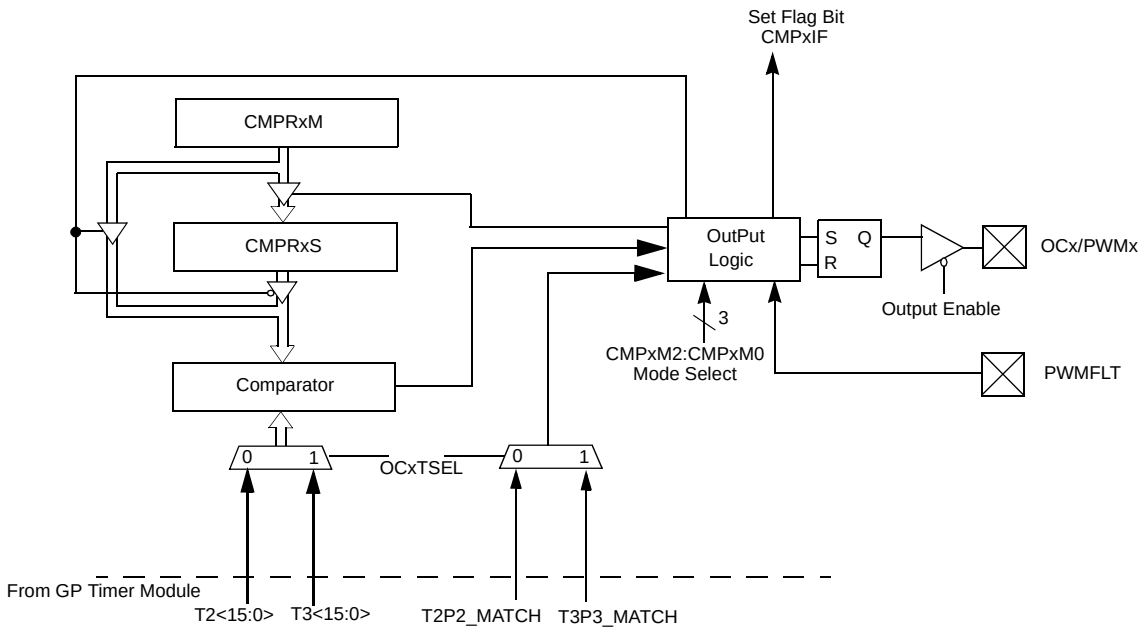
The Output Compare module consists of 1 to “N” output compare channels with the following feature enhancements. The key operational features are, but not limited to:

- Timer2 and Timer3 selection mode
- Simple Output Compare match mode
- Dual Output Compare match mode
- Simple glitchless PWM mode
- Output Compare during sleep mode
- Interrupt on output compare/PWM event

These operating modes are determined by setting the appropriate bits in Output Compare SFR (Special Function Register). Figure 6-7 depicts the output compare mode block diagram.

CMPRxM and CMPRxS in the figure represent the dual compare registers. In the dual compare mode, the CMPRxS register is used for the first compare and CMPRxM is used for the second compare. When configured for the PWM mode of operation, the CMPRxS is the slave latch (read-only) and CMPRxM is the master latch.

FIGURE 6-7: OUTPUT COMPARE MODE BLOCK DIAGRAM



Note 1: Where ‘x’ is shown reference is made to the registers associated to the respective output compare channels 1, 2, 3 or 4.

6.8 SPI™ Module

6.8.1 OPERATING FUNCTION DESCRIPTION

The Serial Peripheral Interface (SPI) module is a synchronous serial interface useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be Serial EEPROMs, shift registers, display drivers, A/D converters, etc.

This SPI module includes all SPI modes. A Frame Synchronization mode is also included for support of voice band CODECs. The following sections describe the basic functionality of the SPI module. Figure 6-8 shows a block diagram of the SPI.

6.8.2 SERIAL PERIPHERAL INTERFACE (SPI)

SPI mode is a high-speed serial I/O interface useful for communicating with peripheral devices (e.g., serial EEPROM, serial A/D) and for I/O expansion. It is compatible with Motorola's SPI™ and SIOP interfaces.

The serial port consists of a 16-bit shift register, SPISR, used for shifting data in and out, and a buffer register, SPIBUF. A control register, SPICON, configures the module. Additionally, a status register, SPISTAT, indicates various status conditions. Five pins make up the serial interface; SDI: serial data input; SDO: serial data output; SCK: shift clock input or output, $\overline{SS}$: active low slave select and FSYNC: frame synchronization pulse. In master mode operation, SCK is clock output, but in slave mode, it is clock input.

The control bit SPIEN along with several control bits enables the serial port and configures SDI, SDO, SCK and $\overline{SS}$ pins as serial port pins.

A series of eight clock pulses shift out 8 bits from the SPISR to SDO pin and simultaneously shift in 8-bit data from SDI pin. An interrupt is generated when the transfer is complete (interrupt flag bit SPIIF). This interrupt can be disabled through the interrupt enable bit SPIIE.

The receive operation is double buffered. When a complete byte is received it is transferred from SPISR to SPIBUF.

Transmit operation is not double buffered. The user writes directly to SPISR. Whereas a read operation will read SPIBUF, a write operation will write to both SPISR and SPIBUF.

In master mode, the clock is generated by prescaling the system clock. When an external clock source is used, a minimum high and low time must be observed.

In master mode, data is transmitted as soon as SPIBUF is written. The interrupt is raised at the middle of the last bit duration (i.e., after the last bit in is latched).

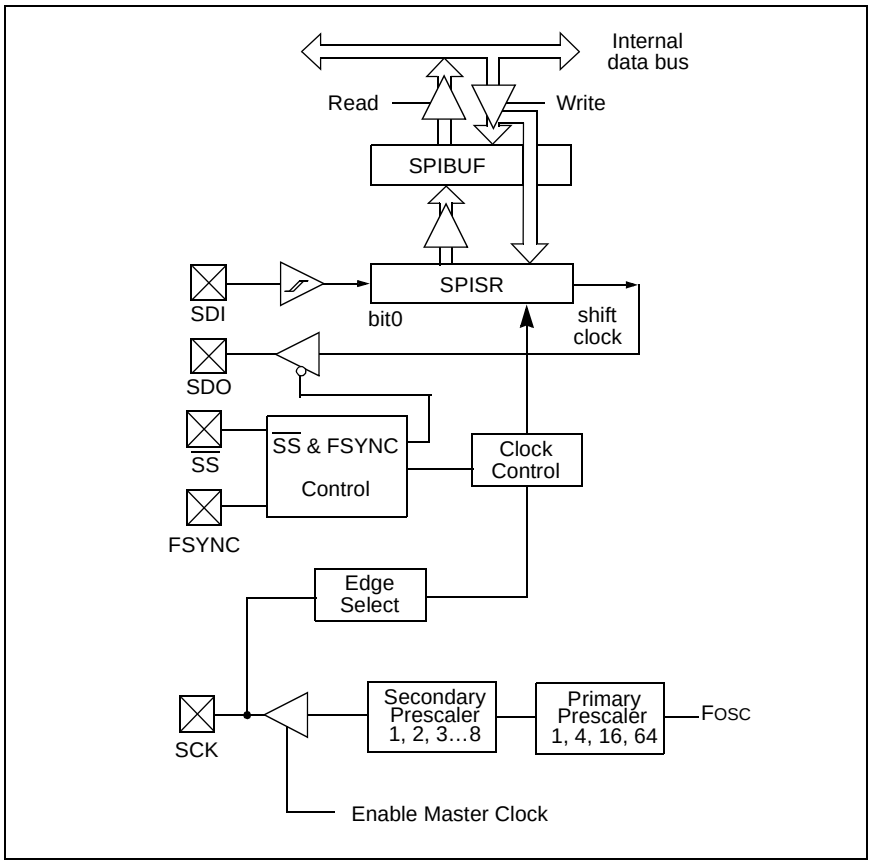
In slave mode, data is transmitted and received as external clock pulses appear on SCK. Again, the interrupt is set as the last bit is latched in. If $\overline{SS}$ control is enabled, then transmission and reception are enabled only when $\overline{SS}$ = low. SDO output will be disabled in $\overline{SS}$ mode with $\overline{SS}$ = high.

6.8.3 DIRECTION CONTROL OF SPI PINS

The input/output direction control on all the SPI pins is controlled by the SPI module. Therefore, control signals generated within the module will override the data direction control register on each SPI pin based on the current operating mode of the module.

The SPI module can give up control of three pins. The $\overline{SS}$ pin is only controlled in slave mode with $\overline{SS}$ enabled. SDO has a control bit in SPICON that allows the module to disable direction control, DISSDO. FSYNC pin is only controlled when the FRMEN bit is high.

FIGURE 6-8: SPI BLOCK DIAGRAM



6.9 UART MODULE

This is the description of a Universal Asynchronous Receiver/Transmitter Communications module. The UART module is defined closely to match the USART module from the PIC18C family with a few key differences. The dsPIC products will have one or more UART's.

6.9.1 OVERVIEW OF FEATURES

The key features of the UART module are:

- Full-duplex operation with 8- or 9-bit data
- Even, Odd or No Parity options (for 8-bit data)
- One or two stop bits
- Hardware flow control option with $\overline{CTS}$ and $\overline{RTS}$ pins
- Fully integrated Baud Rate Generator with 16-bit prescaler
- Baud rates range from up to 2.5Mbps and down to 38Hz at 40MIPS
- 4-byte deep transmit data buffer
- 4-byte deep receive data buffer
- Parity, Framing and Buffer Overrun error detection
- 16X Baud Clock output for IrDA support
- Support for Interrupt only on Address Detect (9th bit=1)
- Separate Transmit and Receive Interrupts
- Loopback mode for diagnostics

6.10 $I^2C^{™}$ MODULE

This document describes the Inter-Integrated Circuit (I^2C) function that offers full hardware support for both slave and multi-master modes, with a 16-bit interface. Figure 6-9 shows an I^2C receive block diagram and Figure 6-10 shows an I^2C transmit block diagram.

6.10.1 FEATURES OVERVIEW

- Inter-Integrated Circuit (I^2C) interface
- I^2C interface supports both master and slave modes.
- I^2C slave mode supports 7- and 10-bit address.
- I^2C master mode supports 7- and 10-bit address.
- I^2C port allows bidirectional transfers between master and slaves.
- Serial clock synchronization for I^2C port can be used as a handshake mechanism to suspend and resume serial transfer.
- I^2C supports multi-master mode. Detects bus collision and will arbitrate accordingly.

6.10.2 OPERATING FUNCTION DESCRIPTION

The I^2C module is a synchronous serial interface useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be Serial EEPROMs, shift registers, display drivers, A/D converters, etc.

6.10.3 INTER-INTEGRATED CIRCUIT (I^2C)

The I^2C module hardware fully implements all the master and slave functions of the I^2C standard and fast mode specifications, as well as 7- and 10-bit addressing.

Thus the I^2C module can operate as a slave, or a master on an I^2C bus.

6.10.4 VARIOUS I^2C MODES

There are no control bits to select a specific mode. However, all of the following modes are supported:

- I^2C slave mode (7-bit address)
- I^2C slave mode (10-bit address)
- I^2C master mode (7- or 10-bit address)

FIGURE 6-9: I²C BLOCK DIAGRAM (I²C RECEIVE)

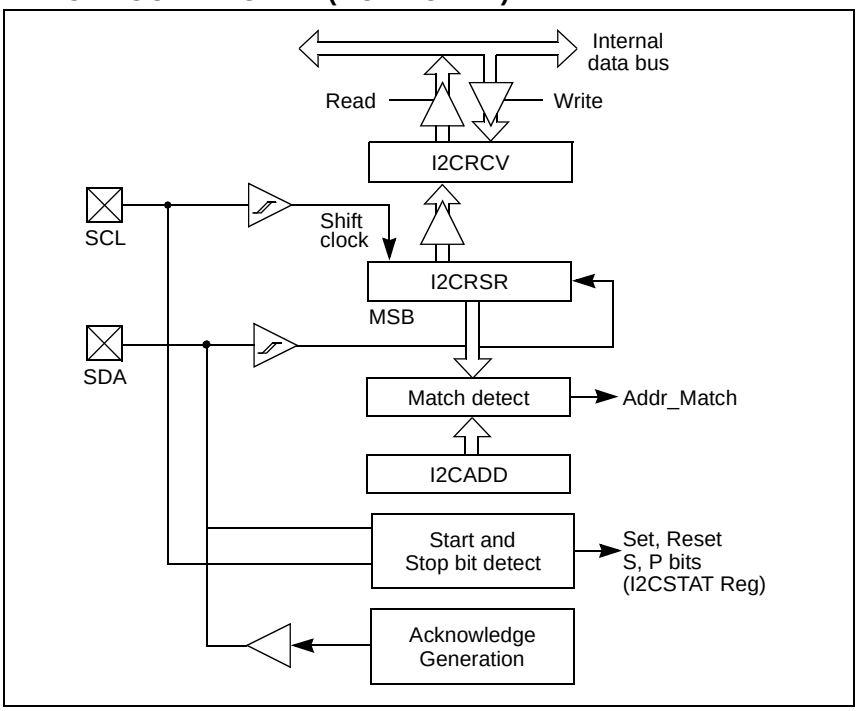
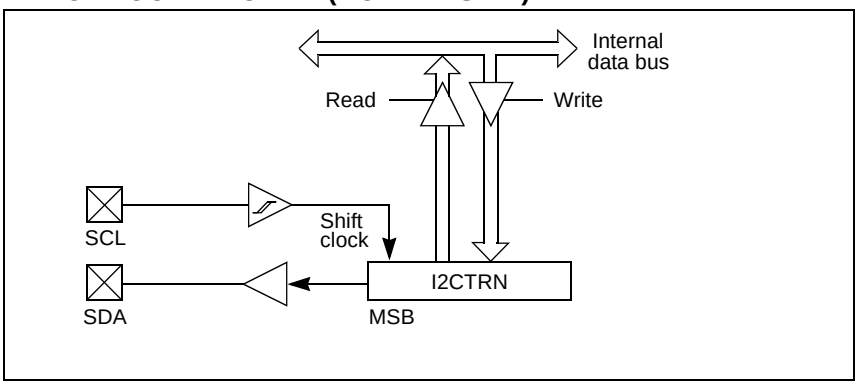


FIGURE 6-10: I²C BLOCK DIAGRAM (I²C TRANSMIT)



6.10.5 PIN CONFIGURATION IN I²C MODE

In I²C mode, pin SCL is clock and pin SDA is data. The module will override the data direction bits for these pins. The pins that are used for I²C modes are configured as open-drain.

6.10.6 I²C REGISTERS

I2CCON, and I2CSTAT are control and status registers, respectively. The I2CCON registers is readable and writable. The lower 6 bits of the I2CSTAT are read-only. The remaining bits of the I2CSTAT are read/write.

I2CRSR is the shift register used for shifting data in Figure 6-9.

I2CRCV is the buffer register to which data bytes are written to or read from. This register is the receive buffer, as shown in Figure 6-9. I2CXMT is the transmit register; bytes are written to this register during a transmit operation, as shown in Figure 6-10.

I2CADD register holds the slave address, and this register is now 10 bits wide to hold the full slave address. If 10-bit mode is desired, the 10-bit address preamble is recognized by the module, which then automatically enables 10-bit addressing mode. A status bit, ADD10, indicates 10-bit address mode. The I2CBRG acts as the baud rate generator reload value. The baud rate generator is a full baud rate generator.

In receive operations, I2CRSR and I2CRCV together create a double buffered receiver. When I2CRSR receives a complete byte, it is transferred to I2CRCV and the i2c_int_flag interrupt is set. During transmission, the I2CTRN is not double buffered.

Note:	Following a RESTART condition in 10-bit mode, the user only needs to match the first 7-bit address.

6.10.7 I^2C 7-BIT SLAVE MODE OPERATION

Once enabled (I2CEN = 1), the slave module will wait for a start bit to occur (IDLE_MODE). Following a start-bit detect, 8 bits are shifted into I2CRSR and the address is compared against I2CADD. In 7-bit mode, bits I2CADD<6:0> are compared against bits I2CRSR<7:1> and bit0 is the R/$\overline{W}$ bit. All incoming bits are sampled with the rising edge of SCL.

If the address matches, an acknowledge will be sent, and on the falling edge of the ninth bit ($\overline{ACK}$ bit) the i2c_int_flag interrupt is set.

6.10.7.1 Slave Mode Transmission

If R/$\overline{W}$ bit received is a '1' then the serial port will go into 'XMIT_MODE'. It will send $\overline{ACK}$ on the ninth bit and then hold SCL to '0' until the CPU responds by writing to I2CTRN. SCL is released and 8 bits of data are shifted out. Data bits are shifted out on the falling edge of SCL such that SDA is valid during SCL high (see timing diagram). Interrupt is set on the falling edge of the ninth clock pulse.

The $\overline{ACK}$ bit from master is latched on the ninth clock pulse. If $\overline{ACK}$ = 1 then 'XMIT_MODE' ends and the serial port resumes 'IDLE_MODE' looking for another start bit.

If $\overline{ACK}$ = 0, then it will again hold SCL low until I2CTRN is full (i.e., written to).

TBF status flag: During transmit, the TBF bit (I2CSTAT<0>) is set when the CPU writes to I2CTRN, and TBF is cleared in hardware when all 8 bits are shifted out.

IWCOL status flag: If the user attempts to write a byte to the I2CTRN register when TBF = 1 (i.e., I2CTRN is still shifting out previous data byte), then IWCOL is set. IWCOL must be cleared in software.

R/$\overline{W}$ status flag: Latches and holds the R/$\overline{W}$ bit received following the last address-match.

6.11 Controller Area Network Module (CAN)

The Controller Area Network (CAN) is a serial communications protocol which efficiently supports distributed real-time control with a very high level of security. Figure 6-11 shows a block diagram of a CAN module.

The DSC CAN module satisfies the Version 2.0B specification, which allows message identifier lengths of 11 and/or 29 bits to be used (an identifier length of 29 bits allows over 536 Million message identifiers). Version 2.0B CAN is also referred to as "Extended CAN".

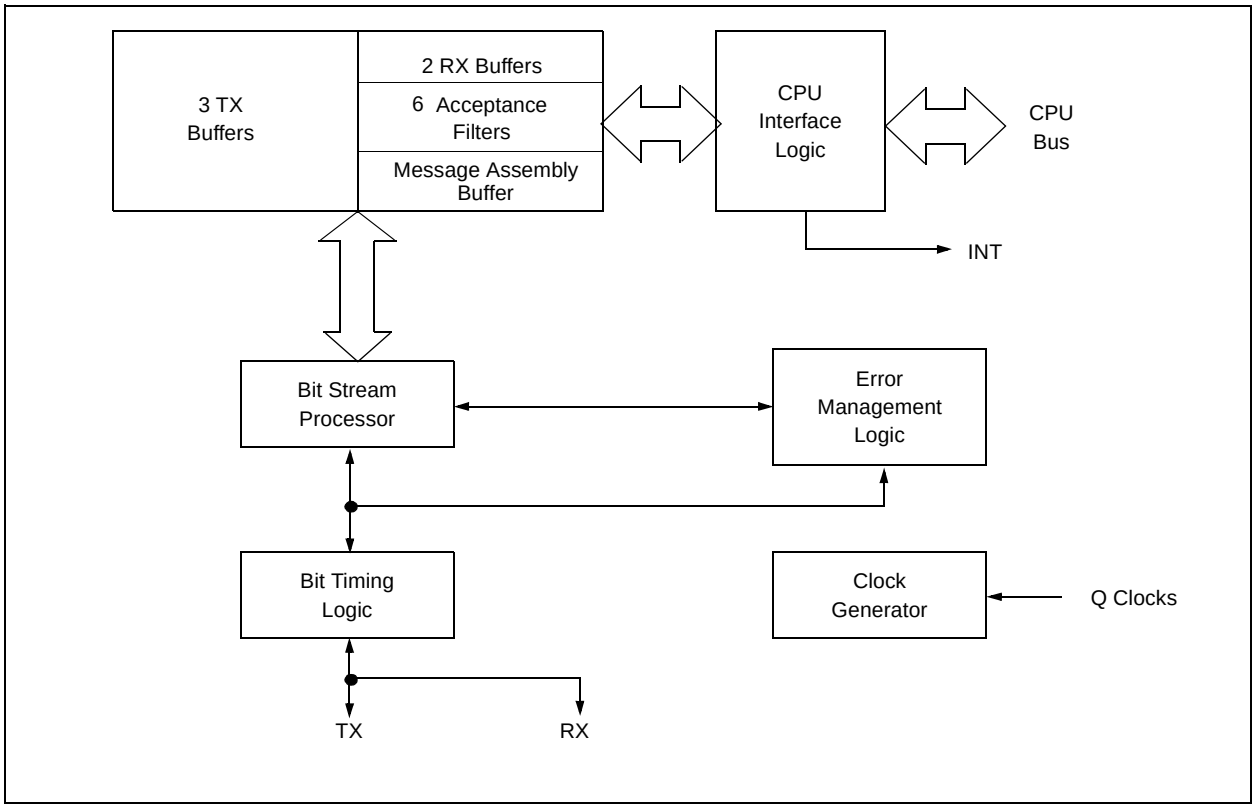
6.11.1 CAN MODULE FEATURES

The CAN module is a communication controller implementing the CAN 2.0 A/B protocol as defined in the BOSCH specification. The module will support CAN 1.2, CAN 2.0A, CAN2.0B Passive, and CAN 2.0B Active versions of the protocol. The module implementation is a Full CAN system. Based on requirements expressed by CAN application software authors for predictable real-time behavior and the need to minimize silicon and to save cost, the module implements an advanced buffer arrangement.

The module features are as follows:

- Implementation of the CAN protocol
- Standard and extended data frames
- 0 - 8 bytes data length
- Programmable bit rate up to 1 Mb/sec
- Support for remote frames
- Double buffered receiver with two prioritized received message storage buffers
- 6 full (standard/extended identifier) acceptance filters, 2 associated with the high priority receive buffer, and 4 associated with the low priority receive buffer
- 2 full acceptance filter masks, one each associated with the high and low priority receive buffers
- Three transmit buffers with application-specified prioritization and abort capability
- Programmable wake-up functionality with integrated low-pass filter
- Programmable loop-back mode and programmable state clocking supports self-test operation
- Signaling via interrupt capabilities for all CAN receiver and transmitter error states
- Programmable clock source
- Programmable link to timer module for time-stamping and network synchronization
- Low power SLEEP mode

FIGURE 6-11: CAN MODULE BLOCK DIAGRAM

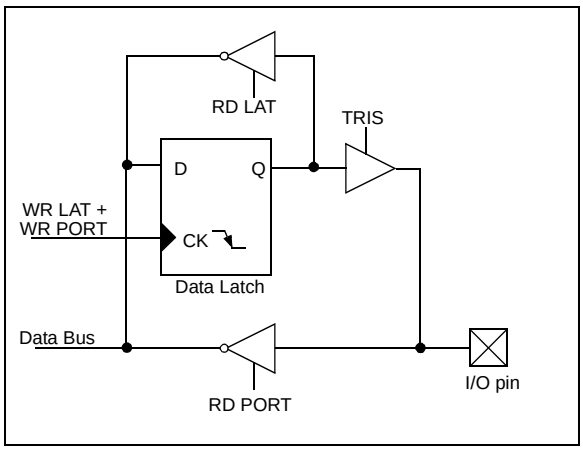


6.12 I/O Pins

Some pins for the I/O pin functions are multiplexed with an alternate function for the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

All I/O port pins have three registers directly associated with the operation of the port pin. The Data Direction Register determines whether the pin is an input or an output. The port Data Latch Register provides latched output data for the I/O pins. The Port Register provides visibility of the logic state of the I/O pins. Reading the Port Register provides the I/O pin logic state, while writes to the Port Register write the data to the port Data Latch Register. Figure 6-12 illustrates a PORT/LAT/TRIS block diagram.

FIGURE 6-12: SIMPLIFIED BLOCK DIAGRAM OF PORT/LAT/TRIS OPERATION



6.12.1 I/O PIN FEATURES

- Schmitt Trigger input
- Open drain output.
- TTL input levels
- CMOS output drivers
- Weak internal pull-up (gated)
- Interrupt on change feature (inputs only)

6.12.2 I/0 Port Latch

Some I/O port pins have latch bits (LATCH register). The LATCH register when read will yield the contents of the I/O latch, and when written will modify the contents of the I/O latch, thus modifying the value driven out on a pin if the corresponding Data Direction Register bit is configured for output. This can be used in read-modify-write instructions that allow the user to modify the contents of the latch register regardless of the status of the corresponding pins.

Trademarks

DNV Certification, Inc. USA
ANSI-RAB QMS ACCREDITED
DNV
DNV MSC The Netherlands Accredited by the RvA
ISO 9001 / QS-9000 REGISTERED FIRM

Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs and microperipheral products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001 certified.

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200 Fax: 480-792-7277
Technical Support: 480-792-7627
Web Address: http://www.microchip.com

Rocky Mountain
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7966 Fax: 480-792-7456

Atlanta
500 Sugar Mill Road, Suite 200B
Atlanta, GA 30350
Tel: 770-640-0034 Fax: 770-640-0307

Austin
Analog Product Sales
8303 MoPac Expressway North
Suite A-201
Austin, TX 78759
Tel: 512-345-2030 Fax: 512-345-6085

Boston
2 Lan Drive, Suite 120
Westford, MA 01886
Tel: 978-692-3848 Fax: 978-692-3821

Boston
Analog Product Sales
Unit A-8-1 Millbrook Tarry Condominium
97 Lowell Road
Concord, MA 01742
Tel: 978-371-6400 Fax: 978-371-0050

Chicago
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071 Fax: 630-285-0075

Dallas
4570 Westgrove Drive, Suite 160
Addison, TX 75001
Tel: 972-818-7423 Fax: 972-818-2924

Dayton
Two Prestige Place, Suite 130
Miamisburg, OH 45342
Tel: 937-291-1654 Fax: 937-291-9175

Detroit
Tri-Atria Office Building
32255 Northwestern Highway, Suite 190
Farmington Hills, MI 48334
Tel: 248-538-2250 Fax: 248-538-2260

Los Angeles
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888 Fax: 949-263-1338

Mountain View
Analog Product Sales
1300 Terra Bella Avenue
Mountain View, CA 94043-1836
Tel: 650-968-9241 Fax: 650-967-1590

New York
150 Motor Parkway, Suite 202
Hauppauge, NY 11788
Tel: 631-273-5305 Fax: 631-273-5335

San Jose
Microchip Technology Inc.
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408-436-7950 Fax: 408-436-7955

Toronto
6285 Northam Drive, Suite 108
Mississauga, Ontario L4V 1X5, Canada
Tel: 905-673-0699 Fax: 905-673-6509

ASIA/PACIFIC

Australia
Microchip Technology Australia Pty Ltd
Suite 22, 41 Rawson Street
Epping 2121, NSW
Australia
Tel: 61-2-9868-6733 Fax: 61-2-9868-6755

China - Beijing
Microchip Technology Beijing Office
Unit 915
New China Hong Kong Manhattan Bldg.
No. 6 Chaoyangmen Beidajie
Beijing, 100027, No. China
Tel: 86-10-85282100 Fax: 86-10-85282104

China - Shanghai
Microchip Technology Shanghai Office
Room 701, Bldg. B
Far East International Plaza
No. 317 Xian Xia Road
Shanghai, 200051
Tel: 86-21-6275-5700 Fax: 86-21-6275-5060

Hong Kong
Microchip Asia Pacific
RM 2101, Tower 2, Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2401-1200 Fax: 852-2401-3431

India
Microchip Technology Inc.
India Liaison Office
Divyasree Chambers
1 Floor, Wing A (A3/A4)
No. 11, O'Shaugnessey Road
Bangalore, 560 025, India
Tel: 91-80-2290061 Fax: 91-80-2290062

Japan
Microchip Technology Intl. Inc.
Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa, 222-0033, Japan
Tel: 81-45-471- 6166 Fax: 81-45-471-6122

ASIA/PACIFIC (continued)

Korea
Microchip Technology Korea
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea
Tel: 82-2-554-7200 Fax: 82-2-558-5934

Singapore
Microchip Technology Singapore Pte Ltd.
200 Middle Road
#07-02 Prime Centre
Singapore, 188980
Tel: 65-334-8870 Fax: 65-334-8850

Taiwan
Microchip Technology Taiwan
11F-3, No. 207
Tung Hua North Road
Taipei, 105, Taiwan
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

EUROPE

Denmark
Microchip Technology Denmark ApS
Regus Business Centre
Lautrup hoj 1-3
Ballerup DK-2750 Denmark
Tel: 45 4420 9895 Fax: 45 4420 9910

France
Arizona Microchip Technology SARL
Parc d'Activite du Moulin de Massy
43 Rue du Saule Trapu
Batiment A - ler Etage
91300 Massy, France
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

Germany
Arizona Microchip Technology GmbH
Gustav-Heinemann Ring 125
D-81739 Munich, Germany
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

Germany
Analog Product Sales
Lochhamer Strasse 13
D-82152 Martinsried, Germany
Tel: 49-89-895650-0 Fax: 49-89-895650-22

Italy
Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Taurus 1 V. Le Colleoni 1
20041 Agrate Brianza
Milan, Italy
Tel: 39-039-65791-1 Fax: 39-039-6899883

United Kingdom
Arizona Microchip Technology Ltd.
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44 118 921 5869 Fax: 44-118 921-5820

01/30/01