Smart Air Handler using ProMPT™ and the PIC18F2539

Author: *Jon Burroughs*
Microchip Technology Inc.

INTRODUCTION

In many heating, ventilation, and air conditioning (HVAC) applications, air handler motors are either off, or on at full speed. However, by adding variable speed control to the air handler, significant energy savings over the standard on/off control can be realized, resulting in significantly reduced cost of operation.

This application note discusses the implementation of a variable speed air handler that utilizes a single phase AC induction motor. The task of designing the variable speed air handler is greatly simplified by using the Microchip Programmable Motor Control Processor Technology (ProMPT) Single Phase Induction Motor Control Evaluation Kit and the PIC18F2539 microcontroller.

The PIC18F2539 microcontroller is an Enhanced FLASH microcontroller that features the Single Phase Induction Motor Control (SPIMC) kernel. The SPIMC kernel enables open loop variable frequency (VF) control and features a programmable voltage versus frequency curve. The PIC18F2539 microcontroller is a natural choice for adding variable speed control to an air handler application.

The ProMPT™ Single Phase Induction Motor Control Evaluation Kit functions as an effective platform for application development. It can be used to control shaded pole and permanent magnet split capacitor type motors. Adding specific features to support the air handler application is accomplished by designing a custom daughter board. In this application, the daughter board merely adds a user interface and temperature sensor to the existing AC induction motor drive.

AC single phase induction motors are used in many household applications, including HVAC, dishwashers, clothes washers and dryers, garage door openers, lawn mowers, and so on. ProMPT technology from Microchip can greatly simplify design and reduce time-to-market for all of these applications.

APPLICATION OVERVIEW

This application note shows how to add variable speed control to an HVAC air handler. The application demonstrates heating only, and does not address control of the heating element, which is assumed to be a simple logic on/off control of a gas furnace or electrical heating coils.

Usually, the target temperature is set by the thermostat installed within a house. In this application, we use a simple user interface consisting of two 8-segment LEDs and two push buttons to set the target temperature. The actual air temperature is measured using the Microchip TC1047 temperature sensor (see Figure 1).

The variable speed functionality is depicted in Figure 2. The air handler operates at full speed when the air temperature is more than 5 degrees Celsius below the target temperature. When the air temperature is within 5 degrees of the target temperature, the air handler speed is proportional to the difference in temperature. As the air temperature within a house falls (for example, because it is cold outside), the air handler speed increases, delivering more hot air into the house.

AC induction motors have a minimum operational speed. To avoid on/off cycling when the actual temperature is near the target temperature, the lower end of the variable speed response curve includes a hysteresis loop. In order for the air handler to turn on at the minimum motor frequency, the actual temperature must be more than 1°C below the target temperature. However, to turn the air handler off, the difference between the two temperatures must be zero. Because on/off cycling is avoided, energy is conserved and air handler operation is quieter and less obtrusive.

FIGURE 1: APPLICATION BLOCK DIAGRAM

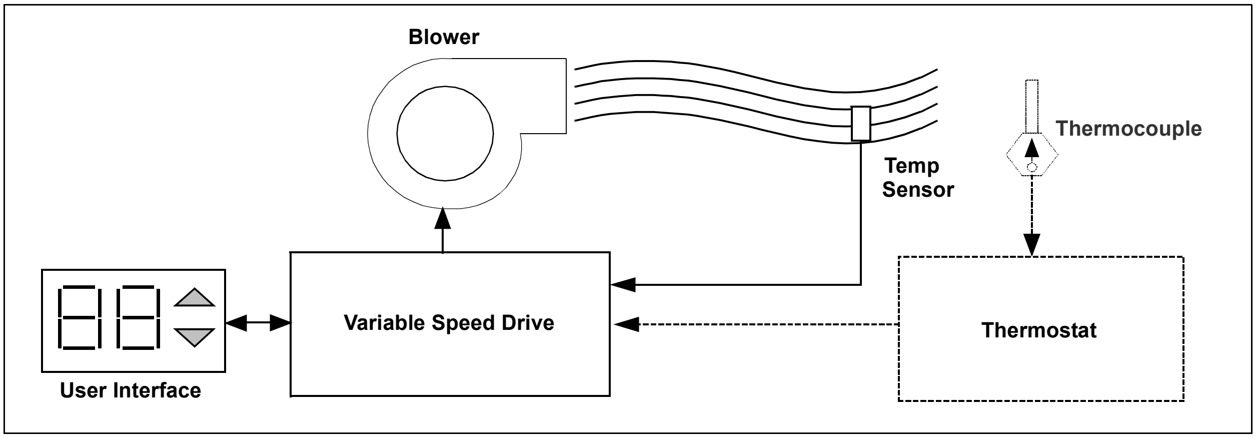
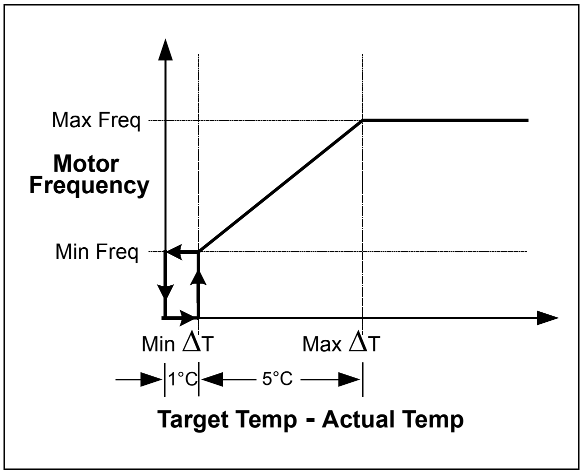


FIGURE 2: VARIABLE SPEED RESPONSE



APPLICATION DESIGN WITH ProMPT

The application design is simplified enormously by using the ProMPT Single Phase Induction Motor Control Evaluation Kit (see Figure 3). With the ProMPT board, the task of designing an efficient AC induction drive has already been accomplished. To create a variable speed air handler, it is necessary only to design a simple daughter board that connects to the ProMPT board's I/O expansion connector and write the necessary firmware (see Figure 4).

All information necessary to use the ProMPT board is published in the following documents:

- ProMPT Single Phase Induction Motor Control Evaluation Kit User's Guide
- PIC18FXX39 Data Sheet

Readers may refer to these documents for more detailed information on the ProMPT evaluation kit and the PIC18F2539 microcontroller. These documents may be obtained from the Microchip web site.

FIGURE 3: ProMPT SINGLE PHASE INDUCTION MOTOR CONTROL EVALUATION KIT

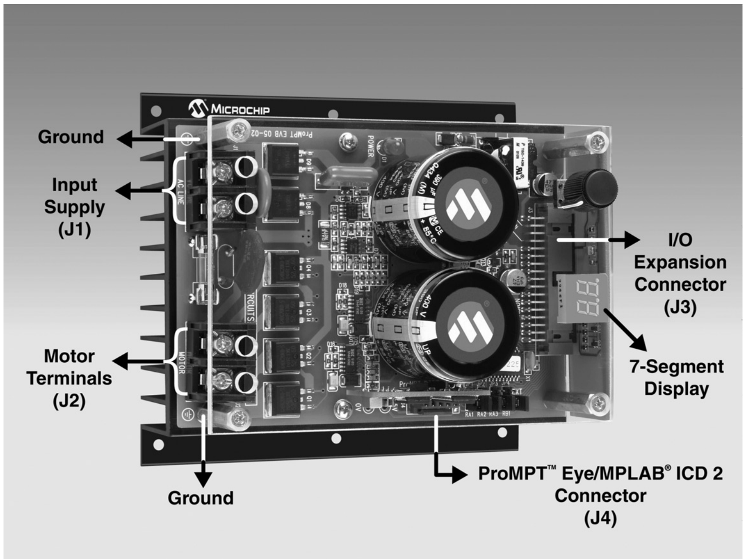
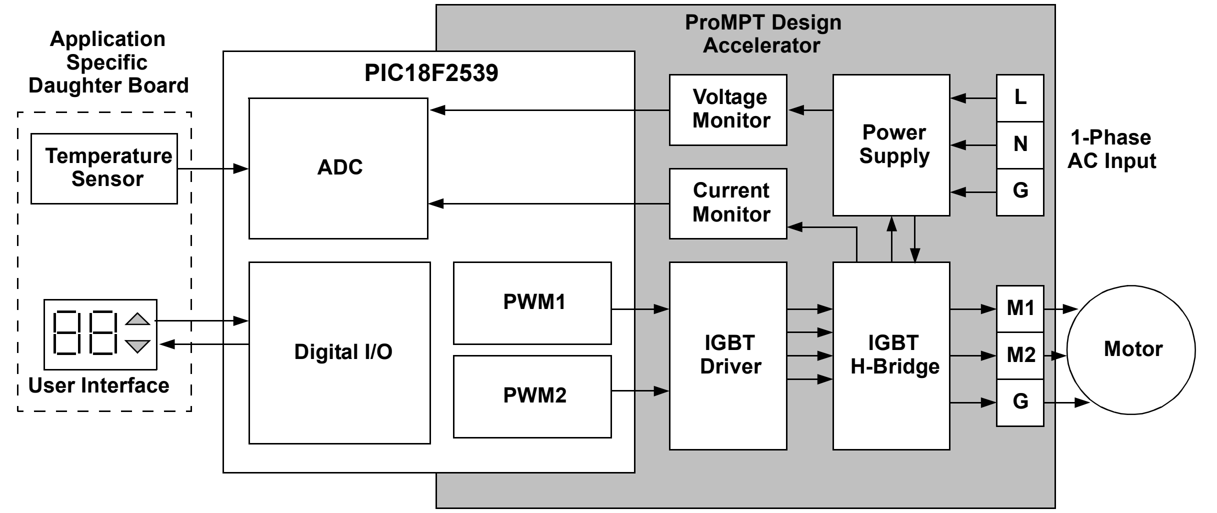


FIGURE 4: BLOCK DIAGRAM OF APPLICATION COMPONENTS



APPLICATION SPECIFIC HARDWARE

The features specific to the air handler application that are designed onto an application specific daughter board are described below:

Display Module

A 2x8 segment display is used to display the temperature in Celsius. To save on I/O pins, control of the two digits is multiplexed. Because of persistence of vision in the human eye, the digits will appear to be illuminated simultaneously, even though they are actually illuminated one at a time. Because the two decimal segments are not required, a total of nine I/O pins are used to control 14 LED segments.

Push Buttons

Two push buttons are used to adjust the temperature.

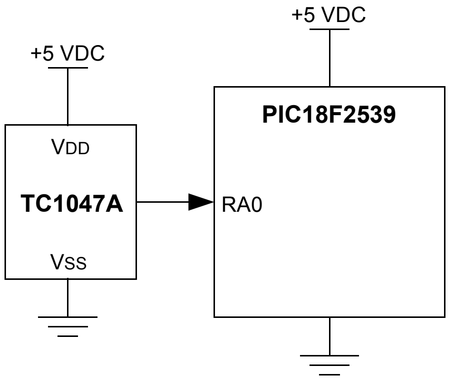
- **Up** - adjusts the target temperature upward.
- **Down** - adjusts the target temperature downward.

When either the **Up** or **Down** button is pressed, the temperature begins blinking to indicate that a new target temperature is being set. The blinking target temperature increments or decrements with each press of the up or down button. After five seconds elapse without a button press, the temperature display returns to the present temperature and stops blinking. Each button requires one I/O pin.

Temperature Sensor

Temperature measurement is made easy with a Precision Temperature-to-Voltage Converter. This solid state temperature sensor eliminates the need to perform calibration that is required when using thermocouples. Microchip's TC1047A is a linear voltage output temperature sensor, whose output is directly proportional to the measured temperature. Temperature is easily calculated without having to construct calibrated lookup tables. The TC1047A can accurately measure temperature from -40°C to 125°C, a range more than adequate for a household HVAC application. Supply voltage can vary from 2.5V to 5.5V (see Figure 5). For more information, see the TC1047/TC1047A data sheet. The temperature sensor requires one analog input pin.

FIGURE 5: PRECISION TEMPERATURE-TO-VOLTAGE CONVERTER

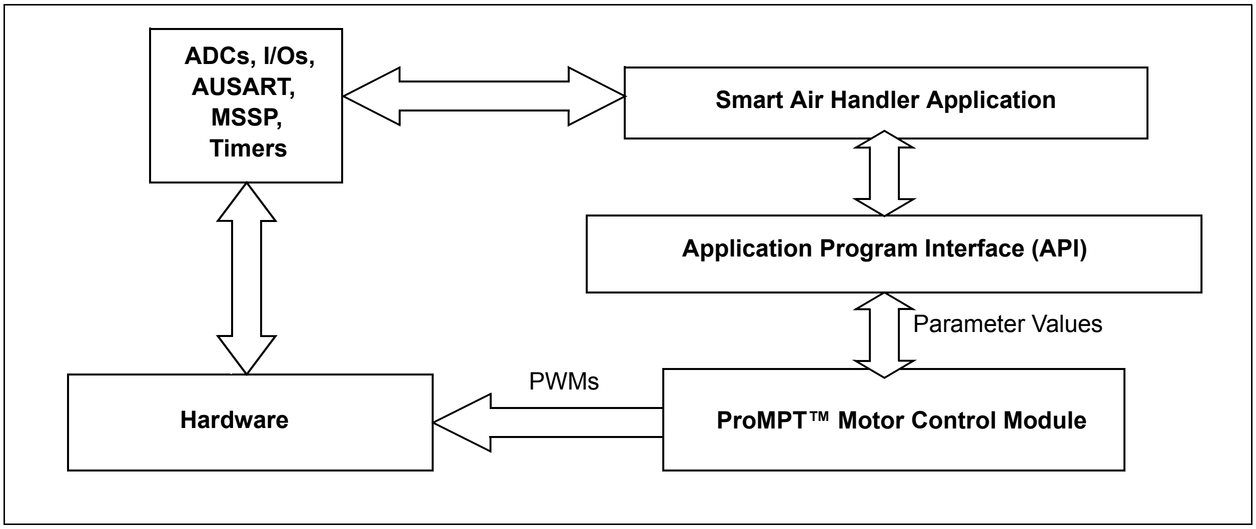


Note: In this application, a low-pass filter and shielded cable are used. See schematic in Appendix A.

APPLICATION FIRMWARE OVERVIEW

ProMPT motor control functionality is accessed by using the pre-defined Application Program Interface (API) described in Appendix B of this document (this information is also available in Appendix E of the ProMPT Design Accelerator Kit User's Guide). By using the defined API, powerful motor control tasks can be realized with no knowledge of the underlying microcontroller activities. Figure 6 illustrates how the user developed application firmware interacts with the ProMPT motor control module through the Application Program Interface (API). In essence, the API consists of the library of ProMPT firmware functions that enables control of the ProMPT module, without needing to know the details of its operation.

FIGURE 6: MOTOR CONTROL ARCHITECTURE USING THE PIC18FXX39


ADCs, I/Os,
AUSART,
MSSP,
Timers
Smart Air Handler Application
Application Program Interface (API)
Parameter Values
PWMs
Hardware
ProMPT™ Motor Control Module

REQUIRED FILES

In order to take advantage of the pre-defined ProMPT API, it is necessary to include several files when creating the project in MPLAB® IDE v6.10 These files are described below.

Application Specific Files:

- `SmartAir.c` Main source code listing
- `SmartAir.h` Definition file for application

Required files when using the PIC18FXX39 device:

- `motor.h` Definition file for the motor
- `ProMPT_c18.h` Prototypes of the API methods used in the application
- `18F2539.lkr` Linker file

APPLICATION FIRMWARE FUNCTIONS

The firmware functions of the variable speed air handler application are outlined below.

> **Note:** Tasks denoted with an asterisk (*) are related to control of the AC induction motor.

1. Initialize the motor control module.*
2. Set the appropriate voltage frequency (VF) curve for the motor.*
3. Execute a continuous loop that performs the following tasks:
   a) Read the temperature sensor connected to ADC channel RA0.
   b) Read the motor current, DC bus voltage, and heatsink temperature.
   c) Read and debounce the button inputs.
   d) If a button is pressed, increment or decrement the target temperature as necessary.
   e) Check for faults.
   f) Compare the actual temperature to the target temperature.
   g) Set the appropriate motor frequency.*
   h) Update the LED display with actual or target temperature.
   i) Continuously control the AC induction motor.*
4. If a fault is present, display fault on LED Fault indicators:

   E1 - Hardware transient current detection

   E2 - Heat sink over temperature set at 70°C

   E3 - Software over current detection set at 6A

   E4 - DC bus over voltage set at 250V

   E5 - DC bus under voltage set at 90V

By using the PIC18F2539 and the ProMPT based Single Phase Induction Motor Control kernel, the biggest tasks (those involving motor control) become the simplest ones. The ProMPT API methods make the development of this application very easy. For example, to initialize the motor control module and set a new VF curve for the motor, the API methods are as shown in Example 1.

Constants like `motorVFCurve` and `ACCELRATE` are defined in the `motor.h` file, and are dependant on the specific motor used in the application.

The ProMPT API helps to make the application code easy to write. See Appendix C for the location of the complete source code with comments. A detailed flow chart of the application firmware is shown in Figure 7.

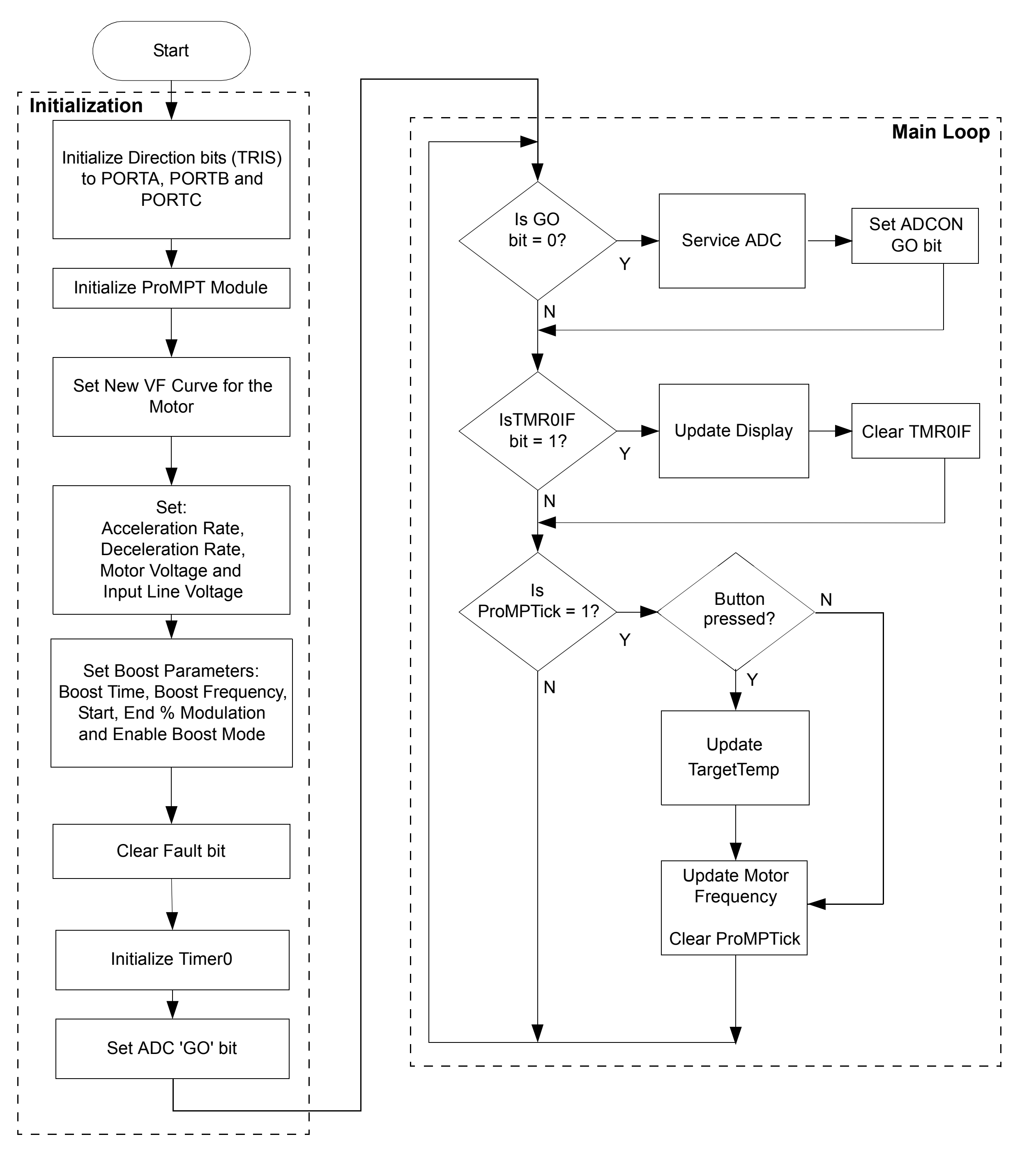
EXAMPLE 1: MOTOR INITIALIZATION API METHODS

```
ProMPT_Init(0);         //Initialize the ProMPT block
                        //0 is the initial motor frequency

for (i=0;i<17;i++) {    //Set the V/F Curve for the motor
        ProMPT_SetVFCurve(i,motorVFCurve[i]);
    }

ProMPT_SetAccelRate(ACCELRATE); //Set other Motor Parameters
ProMPT_SetDecelRate(DECELRATE);
ProMPT_SetMotorVoltage(MOTORVOLTAGE);
ProMPT_SetLineVoltage(LINEVOLTAGE);
```

FIGURE 7: APPLICATION FLOW CHART



DEVELOPMENT TOOL SETUP

The following development tools were used to develop this application:

- MPLAB® IDE v6.10 or later version
- MPLAB® C18 C Compiler
- ProMPT™ Design Accelerator Kit with Single Phase Induction Motor
- MPLAB® ICD 2 Programmer/Debugger

The MPLAB ICD 2 is connected to J4 for programming the PIC18F2539 and to debug the program. The ICD 2 should be disconnected when the ProMPT drive is powered from mains. Powering the ProMPT drive when ICD 2 is connected will damage the ICD 2 or the computer connected to it, unless an isolation transformer is used (see Figure 8).

Application development using the ICD 2 and ProMPT MC Eval Board is simplified by using the isolation transformer. The following steps can be followed to develop and debug an application program on the ProMPT MC Eval Kit with an ICD 2.

Warning 1: Power electronics involve inherent risks, both to equipment and personnel. This document assumes that the user has experience with high voltage electronics. Incorrect use of the ProMPT drive can be hazardous to development staff as well as the user of the equipment.

2: Always disconnect the ProMPT drive from power before making connections or jumper settings. After switching off power, wait until the “Power” LED is completely off before working on the drive or motor. Failure to comply with this warning could result in injury or death.

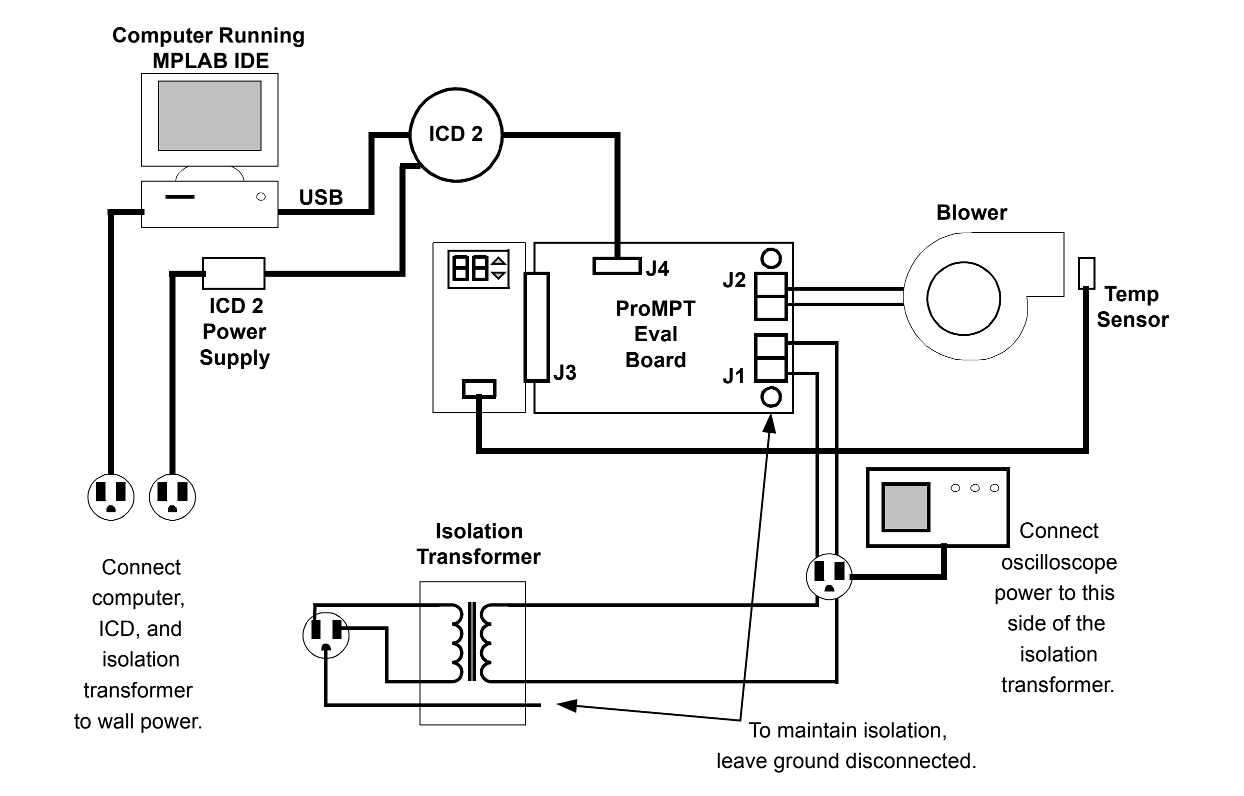
Without the isolation transformer:

1. Open a new project in MPLAB IDE v6.10 or later.
2. Select MPLAB C18 C compiler as the tool suite.
3. Add the application program and header files to the project.
4. Add the appropriate linker file to the project.
5. Compile and link the project.
6. With mains power disconnected from the ProMPT MC Eval Board, connect MPLAB ICD 2 to J4 connector on the board.
7. Enable “Power target circuit from MPLAB ICD 2” in menu *programmer* > *settings* > *power*.
8. Program the target chip and debug the application code.
9. Disconnect MPLAB ICD 2.
10. Power up the ProMPT drive and continue testing.

With the isolation transformer:

1. Open a new project in the MPLAB IDE v6.10 or later.
2. Select MPLAB C18 C compiler as the tool suite.
3. Add the application program and header files to the project.
4. Add the appropriate linker file to the project.
5. Compile and link the project.
6. Disable “Power target circuit from MPLAB ICD 2” in menu *programmer* > *settings* > *power*. MPLAB ICD 2 will be powered from the target board.
7. With the MC Eval kit powered through an isolation transformer (see Figure 8), connect MPLAB ICD 2 to the J4 connector on the board.
8. Program the target chip and debug the application code.
9. If motor frequency is always ‘0’, or motor is left disconnected, MPLAB ICD 2 may be left connected during debugging.
10. To test motor operation, program the target chip with Debug mode disabled, disconnect MPLAB ICD 2, and continue testing.

FIGURE 8: DEVELOPMENT TOOL SETUP WITH ISOLATION TRANSFORMER



Note:	Even with the isolation transformer, the ProMPT drive cannot be operated with MPLAB ICD 2 connected. To operate the motor (motor frequency greater than zero), MPLAB ICD 2 must always be disconnected.

The advantage of the isolation setup shown in Figure 8 is that MPLAB ICD 2 or the computer will not be damaged if MPLAB ICD 2 is connected while the ProMPT drive is powered up. This allows the user to step through code and use other debugging features without disconnecting the ProMPT board from the isolated AC power. In addition, an oscilloscope can be used to look at signals on the ProMPT board.

The jumpers, JP1-4 on the ProMPT board, should be set to “INT” position to read the DC bus voltage (VSENSE), motor current (ISENSE), heat sink temperature (TSENSE) and clear the fault (/CLEAR).

TABLE 1: SUMMARY OF MICROCONTROLLER RESOURCE USE

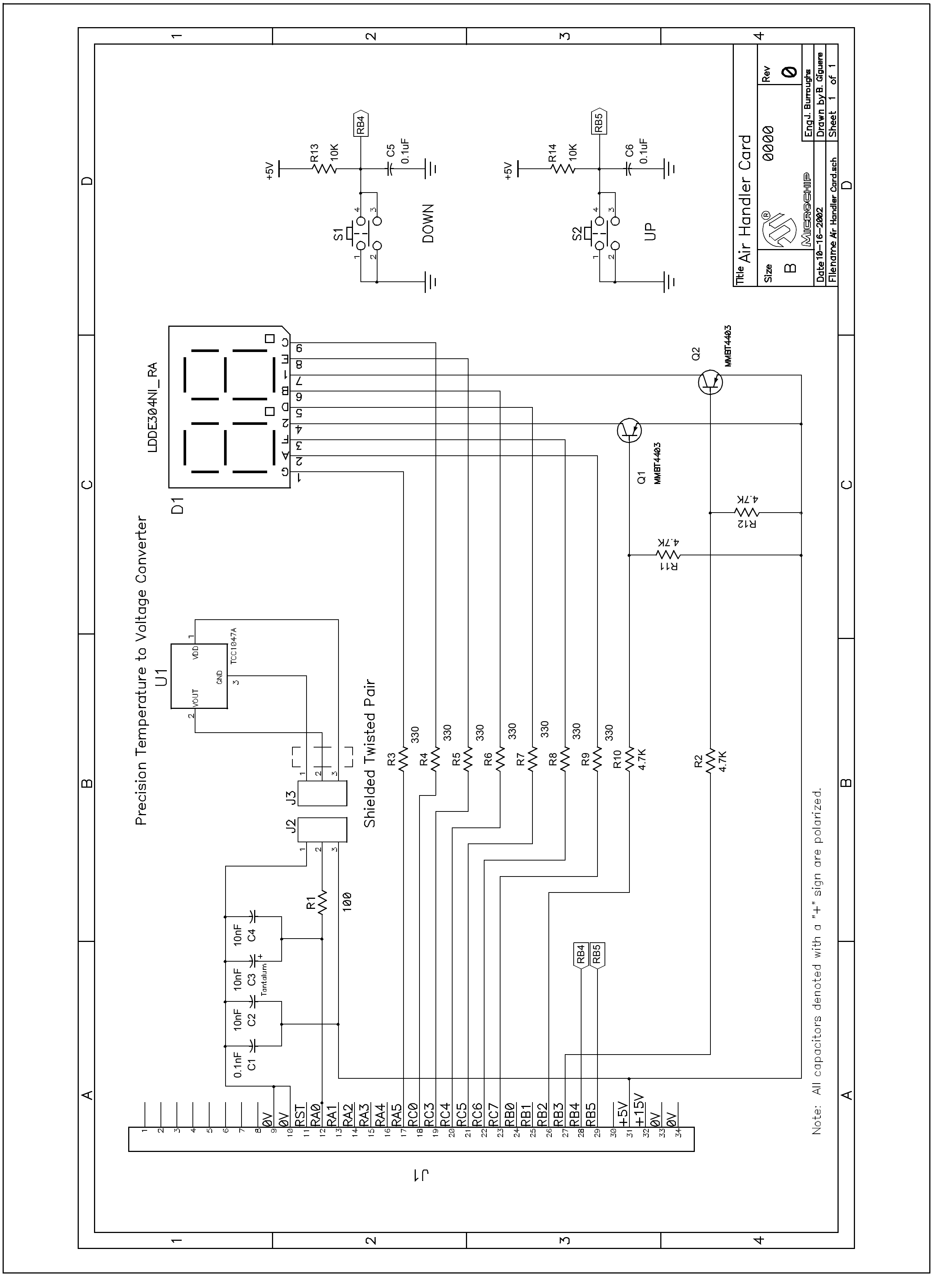
Program Memory:	6184 Words (24%)
Data Memory:	41 bytes (3%)
Peripherals:	
ADC	RA0 Temperature Sensor Input
I/O Port Pins	RB4, RB5 Up and Down Buttons
RA5, RC0, RC3 - RC7, RB2, RB3	LED Display Control
Timers	Timer0 used as a Delay Timer for LED Multiplexing
SPIMC Kernel	Motor Control Functions

CONCLUSION

Variable speed control is easily added to an HVAC air handler by using the ProMPT Single Phase Induction Motor Control Evaluation Kit and the PIC18F2539 microcontroller. The Single Phase Induction Motor Control kernel greatly simplifies the design of a single phase induction motor control application. Microchip's Programmable Motor Processor Technology allows the user to develop applications around the Single Phase Induction Motor Control kernel with little or no knowledge of motor control.

The PIC18F2539 microcontroller is suitable for control of shaded pole and permanent magnet split capacitor type motors. These types of AC single phase induction motors are used in many household applications, including HVAC, dishwashers, clothes washers and dryers, garage door openers, lawn mowers, and so on. ProMPT technology from Microchip has the potential to greatly simplify design and reduce time-to-market for all of these applications.

APPENDIX A: SCHEMATIC



Appendix B: ProMPT APPLICATION PROGRAM INTERFACE (API METHODS)

There are 27 separate API methods for the ProMPT kernel:

Note:	The operation of the Motor Control module and its APIs is based on an assumed clock frequency of 20 MHz. Changing the oscillator frequency will change the timing used in the Motor Control module accordingly. To achieve the best results in motor control applications, a clock frequency of 20 MHz is highly recommended.

`void ProMPT_ClearTick(void)`

Resources used: 0 stack levels

Description: This function clears the Tick (62.5 ms) timer flag returned by `ProMPT_tick()`. This function must be called by any routine that is used for timing purposes.

`void ProMPT_DisableBoostMode(void)`

Resources used: 0 stack levels

Description: This function disables the Boost mode logic. This method should be called before changing any of the Boost mode parameters.

`void ProMPT_EnableBoostMode(void)`

Resources used: 0 stack levels

Description: This function enables the Boost mode logic. Boost mode is entered when a stopped drive is commanded to start. The drive will immediately go to Boost Frequency and ramp from Start Modulation to End Modulation over the time period, Boost Time.

`unsigned char ProMPT_GetAccelRate(void)`

Resources used: 1 stack level

Range of values: 0 to 255

Description: Returns the current Acceleration Rate in Hz/second.

`unsigned char ProMPT_GetBoostEndModulation(void)`

Resources used: 1 stack level

Range of values: 0 to 200

Description: Returns the current End Modulation (in %) used in the Boost logic.

`unsigned char ProMPT_GetBoostFrequency(void)`

Resources used: 1 stack level

Range of values: 0 to 127

Description: Returns the current Boost Frequency in Hz.

`unsigned char ProMPT_GetBoostStartModulation(void)`

Resources used: 1 stack level

Range of values: 0 to `BoostEndModulation`

Description: Returns the Start Modulation (in %) used in the Boost logic.

`unsigned char ProMPT_GetBoostTime()`

Resources used: 1 stack level

Range of values: 0 to 255

Description: Returns the time in seconds for Boost mode.

`unsigned char ProMPT_GetDecelRate()`

Resources used: 1 stack level

Range of values: 0 to 255

Description: Returns the current Deceleration Rate in Hz/second.

`unsigned char ProMPT_GetFrequency(void)`

Resources used: 1 stack level

Range of values: 0 to 127

Description: Returns the current output frequency in Hz. This may not be the frequency commanded due to Boost or Accel/Decel logic.

`unsigned char ProMPT_GetModulation(void)`

Resources used: Hardware Multiplier; 1 stack level

Range of values: 0 to 200

Description: Returns the current output modulation in %.

`unsigned char ProMPT_GetParameter(unsigned char parameter)`

Resources used: 1 stack level

Description: In addition to its pre-defined API methods, the ProMPT kernel allows the user to custom define up to 16 functions for control or communication purposes not covered by the ProMPT APIs. These parameters are used to communicate with motor control GUI evaluation tools, such as Microchip's DashDriveMP™. This method returns the current value of any one of the parameters.

`unsigned char ProMPT_GetVFCurve(unsigned char point)`

Resources used: Hardware Multiplier; 1 stack level

Description: This function returns one of the 17 modulation values (in %) of the V/F curve. Each point represents a frequency increment of 8 Hz, ranging from point 0 (0 Hz) to point 16 (128 Hz).

`void ProMPT_Init(unsigned char PWMfrequency)`

Resources used: 64 Bytes RAM; Timer2; PWM1 and PWM2; High Priority Interrupt Vector; Hardware Multiplier; fast call/return; FSR 0; TBLPTR; 2 stack levels

`PWMfrequency` values: 0 or 1

Description: This function must be called before all other ProMPT methods, and it must be called only once. This routine configures Timer2 and the PWM outputs.

When `PWMfrequency` is '0', the module's operating frequency is 9.75 kHz. When `PWMfrequency` is '1', the module's operating frequency is 19.53 kHz.

Note: Since the high priority interrupt is used, the fast call/return cannot be used by other routines.

`void ProMPT_SetAccelRate(unsigned char rate)`

Resources used: 0 stack level

`rate` range: 0 to 255

Description: Sets the acceleration to the value of rate in Hz/second. The default setting is 10 Hz/s.

`void ProMPT_SetBoostEndModulation(unsigned char modulation)`

Resources used: Hardware Multiplier; 0 stack levels

`modulation` range: 0 to 200

Description: Sets the End Modulation (in %) for the Boost logic. Boost mode operates at Boost Frequency, and the modulation ramps from `BoostStartModulation` to `BoostEndModulation`. This function should not be called while Boost is enabled.

`unsigned char ProMPT_SetBoostFrequency(unsigned char frequency)`

Resources used: 0 stack levels

`frequency` range: 0 to 127

Description: Sets the frequency the drive goes to in Boost mode. Frequency must be < 128. On exit, w = 0 if the command is successful, or w = FFh if the frequency is out of range. This function should not be called while Boost is enabled.

`void ProMPT_SetBoostStartModulation(unsigned char modulation)`

Resources used: Hardware Multiplier; 0 stack levels

`modulation` range: 0 to `BoostEndModulation`

Description: Sets the Start Modulation (in %) for the Boost logic. Boost mode operates at Boost Frequency, and the modulation ramps from `BoostStartModulation` to `BoostEndModulation`. This function should not be called while Boost is enabled.

`void ProMPT_SetBoostTime(unsigned char time)`

Resources used: Hardware Multiplier; 0 stack levels

`time` range: 0 to 255

Description: Sets the amount of time in seconds for the Boost mode. Boost mode operates at Boost Frequency, and the modulation ramps from `BoostStartModulation` to `BoostEndModulation` over `BoostTime`. This function should not be called while Boost is enabled.

`void ProMPT_SetDecelRate(unsigned char rate)`

Resources used: 0 stack levels

`rate` range: 0 to 255

Description: Sets the deceleration to the value of rate in Hz per second. The default setting is 5 Hz/s.

`unsigned char ProMPT_SetFrequency(unsigned char frequency)`

Resources used: 2 stack levels

`frequency` range: 0 to 127

Description: Sets the output frequency of the drive if the drive is running. Frequency is limited to 0 to 127, but should be controlled within the valid operational range of the motor. Modulation is determined from the V/F curve, which is set up with the `ProMPT_SetVFCurve` method. If frequency = 0, the drive will stop. If the drive is stopped and frequency > 0, the drive will start.

void ProMPT_SetLineVoltage(unsigned char voltage)

Resources used: Hardware Multiplier; 0 stack levels

voltage range: 0 to 255

Description: Sets the line voltage for Automatic Voltage Compensation. The units for SetLineVoltage and SetMotorVoltage must be the same for accurate operation. The values passed to SetMotorVoltage and SetLineVoltage can be the same to disable voltage compensation.

void ProMPT_SetMotorVoltage(unsigned char voltage)

Resources used: Hardware Multiplier; 0 stack levels

voltage range: 0 to 255

Description: Sets the motor rating for Automatic Voltage Compensation. The units for SetLineVoltage and SetMotorVoltage must be the same for accurate operation. The values passed to SetMotorVoltage and SetLineVoltage can be the same to disable voltage compensation.

void ProMPT_SetParameter(unsigned char parameter, unsigned char value)

Resources used: 0 stack levels

parameter range:

Description: In addition to its pre-defined API methods, the ProMPT kernel allows the user to custom define up to 16 functions for control or communication purposes not covered by the ProMPT APIs. This function sets the value of the specified user defined function.

void ProMPT_SetPWMfrequency(unsigned char PWMfrequency)

PWMfrequency values: 0 or 1

Resources used: Timer2; 1 stack level

Description: This sets and changes the PWM switching frequency. Typically, this is set with the Init() function. When PWMfrequency is ‘0’, the module’s operating frequency is 9.75 kHz. When PWMfrequency is ‘1’, the module’s operating frequency is 19.53 kHz.

void ProMPT_SetVFCurve(unsigned char point, unsigned char value)

Resources used: Hardware Multiplier; 0 stack level

point range: 0 to 16 (0 = 0 Hz, 1 = 8 Hz, 2 = 16 Hz……. 17 = 128 Hz)

value range: 0 to 200

Description: This sets one of the 17 modulation values (in %) for the V/F curve. Each point represents a frequency increment of 8 Hz, ranging from point 0 (0 Hz) to point 16 (128 Hz).

unsigned char ProMPT_Tick(void)

Resources used: 1 stack level

Description: The value of the Tick timer flag becomes ‘1’ every 62.5 ms (1/16 second). This can be used for timing applications. clearTick must be called in the timing routine when this is serviced.

APPENDIX C: SOURCE CODE

Due to size considerations, the complete source code for this application note is not included in the text. A complete version of the source code, with all required support files, is available for download as a Zip archive from the Microchip web site at:

www.microchip.com

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families 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 Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is 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 products.

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, KEELOQ, MPLAB, PIC, PICmicro, PICSTART and PRO MATE are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

FilterLab, microID, MXDEV, MXLAB, PICMASTER, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

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

Serialized Quick Turn 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.

Printed on recycled paper.

DNV Certification, Inc.
USA

ANSI★RAB
QMS
ACCREDITED

DNV

DNV MSC
The Netherlands
Accredited by the RvA

ISO 9001 / QS-9000
REGISTERED FIRM

Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999 and Mountain View, California in March 2002. The Company’s quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, non-volatile memory and analog 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: 480-792-4338

Atlanta
3780 Mansell Road, Suite 130
Alpharetta, GA 30022
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

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. 2401-2402, 24th Floor,
Ming Xing Financial Tower
No. 88 TIDU Street
Chengdu 610016, China
Tel: 86-28-86766200 Fax: 86-28-86766599

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 - Hong Kong SAR
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

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. 1812, 18/F, Building A, United Plaza
No. 5022 Binhe Road, Futian District
Shenzhen 518033, China
Tel: 86-755-82901380 Fax: 86-755-82966626

China - Qingdao
Rm. B503, Fullhope Plaza,
No. 12 Hong Kong Central Rd.
Qingdao 266071, China
Tel: 86-532-5027355 Fax: 86-532-5027205

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: 65-6334-8870 Fax: 65-6334-8850

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

EUROPE

Austria
Microchip Technology Austria GmbH
Durisolstrasse 2
A-4600 Wels
Austria
Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

Denmark
Microchip Technology Nordic ApS
Regus Business Centre
Lautrup hoj 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 - ler Etage
91300 Massy, France
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

Germany
Microchip Technology GmbH
Steinheilstrasse 10
D-85737 Ismaning, 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
Microchip Ltd.
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44 118 921 5869 Fax: 44-118 921-5820

12/05/02