Readme File for AN1160 Software:
AN1160 MC204 dsPICDEM MCLV

This file contains the following sections:

1. Application Note Software Description

This application note describes a sensorless Brushless Direct Current (BLDC) motor control algorithm that is implemented using dsPIC® digital signal controller (DSC). The algorithm works utilizing a majority function for digitally filtering the Back-Electromotive Force (BEMF). Each phase of the motor is filtered to determine when to commutate the motor drive voltages. This control technique excludes the need for discrete, low-pass filtering hardware and off-chip comparators. It should be pointed out that all the discussions here, and the application software, assume a 3-phase motor has to be used. The motor control algorithm described here has six main parts:

- Sampling trapezoidal BEMF signals using the dsPIC Analog-to-Digital Converter (ADC)
- Reconstructing the Motor Virtual Neutral Point
- Comparing the trapezoidal BEMF signals to the reconstructed motor virtual neutral point to detect the zero crossing points
- Filtering the signals coming from the comparisons using a majority function filter
- Calculate the rotor speed, the commutation delay and the phase advance angle
- Commutate the motor driving voltages
- Control loop

Function: main()
Overview:
Main function used to initialize the ADC, PWM, UART and TIMER2 modules. It also initializes the global variables used in the interrupts and PID controller. The main task executed here is to handle the start and stop the motor as well as setting the ramp-up initial parameters to spin the motor. It also handles the data exchange between the Host PC and the target device.
Note: None

Function: _ADC1Interrupt ()
Overview: ADC interrupt used to measure the BEMF signals, reconstruct the Motor Virtual Neutral Point and compare the BEMF signals against the neutral point reference to detect the zero-crossing event. It also fills the DMCI buffers with the data to plot on the DMCI data view window
Note: None

Function: _MPWM1Interrupt ()
Overview: PWM reload interrupt used to filter the BEMF signals using the Majority detection filter to detect a valid zero-crossing event. If a valid zero-crossing event was detected then it calls the PreCommutationState function. This function also includes the start-up sequence for detecting the initial rotor position
Note: None

Function: _T1Interrupt ()
Overview: Here is where the motor commutation occurs; Timer1 ISR is utilized as the commutation delay used to commutate the motor windings at the right time. It also resets the Blanking Counter. This counter must be cleared after a commutation occurs in order to the detection of false zero-crossing events.
Note: None

Function: PreCommutationState ()
Overview: This function measures the 60 and 30 electrical degrees using the TIMER2. The 60 electrical degrees is proportional to the elapsed time between zero-crossing events. The zero-crossing events occur 30 electrical degrees in advance of the commutation point. Hence a delay proportional to the 30 electrical degrees is added using the TIMER1. This commutation delay can be modified in order to phase advance the commutation. The phase advance angle can be any value within 0 to 30 angles.
This function also calls the open loop function or the close loop function (depending on which mode is activated) when the zero-crossing event is detected.
Note: None

Function: SpeedPILoopController()
Overview: When the macro “CLOSE_LOOP_MODE” is defined the motor operates in close loop mode. The Speed_P, Speed_I and Speed_D parameters were determined using the HURST MOTOR sold through the microchip direct webpage (Part Number:

AC300020). These values should be recalculated according to the motor and load characteristics.
Note: None

Function: OpenLoopController()
Overview: When the the macro "CLOSE_LOOP_MODE" is not defined the motor operates in open loop mode.
Note: None

Function: InitADC10()
Overview: Initializes the ADC module to operate in simultaneous mode sampling terminals AN0, AN1, AN2, AN3 using MUX A. The ADC channels are assigned as follows in the dsPICDEM MCLV board CH0->AN8 (POT), CH1->AN3 PHASE A, CH2->AN4 PHASE B, CH3->AN5 PHASE C, ADC is sync with the PWM. ADC is conversion is triggered every time a PWM reload event occurs. Tadc = 84.75 nSec. ADC resulting samples are formatted as unsigned 10-bits Right-justified.
Note: None

Function: InitMCPWM()
Overview: Initializes the PWM module to operate in center-aligned mode at 20KHz. PWM terminals are configured in independent mode. PWM time base is 67.8 nSec. PDCx value range is 0-1464 for 0%-100% duty cycle ADC reload time is variable according to the PWM duty cycle
Note: None

Function: InitTMR2()
Overview: Initializes the TIMER2 module to operate in free-running up counting mode. The TIMER2 time base is Tcy*64= 2.17uSec. This timer is used to calculate the motor speed.
Note: None

Function: InitTMR1()
Overview: Initializes the TIMER1 module to operate in free-running up counting mode. The TIMER1 time base is Tcy*64 = 2.64uSec. This timer is used to calculate the commutation delay.
Note: None

Function: DelayNmSec(unsigned int N)
Overview: Delay function used for push buttons denounce loop and for the motor start-up sequence.
Note: None

2. Project Files

This project contains the following files:

a. C:\Program Files\Microchip\MPLAB C30\support\gld

This folder will have the device GLD file; it is used for building the project. This file was provided with the MPLAB C30 v3.1 tool suite.

b. C:\Program Files\Microchip\MPLAB C30\support\h

This folder contains C header files useful in building this project. Device register and bit definitions are provided in the *.h file that follows the device name. These files were provided with the MPLAB C30 v3.1 tool suite.

c. C:\Program Files\Microchip\MPLAB C30\lib

This folder contains library archive files, which are a collection of precompiled object files. The file "libdsp-coff.a" contain the C run-time start-up and the PID loop control library. These file were provided with the MPLAB C30 v3.1 tool suite.

d. H

There is a "p33FJ32MC204.h" file; this file has all the dsPIC33F register definitions.

e. SRC

This folder contains all the C and Assembler source files (*.c) used in demonstrating the described example. You will see AN1160 MC204dsPICDEM MCLV rev2.c

f. Linker Script

This folder contains the "p33FJ32MC204.gdl" file; this file has all the dsPIC33F memory section definitions.

3. Suggested Development Resources

a. MPLAB IDE v8.10 or higher
b. MPLAB C30 v3.10 or higher
c. MPLAB ICD2 or MPLAB RealICE
d. dsPICDEM MCLV development board (DM330021)
e. dsPIC33 MC 44p TO 100p Plug In Module (MA330017)
f. 24V 3-phase Brushless DC motor (AC300020)
g. 24V power supply (AC002013)

4. Running the Code Example

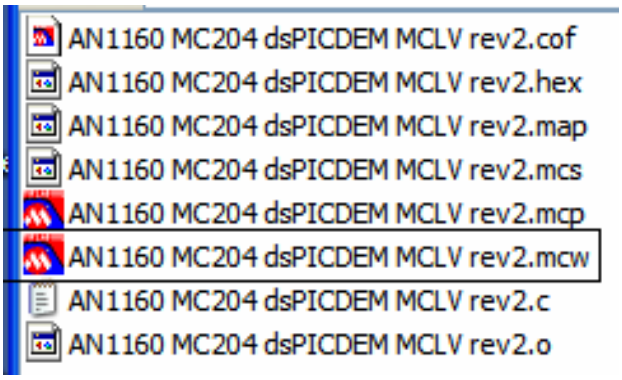
a. On the dsPICDEM MCLV development board (DM330021) make sure that the mini jumpers are configured as follows
   - JP1, JP2 and JP3 on the "VOLT" position
   - JP4 and JP5 open
   - JP11 and J5 open

b. Mount the dsPIC33 MC 44p TO 100P Plug In Module (MA330017) on the dsPICDEM MCLV development board

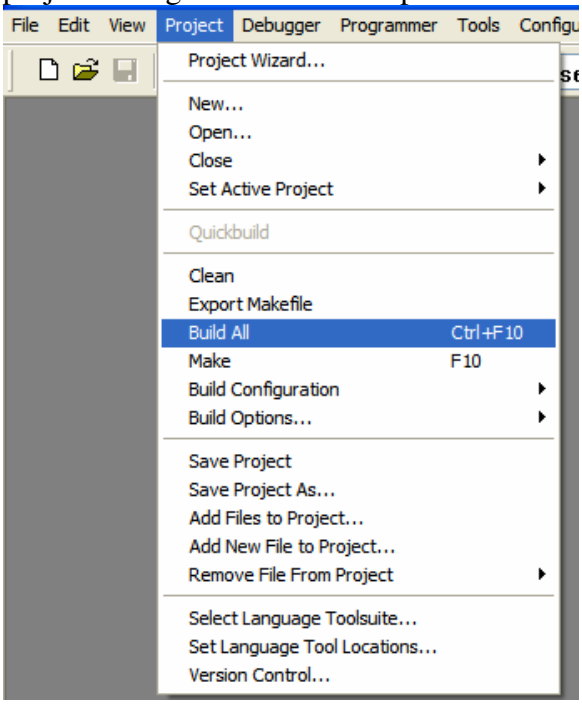
c. Connect the HURST motor (AC300020) to the dsPICDEM MCLV development board

d. Power-On the dsPIC MCLV development board using the suggested 24V power supply (AC002013). To check the possible power supply options please refer to the dsPICDEM MCLV Users Guide.

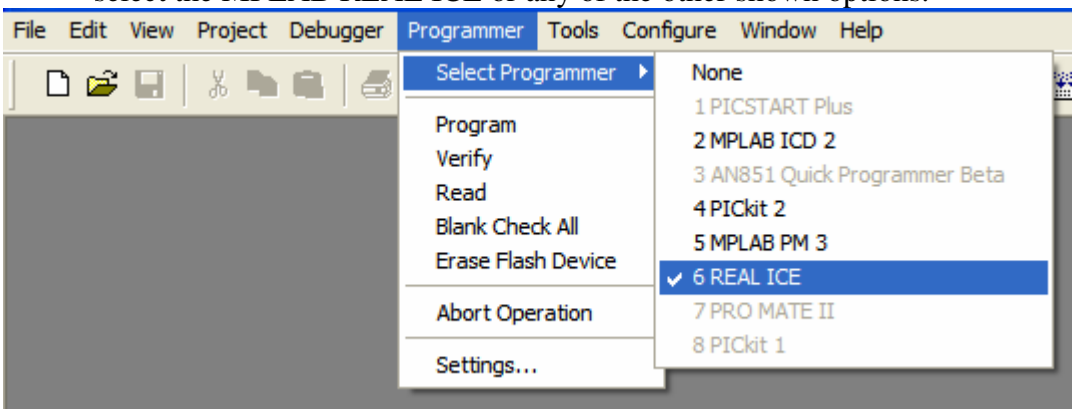
e. Open the AN1160 software by double-clicking on the "RTDM Code AN1160 MC204 dsPICDEM MCLV rev2.mcw" file.



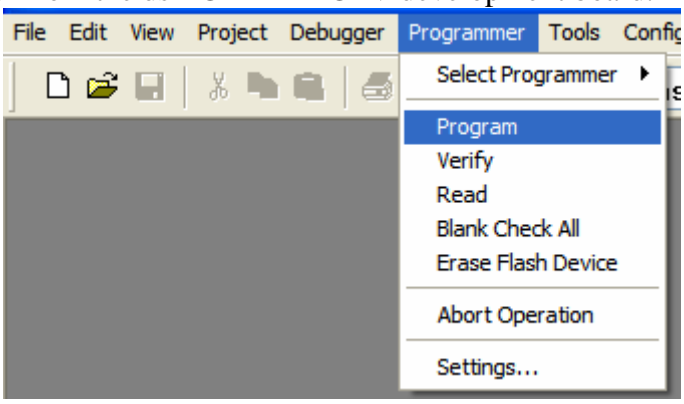
f. Once the MPLAB work-bench is open COMPILE the project. Build the project using "Build All" option from the "Project" drop down menu



g. Select the programmer from the "Programmer" drop down menu. In this case select the MPLAB REAL ICE or any of the other shown options.



h. Connect MPLAB REAL ICE to dsPICDEM MCLV development board using the J11 connector and program the device using the "program" command from the drop down menu. After programming the device, disconnect the debugger from the dsPICDEM MCLV development board.



i. Controlling the motor using AN1160 via push button (S3) and potentiometer (POT1). To run or stop the motor press the S3 button.
j. Move the pot in Clock Wise direction to increase the speed. To reduce the speed move the pot in Counter Clock Wise direction

5. Reconfiguring the project for a different dsPIC device:

The Project/Workspace can be easily reconfigured for any dsPIC device.
Please use the following general guidelines:

1. Change device selection within MPLAB IDE to a dsPIC device of your choice by using the following menu option:
2. MPLAB IDE>>Configure>>Select Device

3. Provide the correct device linker script and header file for your device. Device linker scripts and header files are available in your MPLAB® C30 installation folder under:
4. Device Linker Script-
    a. YourDrive:>Program Files\Microchip\MPLAB C30\support\gld
5. Device C Header file-
    a. YourDrive:>Program Files\Microchip\MPLAB C30\support\h
6. Provide the appropriate path to your MPLAB C30 support file locations using the menu option:
7. MPLAB IDE>>Project>>Build Options>>Project

8. Re-build the MPLAB project using the menu option:
9. MPLAB IDE>>Project>>Build All

10. Download the hex file into the device and run.

6. Revision History:

Daniel Torres	12/15/2007	First Revision
Daniel Torres	08/06/2008	Second Revision