INTRODUCTION

Sinusoidal current drive is known to deliver improvements in performance in terms of motor vibration and acoustic noise. Motor controls may therefore use a sinusoidal current drive in preference to the traditional rectangular/square current drive given that overall system implementation cost should be analogous with the complexity of motor design. In this application note, 8-bit PIC® microcontroller will be used to implement sinusoidal current drive on a low-cost single-phase or 2-phase (Split Phase) BLDC Motor with sensored or sensorless mode.

FIGURE 1: BLOCK DIAGRAM

Figure 1 shows the block diagram of the sinusoidal current drive based on the 8-bit PIC microcontroller. The PIC16F1618/9 is used to drive both single and 2-phase BLDC motor with either sensored or sensorless control. The driver utilizes the Core Independent Peripherals (CIPs) in the microcontroller to perform motor control function from its Central Processing Unit (CPU). The controller is implemented using the following CIPs:

- Angular Timer (AngTMR) for rotor angular position detection
- Signal Measurement Timer (SMT) for desired speed calculation
- Complementary Waveform Generator (CWG) for H-Bridge MOSFET driver
- Math Accelerator (MathAC) for hardware-based mathematical calculations.

These peripherals are internally connected by firmware, significantly reducing the number of external pins required for the implementation. Refer to Appendix A: “Circuit Schematics” for the detailed schematic diagram.

Note: An alternate Micrel MIC4606 H-Bridge driver can also be used with the designed H-bridge circuit in this application note. Refer to www.microchip.com for the complete list of available H-bridge drivers.
SINUSOIDAL DRIVE IMPLEMENTATION

In order to generate the rotating magnetic field required to drive a single or 2-phase BLDC Motor, the excitation on the stator winding must be sequenced in a specific manner while knowing the exact position of the rotor magnets. The rotor magnet position is determined by using the hall effect sensor for sensored control or the motor-generated BEMF signal for the sensorless control. Refer to AN2049: "Single-Phase BLDC Motor Driver" (DS00002049) and AN1083: "Sensorless BLDC Control with Back-EFM Filtering" to know more about driving a sensored or sensorless BLDC motor.

The Complementary Waveform Generator (CWG) peripheral controls the excitation of the stator winding based on the state of the hall sensor output or BEMF signal. In order to control the CWG output, the input hall or BEMF signal is compared to a Fixed Voltage Reference (FVR) by the Comparator 1 (C1). The comparator hysteresis is enabled to disregard the noise that might add to the signal. Refer to Figure 2 for simplified implementation of BEMF signal detection.

During motor initial start-up, Comparator 1 (C1) detects the initial state of the input signal. When the input signal is high, C1 forces the CWG into Full-Bridge Forward mode. Otherwise, when C1 is low, it forces the CWG into Full-Bridge Reverse mode. This will ensure the correct switching of the MOSFET driver based on the rotor angular position. The rate of switching between Forward and Reverse CWG mode depends on the motor desired speed control, which is given by the Signal Measurement Timer (SMT) peripheral. While the motor is spinning, the Angular Timer (AngTMR) detects the angular position of the rotor from the C1 output. The AngTMR subdivides the periodic C1 signal into 360 equally spaced intervals. Each division will trigger an interrupt that will change the modulation of the CWG's PWM input based on the predetermined sine lookup table in firmware. On every AngTMR interrupt, an equivalent pre-calculated value in sine look-up table will be available to Math Accelerator (MathACC) peripheral to multiply with the duty cycle of the external PWM speed control provided by the SMT. The output of MathACC will be used to produce a Sinusoidal PWM (SPWM). This SPWM will be used by the CWG in driving the H-Bridge circuit of Single Phase BLDC motor or MOSFET switches of 2-Phase BLDC Motor and eventually spin the motor with sinusoidal current. Figure 3 shows the resulting signal of the Sinusoidal Pulse Width Modulation. Figure 4 illustrates the flow of mathematical calculation for the system to achieve the sinusoidal current drive.
FIGURE 3: SINUSOIDAL PULSE WITH MODULATION IMPLEMENTATION

FIGURE 4: SINUSOIDAL DRIVE DATA FLOW REPRESENTATION

\[ \text{Angular Timer (AngTMR)} \]

\[ \omega t = \theta = 360^\circ \times \frac{\text{AngTMR}_\text{PHS}}{\text{AngTMR}_\text{RES} + 1} \]

\[ \text{MathACC Multiply Operation} \]

\[ [A \times \sin(\omega t)] \times \text{PWM Reso} \]

\[ \text{SPWM PWM/DC Value} \]

\[ \text{CWG} \]
Motor Speed Adjustment Using the Signal Measurement Timer (SMT)

An external pulse-width modulated signal is commonly used as an input for adjusting the speed of a BLDC Motor. The PWM duty cycle corresponds to the user’s desired speed. Using the SMT Period and Duty Cycle mode, the input duty cycle can be accurately read and translated into user-desired speed even if a change in the input frequency is applied. Equation 1 shows the sample desired speed calculation when a 21 kHz PWM pulse with 50% duty cycle is applied.

EQUATION 1: DESIRED SPEED CALCULATION FORMULAS

\[
SMT_{x\text{CPR}} = \frac{SMT \text{ Clock Source}}{Input \text{ Frequency}} = \frac{HFINTOSC}{21 \text{ kHz}} \approx 3047
\]

\[
SMT_{x\text{CPW}} = \frac{SMT \text{ Clock Source}}{50\% \text{ Duty Cycle Frequency}} = \frac{HFINTOSC}{42 \text{ kHz}} \approx 1523
\]

\[
\text{Speed Magnitude (A)} = \frac{SMT_{\text{CPW}} \times \text{Scaling Factor}}{SMT_{\text{CPR}}} = \frac{1523 \times 1023}{3047} \approx 511
\]

Referring to Equation 1, the motor speed magnitude (A) corresponds to the product of the SMT’s CPW value and scaling factor divided by the SMT’s CPR value. The scaling factor is used to prevent a decimal/fraction number as a result of the speed magnitude. This is done to avoid a complex mathematical calculation of decimal numbers for the firmware and the MathACC peripheral. Table 1 shows the equivalent motor speed for every SMT calculation when a 21 kHz PWM signal and a 1023 scaling factor are used.

TABLE 1: DESIRED SPEED CALCULATION TABLE

<table>
<thead>
<tr>
<th>Input Signal Duty Cycle (%)</th>
<th>SMT Converted Pulse Width (SMT_CPW)</th>
<th>SMT Converted Period (SMT_CPR)</th>
<th>Speed Magnitude (A)</th>
<th>Equivalent Motor Speed (RPM)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>3047</td>
<td>3047</td>
<td>1023</td>
<td>5400 (Rated)</td>
</tr>
<tr>
<td>90</td>
<td>2742</td>
<td>3047</td>
<td>921</td>
<td>4860</td>
</tr>
<tr>
<td>80</td>
<td>2438</td>
<td>3047</td>
<td>818</td>
<td>4320</td>
</tr>
<tr>
<td>70</td>
<td>2133</td>
<td>3047</td>
<td>716</td>
<td>3780</td>
</tr>
<tr>
<td>60</td>
<td>1828</td>
<td>3047</td>
<td>614</td>
<td>3240</td>
</tr>
<tr>
<td>50</td>
<td>1524</td>
<td>3047</td>
<td>512</td>
<td>2700</td>
</tr>
<tr>
<td>40</td>
<td>1219</td>
<td>3047</td>
<td>409</td>
<td>2160</td>
</tr>
<tr>
<td>30</td>
<td>914</td>
<td>3047</td>
<td>307</td>
<td>1620</td>
</tr>
<tr>
<td>20</td>
<td>609</td>
<td>3047</td>
<td>205</td>
<td>1080</td>
</tr>
<tr>
<td>10</td>
<td>305</td>
<td>3047</td>
<td>102</td>
<td>540</td>
</tr>
</tbody>
</table>

For more information regarding the Signal Measurement Timer Peripheral, refer to TB3129, “Signal Measurement Timer Peripheral” (DS90003129).
Motor Driver Using Complementary Waveform Generator (CWG)

A proper excitation sequence is necessary to produce a rotating motion on a BLDC Motor. This excitation sequence is provided by the firmware and used by the Complimentary Waveform Generator (CWG) to drive the MOSFET Drivers. When the C1 output changes from high to low, the CWG drive will also change from Forward-to-Reverse Full Bridge mode. The toggling from Forward-to-Reverse mode produces a clockwise rotation, while toggling from Reverse-to-Forward mode produces a counterclockwise rotation. In addition, aside from toggling the CWG Full-Bridge mode, an input PWM peripheral source is used in the CWG to control the MOSFET switches of a 2-phase BLDC Motor and low-side MOSFETs of the H-Bridge circuit of the single-phase BLDC motor. The input PWM changes its duty cycle value in accordance with the result of the MathACC calculation. The continuous changes in the CWG's PWM duty cycle during motor run time are patterned to produce a Sinusoidal Pulse-Width Modulation (SPWM). More detailed information on the MathACC implementation can be found on Section “Hardware Mathematical Calculation using Math Accelerator (MathACC)”.

For more information regarding the different drive modes and uses of the complementary waveform generator peripheral in driving a motor, refer to TB3118, “Complementary Waveform Generator Technical Brief” (DS90003118).

Motor Angular Position Detection Using Angular Timer (AngTMR)

The angular timer peripheral is used to interpret the periodic signal from the Comparator 1 that came from the hall sensor or motor BEMF as an angle measurement instead of a time measurement. This periodic signal is converted into a 360-degree-angular representation on which all measurements are based. At the start-up procedure, the control is started in an Open Loop mode to allow the periodic square wave signal to build up and adapted by the angular timer. While the BLDC motor is rotating, the rising edge of the C1 output indicates the home position of measurement. As the motor rotates, the C1 output produces a repeating or periodic signal used for the representation of a 360 angular degree. The periodic signal is divided into 360 equidistant angles with each degree of rotation matched to a time element. The AngTMR handles the calculation and will adjust automatically even if the hall or BEMF signal frequency changes due to the change in motor speed. Figure 5 shows a sample operation of the angular timer in detecting the motor angular position on every 30° division or 12 equidistant angles. The 30° division in Figure 5 is just an illustration, the 360 division or one degree/division is used on the actual application.

FIGURE 5: ANGULAR TIMER ANGULAR POSITION DETECTION

Note: 30° division in this figure is just for illustration. 360 division or 1 degree/division is used on the actual application.
On every represented angular degree of the angular timer there is an equivalent precalculated value on the sine look-up table. This value is used as an input to the Math Accelerator (MathACC) peripheral together with the SMT calculated output.

**Hardware Mathematical Calculation using Math Accelerator (MathACC)**

The MathACC peripheral is used to provide a hardware-based multiply function to speed up math performance and reduce code size. It is used to multiply the precalculated sine look-up table value of every AngTMR interrupt with the calculated SMT desired speed to derive the desired Sinusoidal Pulse-Width Modulation (SPWM) value. Equation 2 shows the calculation of the SPWM value. Table 2 shows the angular degree and the equivalent precalculated sine look-up table for 100% desired speed.

**EQUATION 2: SINUSOIDAL PULSE-WIDTH MODULATION CALCULATION**

\[
\text{Speed Magnitude (A)} = \frac{\text{SMT}_{\text{CPW}}}{\text{SMT}_{\text{CPR}}} \times \text{Scaling Factor}
\]

\[
\text{Sine Table Value} = \sin(\omega t) \times \text{CWG's PWM Resolution} = \sin\left(\frac{360^\circ \times \text{AT}_{\text{PHS}}}{\text{AT}_{\text{RES}} + 1}\right) \times \text{CWG's PWM Resolution}
\]

\[
\text{MATHACC Output} = \text{Desired Speed (A)} \times \text{Sine Table Value}
\]

\[
\text{SPWM} = \text{MATHACC Output} \div \text{Scaling Factor}
\]

**SYSTEM Firmware**

In implementing the sinusoidal drive, the maximum allowable speed and the user's designed code execution need to be taken into consideration to avoid missing any AngTMR Interrupt. The actual motor speed in RPM is derived from the hall sensor signal. For example, 4-pole single-phase or 2-phase BLDC Motor running at rated 10000 RPM will produce a maximum hall signal frequency of 333 Hz. If a sinusoidal drive with 1° angle division is used on a 333 Hz hall frequency, the AngTMR interrupt is expected to occur every 8.34 uS. Therefore, the code execution of deriving the SPWM value should be less than 8.34 uS to avoid missing any AngTMR interrupt. For this application note, the designed maximum code execution time of sinusoidal drive is at 6 uS. Since the designed code execution time is lower than the time occurrence of the AngTMR interrupt, the firmware design can successfully implement the sinusoidal drive. Refer to Figure 6 for the event timeline of firmware execution. In addition, the designed code execution time limits the maximum attainable speed of the motor, since the time between AngTMR interrupt events decreases as the hall frequency or motor speed increases. Equation 3 shows the firmware limitation for the maximum attainable speed of the sinusoidal drive.

**TABLE 2: CALCULATED SINE LOOK-UP TABLE BASED ON MOTOR ANGULAR DEGREE**

<table>
<thead>
<tr>
<th>AngTMR Phase (°)</th>
<th>IA/IB Modulation (%)</th>
<th>10Bit-PWM (Sine Look-up Table)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.75%</td>
<td>17</td>
</tr>
<tr>
<td>2</td>
<td>3.49%</td>
<td>35</td>
</tr>
<tr>
<td>3</td>
<td>5.23%</td>
<td>53</td>
</tr>
<tr>
<td>4</td>
<td>6.98%</td>
<td>71</td>
</tr>
<tr>
<td>5</td>
<td>8.72%</td>
<td>89</td>
</tr>
<tr>
<td>6</td>
<td>10.45%</td>
<td>107</td>
</tr>
<tr>
<td>7</td>
<td>12.19%</td>
<td>124</td>
</tr>
<tr>
<td>8</td>
<td>13.92%</td>
<td>142</td>
</tr>
<tr>
<td>9</td>
<td>15.64%</td>
<td>160</td>
</tr>
<tr>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>90</td>
<td>100.00%</td>
<td>1023</td>
</tr>
<tr>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>180</td>
<td>0.00%</td>
<td>0</td>
</tr>
</tbody>
</table>
FIGURE 6: EXECUTION TIMELINE

The 333 Hz Hall Sensor Signal is divided into 360 by Angular Timer and 1 division is equivalent to 8.36 us

Loading of the new CWG’s PWM Value should be done before the next AngTMR interrupt occurs or before the 1 AngTMR division time of 8.36 us expires to successfully implement the sinusoidal drive.
**EQUATION 3: MAXIMUM ATTAINABLE SPEED**

Given that:
- \( F_{OSC} = 32 \text{ MHz} \)
- \( \text{AngTMR Division} = 360 \text{ or } 1^\circ \text{ angle AngTMR Interrupt} \)

The maximum time for sinusoidal drive code execution is approximately 6 uS

\[
 Hall \ Frequency \ (MAX) = \frac{1}{\text{Code Execution Time(s) } \times \text{AT Division}} \\
 Hall \ Frequency \ (MAX) = \frac{1}{6 \text{ uS } \times 360} = 462 \text{ Hz} \\
 Motor \ Speed \ (RPM) = \frac{120 \times \text{Hall Frequency}}{\text{No. of Poles}} \\
 Motor \ Speed \ (MAX \ RPM) = \frac{120 \times 462 \text{ Hz}}{4} = 13860 \text{ RPM} \\
\]

The maximum attainable speed for this specification is 13860 RPM

If \( \text{AngTMR Division} = 180 \text{ or } 2^\circ \text{ angle AngTMR Interrupt} \)

\[
 Hall \ Frequency \ (MAX) = \frac{1}{6 \text{ uS } \times 180} = 925 \text{ Hz} \\
 Motor \ Speed \ (MAX \ RPM) = \frac{120 \times 925 \text{ Hz}}{4} = 27750 \text{ RPM} \\
\]

The maximum attainable speed for this specification is 27750 RPM

To illustrate the exact code flow chart for the sinusoidal drive, Figure 7 shows the main code and interrupt service routine (ISR) used on the design. All peripherals used in the firmware are configured and initialized using the MPLAB® Code Configurator (MCC). Appendix B: “MPLAB® Code Configurator (MCC) Peripheral Initialization” provides the procedures on how the peripherals are initialized using MCC. For the complete source code, refer to Appendix C: “Source Code Listing”.

---

Hall Frequency (MAX) = \( \frac{1}{6 \text{ uS } \times 360} \) = 462 Hz

Motor Speed (RPM) = \( \frac{120 \times 462 \text{ Hz}}{4} \) = 13860 RPM

Motor Speed (MAX RPM) = \( \frac{120 \times 925 \text{ Hz}}{4} \) = 27750 RPM
Figure 7: Flow Chart

Initialization

- Check Hall Sensor Signal
- Determine the Desired Speed
- Initialize Sine Look-Up Table
- Divide Hall Signal into 360 Division

ISR

AngTMR Period Interrupt?

- YES PhaseCount = 0
- NO

AngTMR Phase Interrupt?

- YES PhaseCount++
- NO

MathACC Multiply (SineTable[PhaseCount] x Desired Speed)

CWG’s PWM Value = MathACC Result

Return

MOTOR DRIVER PERFORMANCE

Figure 8 and Figure 9 show the actual captured waveform of the rectangular current drive together with its resulting motor torque/power, while Figure 10 and Figure 11 show the actual captured waveform of sinusoidal drive together with its resulting motor torque/power.

Figure 8: Winding Voltage and Current for Rectangular Current Drive

Winding A

- BEMF
- Current

Power/Torque

Winding B

- Current
- BEMF

Power/Torque
FIGURE 9:   WINDING RESULTING TORQUE FOR RECTANGULAR CURRENT DRIVE

FIGURE 10:   WINDING VOLTAGE AND CURRENT FOR SINUSOIDAL CURRENT DRIVE

FIGURE 11:   WINDING RESULTING TORQUE FOR SINUSOIDAL CURRENT DRIVE
CONCLUSION

In conclusion, this method generates a sinusoidal current drive that has a better motor torque response than the traditional square/rectangular current drive method. By reducing the AngTMR interrupt or angular resolution, a higher speed range also becomes possible. In addition, the solution also has the capability to create a wave shape tailored for a particular application by changing the PWM modulation on the look-up table. Reduction of overall system cost can add to its advantage if sensorless mode is implemented instead of sensored mode.
FIGURE A-1: SENSORED SINGLE-PHASE BRUSHLESS DC MOTOR USING PIC16F1618/9

APPENDIX A: CIRCUIT SCHEMATICS
FIGURE A-2: SENSORLESS 2-PHASE BRUSHLESS DC MOTOR USING PIC16F1618/9

[Diagram showing the sensorless 2-phase brushless DC motor circuit using PIC16F1618/9, with labeled components and connections.]
APPENDIX B: MPLAB® CODE CONFIGURATOR (MCC) PERIPHERAL INITIALIZATION

In this application note, the MPLAB® Code Configurator (MCC) is utilized to easily configure the peripherals used in this motor control application. The MCC is a user-friendly plug-in tool for MPLAB® X IDE which generates drivers for controlling and driving peripherals of PIC® microcontrollers, based on the settings and selections made in its Graphical User Interface (GUI). Refer to the MPLAB® Code Configurator User’s Guide (DS40001725) for further information on how to install and set up the MCC in MPLAB X IDE. The latest MCC file contains the MCC setup and configuration for this application and can be downloaded from the Microchip web site (www.microchip.com). The user will find the MC3 file appended to the electronic version of this application note.

**Note:** MCC Version 3.16 is used for writing the application note. The latest software version can be downloaded from the Microchip website (http://www.microchip.com/mplab/mplab-code-configurator).
APPENDIX C: SOURCE CODE LISTING

The latest software version can be downloaded from the Microchip web site (www.microchip.com). The user will find the source code appended to the electronic version of this application note. The latest version is v1.0.
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 unless otherwise stated.

Trademarks
The Microchip name and logo, the Microchip logo, AnyRate, AVR, AVR logo, AVR Freaks, BeaconThings, BitCloud, chipKIT, chipKIT logo, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, Heldo, JukeBlox, KEELoq, KEELoq logo, Kleer, LANCheck, LINK MD, maXStylus, maXTouch, MediaLB, megaAVR, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picPower, PICSTART, Pic32 logo, Prochip Designer, QTouch, RightTouch, SAM-BA, SpyNICE, SST, SST Logo, SuperFLASH, tinyAVR, UNI/O, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, EtherSynch, Hyper Speed Control, HyperLight Load, IntelliMOS, mTouch, Precision Edge, and Quiet-Wire are registered trademarks of Microchip Technology Incorporated in the U.S.A.


SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestiC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

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

© 2017, Microchip Technology Incorporated, All Rights Reserved.

Microchip received ISO/TS-16949:2009 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 =