1.1 SETTING HARDWARE PARAMETERS

It is required to scale the motor phase voltages through a resistor network in order to make them compatible with the maximum and minimum ADC input voltage levels as specified in the device data sheet.

Figure 1-1 shows a typical circuit for a Vbus equal to 48V. Figure 1-2 shows a typical circuit for a 280V motor.

FIGURE 1-1: SIGNAL CONDITIONING CIRCUITRY FOR THE LOW-VOLTAGE MOTOR PHASE

This is the typical signal conditioning circuit for a Vbus equal to 48V.
This is the typical signal conditioning circuit for a VBUS equal to 400V. It is required to sample the motor voltage phase signals simultaneously; therefore, they have to be connected to ADC input pins in which the CH1, CH2, CH3 sample and hold circuits can be used. These channels can be AN0, AN1, AN2 or AN3, AN4 and AN5.

1.2 SOFTWARE PARAMETERS

All of the configurable parameters are defined in the UserParameters.h file. The parameters are classified into three different categories: basic, advance, and derived. Figure 1-3 shows these parameters and Table 1-1 provides descriptions of each parameter.
FIGURE 1-3: SOFTWARE PARAMETERS

/*Timing Defines*/
#define DESIRED_MIPS 40
#define FIN 7.3728e6
#undef XTAL
#define PWM_FREQUENCY 20e3

/*Motor Control Definitions*/
/*START-UP SEQUENCE PARAMETERS*/
#define MAX_PWM_TICKS 1230
#define RAM_UP_DELAY 1
#define MAX_DUTY_CYCLE_PERCENTAGE 100
#define MIN_DUTY_CYCLE_PERCENTAGE 2
#define PHASE_ADVANCE_DEGREES 5
#define BLANKING_COUNT 5

/*Close Loop Defines*/
#define CLOSELOOPMODE
#define PROPORTINAL_GAIN 2500
#define INTEGRAL_GAIN 256
#define DIFFERENTIAL_GAIN 0

/*Hardware definitions*/
#define S3 !PORTBbits.RB4
#define S2 !PORTAbits.RA3

/********************ADVANCE DEFINITIONS*****************
#define TIMER1_PRESCALER 256
#define TIMER2_PRESCALER 256
#define SAMPLING_POINT_AT_LOWSPEEDS 25
#define SAMPLING_POINTS_AT_HIGHSPEEDS 75
<table>
<thead>
<tr>
<th>Software Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESIRED_MIPS</td>
<td>This macro defines the operating MIPS of the dsPIC device.</td>
</tr>
<tr>
<td>FIN</td>
<td>This macro defines the input clock frequency utilized by the system clock circuitry.</td>
</tr>
<tr>
<td>XTAL</td>
<td>If this macro is defined, the dsPIC device utilizes an external crystal as the source for the input clock. The FRC clock is selected when this macro is not defined.</td>
</tr>
<tr>
<td>PWM_FREQUENCY</td>
<td>This macro sets the output frequency of the PWM module.</td>
</tr>
<tr>
<td>MAX_PWM_TICKS</td>
<td>This macro sets the number of pulses applied in the forced commutation process utilized for breaking the motor's stand-still inertia. The possible value could be from 1 to 65535. This value is set according to the motor and load characteristics.</td>
</tr>
<tr>
<td>RAMP_UP_DELAY</td>
<td>This macro sets the delay required to switch from the forced commutation mode to sensorless mode. This delay is useful when the motor is attached to high-inertia loads such as fans with large blades.</td>
</tr>
<tr>
<td>MAX_DUTY_CYCLE_PERCENTAGE</td>
<td>This macro limits the maximum PWM duty cycle. It is expressed in PWM cycles.</td>
</tr>
<tr>
<td>MIN_DUTY_CYCLE_PERCENTAGE</td>
<td>This macro sets the duty cycle for the start-up sequence. It is defined in PWM cycles. Once the motor is running this number also limits the minimum duty cycle that can be applied to the motor.</td>
</tr>
<tr>
<td>CLOSELOOPMODE</td>
<td>When defined, this macro enables the speed controller. If it is disabled, the motor will run without the speed controller.</td>
</tr>
<tr>
<td>PROPORTIONAL_GAIN</td>
<td>This macro sets the P gain for the speed PID controller.</td>
</tr>
<tr>
<td>INTEGRAL_GAIN</td>
<td>This macro sets the I gain for the speed PID controller.</td>
</tr>
<tr>
<td>PHASE_ADVANCE DEGREES</td>
<td>Phase advance angles to get the best motor performance or surpass the nominal speed.</td>
</tr>
<tr>
<td>BLANKING_COUNT</td>
<td>Sets the blanking count expressed in PWM cycles used to avoid false zero-crossing detection.</td>
</tr>
<tr>
<td>S3</td>
<td>This macro defines the I/O port used to poll the push button S3.</td>
</tr>
<tr>
<td>S2</td>
<td>This macro defines the I/O port used to poll the push button S2.</td>
</tr>
</tbody>
</table>
1.3 SETTING START-UP PARAMETERS

In order to determine the start-up parameters it is required to follow the described procedure.

1. Disable the speed loop controller. It is possible to disable this controller by removing the CLOSELOOPMODE definition. If DMCI with RTDM is used to control the motor, make sure the “ENABLE PI” button is OFF.

2. Set the reference speed (using the pot or the DMCI slider, depending on which SW version is running), “maximum PWM ticks”, “Phase Advance”, and “ramp up delay” values to 0. Set “Blanking Count” to 5.

3. Next, we are going to determine the correct values for “minimum duty cycle” and “maximum PWM ticks”. This is a trial-and-error process that depends on the motor and load characteristics.

4. Turn the motor ON by pressing S2 (non-DMCI version) or by turning ON the START/STOP button on DMCI.

5. Slowly increase (steps equal to 10 units) the “maximum PWM ticks” value. Stop increasing the “maximum PWM ticks” value when the motor shaft starts bouncing backward and forward.

6. If the motor shaft is not bouncing backward and forward, but instead is emitting a high-pitched sound, increase the “minimum duty cycle” by 10 unit steps and repeat steps 4-6 until the motor shaft is bouncing backward and forward.

7. The value “minimum PWM duty cycle” is close to the correct the start-up value when the motor shaft is bouncing back and forward. At this point it is required to find the minimum duty cycle in order to completely break the stand-still inertia.

8. Increase (using one step at a time) the “minimum PWM duty cycle”, and stop when the motor is continuously spinning.

9. Save the current “minimum PWM duty cycle” and “maximum PWM ticks”.

10. Stop the motor.

11. Start the motor using the saved “minimum PWM duty cycle” and “maximum PWM ticks” for the start-up sequence. The motor should be running.

Note: In some applications where high-inertia loads are attached to the motor is required to use a delay allowing the software make a smooth transition from the start-up sequence to sensorless mode. This delay is set by the “ramp-up delay” value. Increase or decrease this value as required by the application, specifically when the motor shaft it is not bouncing after trying different combinations of “minimum PWM duty cycle” and “maximum PWM ticks” values.

Figure 1-4 shows the start-up tuning procedure flow chart
FIGURE 1-4: TUNING PROCEDURE FOR THE START-UP SEQUENCE

Disable Close Loop Controller (Speed Controller)

RAMP_UP_DELAY = 0
MIN_DUTY_CYCLE_PERCENTAGE = 10
MAX_PWM_TICKS = 0
PHASE_ADVANCE_DEGREES = 0
BLANKING_COUNT = 5

Start Motor

Increase MAX_PWM_TICKS

Increase MIN_DUTY_CYCLE

Is the motor emitting a high-pitched sound?

NO

Is the motor shaft bouncing back and forward? (~every 500 mS)

NO

YES

Increase MIN_DUTY_CYCLE_PERCENTAGE

Is the motor running?

NO

YES

END
1.4 ADVANCE TUNING PARAMETERS

The other tuning parameters such as the PI gains, the Phase Advance Angle, and the Blanking count number requires a better understanding of the algorithm used in the application note. Please refer to application note AN1160 “Sensorless BLDC Control with Back-EMF Filtering Using a Majority Function”.

Tuning the close loop gains P and I can be done using multiple methods. Please refer to Wikipedia and search for “PID controller” and go to the section “Loop Tuning” for more information, or use the following link to access the information directly:

http://en.wikipedia.org/wiki/PID_controller#Loop_tuning

It is possible to improve the performance of the algorithm by increasing the phase advance angle and the blanking counter. To do so, it is required to observe the current waveform using a current probe and a oscilloscope.

1. Attach the current probe to any of the motor phases.
2. Start the motor.
3. Increase the motor speed slowly and stop at ¼ point of the maximum rated speed.
4. Change the phase advance and blanking counter values in order to get the current waveform as close as possible to Figure 1-5 when full load is applied to the motor.
5. Observe the current waveform while increasing the speed to ¼ of full speed.
6. Repeat steps 4-5 until the motor reaches the maximum rated speed.
7. The current waveform at this point should be similar in shape as Figure 1-5 in a wide speed range.

FIGURE 1-5: DESIRED PHASE CURRENT WAVEFORM
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. Attempts to break Microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications.

MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights.

Trademarks

The Microchip name and logo, the Microchip logo, dsPIC, KEELOG, KEELOG logo, MPLAB, PIC, PICmicro, PICSTART, PIC® logo, rPIC and Uni/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

FilterLab, Hampshire, HI-TECH C, Linear Active Thermistor, MXDEV, MXLAB, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, Application Maestro, CodeGuard, dsPICDEM, dsPICDEM.net, dsPICworks, dsSPEAK, ECAN, ECONOMONITOR, FanSense, HI-TIDE, In-Circuit Serial Programming, ICSP, Mindi, MiWi, MPASM, MPLAB Certified logo, MPLIB, MPLINK, mTouch, Octopus, Omniscient Code Generation, PICC, PICC-18, PICDEM, PICDEM.net, PICKit, PICtail, REAL ICE, rILAB, Select Mode, Total Endurance, TSHARC, UniWinDriver, WiperLock and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

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.

© 2010, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

Printed on recycled paper.

ISBN: 978-1-60932-342-4

Microchip received ISO/TS-16949:2002 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV

ISO/TS 16949:2002

© 2010 Microchip Technology Inc.