

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

High Performance Modified RISC CPU:

- Modified Harvard architecture
- C compiler optimized instruction set architecture
- 89 base instructions
- 24-bit wide instructions, 16-bit wide data path
- Linear program memory addressing up to 4M Instruction Words
- Linear data memory addressing up to 64 Kbytes
- Up to 144 Kbytes on-chip FLASH program space
 - Up to 48K Instruction Words
- Up to 8 Kbytes of on-chip data RAM
- Up to 4 Kbytes of non-volatile data EEPROM
- 16 x 16-bit working register array
- Three Address Generation Units that enable:
 - Dual data fetch
 - Accumulator writeback for DSP operations
- Flexible addressing modes supporting:
 - Indirect, Modulo and Bit-Reversed modes
- Two, 40-bit wide accumulators with optional saturation logic
- 16-bit x 16-bit single cycle hardware fractional/integer multiplier
- Single cycle Multiply-Accumulate (MAC) operation
- 40-stage Barrel Shifter
- Up to 30 MIPS operation:
 - DC to 40 MHz External Clock Input mode
 - 4 MHz - 10 MHz Crystal mode with PLL active (4X, 8X, 16X)
- Up to 45 interrupt sources
 - 7 user selectable priority levels
 - 8 processor exceptions and software traps
 - Vector table with up to 62 vectors

Peripheral Features (see Note 1):

- High current sink/source I/O pins: 25 mA/25 mA
- Up to 5 external interrupt sources
- Timer module with programmable prescaler:
 - Up to five 16-bit timers/counters; optionally pair up 16-bit timers into 32-bit timer modules
- 16-bit Capture input functions
- 16-bit Compare/PWM output functions
 - Dual Compare mode available

Peripheral Features (Continued):

- Data Converter Interface (DCI), supports common audio CODEC protocols, including I²S and AC'97
- 3-wire SPI™ modules (supports 4 SPI modes and Frame Sync mode)
- I²C™ module supports Multi-Master/Slave mode and 7-bit/10-bit addressing
- Addressable UART modules supporting:
 - Interrupt-on-address bit
 - Wake-up-on-START bit
 - 4 characters deep TX and RX FIFO buffers
 - CAN bus modules
- As many as 54 programmable digital I/O pins
 - Some with interrupt-on-change (up to 24)

Note 1: Refer to Table 1-1, Table 1-2 and Table 2-1 for exact peripheral features per device.

Advanced Analog Features:

- 10-bit Analog-to-Digital Converters (A/D) with:
 - 16 input channels, typically
 - 500 ksps conversion rate
 - Automated input scanning
 - 2 or 4 simultaneous samples
 - Conversion available during SLEEP
- 12-bit Analog-to-Digital Converters (A/D) with:
 - 16 input channels, typically
 - 100 ksps conversion rate
 - Automated input scanning
 - Conversion available during SLEEP
- Programmable Low Voltage Detection (LVD)
 - Supports interrupt on low voltage detection
- Programmable Brown-out Reset generation

Motor Control PWM Module Features:

- Up to 8 PWM output channels
 - Complementary or Independent Output modes
 - Edge and Center Aligned modes
- 4 duty cycle generators
- Dedicated time-base with 4 modes
- Programmable output polarity
- Dead-time control for Complementary mode
- Manual output control
- Trigger for A/D conversions

dsPIC30F

Quadrature Encoder Interface Module Features:

- Phase A, Phase B and Index Pulse input
- 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
- Interrupt on position counter rollover/underflow

Special Microcontroller Features:

- Enhanced FLASH program memory
 - 100,000 write/erase cycle (typical) for industrial temperature range
- Data EEPROM memory
 - 1,000,000 write/erase cycle (typical) industrial temperature range
 - Data EEPROM Retention > 20 years
- Self-reprogrammable under software control
- Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Flexible Watchdog Timer (WDT) with on-chip low power RC oscillator for reliable operation

Special Microcontroller Features (Continued):

- Fail safe clock monitor operation
 - Detects clock failure and switches to on-chip 8 MHz RC oscillator
- Programmable code protection
- In-Circuit Serial Programming™ (ICSP™) via 3 pins and power/ground
- Selectable Power Management modes
 - SLEEP, IDLE and Alternate Clock modes

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

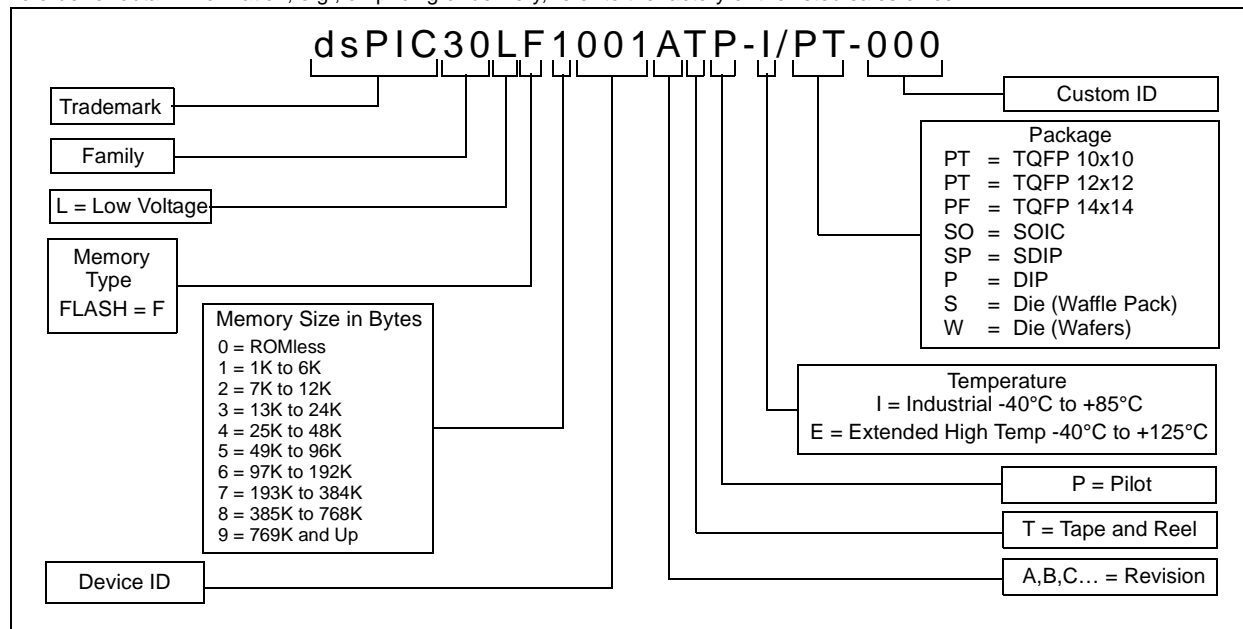
Packaging:

- 80-pin TQFP
- 64-pin TQFP
- 40-pin DIP, 44-pin TQFP
- 28-pin DIP (300 mil), 28-pin SOIC
- 18-pin DIP (300 mil), 18-pin SOIC

The following figure defines the part number structure.

PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.



1.0 GENERAL PURPOSE AND SENSOR FAMILY PRODUCT INFORMATION

TABLE 1-1: dsPIC30F SENSOR PROCESSOR FAMILY VARIANTS

Device	Pins	Program Memory		SRAM Bytes	EEPROM Bytes	Timer 16-bit	Input Cap	Output Comp/ Std PWM	A/D 12-bit 100 Ksps	UART	SPI™	I²C™
		Bytes	Instructions									
dsPIC30F2011	18	12K	4K	1024	0	3	2	2	8 ch	1	1	1
dsPIC30F3012	18	24K	8K	2048	1024	3	2	2	8 ch	1	1	1
dsPIC30F2012	28	12K	4K	1024	0	3	2	2	10 ch	1	1	1
dsPIC30F3013	28	24K	8K	2048	1024	3	2	2	10 ch	2	1	1

TABLE 1-2: dsPIC30F GENERAL PURPOSE CONTROLLER FAMILY

Device	Pins	Program Memory		SRAM Bytes	EEPROM Bytes	Timer 16-bit	Input Cap	Output Comp/ Std PWM	Codec Interface	A/D 12-bit 100 Ksps	UART	SPI™	I²C™	CAN
		Bytes	Instructions											
dsPIC30F3014**	40/44	24K	8K	2048	1024	3	2	2	-	13 ch	2	1	1	-
dsPIC30F4013**	40/44	48K	16K	2048	1024	5	4	4	AC97, I²S	13 ch	2	1	1	1
dsPIC30F4014**	64	36K	12K	2048	1024	5	8	8	AC97, I²S	16 ch	2	2	1	1
dsPIC30F5011	64	66K	22K	4096	1024	5	8	8	-	16 ch	2	2	1	2
dsPIC30F5012	64	96K	32K	4096	2048	5	8	8	AC97, I²S	16 ch	2	2	1	2
dsPIC30F6011	64	132K	44K	6144	2048	5	8	8	-	16 ch	2	2	1	2
dsPIC30F6012	64	144K	48K	8192	4096	5	8	8	AC97, I²S	16 ch	2	2	1	2
dsPIC30F4015**	80	36K	12K	2048	1024	5	8	8	AC97, I²S	16 ch	2	2	1	1
dsPIC30F5013	80	66K	22K	4096	1024	5	8	8	-	16 ch	2	2	1	2
dsPIC30F5014	80	96K	32K	4096	2048	5	8	8	AC97, I²S	16 ch	2	2	1	2
dsPIC30F6013	80	132K	44K	6144	2048	5	8	8	-	16 ch	2	2	1	2
dsPIC30F6014	80	144K	48K	8192	4096	5	8	8	AC97, I²S	16 ch	2	2	1	2

** Proposed Products (others are committed)

dsPIC30F

FIGURE 1-1: PIN DIAGRAMS (18-Pin SOIC, 18-Pin PDIP)

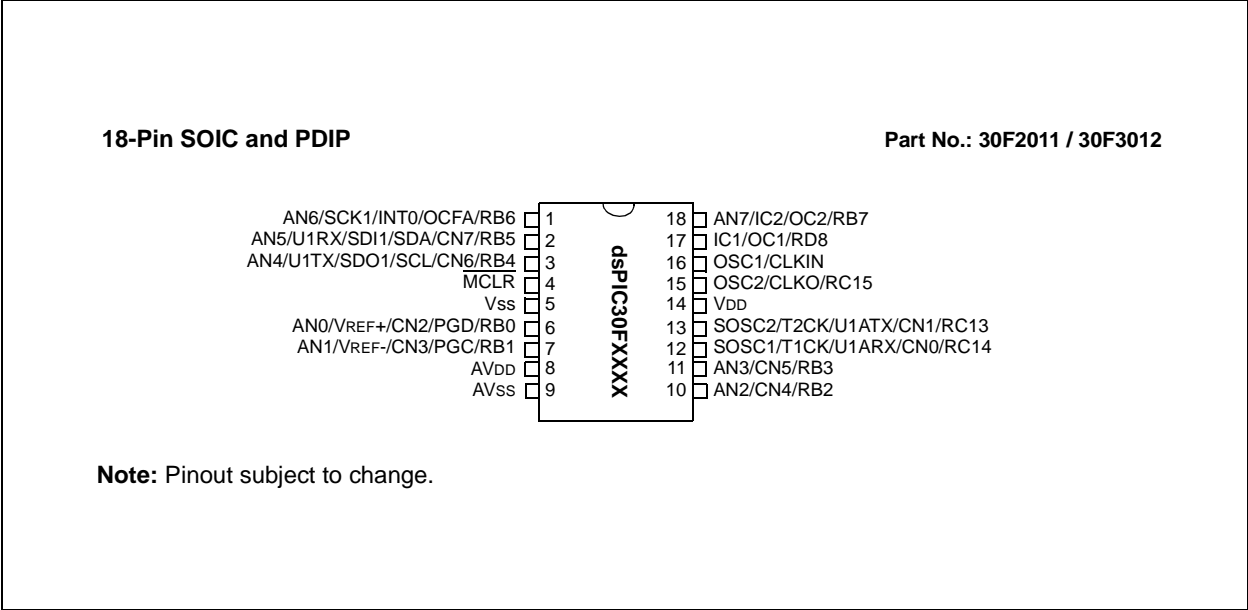
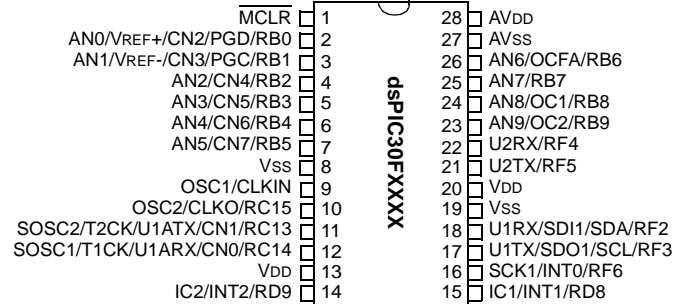


FIGURE 1-2: PIN DIAGRAMS (28-Pin SDIP, 40-Pin PDIP)

28-Pin SDIP

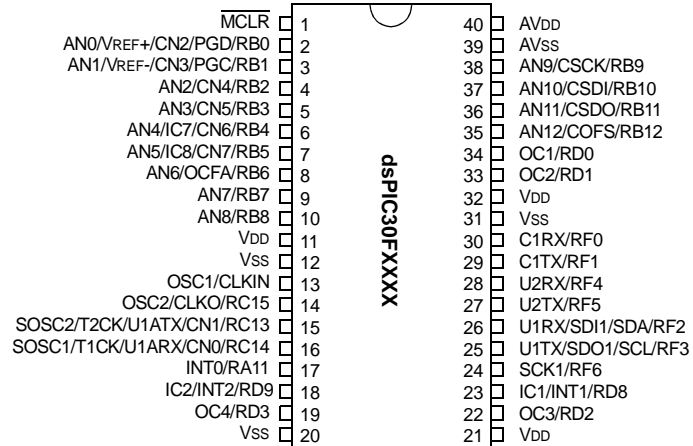
Part No.: 30F2012 / 30F3013



Note: Pinout subject to change.

40-Pin PDIP

Part No.: 30F3014 / 30F4013



Note: Pinout subject to change.

FIGURE 1-3: PIN DIAGRAMS (44-Pin TQFP)

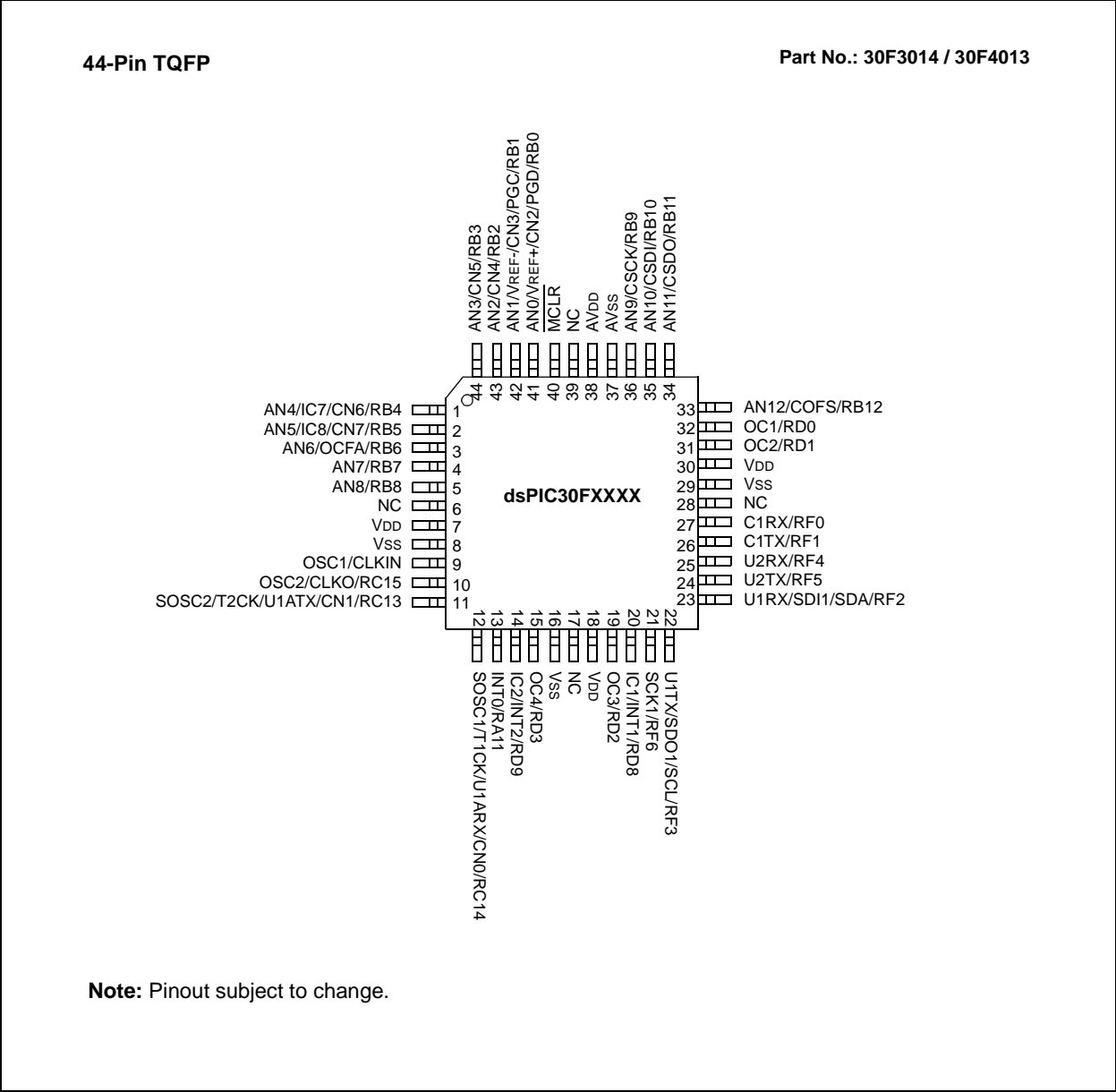


FIGURE 1-4: PIN DIAGRAMS (64-Pin TQFP)

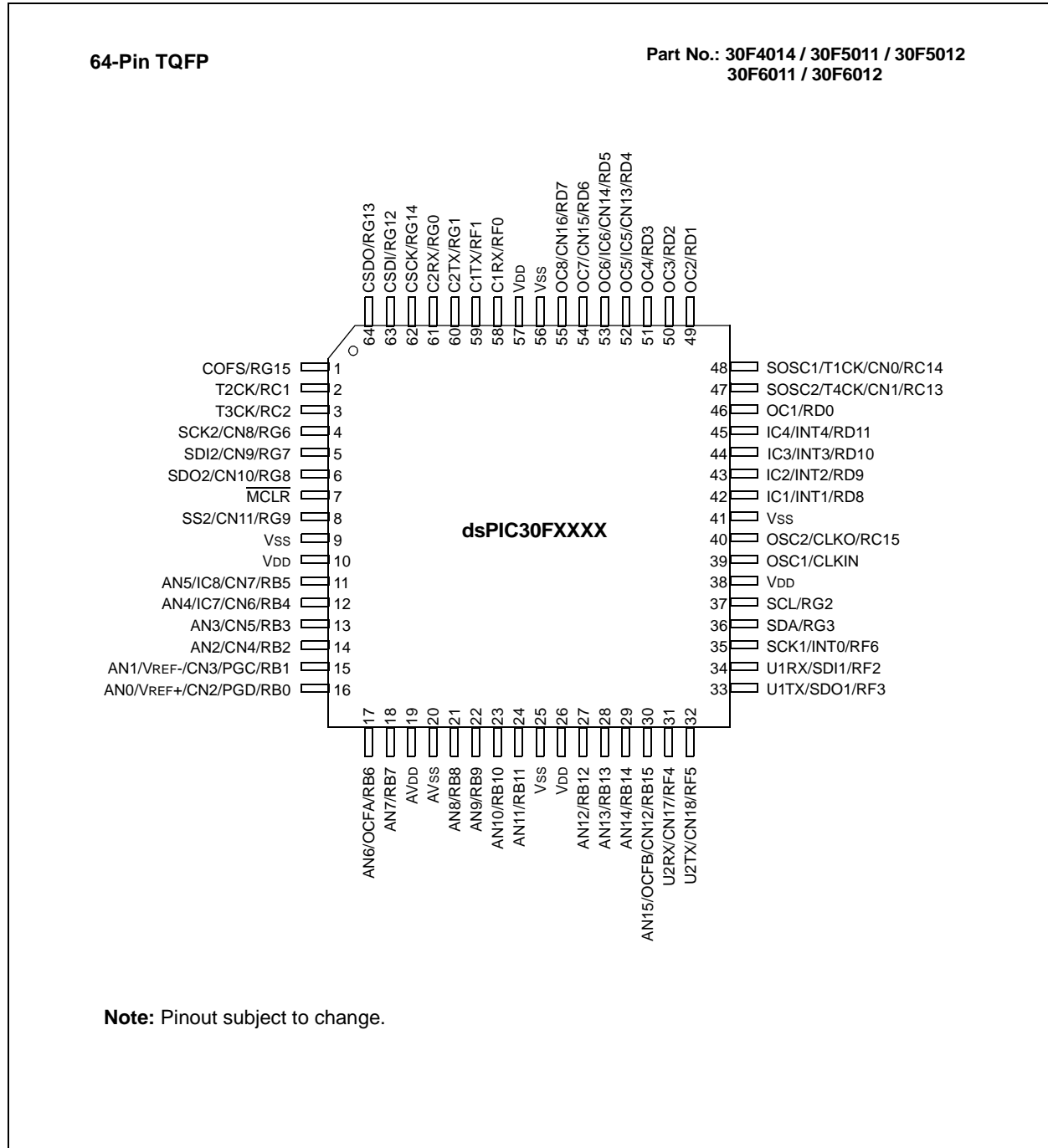
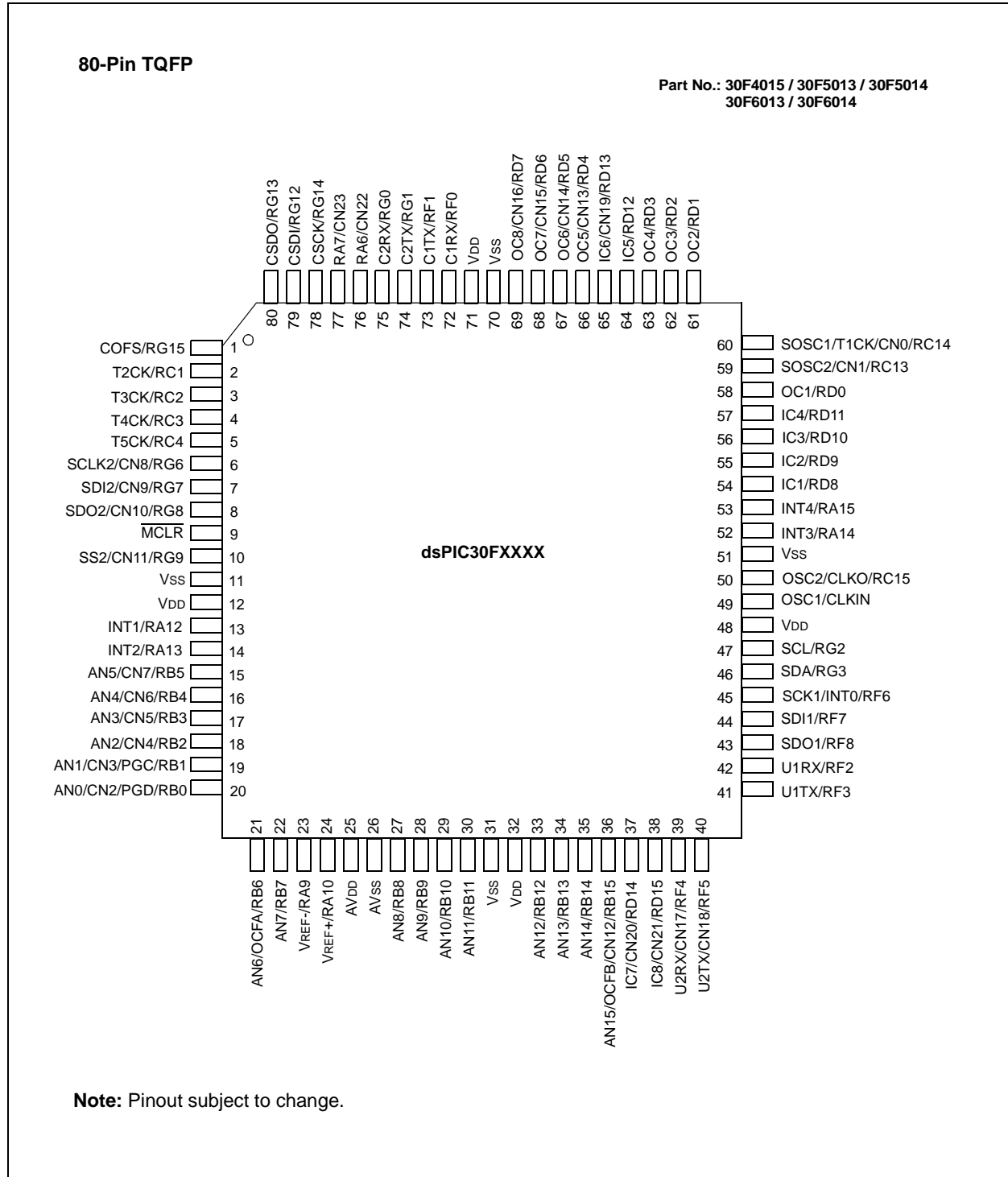


FIGURE 1-5: PIN DIAGRAMS (80-Pin TQFP)

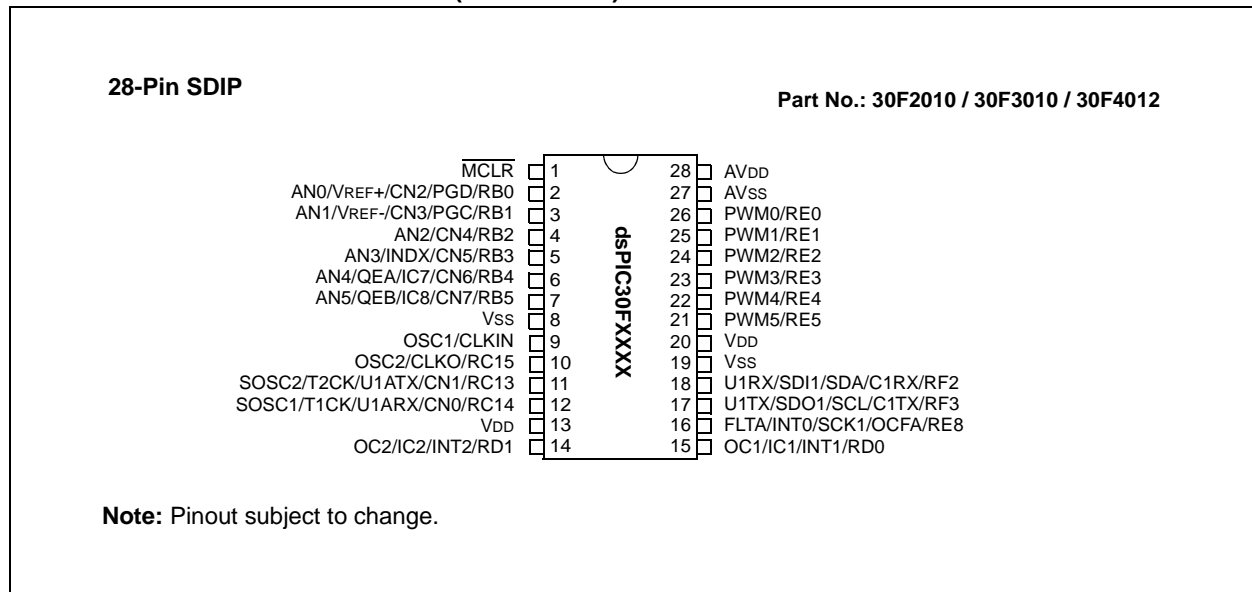


2.0 MOTOR CONTROL FAMILY PRODUCT INFORMATION

TABLE 2-1: dsPIC30F POWER CONVERSION AND MOTION CONTROL FAMILY VARIANTS

Device	Pins	Program Mem. Bytes/ Instructions		SRAM Bytes	EEPROM Bytes	Timer 16-bit	Input Cap	Output Comp/ Std PWM	Motor Cntrl PWM	A/D 10-bit 500 Ksps	Quad Enc	UART	SPI™	I ² C™	CAN
dsPIC30F2010	28	12K	4K	512	1024	3	4	2	6 ch	6 ch	1	1	1	1	-
dsPIC30F3010	28	24K	8K	1024	1024	5	4	2	6 ch	6 ch	1	1	1	1	-
dsPIC30F4012	28	48K	16K	2048	1024	5	4	2	6 ch	6 ch	1	1	1	1	1
dsPIC30F3011	40/44	24K	8K	1024	1024	5	4	4	6 ch	9 ch	1	2	1	1	-
dsPIC30F4011	40/44	48K	16K	2048	1024	5	4	4	6 ch	9 ch	1	2	1	1	1
dsPIC30F4010	64	36K	12K	2048	1024	5	8	8	8 ch	16 ch	1	2	2	1	1
dsPIC30F5010	64	96K	32K	4096	2048	5	8	8	8 ch	16 ch	1	2	2	1	2
dsPIC30F6010	80	144K	48K	8192	4096	5	8	8	8 ch	16 ch	1	2	2	1	2

FIGURE 2-1: PIN DIAGRAMS (28-Pin SDIP)

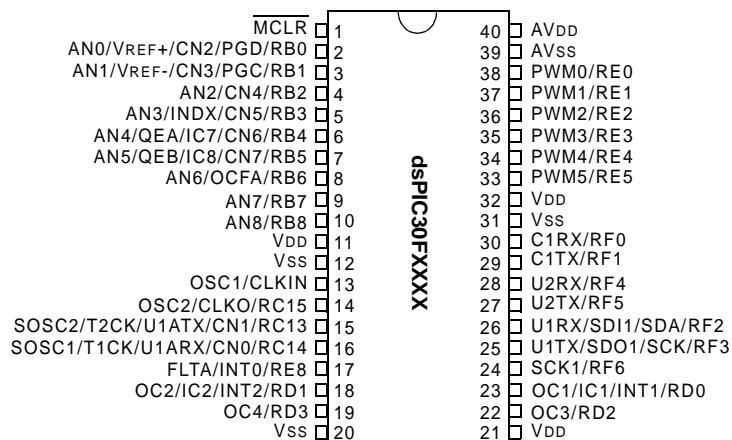


dsPIC30F

FIGURE 2-2: PIN DIAGRAMS (40-Pin PDIP, 44-Pin TQFP)

40-Pin PDIP

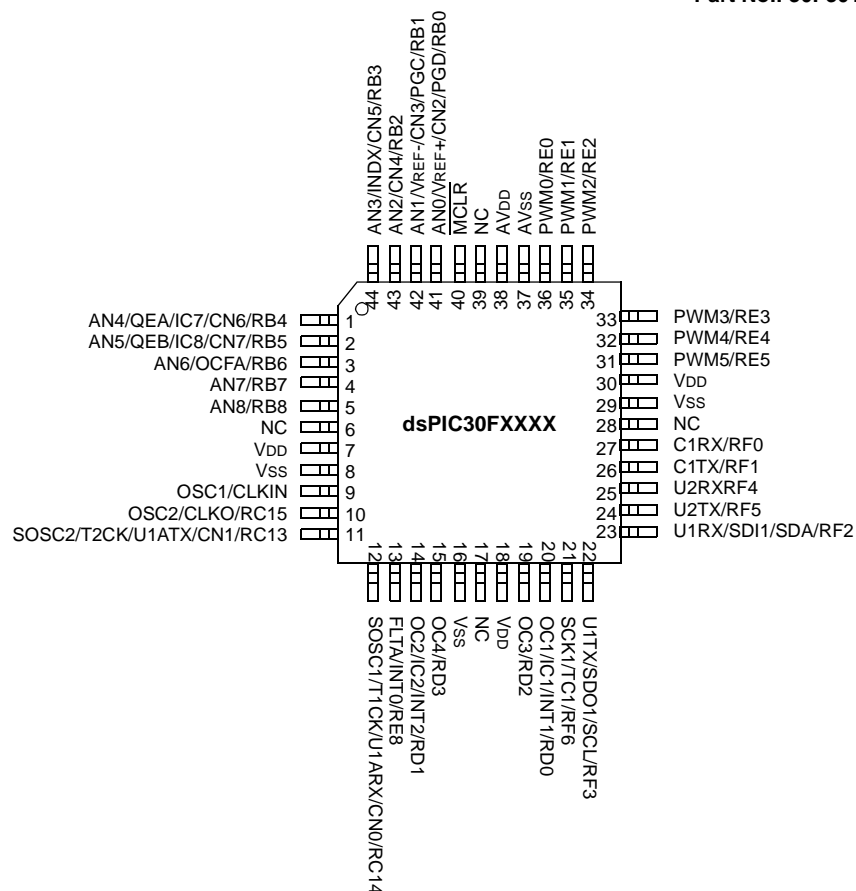
Part No.: 30F3011 / 30F4011



Note: Pinout subject to change.

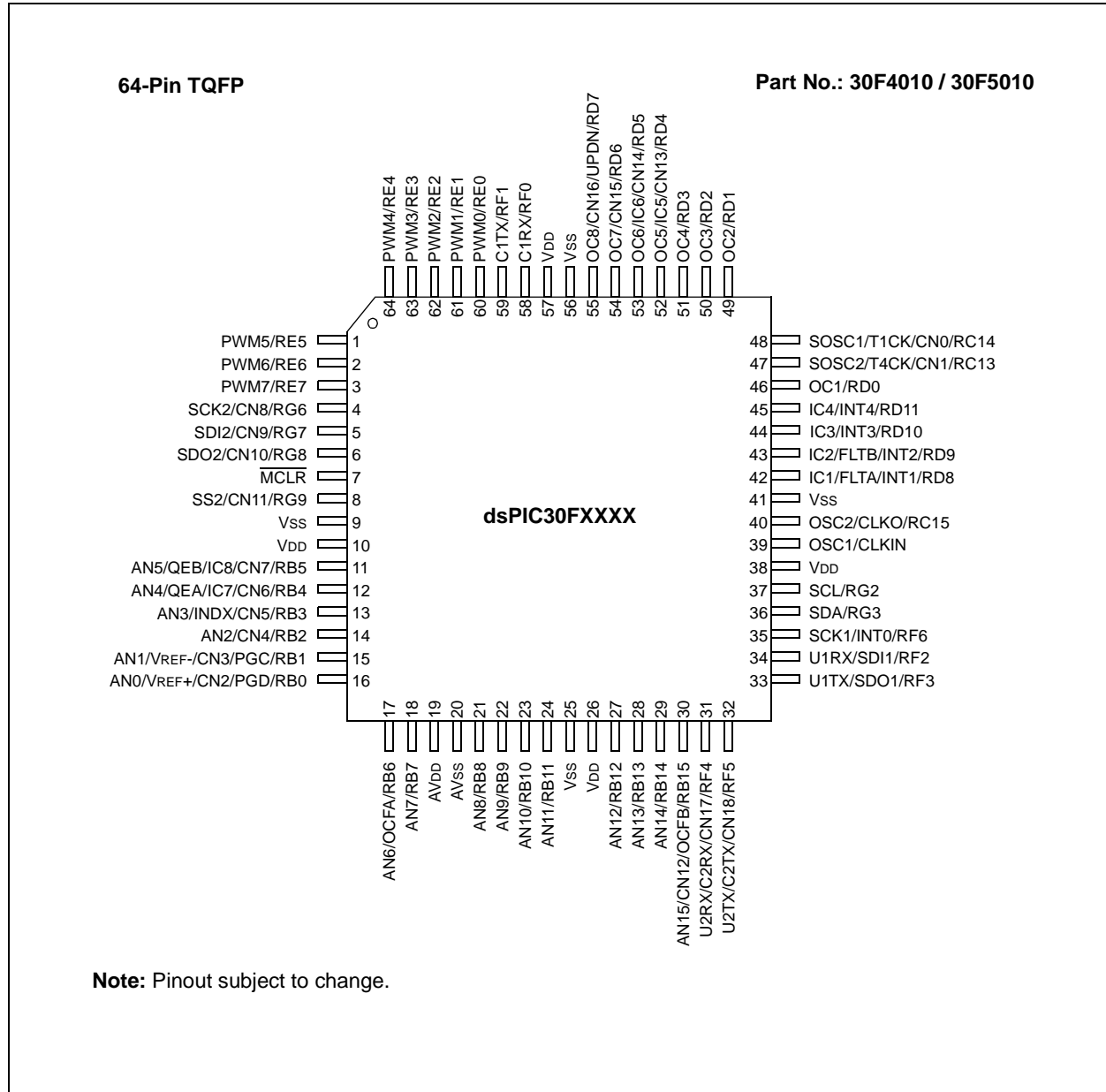
44-Pin TQFP

Part No.: 30F3011 / 30F4011



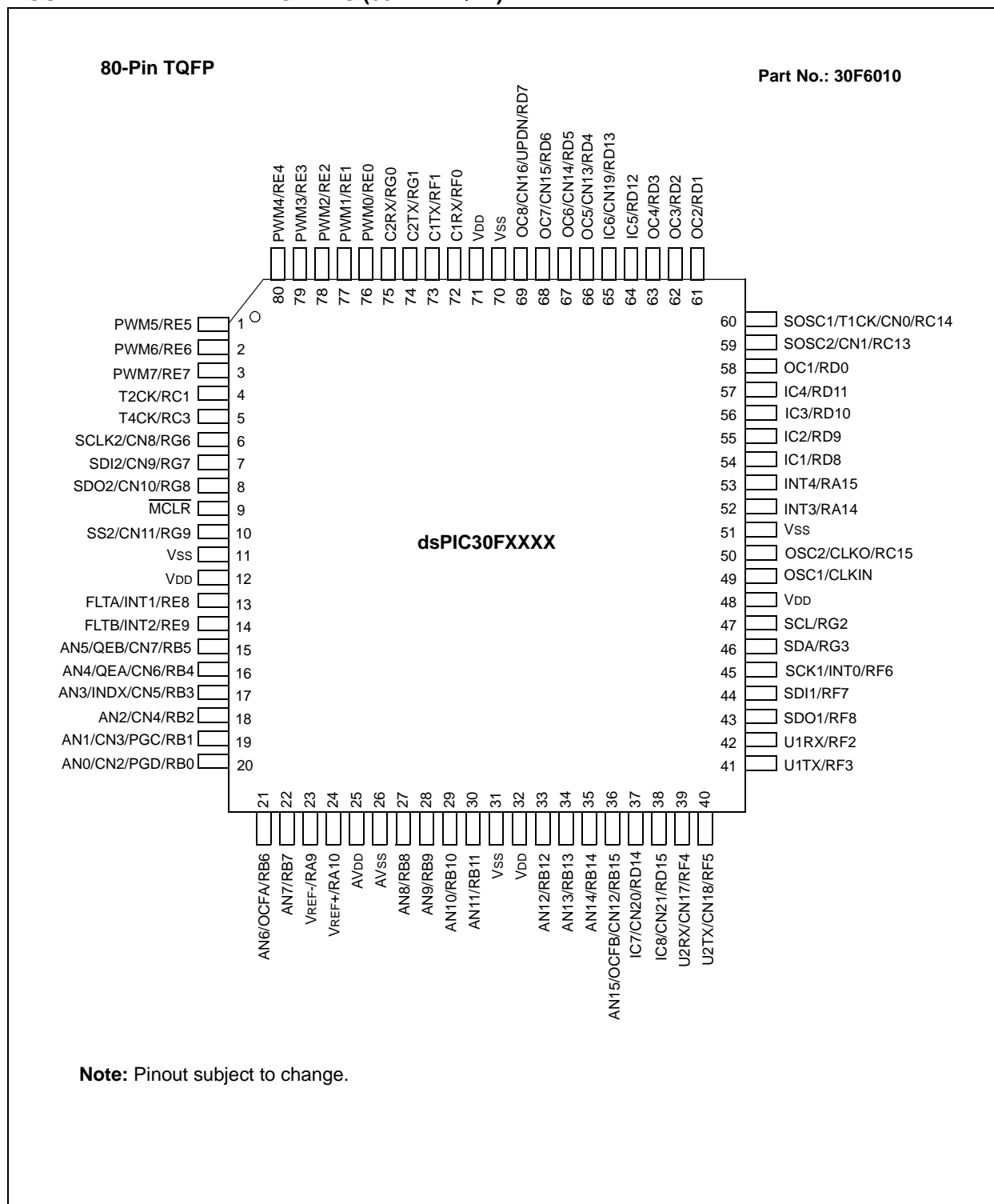
Note: Pinout subject to change.

FIGURE 2-3: PIN DIAGRAMS (64-Pin TQFP)



dsPIC30F

FIGURE 2-4: PIN DIAGRAMS (80-Pin TQFP)



3.0 CORE ARCHITECTURE OVERVIEW

3.1 Core Overview

The dsPIC30FXXX core is a 16-bit (data) modified Harvard architecture with an enhanced instruction set, including significant support for DSP. The core has a 24-bit instruction word. The Program Counter (PC) is 24-bits wide, with the Least Significant (LS) bit always clear and the Most Significant (MS) bit is ignored during normal program execution, except for certain specialized instructions. Thus, the PC can address up to 4M instruction words of user program space. An instruction pre-fetch mechanism is used to help maintain throughput. Unconditional overhead free program loop constructs are supported using the `DO` and `REPEAT` instructions, both of which are interruptible at any point.

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

The data space is 64 Kbytes (32K words), and 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 combines the X and Y memory into a single unified data space. The Multiply-Accumulate (MAC) class of dual source DSP instructions operate through both the X and Y AGUs, splitting the data address space into two parts. The X and Y data space boundary is device specific and cannot be altered by the user. Each data word consists of 2 bytes, and most instructions can address data either as words or bytes.

There are two methods of accessing data stored in program memory:

- The upper 32 Kbytes of data space memory can optionally be mapped into the lower half (user space) of program space at any 16K program word boundary defined by the 8-bit Program Space Visibility Page (PSVPAG) register. This lets any instruction access program space as if it were data space, with the sole limitation that the access requires an additional cycle. Moreover, only the lower 16 bits of each instruction word can be accessed using this method.
- Linear indirect access of 32K word pages within program space is also possible, using any working register via table read and write instructions. Table read and write instructions can be used to access all 24 bits of an instruction word.

Overhead-free circular buffers (modulo addressing) are supported in both X and Y address spaces. This is primarily intended to remove the loop overhead for DSP algorithms.

The X AGU also supports bit-reversed addressing on destination effective addresses, to greatly simplify input or output data reordering for radix-2 FFT algorithms.

The core supports Inherent (no operand), Relative, Literal, Memory Direct, Register Direct, Register Indirect and Register Offset Addressing modes. Instructions are associated with predefined addressing modes depending upon their functional requirements.

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 are supported, allowing $C=A+B$ 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. Data in the accumulator, or any working register, can be shifted up to 15 bits right or 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 in one instruction cycle. To enable this concurrent fetching of data operands, the data space is 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 does not support a multi-stage instruction pipeline. However, a single-stage instruction pre-fetch mechanism is used, which accesses and partially pre-decodes instructions a cycle ahead to maximize available execution time. Most instructions execute in a single cycle, with certain exceptions, as outlined in Section 3.1.2.

The core features a vectored exception processing structure for traps and interrupts, with 62 independent vectors. The exceptions consist of up to 8 traps (of which 3 are reserved) and 54 interrupts. Each interrupt is prioritized, based on a user assigned priority between 1 and 7 (1 being the lowest priority and 7 being the highest) in conjunction with a predetermined 'natural order'.

3.1.1 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 $C = A + B$ 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 8 Kbytes. This, together with the addition of 16-bit direct address MOV based instructions, has provided a contiguous linear addressing space.
5. Linear indirect access of 32K word (64 Kbyte) 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.

3.1.2 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 instructions executed within a REPEAT loop, which executes in 1 cycle).
3. Double-word move based instructions.

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 pre-fetched, usually by the preceding instruction, resulting in a simple out of order data processing model.

3.2 Programmer's Model

The programmer's model is shown in Figure 3-1 and consists of 16 x 16-bit working registers (W0 through W15), 2 x 40-bit accumulators (ACCA and ACCB), Status register (SR), Data Table Page register (TBLPAG), Program Space Visibility Page register (PSVPAG), DO and REPEAT registers (DOSTART, DOEND, DCOUNT and RCOUNT), and Program Counter (PC). The working registers can act as data, address or offset registers. All registers are memory mapped. W0 is the W register for all instructions that perform file register addressing.

Most of these registers have a shadow register associated with them, as shown in Figure 3-1. The shadow register is used as a temporary holding register and can transfer its contents to or from its host register upon some event occurring. None of the shadow registers are accessible directly. The following rules apply for transfer of registers into and out of shadows.

- PUSH.S and POP.S
W0...W14, TBLPAG, PSVPAG, SR (DC, N, OV, SZ and C bits only) transferred
- DO instruction
DA bit, DOSTART, DOEND, DCOUNT shadows pushed on loop start, popped on loop end

When a byte operation is performed on a working register, only the Least Significant Byte of the target register is affected. However, a benefit of memory mapped working registers is that both the Least and Most Significant Bytes can be manipulated through byte wide data memory space accesses.

3.2.1 SOFTWARE STACK POINTER/FRAME POINTER

W15 is the dedicated software stack pointer (SP), and will be automatically modified by exception processing and subroutine calls and returns. However, W15 can be referenced by any instruction in the same manner as all other W registers. This simplifies the reading, writing and manipulation of the stack pointer (e.g., creating stack frames).

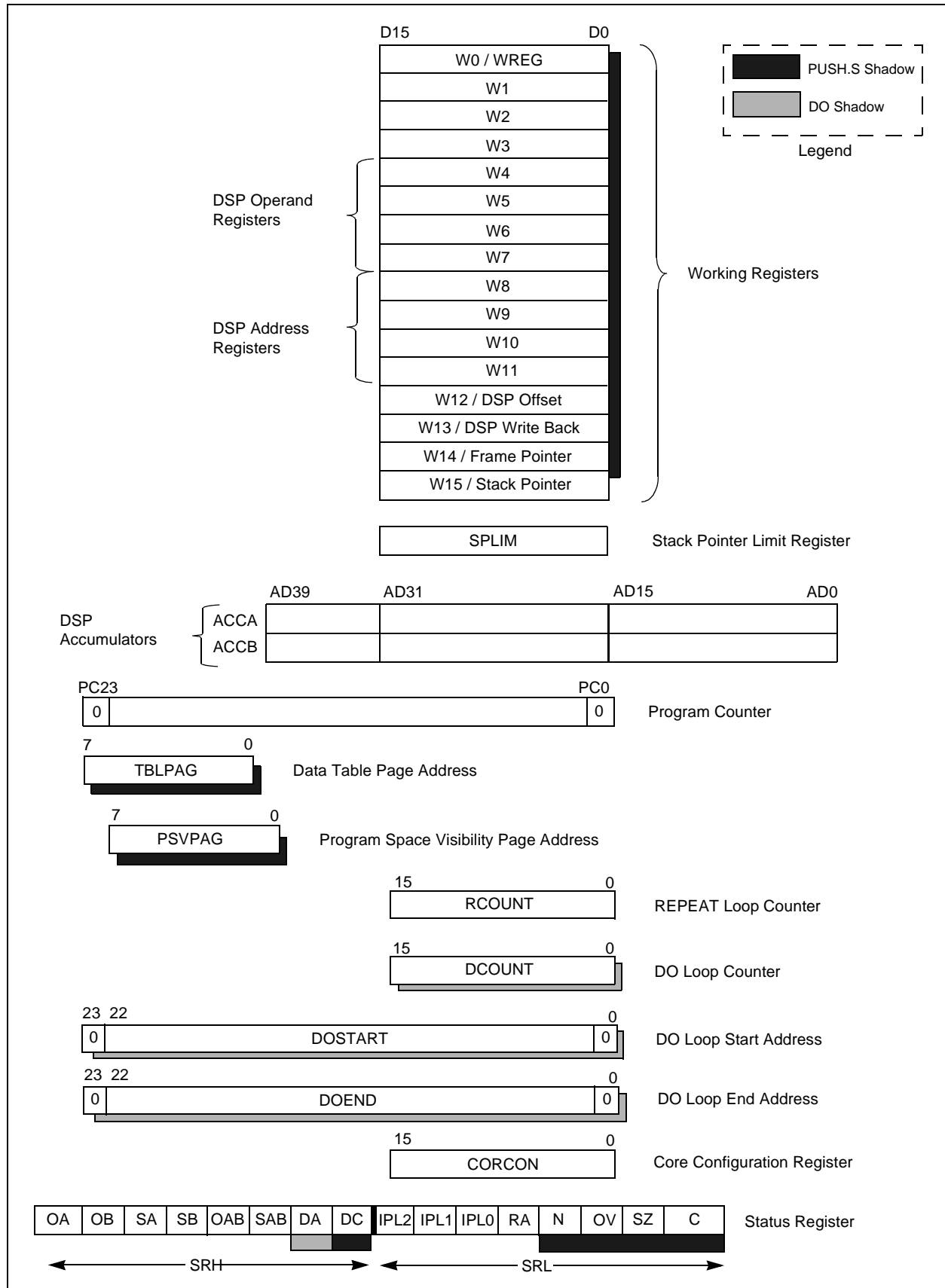
In order to protect against misaligned stack accesses, W15<0> is always clear.

W15 is initialized to 0x0800 during a RESET. The user may reprogram the SP during initialization to any location within data space greater than 0x0800.

W14 has been dedicated as a stack frame pointer, as defined by the LNK and ULNK instructions. However, W14 can be referenced by any instruction in the same manner as all other W registers.

The stack pointer always points to the first available free word and grows from lower addresses towards higher addresses. It pre-decrements for stack pops (reads) and post increments for stack pushes (writes).

FIGURE 3-1: PROGRAMMER'S MODEL



4.0 DATA ADDRESS SPACE

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

4.1 Data Spaces

The X AGU is used by all instructions and supports all addressing modes. It consists of a read AGU (X RAGU) and a write AGU (X WAGU), which operate independently during different phases of the instruction cycle. There are separate read and write data buses. The X read data bus is the return data path for all instructions that view data space as combined X and Y address space. It is also the X address space data path for the dual operand read instructions (MAC class). The X write data bus is the only write path to data space for all instructions.

The X RAGU and X WAGU also support modulo addressing for any instructions subject to addressing mode restrictions. Bit-reversed addressing is only supported by X WAGU.

The Y AGU and data path are used in concert with the X AGU by the MAC class of instructions (CLR, ED, EDAC, MAC, MOV SAC, MPY, MPY.N and MSC) to provide two concurrent data read paths. No writes occur across the Y-bus. This class of instructions dedicates two W register pointers, W10 and W11, to always operate through the Y AGU and address Y data space independent of X data space, whereas W8 and W9 operate through the X RAGU and address X data space. Note that during accumulator write back, the data address space is considered a combination of X and Y, so the write occurs across the X-bus. Consequently, it can be to any address in the entire data space.

The Y AGU only supports the 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 not user programmable. Should an EA point to data outside its own assigned address space, or to a location outside physical memory, an all zero word/byte will be returned. For example, although Y address space is visible by all non-MAC instructions using any addressing mode, an attempt by a MAC instruction to fetch data from that space using W8 or W9 (X space pointers) will return 0x0000.

TABLE 4-1: EFFECT OF INVALID MEMORY ACCESSES

Attempted Operation	Data Returned
EA=an unimplemented address	0x0000
W8 or W9 used to access Y data space in a MAC instruction	0x0000
W10 or W11 used to access X data space in a MAC instruction	0x0000

All effective addresses are 16-bits wide and point to bytes within the data space. Therefore, the data space address range is 64 Kbytes or 32K words.

4.2 Data Space Width

The core data width is 16-bits. All internal registers are organized as 16-bit wide words. Data space memory is organized in byte addressable, 16-bit wide blocks. Figure 4-1 depicts a sample data space memory map for the dsPIC30F.

4.3 Data Alignment

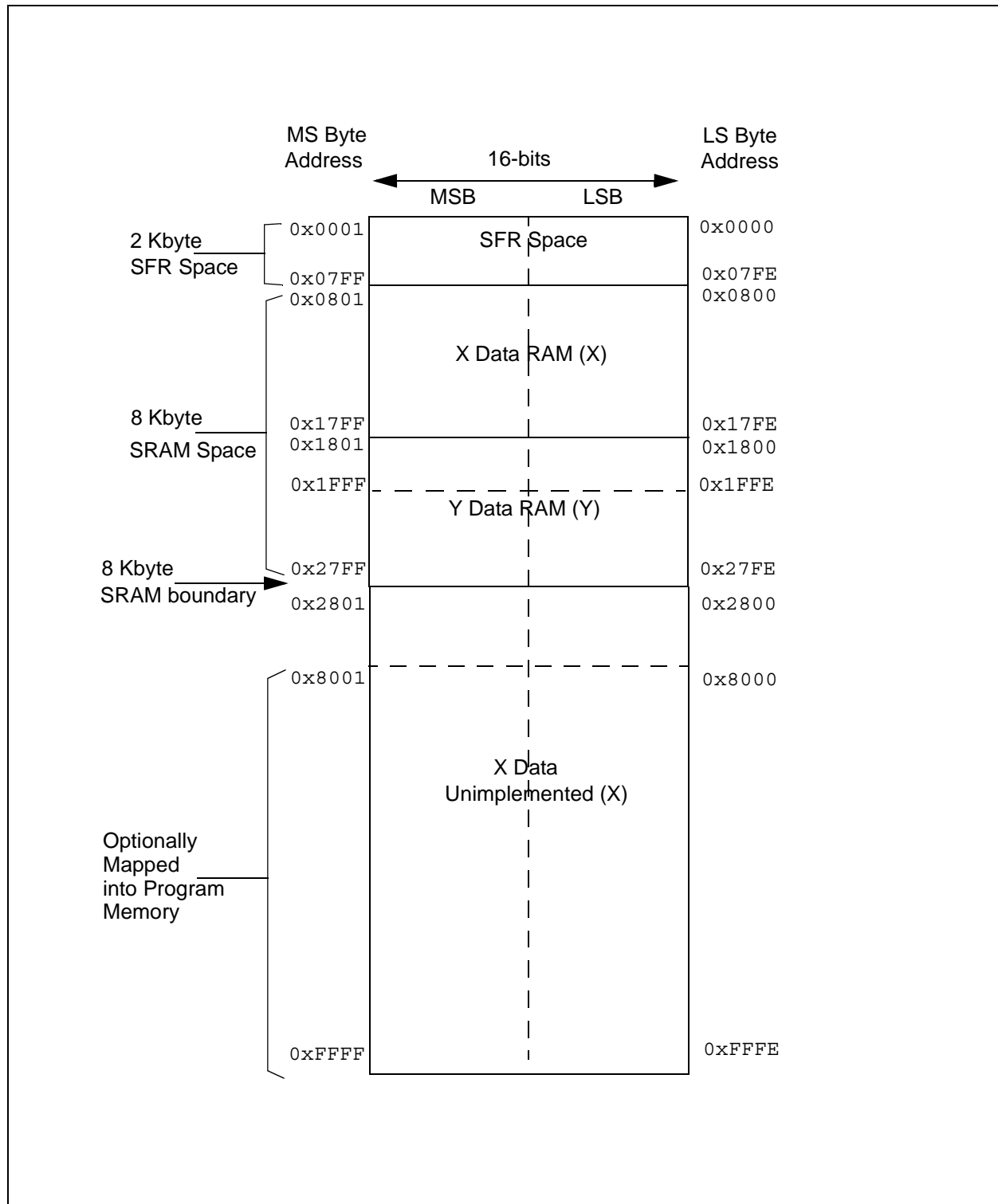
To help maintain backward compatibility with PICmicro® devices and improve data space memory usage efficiency, the dsPIC30F instruction set supports both word and byte operations. Data is aligned in data memory and registers as words, but all data space EAs resolve to bytes. Data byte reads will read 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 only write to the corresponding side of the array or register which matches the byte address.

As a consequence of this byte accessibility, all effective address calculations (including those generated by the DSP operations, which are restricted to word-sized data) are internally scaled to step through word aligned memory. For example, the core would recognize that Post-Modified Register Indirect Addressing mode, [Ws++], 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 be taken when mixing byte and word operations, or translating from 8-bit MCU code. Should a misaligned read or write be attempted, an address error trap will be forced. If the error occurred on a read,

the instruction underway is completed, whereas if it occurred on a write, the instruction will be inhibited and the PC will not be incremented. In either case, a trap will then be executed, allowing the system and/or user to examine the machine state prior to execution of the address fault.

FIGURE 4-1: SAMPLE DATA SPACE MEMORY MAP



5.0 DSP ENGINE

Concurrent operation of the DSP engine with MCU instruction flow is not possible, though both the MCU ALU and DSP engine resources may be used concurrently by the same instruction (e.g., ED and EDAC instructions).

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 accumulators, round and saturation logic. The 16-bit x 16-bit multiplier is also utilized for MCU based multiply instructions.

Data input to the DSP engine is derived from one of the following:

1. Directly from the W array (registers W4, W5, W6 or W7) via the X and Y data buses for the MAC class of instructions (MAC, MSC, MPY, MPY.N, ED, EDAC, CLR and MOVSAAC)
2. From the X-bus for all other DSP instructions
3. From the X-bus for all MCU instructions which use the barrel shifter

Data output from the DSP engine is written to one of the following:

1. The target accumulator, as defined by the DSP instruction being executed
2. The X-bus for MAC, MSC, CLR and MOVSAAC accumulator writes, where the EA is derived from W13 only (MPY, MPY.N, ED and EDAC do not offer an accumulator write option)
3. The X-bus for all MCU instructions which use the barrel shifter

The DSP engine also has the capability to perform inherent accumulator-to-accumulator operations, which require no additional data. These instructions are ADD, SUB and NEG.

A block diagram of the DSP engine is shown in Figure 5-1.

5.1 Multiplier

The 16 x 16-bit multiplier is capable of signed or unsigned operation and can multiplex its output using a scaler to support either 1.31 fractional (Q31), or 32-bit integer results. Integer data is inherently represented as a signed two's-complement value, where the MSB is defined as a sign bit. Generally speaking, the range of an N-bit two's-complement integer is -2^{N-1} to $2^{N-1}-1$. For a 16-bit integer, the data range is -32768 (0x8000) to 32767 (0x7FFF), including 0. For a 32-bit integer, the data range is -2,147,483,648 (0x8000 0000) to 2,147,483,645 (0x7FFF FFFF).

5.2 Data Accumulators and Adder/Subtractor

The data accumulator consists of a 40-bit adder/subtractor with automatic sign extension logic. It can select one of two accumulators (A or B) as its pre-accumulation source and post-accumulation destination. For the ADD and LAC instructions, the data to be accumulated or loaded can be optionally scaled via the barrel shifter prior to accumulation.

5.2.1 ADDER/SUBTRACTOR, OVERFLOW AND SATURATION

The adder/subtractor is a 40-bit adder with an optional zero input into one side and either true or complement data into the other input. In the case of addition, the carry/borrow input is active-high and the other input is true data (not complemented), whereas in the case of subtraction, the carry/borrow input is active-low and the other input is complemented. The adder/subtractor generates overflow status bits SA/SB and OA/OB, which are latched and reflected in the Status Register:

- Overflow from bit 39. This is a catastrophic overflow in which the sign of the accumulator is destroyed.
- Overflow into guard bits 32 through 39. This is a recoverable overflow. This bit is set whenever all the guard bits are not identical to each other.

The adder has an additional saturation block which controls accumulator data saturation, if selected. It uses the result of the adder, the overflow status bits described above, and the SATA/B and ACCSAT mode control bits to determine when to saturate and to what value to saturate

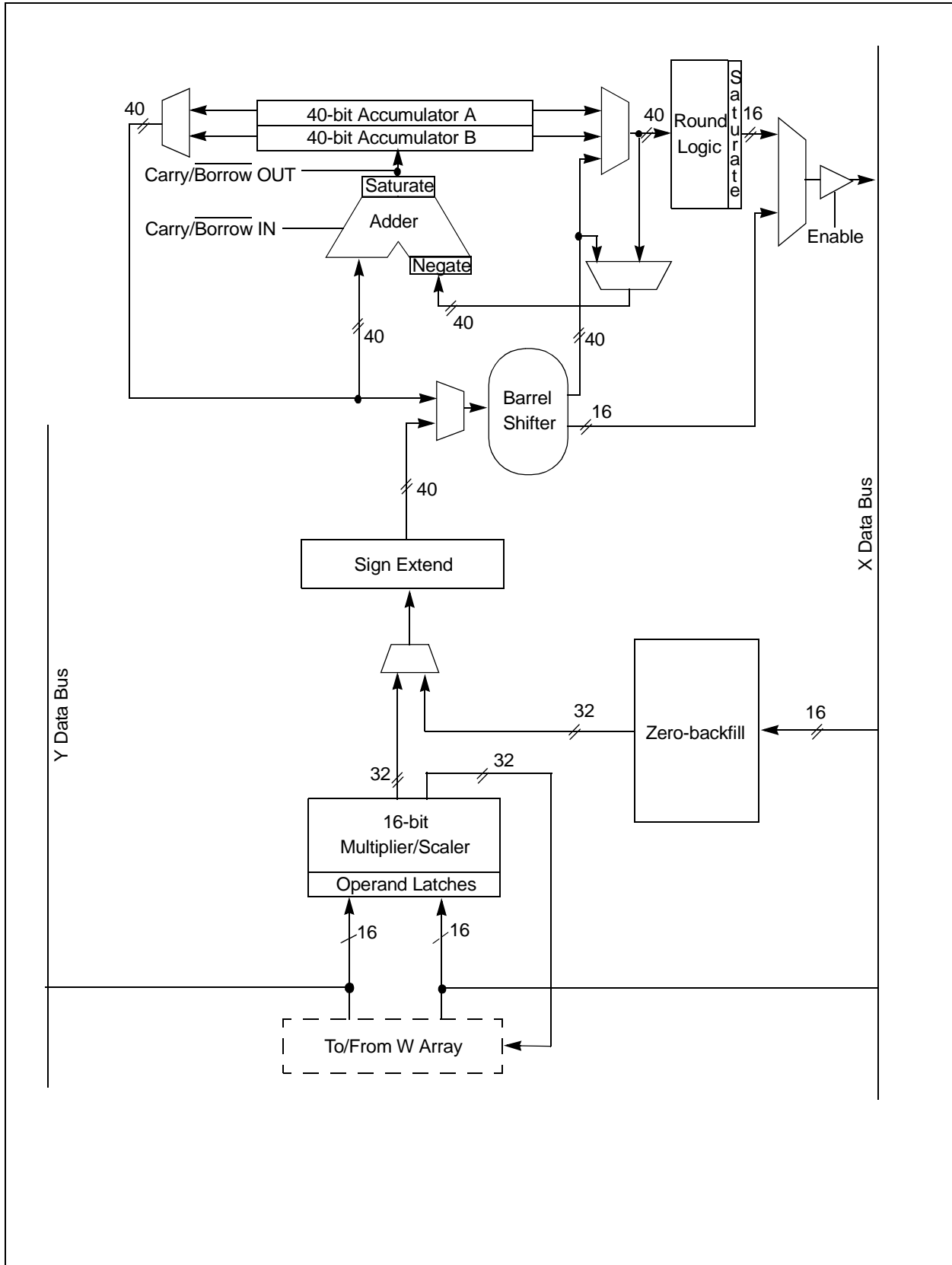
5.2.2 ROUND LOGIC

The round logic is a combinational block, which performs a conventional (biased) or convergent (unbiased) round function during an accumulator write (store). The Round mode is determined by the state of the RND bit in the CORCON register. It generates a 16-bit 1.15 data value, which is passed to the data space write saturation logic. If rounding is not indicated by the instruction, a truncated 1.15 data value is stored and the LS word is simply discarded.

5.2.3 DATA SPACE WRITE SATURATION

In addition to adder/subtractor saturation, writes to data space may also be saturated, but without affecting the contents of the source accumulator. The data space write saturation logic block accepts a 16-bit 1.15 fractional value from the round logic block as its input, together with overflow status from the original source (accumulator) and the 16-bit round adder. These are combined and used to select the appropriate 1.15 fractional value as output to write to data space memory.

FIGURE 5-1: DSP ENGINE BLOCK DIAGRAM



6.0 EXCEPTION PROCESSING

The dsPIC30F has 45 interrupt sources and 8 processor exceptions (traps), which must be arbitrated based on a priority scheme.

The processor core is responsible for reading the Interrupt Vector Table (IVT) and transferring the address contained in the interrupt vector to the program counter. The interrupt vector is transferred from the program data bus into the program counter, via a 24-bit wide multiplexer on the input of the program counter.

The Interrupt Vector Table (IVT) and Alternate Interrupt Vector Table (AIVT) are placed near the beginning of program memory (0x000004).

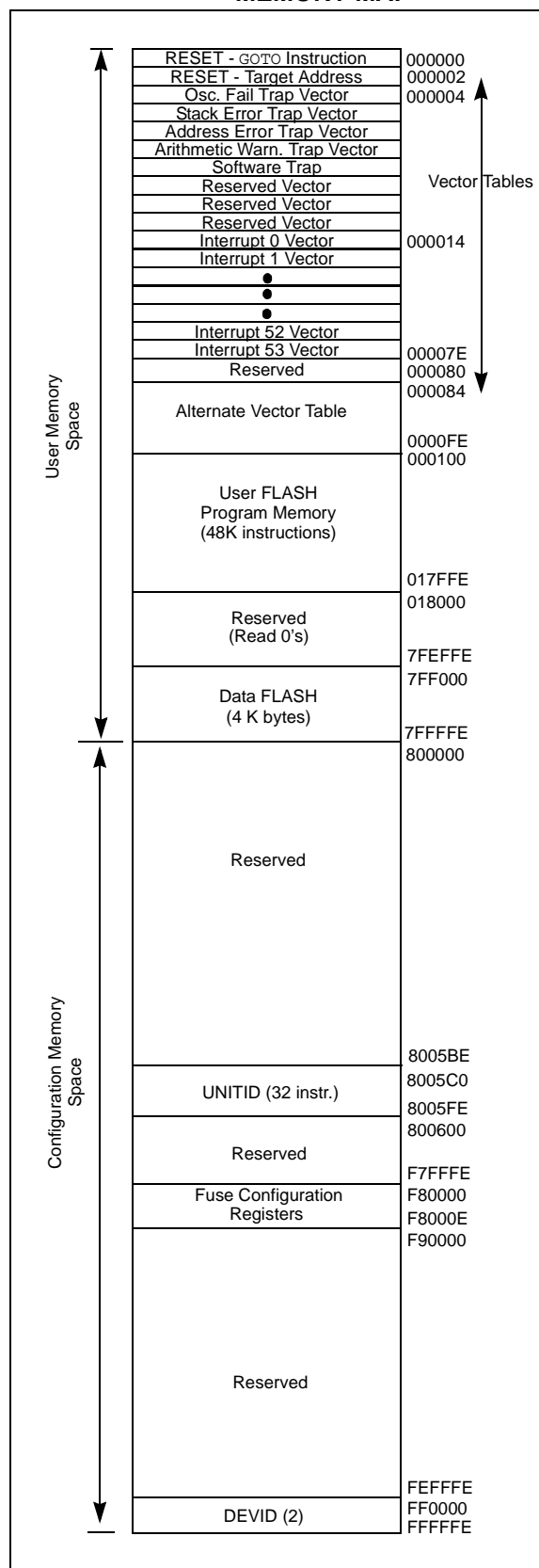
The interrupt controller and processor exceptions are responsible for pre-processing the interrupts prior to them being presented to the processor core. The interrupts and traps are enabled, prioritized, and controlled using centralized special function registers:

- IFS0<15:0>, IFS1<15:0>, IFS2<15:0>
All interrupt request flags are maintained in these three registers. The flags are set by their respective peripherals or external signals, and they are cleared via software.
- IEC0<15:0>, IEC1<15:0>, IEC2<15:0>
All interrupt enable control bits are maintained in these three registers. These control bits are used to individually enable interrupts from the peripherals or external signals.
- IPC0<15:0>... IPC11<7:0>
The user assignable priority level associated with each of these 45 interrupts is held centrally in these twelve registers.
- IPL<2:0> The current CPU priority level is explicitly stored in the 16-bit Status register that resides in the processor core.
- INTCON1<15:0>, INTCON2<15:0>
Global interrupt control functions are derived from these two registers. INTCON1 contains the control and status flags for the processor exceptions. The INTCON2 register controls the external interrupt request signal behavior and the use of the alternate vector table.

Note: Interrupt flag bits get set when an interrupt condition occurs, regardless of the state of its corresponding enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

All interrupt sources can be user assigned to one of 8 priority levels, 0 through 7 via the IPCx registers. Each interrupt source is associated with an interrupt vector. Levels 7 and 0 represent the highest and lowest maskable priorities, respectively.

FIGURE 6-1: PROGRAM SPACE MEMORY MAP



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.

The `DISI` instruction can be used to disable the processing of interrupts of priorities 6 and lower for a certain number of instructions, during which the `DISI` bit remains set.

When an interrupt is serviced, the PC is loaded with the address stored in the vector location in Program memory that corresponds to the interrupt. There are 63 different vectors within the IVT. These vectors are contained in locations 0x000004 through 0x0000FE 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 through accidentally decrementing a PC into vector space, accidentally mapping a data space address into vector space, or the PC rolling over to 0x000000 after reaching the end of implemented program memory space. Execution of a `GOTO` instruction to this vector space will also generate an address error trap.

6.1 Interrupt Priority

The user assignable Interrupt Priority Control (IPC<2:0>) bits for each individual interrupt source are located in the three LS bits of each nibble within the IPCx register(s). Bit 3 of each nibble is not used and is read as a 0. These bits define the priority level assigned to a particular interrupt by the user.

Note: The user selectable priority levels start at 0 as the lowest priority and level 7 as the highest priority.

Since more than one interrupt request source may be assigned to a specific user specified priority level, a means is provided to assign priority within a given level. This method is called "Natural Order Priority"

The Natural Order Priority of an interrupt is numerically identical to its Vector Number.

Note 1: The natural order priority scheme has 0 as the highest priority and 53 as the lowest priority.

2: The natural order priority number is the same as the vector number.

The ability for the user to assign every interrupt to one of eight priority levels implies that the user can assign a very high overall priority level to an interrupt with a low natural order priority. For example: the PLVD (Low Voltage Detect) can be given a priority of 7, and the INT0 (external interrupt 0) may be assigned to priority level 1, thus giving it a very low effective priority.

7.0 LOW VOLTAGE DETECT

In many applications, the ability to determine if the device voltage (VDD) 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 (can be any user assigned priority interrupt level) 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.

8.0 dsPIC30F PERIPHERALS

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

- System Integration Block
- 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
- Data Converter Interface
- Serial Peripheral Interface (SPI™) module
- UART module
- I²C™ module
- Controller Area Network (CAN) module
- I/O pins

8.0.1 SYSTEM INTEGRATION BLOCK

The System Integration Block contains many of the smaller functions and peripherals that provide essential services to the dsPIC30F device. The SIB contains:

- Primary Clock Oscillators (XTL, XT, HS)
- 32 kHz Clock Oscillator (LP)
- PLL Clock Multiplier
- High Speed Internal RC Oscillator (FRC)
- Watchdog Timer with low power RC Oscillator (LPRC)
- External RC Oscillator (EXTRC)
- Power-on Reset Circuit (POR)
- Programmable Brown-out Detector (BOR)
- Programmable Low Voltage Detector (LVD)

There are three primary clock oscillators: XTL, XT, HS. The XTL oscillator is designed for crystals or ceramic resonators in the range of 200 kHz to 4 MHz. The XT oscillator is designed for crystals and ceramic resonators in the range of 4 to 10 MHz. The HS (high Speed) oscillator is for crystals in the 10 to 25 MHz range. These oscillators use the OSC1 and OSC2 pins.

The secondary (LP) oscillator is designed for low power and uses a 32 kHz crystal or ceramic resonator. The LP oscillator uses the SOSC1 and SOSC2 pins.

The FRC (Fast RC) internal oscillator runs at a nominal 8 MHz.

The LPRC (Low Power RC) internal oscillator is connected to the Watchdog Timer, and it runs at a nominal 512 kHz.

The External RC (EXTRC) oscillator uses an external resistor and capacitor connected to the OSC1 pin. Frequency of operation is up to 4 MHz.

The OSC1 pin may also be used as an input from an external clock source, this mode is called "EC".

8.1 A/D Module Overview

There will be 2 versions of A/D converters available for the dsPIC30F family of devices. There is a 10-bit high speed A/D module and a 12-bit high resolution A/D module. Some of the key features for the 10 and 12-bit ADC are presented below.

8.1.1 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
- Four on-chip sample and hold amplifiers
- Single supply operation: 2.7V - 5.5V
- 500 kps sampling rate at 5V
- 100 kps sampling rate at 2.7V
- Ability to convert while the device SLEEPS
- Low power CMOS technology
- 5 nA typical standby current, 2 μ A max
- 2.5 mA typical active current at 5V

8.1.2 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
- One on-chip sample and hold amplifier
- Automated input scanning
- Single supply operation: 2.7V - 5.5V
- 100 kps sampling rate at 5V
- 50 kps sampling rate at 2.7V
- Ability to convert while the device SLEEPS
- Low power CMOS technology
- 5 nA typical standby current, 2 μ A max
- 2.5 mA max active current at 5V

8.1.3 A/D 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.

1. Result alignment options
2. Automated sampling
3. Automated input scanning
4. Dual Port data buffer
5. External conversion start control

8.2 General Purpose Timer Overview

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.

The dsPIC30F device will support up to five 16-bit timers. Four of the 16-bit timers can be configured as two 32-bit timers. Each timer has several selectable operating modes.

8.3 Watchdog Timer Module 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.

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, the Watchdog Timer is enabled. 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.

8.4 Motor Control PWM Module Overview

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)

The PWM module has the following features:

- 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
- Dead-time control for Complementary mode
- 'Special Event' comparator for scheduling other peripheral events, such as A/D conversions

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 pins 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.

8.4.1 PWM TIME-BASE

The PWM time-base is provided by a 15-bit timer with a prescaler and postscaler. Bit 15 of the PTMR SFR contains a read only status bit, PTDIR, that indicates the present count direction of the PWM time-base. If the PTDIR status bit is cleared, PTMR is counting upwards. If PTDIR is set, PTMR is counting downwards. The PWM time-base is configured via a special function register (SFR).

The PWM time-base 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 PTMOD<1:0> 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).

8.5 QEI Module Overview

The Quadrature Encoder Interface (QEI) module provides the interface to incremental encoders for obtaining motor positioning data. Incremental encoders are very useful and specific to motor control applications.

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

8.6 Input Capture Module Overview

The Input Capture module is useful in applications requiring Frequency (Period) and Pulse measurement. The dsPIC30F device will support up to eight input capture channels. The key operational features are:

- Capture every falling edge
- Capture every rising edge
- Capture every 4th rising edge
- Capture every 16th rising edge
- Capture every rising and falling edge
- Capture timer values based on internal or external clocks
- Timer2 or Timer3 time-base selection
- Device wake-up from capture pin during CPU SLEEP and IDLE modes
- Interrupt on input capture event
- 4-word FIFO buffer for capture values

These operating modes are determined by setting the appropriate control and configuration bits.

Input capture is useful for such modes as:

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

8.7 Output Compare/PWM Module Overview

The Output Compare module features are quite useful in applications requiring operational modes such as:

- Generation of variable width output pulses
- Simple PWM Operation

The dsPIC30F device will support up to eight Output Compare channels with the following key operational features:

- Timer2 and Timer3 Selection mode
- Simple Output Compare Match mode
- Dual Output Compare Match mode
- Simple Glitchless PWM mode
- Output Compare during CPU SLEEP and IDLE mode
- Interrupt on output compare/PWM event
- Interrupt on PWM fault detect condition

8.8 Data Converter Interface

The dsPIC Data Converter Interface (DCI) module allows simple interfacing of devices, such as audio coder/decoders (CODECs), A/D converters, and D/A converters. The following interfaces are supported:

- Framed Synchronous Serial Transfer (Single or Multi-Channel)
- Inter-IC Sound (I²S) Interface
- AC-link Compliant mode

The DCI module has the following hardware features:

- Programmable word size up to 16-bits.
- Support for up to 16 time slots, for a maximum frame size of 256 bits.
- Data buffering for up to 4 samples without CPU overhead

8.9 SPI™ Module Overview

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. It is compatible with Motorola's SPI and SIOP interfaces.

This SPI module includes all SPI modes. A Frame Synchronization mode is also included for support of voice band CODECs.

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.

Four 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, which also serves as the FSYNC, frame synchronization pulse.

In Master mode operation, SCK is clock output, but in Slave mode, it is clock input.

A series of eight or sixteen clock pulses (depending on mode) shift out the 8- or 16-bits from the SPISR to SDO pin and simultaneously shift in 8- or 16-bits 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.

If the receive buffer is full when the protocol needs to transfer data from SPISR to SPIBUF, the module will set the SPIROV bit, indicating an overflow condition. The module will not complete the transfer of the data from SPISR to SPIBUF. The module will not respond to SCL transitions while SPIROV is '1', effectively disabling the module until software reads SPIBUF.

Transmit writes are double buffered. The user writes to the SPIBUF. When the master or slave transfer finishes, the SPISR is swapped with the SPIBUF. This places the receive data in the SPIBUF and the transmit data in the SPISR, ready for the next transfer.

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 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 bit 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.

8.10 UART Module Overview

The dsPIC products will have one or more UART's.

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
- Fully integrated Baud Rate Generator with 16-bit prescaler
- Baud rates range from up to 2.5 Mbps and down to 38 Hz at 30 MIPs
- 4-character deep transmit data buffer
- 4-character 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
- Alternate TX/RX pins

8.11 I²C™ Module Overview

The I²C 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.

The Inter-Integrated Circuit (I²C) module offers full hardware support for both Slave and Multi-Master operations.

The key features of the I²C module are:

- I²C interface supports both Master and Slave operation.
- I²C Slave operation supports 7- and 10-bit address.
- I²C Master operation supports 7- and 10-bit address.
- I²C port allows bi-directional transfers between master and slaves.
- Serial clock synchronization for I²C port can be used as a handshake mechanism to suspend and resume serial transfer (SCLREL control).
- I²C supports Multi-Master operation. Detects bus collision and will arbitrate accordingly.
- Slew Rate Control for 100 kHz and 400 kHz bus speeds.

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.

8.12 Controller Area Network (CAN) Module Overview

The Controller Area Network (CAN) is a serial communications protocol, which efficiently supports distributed real-time control.

The dsPIC30F 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".

The module will support CAN 1.2, CAN 2.0A, CAN2.0B Passive, and CAN 2.0B Active versions of the protocol. The module features are:

- 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 Loopback mode and programmable state clocking supports self-test operation
- Signaling via interrupt capabilities for all CAN receiver and transmitter error states

8.13 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.

8.13.1 I/O PIN FEATURES

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

8.13.2 I/O Port Latch

I/O port pins have latch bits (LAT register). The LAT 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.

9.0 dsPIC30F INSTRUCTION SET

9.1 Introduction

The dsPIC30F instruction set provides a broad suite of instructions, which supports traditional microcontroller applications and a class of instructions, which supports math intensive applications. Since almost all of the functionality of the PICmicro instruction set has been maintained, this hybrid instruction set allows a friendly DSP migration path for users already familiar with the PICmicro microcontroller.

9.2 Instruction Set Overview

The dsPIC30F instruction set contains 89 instructions, which can be grouped into the ten functional categories shown in Table 9-1. Table 9-1 defines the symbols used in the instruction summary tables, Table 9-3 through Table 9-12. These tables define the syntax, description, storage and execution requirements for each instruction. Storage requirements are represented in 24-bit instruction words, and execution requirements are represented in instruction cycles. Most instructions have several different Addressing modes and execution flows, which require different instruction variants. For instance, there are six unique ADD instructions and each instruction variant has its own instruction encoding.

TABLE 9-1: dsPIC30F INSTRUCTION GROUPS

Functional Group	Summary Table
Move Instructions	Table 9-3
Logic Instructions	Table 9-4
Rotate/Shift Instructions	Table 9-5
Bit Instructions	Table 9-6
Compare/Skip Instructions	Table 9-7
Flow Instructions	Table 9-8
Shadow/Stack Instructions	Table 9-9
Control Instructions	Table 9-10
Control Instructions	Table 9-11
DSP Instructions	Table 9-12

9.2.1 MULTI-CYCLE INSTRUCTIONS

As the instruction summary tables show, most instructions execute in a single cycle, with the following exceptions:

- Instructions `DO`, `MOV.D`, `POP.D`, `PUSH.D`, `TBLRDH`, `TBLRDL`, `TBLWTH`, and `TBLWTL` require 2 cycles to execute.
- Instructions `DIVF`, `DIV.S`, `DIV.U` are single cycle instructions, which should be executed 18 consecutive times as the target `REPEAT` instruction.
- Instructions that change the program counter also require 2 cycles to execute, with the extra cycle executed as a `NOP`. Skip instructions, which skip over a 2-word instruction, require 3 instruction cycles to execute, with 2 cycles executed as a `NOP`.
- The `RETFIE`, `RETLW`, and `RETURN` are a special case of an instruction that changes the program counter. These execute in 3 cycles, unless an exception is pending and then they execute in 2 cycles.
- The `TRAP` instruction is a special case of an instruction that changes the program counter. It executes in 5 cycles, which includes the pre-fetch of the first instruction of the exception handler.

Note: Instructions which access program memory as data, using Program Space Visibility, will incur a one cycle delay. However, when the target instruction of a `REPEAT` loop accesses program memory as data, only the first execution of the target instruction is subject to the delay. See the dsPIC30F Data Sheet for details.

9.2.2 MULTI-WORD INSTRUCTIONS

As the instruction summary tables show, almost all instructions consume one instruction word (24 bits), with the exception of the `CALL`, `DO` and `GOTO` instructions, which are Flow Instructions listed in Table 9-9. These instructions require two words of memory, because their opcodes embed large literal operands.

TABLE 9-2: SYMBOLS USED IN SUMMARY TABLES

Symbol	Description
#	Literal operand designation
Acc	Accumulator A or Accumulator B
AWB	Accumulator Write Back
bit3	3-bit wide bit position (0:7)
bit4	4-bit wide bit position (0:15)
Expr	Absolute address, label or expression (resolved by the linker)
f	File register address
lit1	1-bit literal (0:1)
lit4	4-bit literal (0:15)
lit5	5-bit literal (0:31)
lit8	8-bit literal (0:255)
lit10	10-bit literal (0:255 for Byte mode, 0:1023 for Word mode)
lit14	14-bit literal (0:16383)
lit16	16-bit literal (0:65535)
lit23	23-bit literal (0:8388607)
Slit4	Signed 4-bit literal (-8:7)
Slit5	Signed 5-bit literal (-16:15)
Slit10	Signed 10-bit literal (-512:511)
Slit16	Signed 16-bit literal (-32768:32767)
TOS	Top-of-Stack
Wb	Base working register
Wd	Destination working register (direct and indirect addressing)
Wdo	Destination working register (direct and indirect addressing with Offset mode)
Wm, Wn	Working register divide pair (dividend, divisor)
Wm*Wm	Working register multiplier pair (same source register)
Wm*Wn	Working register multiplier pair (different source registers)
Wn	Both source and destination working register (direct addressing)
Wnd	Destination working register (direct addressing)
Wns	Source working register (direct addressing)
WREG	Default working register
Ws	Source working register (direct and indirect addressing)
Wso	Source working register (direct and indirect addressing with Offset mode)
Wx	Source Addressing mode and working register for X data bus pre-fetch
Wxd	Destination working register for X data bus pre-fetch
Wy	Source Addressing mode and working register for Y data bus pre-fetch
Wyd	Destination working register for Y data bus pre-fetch

TABLE 9-3: MOVE INSTRUCTIONS

Assembly Syntax	Description	Words	Cycles
EXCH Wns,Wnd	Swap Wns and Wnd	1	1
MOV f {,WREG}	Move f to destination	1	1
MOV WREG,f	Move WREG to f	1	1
MOV f,Wnd	Move f to Wnd	1	1
MOV Wns,f	Move Wns to f	1	1
MOV.b #lit8,Wnd	Move 8-bit literal to Wnd	1	1
MOV #lit16,Wnd	Move 16-bit literal to Wnd	1	1
MOV [Ws+Slit10],Wnd	Move [Ws + signed 10-bit offset] to Wnd	1	1
MOV Wns,[Wd+Slit10]	Move Wns to [Wd + signed 10-bit offset]	1	1
MOV Wso,Wdo	Move Wso to Wdo	1	1
MOV.D Ws,Wnd	Move double Ws to Wnd:Wnd+1	1	2
MOV.D Wns,Wd	Move double Wns:Wns+1 to Wd	1	2
SWAP Wn	Wn = byte or nibble swap Wn	1	1
TBLRDH Ws,Wd	Read high program word to Wd	1	2
TBLRDL Ws,Wd	Read low program word to Wd	1	2
TBLWTH Ws,Wd	Write Ws to high program word	1	2
TBLWTL Ws,Wd	Write Ws to low program word	1	2
Note: When the optional {,WREG} operand is specified, the destination of the instruction is WREG. When {,WREG} is not specified, the destination of the instruction is the file register f.			

TABLE 9-4: MATH INSTRUCTIONS

Assembly Syntax	Description	Words	Cycles
ADD f {,WREG}	Destination = f + WREG	1	1
ADD #lit10,Wn	Wn = lit10 + Wn	1	1
ADD Wb,#lit5,Wd	Wd = Wb + lit5	1	1
ADD Wb,Ws,Wd	Wd = Wb + Ws	1	1
ADDC f {,WREG}	Destination = f + WREG + (C)	1	1
ADDC #lit10,Wn	Wn = lit10 + Wn + (C)	1	1
ADDC Wb,#lit5,Wd	Wd = Wb + lit5 + (C)	1	1
ADDC Wb,Ws,Wd	Wd = Wb + Ws + (C)	1	1
DAW.B Wn	Wn = decimal adjust Wn	1	1
DEC f {,WREG}	Destination = f - 1	1	1
DEC Ws,Wd	Wd = Ws - 1	1	1
DEC2 f {,WREG}	Destination = f - 2	1	1
DEC2 Ws,Wd	Wd = Ws - 2	1	1
DIV.S Wm,Wn	Signed 16/16-bit integer divide	1	18
DIV.SD Wm,Wn	Signed 32/16-bit integer divide	1	18
DIV.U Wm,Wn	Unsigned 16/16-bit integer divide	1	18
DIV.UD Wm,Wn	Unsigned 32/16-bit integer divide	1	18
DIVF Wm,Wn	Signed 16/16-bit fractional divide	1	18
INC f {,WREG}	Destination = f + 1	1	1
INC Ws,Wd	Wd = Ws + 1	1	1
INC2 f {,WREG}	Destination = f + 2	1	1
INC2 Ws,Wd	Wd = Ws + 2	1	1
MUL f	W3:W2 = f * WREG	1	1
MUL.SS Wb,Ws,Wnd	{Wnd+1,Wnd} = sign(Wb) * sign(Ws)	1	1
MUL.SU Wb,#lit5,Wnd	{Wnd+1,Wnd} = sign(Wb) * unsign(lit5)	1	1
MUL.SU Wb,Ws,Wnd	{Wnd+1,Wnd} = sign(Wb) * unsign(Ws)	1	1
MUL.US Wb,Ws,Wnd	{Wnd+1,Wnd} = unsign(Wb) * sign(Ws)	1	1
MUL.UU Wb,#lit5,Wnd	{Wnd+1,Wnd} = unsign(Wb) * unsign(lit5)	1	1
MUL.UU Wb,Ws,Wnd	{Wnd+1,Wnd} = unsign(Wb) * unsign(Ws)	1	1
SE Ws,Wnd	Wnd = sign-extended Ws	1	1
SUB f {,WREG}	Destination = f - WREG	1	1
SUB Wn,#lit10	Wn = Wn - lit10	1	1
SUB Wb,#lit5,Wd	Wd = Wb - lit5	1	1
SUB Wb,Ws,Wd	Wd = Wb - Ws	1	1
SUBB f {,WREG}	Destination = f - WREG - (C)	1	1
SUBB Wn,#lit10	Wn = Wn - lit10 - (C)	1	1
SUBB Wb,#lit5,Wd	Wd = Wb - lit5 - (\overline{C})	1	1
SUBB Wb,Ws,Wd	Wd = Wb - Ws - (\overline{C})	1	1
SUBBR f {,WREG}	Destination = WREG - f - (\overline{C})	1	1
SUBBR Wb,#lit5,Wd	Wd = lit5 - Wb - (\overline{C})	1	1
SUBBR Wb,Ws,Wd	Wd = Ws - Wb - (\overline{C})	1	1
SUBR f {,WREG}	Destination = WREG - f	1	1
SUBR Wb,#lit5,Wd	Wd = lit5 - Wb	1	1
SUBR Wb,Ws,Wd	Wd = Ws - Wb	1	1
ZE Ws,Wnd	Wnd = zero-extended Ws	1	1

TABLE 9-5: LOGIC INSTRUCTIONS

Assembly Syntax	Description	Words	Cycles
AND f {,WREG}	Destination = f .AND. WREG	1	1
AND #lit10,Wn	Wn = lit10 .AND. Wn	1	1
AND Wb,#lit5,Wd	Wd = Wb .AND. lit5	1	1
AND Wb,Ws,Wd	Wd = Wb .AND. Ws	1	1
CLR f	f = 0x0000	1	1
CLR WREG	WREG = 0x0000	1	1
CLR Wd	Wd = 0x0000	1	1
COM f {,WREG}	Destination = \overline{f}	1	1
COM Ws,Wd	Wd = \overline{Ws}	1	1
IOR f {,WREG}	Destination = f .IOR. WREG	1	1
IOR #lit10,Wn	Wn = lit10 .IOR. Wn	1	1
IOR Wb,#lit5,Wd	Wd = Wb .IOR. lit5	1	1
IOR Wb,Ws,Wd	Wd = Wb .IOR. Ws	1	1
NEG f {,WREG}	Destination = $\overline{f} + 1$	1	1
NEG Ws,Wd	Wd = $\overline{Ws} + 1$	1	1
SETM f	f = 0xFFFF	1	1
SETM WREG	WREG = 0xFFFF	1	1
SETM Wd	Wd = 0xFFFF	1	1
XOR f {,WREG}	Destination = f .XOR. WREG	1	1
XOR #lit10,Wn	Wn = lit10 .XOR. Wn	1	1
XOR Wb,#lit5,Wd	Wd = Wb .XOR. lit5	1	1
XOR Wb,Ws,Wd	Wd = Wb .XOR. Ws	1	1
Note: When the optional {,WREG} operand is specified, the destination of the instruction is WREG. When {,WREG} is not specified, the destination of the instruction is the file register f.			

TABLE 9-6: ROTATE/SHIFT INSTRUCTIONS

Assembly Syntax	Description	Words	Cycles
ASR f {,WREG}	Destination = arithmetic right shift f	1	1
ASR Ws,Wd	Wd = arithmetic right shift Ws	1	1
ASR Wb,#lit4,Wnd	Wnd = arithmetic right shift Wb by lit4	1	1
ASR Wb,Wns,Wnd	Wnd = arithmetic right shift Wb by Wns	1	1
LSR f {,WREG}	Destination = logical right shift f	1	1
LSR Ws,Wd	Wd = logical right shift Ws	1	1
LSR Wb,#lit4,Wnd	Wnd = logical right shift Wb by lit4	1	1
LSR Wb,Wns,Wnd	Wnd = logical right shift Wb by Wns	1	1
RLC f {,WREG}	Destination = rotate left through Carry f	1	1
RLC Ws,Wd	Wd = rotate left through Carry Ws	1	1
RLNC f {,WREG}	Destination = rotate left (no Carry) f	1	1
RLNC Ws,Wd	Wd = rotate left (no Carry) Ws	1	1
RRC f {,WREG}	Destination = rotate right through Carry f	1	1
RRC Ws,Wd	Wd = rotate right through Carry Ws	1	1
RRNC f {,WREG}	Destination = rotate right (no Carry) f	1	1
RRNC Ws,Wd	Wd = rotate right (no Carry) Ws	1	1
SL f {,WREG}	Destination = left shift f	1	1
SL Ws,Wd	Wd = left shift Ws	1	1
SL Wb,#lit4,Wnd	Wnd = left shift Wb by lit4	1	1
SL Wb,Wns,Wnd	Wnd = left shift Wb by Wns	1	1
Note: When the optional {,WREG} operand is specified, the destination of the instruction is WREG. When {,WREG} is not specified, the destination of the instruction is the file register f.			

TABLE 9-7: BIT INSTRUCTIONS

Assembly Syntax	Description	Words	Cycles
BCLR.b f,#bit3	Bit clear f	1	1
BCLR Ws,#bit4	Bit clear Ws	1	1
BSET.b f,#bit3	Bit set f	1	1
BSET Ws,#bit4	Bit set Ws	1	1
BSW.C Ws,Wb	Write C bit to Ws<Wb>	1	1
BSW.Z Ws,Wb	Write \overline{SZ} bit to Ws<Wb>	1	1
BTG.b f,#bit3	Bit toggle f	1	1
BTG Ws,#bit4	Bit toggle Ws	1	1
BTST.b f,#bit3	Bit test f	1	1
BTST.C Ws,#bit4	Bit test Ws to C	1	1
BTST.Z Ws,#bit4	Bit test Ws to SZ	1	1
BTST.C Ws,Wb	Bit test Ws<Wb> to C	1	1
BTST.Z Ws,Wb	Bit test Ws<Wb> to SZ	1	1
BTSTS.b f,#bit3	Bit test f then set f	1	1
BTSTS.C Ws,#bit4	Bit test Ws to C then set Ws	1	1
BTSTS.Z Ws,#bit4	Bit test Ws to SZ then set Ws	1	1
FBCL Ws,Wnd	Find bit change from left (MSb) side	1	1
FF1L Ws,Wnd	Find first one from left (MSb) side	1	1
FF1R Ws,Wnd	Find first one from right (LSb) side	1	1
Note: Bit positions are specified by bit3 (0:7) for byte operations, and bit4 (0:15) for word operations.			

TABLE 9-8: COMPARE/SKIP INSTRUCTIONS

Assembly Syntax		Description	Words	Cycles
BTSC.b	f,#bit3	Bit test f, skip if clear	1	1 (2 or 3)
BTSC	Ws,#bit4	Bit test Ws, skip if clear	1	1 (2 or 3)
BTSS.b	f,#bit3	Bit test f, skip if set	1	1 (2 or 3)
BTSS	Ws,#bit4	Bit test Ws, skip if set	1	1 (2 or 3)
CP	f	Compare (f - WREG)	1	1
CP	Wb,#lit5	Compare (Wb - lit5)	1	1
CP	Wb,Ws	Compare (Wb - Ws)	1	1
CP0	f	Compare (f - 0x0000)	1	1
CP0	Ws	Compare (Ws - 0x0000)	1	1
CPB	f	Compare with Borrow (f - WREG - \overline{C})	1	1
CPB	Wb,#lit5	Compare with Borrow (Wb - lit5 - \overline{C})	1	1
CPB	Wb,Ws	Compare with Borrow (Wb - Ws - \overline{C})	1	1
CPSEQ	f	Compare (f - WREG), skip if =	1	1 (2 or 3)
CPSGT	f	Compare (f - WREG), skip if >	1	1 (2 or 3)
CPSLT	f	Compare (f - WREG), skip if <	1	1 (2 or 3)
CPSNE	f	Compare (f - WREG), skip if \neq	1	1 (2 or 3)
DECSNZ	f {,WREG}	Destination = f-1, skip if not 0	1	1 (2 or 3)
DECSZ	f {,WREG}	Destination = f-1, skip if 0	1	1 (2 or 3)
INCSNZ	f {,WREG}	Destination = f+1, skip if not 0	1	1 (2 or 3)
INCSZ	f {,WREG}	Destination = f+1, skip if 0	1	1 (2 or 3)
<p>Note 1: Bit positions are specified by bit3 (0:7) for byte operations, and bit4 (0:15) for word operations.</p> <p>2: Conditional skip instructions execute in 1 cycle if the skip is not taken, 2 cycles if the skip is taken over a one-word instruction and 3 cycles if the skip is taken over a two-word instruction.</p> <p>3: When the optional {,WREG} operand is specified, the destination of the instruction is WREG. When {,WREG} is not specified, the destination of the instruction is the file register f.</p>				

TABLE 9-9: FLOW INSTRUCTIONS

Assembly Syntax		Description	Words	Cycles
BRA	Expr	Branch unconditionally	1	2
BRA	Wn	Computed branch	1	2
BRA	C,Expr	Branch if Carry (no Borrow)	1	1 (2)
BRA	GE,Slit16	Branch if greater than or equal	1	1 (2)
BRA	GEU,Expr	Branch if unsigned greater than or equal	1	1 (2)
BRA	GT,Expr	Branch if greater than	1	1 (2)
BRA	GTU,Expr	Branch if unsigned greater than	1	1 (2)
BRA	LE,Expr	Branch if less than or equal	1	1 (2)
BRA	LEU,Expr	Branch if unsigned less than or equal	1	1 (2)
BRA	LT,Expr	Branch if less than	1	1 (2)
BRA	LTU,Expr	Branch if unsigned less than	1	1 (2)
BRA	N,Expr	Branch if Negative	1	1 (2)
BRA	NC,Expr	Branch if not Carry (Borrow)	1	1 (2)
BRA	NN,Expr	Branch if not Negative	1	1 (2)
BRA	NOV,Expr	Branch if not Overflow	1	1 (2)
BRA	NZ,Expr	Branch if not Zero	1	1 (2)
BRA	OA,Expr	Branch if Accumulator A Overflow	1	1 (2)
BRA	OB,Expr	Branch if Accumulator B Overflow	1	1 (2)
BRA	OV,Expr	Branch if Overflow	1	1 (2)
BRA	SA,Expr	Branch if Accumulator A Saturate	1	1 (2)
BRA	SB,Expr	Branch if Accumulator B Saturate	1	1 (2)
BRA	Z,Expr	Branch if Zero	1	1 (2)
CALL	Expr	Call subroutine	2	2
CALL	Wn	Call indirect subroutine	1	2
DO	#lit14,Expr	Do code through PC+Expr, (lit14+1) times	2	2+loop
DO	Wn,Expr	Do code through PC+Expr, (Wn+1) times	2	2+loop
GOTO	Expr	Go to address	2	2
GOTO	Wn	Go to address indirectly	1	2
RCALL	Expr	Relative call	1	2
RCALL	Wn	Computed call	1	2
REPEAT	#lit14	Repeat next instruction (lit14+1) times	1	2 + lit14
REPEAT	Wn	Repeat next instruction (Wn+1) times	1	2 + (Wn)
RETFIE		Return from interrupt enable	1	3 (2)
RETLW	#lit10,Wn	Return with lit10 in Wn	1	3 (2)
RETURN		Return from subroutine	1	3 (2)
TRAP	#lit16	Software trap with lit16	1	5

Note 1: Conditional branch instructions execute in 1 cycle if the branch is not taken, or 2 cycles if the branch is taken.

2: RETURN normally executes in 3 cycles. However, it executes in 2 cycles if an interrupt is pending.

3: TRAP execution time includes exception processing.

TABLE 9-10: SHADOW/STACK INSTRUCTIONS

Assembly Syntax	Description	Words	Cycles
LNK #lit14	Link frame pointer	1	1
POP f	Pop TOS to f	1	1
POP Wdo	Pop TOS to Wdo	1	1
POP.D Wnd	Double pop from TOS to Wnd:Wnd+1	1	2
POP.S	Pop shadow registers	1	1
PUSH f	Push f to TOS	1	1
PUSH Wso	Push Wso to TOS	1	1
PUSH.D Wns	Push double Wns:Wns+1 to TOS	1	2
PUSH.S	Push shadow registers	1	1
ULNK	Unlink frame pointer	1	1

TABLE 9-11: CONTROL INSTRUCTIONS

Assembly Syntax	Description	Words	Cycles
CLRWDT	Clear Watchdog Timer	1	1
DISI #lit14	Disable interrupts for (lit14+1) instruction cycles	1	1
NOP	No operation	1	1
NOPR	No operation	1	1
PWRSV #lit1	Enter Power Saving mode lit1	1	1
RESET	Software device RESET	1	1+T
Note: T for RESET execution time is 250 nsec nominal.			

TABLE 9-12: DSP INSTRUCTIONS

Assembly Syntax	Description	Words	Cycles
ADD Acc	Add accumulators	1	1
ADD Wso,#Slit4,Acc	16-bit signed add to Acc	1	1
CLR Acc,Wx,Wxd,Wy,Wyd,AWB	Clear Acc	1	1
ED Wm*Wm,Acc,Wx,Wy,Wxd	Euclidean distance	1	1
EDAC Wm*Wm,Acc,Wx,Wy,Wxd	Euclidean distance and accumulate	1	1
LAC Wso,#Slit4,Acc	Load Acc	1	1
MAC Wm*Wn,Acc,Wx,Wxd,Wy,Wyd,AWB	Multiply and accumulate	1	1
MAC Wm*Wm,Acc,Wx,Wxd,Wy,Wyd	Square and accumulate	1	1
MOVSAC Acc,Wx,Wxd,Wy,Wyd,AWB	Move Wx to Wxd and Wy to Wyd	1	1
MPY Wm*Wn,Acc,Wx,Wxd,Wy,Wyd	Multiply Wn by Wm to Acc	1	1
MPY Wm*Wm,Acc,Wx,Wxd,Wy,Wyd	Square to Acc	1	1
MPY.N Wm*Wn,Acc,Wx,Wxd,Wy,Wyd	-(Multiply Wn by Wm) to Acc	1	1
MSC Wm*Wn,Acc,Wx,Wxd,Wy,Wyd,AWB	Multiply and subtract from Acc	1	1
NEG Acc	Negate Acc	1	1
SAC Acc,#Slit4,Wdo	Store Acc	1	1
SAC.R Acc,#Slit4,Wdo	Store rounded Acc	1	1
SFTAC Acc,#Slit5	Arithmetic shift Acc by Slit5	1	1
SFTAC Acc,Wn	Arithmetic shift Acc by (Wn)	1	1
SUB Acc	Subtract accumulators	1	1

10.0 DEVELOPMENT TOOLS SUPPORT

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

- MPLAB® GL 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

10.1 MPLAB GL Integrated Development Environment Software

The MPLAB GL Integrated Development Environment is available at no cost. MPLAB GL Integrated Development Environment (IDE) software is a desktop development environment and tools set for developing and debugging a microcontroller design application. MPLAB GL IDE allows quick changes between different development and debugging activities. Designed for use with the Windows® operating environment, it is a powerful, affordable, run-time development tool. It is the common user interface for Microchip's development systems tools, including MPLAB Editor, MPLAB ASM30 Assembler, MPLAB SIM software simulator, MPLAB LIB30 Library, MPLAB LINK30 Linker, MPLAB ICE 4000 In-Circuit Emulators, PRO MATE II programmer and In-Circuit Debugger (ICD 2). The MPLAB 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 devices in the same design environment that they have used for PICmicro microcontrollers.

The MPLAB GL 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 GL IDE 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 (sold separately)

The MPLAB GL 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 GL 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.

10.2 dsPIC Language Suite

The Microchip Technology MPLAB C30 C compiler is a complete, easy to use language product. It allows dsPIC applications code to be written in high level C language, and then be fully converted into machine object code for programming of the microcontroller. It simplifies development of code by removing code obstacles and allowing the designer to focus on program flow and not program elements. Several options for compiling are available, so the user can select those that will maximize the efficiency of the code characteristics.

It is a fully ANSI compliant product with standard libraries for the dsPIC family of microcontrollers. It uses the many advanced features of the dsPIC devices to provide very efficient assembly 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 GL IDE for high level, source debugging. Some features include:

- 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 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 GL IDE.

10.3 MPLAB SIM Software Simulator

The MPLAB SIM software simulator allows code development in a PC-hosted environment by simulating the dsPIC device 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.

10.4 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 device. Software control of the emulator is provided by MPLAB GL IDE.

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 4000 in-circuit emulator allows expansion to support new devices.

The MPLAB ICE 4000 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. Features include:

- Full speed emulation, up to 50 MHz bus speed, or 200 MHz external clock speed
- Low voltage emulation down to 1.8 volts
- Configured with 2 Mb program emulation memory, additional modular memory up to 16 Mb
- 64K x 256-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
- US and parallel printer port PC connection

10.5 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 2 utilizes the in-circuit debugging capability built into the various devices. This feature, along with Microchip's In-Circuit Serial Programming protocol (ICSP), offers cost effective, in-circuit debugging from the graphical user interface of MPLAB GL IDE. 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. Some of its features include:

- Full speed operation to the range of the device
- Serial or US PC connector
- Serial interface externally powered
- US 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 inexpensive 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)

10.6 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 has programmable VDD and VPP supplies, which allow it to verify programmed memory at VDDMIN and VDDMAX, for maximum reliability when programming requiring this capability. It has an LCD display for instructions and error messages, keys to enter commands. Interchangeable optional socket modules support all package types.

In Stand-Alone mode, the PRO MATE II device programmer can read, verify, or program PICmicro and dsPIC30F devices. It can also set code protection in this mode. PRO MATE II features include:

- Runs under MPLAB GL IDE
- Field upgradable firmware
- DOS Command Line interface for production
- Host, Safe, and "Stand-Alone" operation
- Automatic downloading of object file

SQTPSM 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).

11.0 THIRD PARTY HW/SW TOOLS AND APPLICATION LIBRARIES

Microchip is partnering with key third party tool manufacturers for the development of quality hardware and software tools, in support of the dsPIC30F product family. Microchip is offering this initial set of tools and libraries, which will enable customers to rapidly develop their dsPIC30F based application(s).

Microchip will expand this current list to provide our customers with additional value added services, i.e., repository of skilled/certified technical applications contacts, reference designs, hardware and software developers.

The dsPIC30F software tools and libraries include:

- Third Party C compilers
- Floating Point and Double Precision Math Library
- DSP Algorithm Library
- DSP Filter Design Software Utility
- Peripheral Driver Library
- CAN Library
- Real-Time Operating Systems (RTOS)
- OSEK Operating Systems
- TCP/IP Protocol Stacks
- V.22/V.22bis and V.32 ITU Specifications

The dsPIC30F hardware development board tools include:

- General Purpose Development Board
- Motor Control Development Board
- Connectivity Development Board

Table 11-1 presents a summary of the tools with the respective provider and availability dates.

TABLE 11-1: THIRD PARTY TOOLS, APPLICATIONS LIBRARIES AND HARDWARE DEVELOPMENT BOARDS PLANNED

Product	From	Availability
Third Party C compilers	IAR, Hi-Tech, CCS	2002
Math Library (includes double-precision Floating Point routines)	Microchip	2002
DSP Algorithm Library	Microchip	2002
Digital Filter Design Software Utility	Microchip	2002
Device Initialization Software Utility	Microchip	2002
Peripheral Driver Library	Microchip	2002
General Purpose Development Board	Microchip	2002
Motor Control Development Board and Application Library	Microchip	2002
Connectivity Development Board and Application Library	Microchip	2002
RTOS	Multiple Third Parties	2002
OSEK Operating Systems	Third Party	2002
TCP/IP Stack	Multiple Third Parties	2002
CAN Library	Third Party	2002
V.22/V.22bis and V.32 ITU Specifications	Microchip and Third Party	2002

11.1 Third Party C compilers

In addition to the Microchip MPLAB C30 C compiler, the dsPIC30F will be supported by ANSI C compilers developed by IAR, Hi-Tech and Custom Computer Services (CCS).

The compilers allow dsPIC application code to be written in high level C language, and then be fully converted into machine object code for programming of the microcontroller. Each compiler tool provides several options for compiling, so the user can select those that will maximize the efficiency of the generated code characteristics.

Multiple C compiler solutions come with different price targets and features, enabling the customer to select the compiler best suited for their application requirements.

11.2 Math Library

The Math Library will support several standard C functions, such as, but not limited to:

- sin(), cos(), tan()
- asin(), acos(), atan(),
- log(), log(10)
- sqrt(), power()
- ceil(), floor()
- fmod(), frexp()

The math function routines will be developed and optimized in dsPIC30F assembly language and will be callable from both assembly and C language. Floating point and double precision versions of each function shall be provided. The Microchip MPLAB-C30 and IAR C compilers will be supported.

Electronic documentation will accompany the math library, enabling the user to efficiently understand and implement the library functions.

11.3 DSP Algorithm Library

The DSP library will support multiple filtering, convolution, vector and matrix functions. Some of the functions include, but are not limited to:

- Cascaded Infinite Impulse Response (IIR) Filters
- Correlation
- Convolution
- Finite Impulse Response (FIR) Filters
- Windowing Functions
- FFTs
- LMS Filter
- Vector Addition and Subtraction
- Vector Dot Product
- Vector Power
- Matrix Addition and Subtraction
- Matrix Multiplication

Some DSP functions will be implemented, utilizing double precision and floating point arithmetic. All DSP routines will be developed and optimized in dsPIC30F assembly language and will be callable from both assembly and C language. The Microchip MPLAB C30 and IAR C compilers will be supported.

Electronic documentation will accompany the DSP library, enabling the user to efficiently understand and implement the library functions.

11.4 DSP Filter Design Software Utility

Microchip will offer a digital filter design software tool which enables the user to develop optimized assembly code for Lowpass, Highpass, Bandpass and Bandstop IIR and FIR filters, including 16-bit fractional data size filter coefficients, from a graphical user interface. The application developer enters the required filter frequency specifications and the software tool develops the filter code and coefficients. Ideal filter frequency response and time domain plots are generated for analysis.

FIR filter lengths up to 513 taps, and IIR filter lengths up to 10 cascaded sections, are supported.

All IIR and FIR routines are generated in assembly language and will be callable from both assembly and C language. The Microchip MPLAB C30 C compiler will be supported.

11.5 Peripheral Driver Library

Microchip will offer a peripheral driver library, which will support the setup and control of dsPIC30F hardware peripherals, such as, but not limited to:

- Analog-to-Digital Converter
- Motor Control PWM
- Quadrature Encoder Interface
- UART
- SPI
- Data Converter Interface
- I²C
- General Purpose Timers
- Input Capture
- Output Compare/Simple PWM

In addition to the hardware peripherals, the library will support software generated peripherals, such as I²C and SPI and standard LCDs which support a Hitachi style controller.

All peripheral driver routines will be developed and optimized in dsPIC30F assembly language and will be callable from both assembly and C language. The Microchip MPLAB C30 C compiler will be supported.

Electronic documentation will accompany the peripheral library, enabling the user to efficiently understand and implement the library functions.

11.6 CAN Library

Microchip will offer a CAN driver library, which will support the dsPIC30F CAN peripheral. Some of the CAN functions which will be supported are:

- Initialize CAN Module
- Set CAN Operational Mode
- Set CAN Baud Rate
- Set CAN Masks
- Set CAN Filters
- Send CAN Message
- Receive CAN Message
- Abort CAN Sequence
- Get CAN TX Error Count
- Get CAN RX Error Count

All CAN driver routines will be developed and optimized in dsPIC30F assembly language and will be callable from both assembly and C language. Support for the Microchip MPLAB C30 C compiler will be provided.

Electronic documentation will accompany the CAN library, enabling the user to efficiently understand and implement the library functions.

11.7 Real-Time Operating System (RTOS)

Real-Time Operating System (RTOS) solutions for the dsPIC30F product family will be provided. The RTOS solutions provide the necessary function calls and operating system routines to write efficient C and/or assembly code, to create well designed multi-tasking applications. In addition, RTOS solutions will be provided, which will address those applications in which program and more importantly, data memory resources are limited. Configurable and optimized kernels will be available to support various RTOS application requirements.

The RTOS solutions will range from a fully true preemptive and multi-tasking scheduler, to a cooperative type scheduler, of which both are designed to optimally run on the dsPIC30F devices. Depending on the RTOS implementation, some of the function calls provided in the system kernel are:

- control tasks
- send and receive messages
- handle events
- control resources
- control semaphores
- regulate timing in a variety of ways
- provide memory management
- handle interrupts and swap tasks

Most functions are written in ANSI C, with the exception of time critical functions which are optimized in assembly, thereby reducing execution time for maximum code efficiency. The ANSI C and assembly routines are supported by the Microchip MPLAB C30 C compiler.

Electronic documentation will accompany the RTOS, enabling the user to efficiently understand and implement the RTOS in their application.

11.8 OSEK Operating Systems

Operating Systems for the vehicle software standard OSEK/VDX will be developed for support in the dsPIC30F product family. The functionality of OSEK "Offene Systeme und deren Schnittstellen für die Elektronik im Kraftfahrzeug" (Open systems and the corresponding interfaces for automotive electronics) is harmonized with VDX "Vehicle Distributed eXecutive" yielding OSEK/VDX.

Structured and modular RTOS software implementations, based on standardized interfaces and protocols will be provided. Structured and modular implementations provide for portability and extendability for distributed control units for vehicles.

Various OSEK COM modules will be provided, such as:

- OSEK/COM Standard API
- OSEK/COM Communication API
- OSEK/COM Network API
- OSEK/COM Standard Protocols
- OSEK/COM Device Driver Interface

Microchip will also provide Internal and External CAN driver support. The physical layer is integrated into the communication controller's hardware and is not covered by the OSEK specifications.

Most module functions will be developed in ANSI C with the exception of time critical functions and peripheral utilization, which are optimized in assembly, thereby reducing execution time for maximum code efficiency. The Microchip MPLAB C30 C compiler will be supported.

11.9 TCP/IP Protocol Stack

Microchip will offer various Transmission Control Protocol/Internet Protocol (TCP/IP) Stack Layer solutions for Internet connectivity solutions implemented on the dsPIC30F product family. Both reduced and full stack implementations will be provided, which will allow the user to select the optimum TCP/IP stack solution for their application.

Application protocol layers, such as FTP, TFTP and SMTP, Transport and Internet layers, such as ICMP, IP, TCP and UDP and Network Access layers, such as PPP, SLIP, ARP and DHCP will be provided. Various configurations, such as a minimal UDP/IP stack will be available for limited connectivity requirements.

Most stack protocol functions will be developed and optimized in Microchip's MPLAB C30 C language. Assembly language coding may be developed for specific dsPIC30F hardware peripherals and Ethernet drivers to optimize code size and execution time. These assembly language specific routines will be assembly and C callable.

Electronic documentation will accompany the TCP/IP protocol stack, enabling the user to efficiently understand and implement the protocol stack in their application.

11.10 V.22/V.22bis and V.32 Specification

Microchip will offer ITU compliant V.22/V.22bis (1200/2400 bps) and V.32 (non-trellis coding at 9600 bps) modem specifications to support a range of "connected" applications.

Applications which will benefit from these modem specifications are numerous and fall into many applications, some of which are listed here:

- Internet enabled home security systems
- Internet connected power, gas and water meters
- Internet connected vending machines
- Smart Appliances
- Industrial monitoring
- POS Terminals
- Set Top Boxes
- Drop Boxes
- Fire Panels

Most ITU specification modules will be developed and optimized in Microchip's MPLAB C30 C language. Assembly language coding may be developed for specific dsPIC30F hardware peripherals and key transmitter and receiver filtering routines to optimize code size and execution time. These assembly language specific routines will be assembly and C callable.

Electronic documentation will accompany the modem library, enabling the user to efficiently understand and implement the library functions.

12.0 dsPIC30F HARDWARE DEVELOPMENT BOARDS

Microchip will initially provide three hardware development boards, which will provide the application developer with a tool in which to quickly prototype and validate key design requirements. Each board will feature key dsPIC30F peripherals and support Microchip's MPLAB In-Circuit Debugger (ICD 2) tool for cost effective debugging and programming of the dsPIC30F device. The three initial boards to be provided are:

- General Purpose Development Board
- Motor Control Development Board
- Connectivity Development Board

12.1 General Purpose Development Board

The dsPIC30F general purpose development board provides the application designer with a low cost development tool in which to become familiar with the dsPIC30F 16-bit architecture, high-performance peripherals and powerful instruction set. The development board serves as an ideal prototyping tool in which to quickly develop and validate key design requirements.

Some key features and attributes of the general purpose development board are:

- Supports various dsPIC30F packages through optimized daughter boards
- CAN communication channel
- RS-232 and RS-485 communication channels
- CODEC interface with line in/out jacks
- In-Circuit Debugger interface
- Microchip temperature sensor
- Microchip Op Amp circuit, supporting user input signals
- Microchip Digital-to-Analog Converter
- 2x16 LCD
- General purpose prototyping area
- Various LEDS, switches and potentiometers
- 4x4 keypad interface

The general purpose development board is shipped with a 9V power supply, RS-232 I/O cable, pre-programmed dsPIC30F device, example software and appropriate documentation to enable the user to exercise the development board demonstration programs.

12.2 Motor Control Development Board

The dsPIC30F motor control development board provides the application developer with three main components for quick prototyping and validation of BLDC, PMAC and ACIM applications. The three main components are:

- dsPIC30F Motor Control Main Board
- 3-phase low voltage power module
- 3-phase high voltage power module

The main control board supports the dsPIC30F6010 device, various peripheral interfaces and a custom interface header system, which allows different motor power modules to be connected to the PCB. The control board also has connectors for mechanical position sensors, such as incremental rotary encoders and hall effect sensors, and a breadboard area for custom circuits. The main control board receives its power from a standard plug-in transformer.

The low voltage power module is optimized for 3-phase motor applications that require a DC bus voltage less than 50 volts and can deliver up to 400W power output. The 3-phase low voltage power module is intended to power BLDC and PMAC motors.

The high voltage power module is optimized for 3-phase motor applications that require DC bus voltages up to 400 volts and can deliver up to 1 kW power output. The high voltage module has an active power factor correction circuit that is controlled by the dsPIC30F device. This power module is intended for AC induction motor and power inverter applications.

Both power modules have automatic fault protection and the high voltage module is electrically isolated from the control interface. Both power module boards provide pre-conditioned voltage and current signals to the main control board. All position feedback devices that are isolated from the motor control circuitry, such as incremental encoders, hall-effect sensors, or tachometer sensors, can be directly connected to the main control board. Both modules are equipped with motor braking circuits.

Some key features and attributes of the motor control main development board are:

- dsPIC30F Motor Control Main Board supporting the dsPIC30F6010
- RS-232 and RS-485 interface channels
- 2x16 LCD
- In-Circuit Debugger support
- General purpose prototyping area
- Custom interface header system
- Various LEDS, switches and potentiometers

The motor control development systems is shipped with a 9V power supply for the control board, RS-232 I/O cable, pre-programmed dsPIC30F device, example software and documentation that allows the user to exercise the development board demo programs. Furthermore, a list of motors compatible with the development system is provided to allow rapid evaluation.

12.3 Connectivity Development Board

The dsPIC30F connectivity development board provides the application developer a basic connectivity platform for developing and evaluating various connectivity solutions, implementing TCP/IP protocol layers combined with V.22/V.22bis and V.32 (non-trellis coding) ITU specifications across PSTN or Ethernet communication channels.

Some key features and attributes of the connectivity development board are:

- Supports dsPIC30F6014 device with optimized daughter boards.
- Media Access Control (MAC) and PHY interface
- PSTN interface with DAA/AFE
- RS-232 and RS-485 communication channels
- In-Circuit Debugger interface
- Microchip temperature sensor
- Microchip Digital-to-Analog Converter
- 2x16 LCD
- General purpose prototyping area
- Various LEDs, switches and potentiometers
- 4x4 keypad interface

The connectivity development board will be shipped with a 9V power supply, RS-232 I/O cable, pre-programmed dsPIC30F devices with example connectivity software and appropriate documentation to enable the user to exercise the development board connectivity demo program.

Note the following details of the code protection feature on PICmicro® MCUs.

- The PICmicro family meets the specifications contained in the Microchip Data Sheet.
- Microchip believes that its family of PICmicro microcontrollers is one of the most secure products of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the PICmicro microcontroller in a manner outside the operating specifications contained in the data sheet. The person doing so may be engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable”.
- Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our product.

If you have any further questions about this matter, please contact the local sales office nearest to you.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.

Trademarks


The Microchip name and logo, the Microchip logo, FilterLab, KEELOQ, MPLAB, PIC, PICmicro, PICMASTER, PICSTART, PRO MATE, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

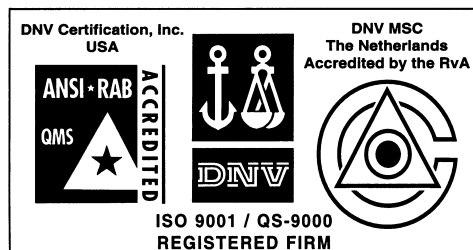
dsPIC, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, microID, microPort, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, MXDEV, PICC, PICDEM, PICDEM.net, rPIC, Select Mode and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A.

Serialized Quick Term Programming (SQTP) is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2002, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

 Printed on recycled paper.



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

Boston

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

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

Detroit

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

Kokomo

2767 S. Albright Road
Kokomo, Indiana 46902
Tel: 765-864-8360 Fax: 765-864-8387

Los Angeles

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

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 Consulting (Shanghai)
Co., Ltd., Beijing Liaison Office
Unit 915
Bei Hai Wan Tai Bldg.
No. 6 Chaoyangmen Beidajie
Beijing, 100027, No. China
Tel: 86-10-85282100 Fax: 86-10-85282104

China - Chengdu

Microchip Technology Consulting (Shanghai)
Co., Ltd., Chengdu Liaison Office
Rm. 2401, 24th Floor,
Ming Xing Financial Tower
No. 88 TIDU Street
Chengdu 610016, China
Tel: 86-28-6766200 Fax: 86-28-6766599

China - Fuzhou

Microchip Technology Consulting (Shanghai)
Co., Ltd., Fuzhou Liaison Office
Unit 28F, World Trade Plaza
No. 71 Wusi Road
Fuzhou 350001, China
Tel: 86-591-7503506 Fax: 86-591-7503521

China - Shanghai

Microchip Technology Consulting (Shanghai)
Co., Ltd.
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

China - Shenzhen

Microchip Technology Consulting (Shanghai)
Co., Ltd., Shenzhen Liaison Office
Rm. 1315, 13/F, Shenzhen Kerry Centre,
Renminnan Lu
Shenzhen 518001, China
Tel: 86-755-2350361 Fax: 86-755-2366086

Hong Kong

Microchip Technology Hongkong Ltd.
Unit 901-6, 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 Japan K.K.
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

Korea

Microchip Technology Korea
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea 135-882
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 Nordic ApS
Regus Business Centre
Lautrup høj 1-3
Ballerup DK-2750 Denmark
Tel: 45 4420 9895 Fax: 45 4420 9910

France

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

Germany

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

Italy

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

01/18/02