INTRODUCTION

This application note describes a driver solution for a high-torque bipolar stepper motor. The feature-rich peripherals of Microchip’s PIC16F1776/9 allows the two H-Bridge switches to control different driving techniques for high- and low-power stepper motor, constant or high-torque microstepping, current limiting, motor step rate setting and motor Fault event detection.

The solution described in this application note has the following key features:

- Control Two H-Bridge Circuits with Shoot-Through Prevention
- Driving in Wave Step, Full-Step, Half-Step and Microstepping mode
- Capable of Constant or High-Torque Microstepping Drive
- Motor Current Limiting
- Microcontroller Implementation of Chopper Drive
- Chopper Drive Microstepping
- Motor Fault Detection
- Capable of Real-Time Motor Parameter Monitoring through Serial Communication (I²C, SPI, EUSART)

FIGURE 1: BLOCK DIAGRAM
Figure 1 shows the block diagram of a High-Torque/High-Power Bipolar Stepper Motor Driver based on the PIC16F1776/9 microcontroller. The motor driver utilizes different Core Independent Peripherals (CIP) in the microcontroller to perform complete stepper motor drive with minimum intervention from its CPU. These are the CIPs used in the design:

- Complementary Output Generator (COG)
- Hardware Limit Timer (HLT)
- High-Speed Comparator
- 16-bit Pulse-Width Modulation (16-bit PWM)
- Temperature Indicator (TempIND)
- Peripheral Pin Select (PPS)

The novelty of this solution is the way CIPs are combined with other on-chip peripherals such as I/O ports, Analog-to-Digital Converter (ADC), Fixed Voltage Reference (FVR), Digital-to-Analog Converter (DAC) and Timers.

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

CONTROLLING THE STEPPER MOTOR

A stepper motor distinguishes itself from other motors by its ability to move in a discrete number of angular increments or steps. It is a digital version of an electric motor that divides a full rotation into a number of equal steps and moves one step at a time. It can be easily configured to move some specific number of steps as well as to create a precise Stop command. Depending on the application’s speed and torque requirements, stepper motor usage varies from low up to high-power precision control applications. To know more about the basics and fundamentals of stepper motors refer to AN907, “Stepper Motors Fundamentals”.

Drive Circuit and Control Mechanism

To make the rotor spin, a rotating magnetic field must be generated. The two stepper motor windings (Winding A and B) are electrically energized to create a rotating field and a varying magnetic pole polarity on the stator (North and South) through the use of the H-Bridge circuit with the Complementary Output Generator (COG) as the control signal. Figure 2 illustrates the current flow through the H-bridge circuit. The current will flow from left to right in Winding A when MOSFETs Q1 and Q4 are turned ON while Q2 and Q3 are OFF. On the other hand, the current will flow from right to left when Q2 and Q3 are ON while Q1 and Q4 are OFF. The same principle applies with the MOSFETs Q5, Q6, Q7 and Q8 on Winding B. The turning ON and OFF of the MOSFETs is implemented through the use of the COG Forward and Reverse Full-Bridge mode output as a control signal. Refer to TB3119, “Complimentary Output Generator Technical Brief” for more information regarding the COG peripheral of PIC® microcontrollers.

FIGURE 2: COG AND H-BRIDGE CIRCUIT STEPPING ALGORITHM

<table>
<thead>
<tr>
<th></th>
<th>FORWARD MODE</th>
<th>REVERSE MODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1/Q5</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>Q2/Q6</td>
<td>OFF</td>
<td>MODULATED</td>
</tr>
<tr>
<td>Q3/Q7</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>Q4/Q8</td>
<td>MODULATED</td>
<td>OFF</td>
</tr>
</tbody>
</table>
Due to the constantly alternating flow of current, one inherent danger that should be monitored in using an H-Bridge circuit is the occurrence of a cross conduction. Cross conduction is a condition where the high- and low-side MOSFETs are both switched ON at the same time. This scenario will cause a shoot-through current, which could damage the driver’s components. Using the COG’s Counter registers, a dead-band delay can be imposed on the COG outputs. This provides non-overlapping output signals that will prevent shoot-through. The COG contains two 6-bit dead-band delay counters, one for the rising edge of the input source and the other for the falling edge of the input source. This dead-band delay is timed by counting COG clock periods from zero up to the specified value in the two COG Counter registers (COG1DBR and COG1DBF). Figure 3 illustrates the COG outputs with dead-band delay.

To continuously step the stepper motor to its desired position, the excitation on the stator windings must be sequenced in a specific sequence. Figure 4 illustrates the basic block diagram in driving the windings of a bipolar stepper motor. A Timer0 (TMR0) peripheral is used to provide the time interval between changes of steps throughout the excitation sequence. There are different stepping sequences or algorithms that can be implemented on a stepper motor. The choice of which algorithm to use depends primarily on the application’s requirements for motor operational speed, torque and step resolution. A detailed explanation of the different stepping algorithms will be given in the next Section “Step Mode Implementation”. The motor stepping rate and the rotation is implemented using the Complementary Output Generator (COG). The COG produces multiple output complementary signals, suitable to drive a Full-Bridge or H-Bridge circuit.
STEPPER MOTOR CHARACTERISTICS

To ensure the motor's optimal performance, different motor characteristics such as torque, speed, and stepping rate must be considered. This chapter discusses these characteristics and how they influence the final implementation.

Torque Generation

The first thing to understand in designing a stepper motor drive is how the torque is generated. A torque is developed when the magnetic fluxes of the rotor and stator are displaced relative to each other. When the winding is being energized with current, a magnetic flux is developed in the stator. Due to the high magnetic permeability of the material in the stator, the developed magnetic flux is confined and creates a strong flux concentration on the stator pole. This magnetic flux causes the rotor to be attracted and eventually rotate towards the energized stator. Motor torque can be generated by forcing the motor shaft out of its stable position. It can be done by either manually twisting the shaft or by electrically driving the motor to step to a new position, and the amount of torque produced depends on factors such as motor stepping rate, winding drive current and the drive design.

Stepping Rate

Motor stepping rate is the number of steps through which the shaft rotates during a specific time interval and is commonly referred to as PPS (Pulse per Second). It dictates the running speed of a motor. Example 1 shows the sample calculation for the motor stepping rate.

In a 1.8° step resolution motor, a rotational velocity of 120RPM can be generated by applying a 400PPS step rate using a Full-Step mode or an 800PPS step rate using a Half-Step mode. In the case of Microstepping mode the calculated step count should be divided by 1/4, 1/8 or 1/16 depending on the microstep resolution before calculating for the step rate in order to retain the same speed calculation as the full step. The calculated PPS will then be used by the TMR0 peripheral as a reference in producing the drive pulse for the motor winding. More detailed information on how the PPS is translated into the necessary driving pulse can be found on the Section “Step Mode Implementation”.

FIGURE 4: STEPPER MOTOR DRIVE CIRCUIT

![Stepper Motor Drive Circuit Diagram]
EXAMPLE 1: STEPPING RATE CALCULATION

It is also important to understand that the stepping rate affects the amount of motor torque generated. A higher stepping rate will result in a lower torque output. In solving motor torque limitation at higher speed, the current limiting technique can be implemented on the design. More detailed information about the current limiting technique implementation can be found in the Section “Current Limiting”.

STEP MODE IMPLEMENTATION

Step mode refers to the method or technique of stepping the motor each time the polarity of the current in the stator winding changes. It is simply a method of rotating a stepper motor. Depending on the required application, different step modes can be implemented to vary motor output resolution and torque. The following are the step modes that can be implemented on the PIC16F177X microcontroller:

- Wave Drive
- Full-Step Drive
- Half-Step Drive
- Microstepping Drive

Wave Drive

In wave drive mode, the stepper motor is driven by energizing only one winding at a time. Figure 5 shows the implementation of wave drive and its corresponding stepping algorithm.
FIGURE 5: WAVE DRIVE IMPLEMENTATION

In Figure 5, both Winding A and Winding B are connected to the H-Bridge drive circuit which is connected to the COG peripheral of a PIC MCU. Step 1 on the algorithm table applies a positive voltage or logic High to winding lead A while driving the winding lead A' low, and the current is generated in the direction as shown in Figure 5 creating a magnetic north and south on the stator poles accordingly and a rotation towards the next stator pole. On Step 2, the voltage applied in winding lead A is removed and put on the winding lead B. The winding lead B is driven high while winding lead B' is driven low, again creating a rotation towards the next stator pole. The process will continue up to Step 4 then repeating the cycle. Notice that the wave drive is done by turning ON one winding at a time. This is why it is also referred to as a One-Phase ON voltage sequence. The term wave is derived due to the generated voltage sequence that resembles a wave. Figure 6 shows the generated voltage drive sequence on the motor windings.

FIGURE 6: WAVE DRIVE WINDING VOLTAGE SEQUENCE

This stepping algorithm can be easily implemented through the use of a COG peripheral. As discussed in the Section "Drive Circuit and Control Mechanism", this peripheral is used to drive the MOSFETs, which in turn drive the corresponding motor windings. The COG mode is toggled from Forward to Reverse mode, as dictated by the stepping algorithm. The rate at which the mode is toggled depends mainly on the desired stepping rate (Refer to Section "Stepping Rate" for the calculation). The pre-calculated value of the stepping rate is loaded into the TMR0 register that interrupts the CPU at the appropriate time intervals to perform the subsequent steps. Figure 7 shows the step implementation of wave drive and Figure 8 shows its software implementation.
FIGURE 7: WAVE DRIVE CWG STEPPING ALGORITHM IMPLEMENTATION

Winding A

Q1
COG1A

Q2
COG1B

Q3
COG1C

Q4
COG1D

COG Drive

Forward OFF Reverse OFF Forward OFF Reverse OFF

Winding B

Q5
COG2A

Q6
COG2B

Q7
COG2C

Q8
COG2D

COG Drive

OFF Forward OFF Reverse OFF Forward OFF Reverse
One advantage of this method is that it has the simplest drive implementation, while the disadvantage is reduced torque output performance. Since the windings are energized one at a time, only half of the overall motor torque can be used. More degradation of torque performance can be observed with this method when using a unipolar motor construction because of its multiple winding constructions.

Full-Step Drive

On a full-step drive, two phases are always energized. Both Winding A and Winding B are energized simultaneously to rotate the motor. Figure 9 shows the implementation of full-step drive and its corresponding stepping algorithm.
In Figure 9, two windings are connected to a motor drive circuit which is connected to the COG peripheral of a PIC MCU. Step 1 of the algorithm table applies a positive voltage or logic High to both winding leads A and B while driving the winding leads A’ and B’ low. Notice that when both Windings A and B are energized, it creates the same polarity on adjacent poles which results in the rotor being equally attracted by both poles and lining up directly in or with the middle as opposed to being lined up with a specific stator pole as seen in the wave drive. Step 2 in the algorithm maintains the current flow direction in Winding B while reversing the current direction in Winding A. This causes the rotor to rotate 90 degrees and lie in between the next two stator poles. Continuing the algorithm up to Step 4 produces a complete motor rotation. Since it energizes both windings at the same time, a full-step drive is also referred as a Two-Phase ON voltage sequence. Figure 10 shows the voltage drive sequence on the motor windings. The voltage sequence of full-step drive clearly demonstrates that at any given time the current is flowing through both windings.

**FIGURE 10: FULL-STEP DRIVE VOLTAGE SEQUENCE**

Just like the wave drive, the full stepping algorithm can be easily implemented through the use of COG peripheral’s Forward and Reverse Full-Bridge mode. Figure 11 shows the step implementation of full-step drive and Figure 12 shows its software implementation.
FIGURE 11: FULL-STEP DRIVE STEPPING ALGORITHM IMPLEMENTATION

Winding A

Q1 COG1A
Q2 COG1B
Q3 COG1C
Q4 COG1D

Winding B

Q5 COG2A
Q6 COG2B
Q7 COG2C
Q8 COG2D

COG Drive

STEP 1 STEP 2 STEP 3 STEP 4 STEP 1 STEP 2 STEP 3
Forward Reverse Reverse Forward Forward Reverse Reverse Forward

Winding B

Forward Forward Reverse Reverse Forward Forward Reverse Reverse
Compared with the wave drive method, the advantage of this method is an increased output. Since two windings are energized at the same time, greater torque can be produced using the full step while maintaining the same amount of step angle or resolution.

Resonance on Wave and Full-Step Drive

**FIGURE 13: SINGLE STEP RESPONSE**

<table>
<thead>
<tr>
<th>Angle</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>90°</td>
<td>$t_1$</td>
</tr>
<tr>
<td>180°</td>
<td>$T_1$</td>
</tr>
<tr>
<td>Ringing</td>
<td>$t_2$</td>
</tr>
<tr>
<td>Ringing</td>
<td>$T_2$</td>
</tr>
</tbody>
</table>

**FULL STEP DRIVE TABLE**

- **Step_One:**
  - COG1CON0.bits.MD0 = 0 (Forward)
  - COG2CON0.bits.MD0 = 0 (Forward)
- **Step_Two:**
  - COG1CON0.bits.MD0 = 1 (Reverse)
  - COG2CON0.bits.MD0 = 0 (Forward)
- **Step_Three:**
  - COG1CON0.bits.MD0 = 1 (Reverse)
  - COG2CON0.bits.MD0 = 1 (Reverse)
- **Step_Four:**
  - COG1CON0.bits.MD0 = 0 (Forward)
  - COG2CON0.bits.MD0 = 1 (Reverse)
In Figure 13, the single-step response characteristic vs. time of a wave and full-step drive is illustrated. The step time "t1" is the time it takes the motor shaft to rotate one step angle once the first step pulse is applied. This step time is dependent on the ratio of torque and the applied load on the motor. Since the torque is a function of the displacement it follows what the acceleration will also be. Therefore, when moving in large step increments such as in Wave or Full-Drive mode a high torque is developed and consequently a high acceleration. This can cause overshoots and ringing on the motor. The settling time T1 is the time it takes these oscillations or ringing to cease. Subsequent steps such as t2 to T2 will also suffer from the same ringing and oscillation. In severe cases, this ringing could be so pronounced that the rotor will not have time to settle before the next step pulse is applied.

In certain applications where the motor is operated at a lower speed, the resonance phenomenon can be undesirable. To eliminate such behavior, other stepping algorithms such as half-step and microstepping drive can be implemented.

**Half-Step Drive**

Half-step drive is an algorithm that is derived by combining both wave and full-step drive techniques. It is a drive technique that increases the motor resolution by reducing the motor’s default stepping angle by half. For example, a 90° per step motor will have a new step angle of 45° when half-step drive is used. Since the rotor shaft travels less distance in a 45° step compared to the original 90° step, the ringing produced at each step is minimized, thereby reducing the resonance effects. Figure 14 shows the implementation of half-step drive and its corresponding stepping algorithm.

**FIGURE 14: HALF-STEP DRIVE IMPLEMENTATION**

<table>
<thead>
<tr>
<th>Step</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Winding A</td>
<td>A</td>
<td>A</td>
<td>0</td>
<td>A'</td>
<td>A'</td>
<td>0</td>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>Winding B</td>
<td>0</td>
<td>B</td>
<td>B</td>
<td>0</td>
<td>B'</td>
<td>B'</td>
<td>B'</td>
<td>B'</td>
</tr>
</tbody>
</table>
In Figure 14, no significant changes from wave and full-step drive can be observed in the motor connection and implementation. However, the step algorithm is now twice as long as in the wave or full-step drive. This makes sense considering that reducing the step angle by half will take twice as many steps to complete a 360 degree rotation. The first step used in the algorithm is actually the first step of the wave drive in which current flow occurs only in the Winding A and the rotor responds by aligning itself with the stator poles. Likewise, the second step used is the first step of the full-step drive in which the Winding A and Winding B are energized at the same time, resulting the rotor to position itself between stator poles. The drive algorithm of wave and full-step are alternately used in the Half-Step mode to generate the eight-step algorithm. Figure 15 shows the actual winding voltage sequence produced on the motor windings.

The same as the wave and Full-Step mode, the half-step algorithm can be implemented easily by using the COG peripheral. Figure 16 shows the step implementation of half-step drive and Figure 17 shows its software implementation.
When using the half-step algorithm there are some important additional considerations. Motor speed will be reduced to half the speed of the Full-Step mode. This means that the stepping rate should be doubled to retain the same amount of speed. Also, since half the time only one winding is energized, torque will be dramatically reduced in Half-Stepping mode.

**Microstepping**

Microstepping is a way of moving the stator flux of a stepper motor smoothly compared to the Wave, Full and Half-Step Drive modes. It offers an increase in overall system performance by further reducing the motor’s resonance problem in exchange for more processing power and a more complex control algorithm. The results are a significant reduction in motor vibration and stepping noises at lower speed while at the same time producing a much higher resolution and smaller step angles. Microstepping works on the principle of gradually increasing and decreasing the current in each winding. The currents in the windings are continuously varied to break up one full step into many small discrete steps. For example, using a basic Full-Step mode, one electrical cycle always consists of four full steps. Hence, one full step of any stepper motor with any value of step angle corresponds to 360/4 or 90 degrees of electrical angle. If the 90° electrical cycle is further subdivided into much smaller steps, normally in multiples of 8, 16, 32 or 64, it is referred to as a microstepping drive. This technique of subdividing steps is achieved by pulse-width modulating the voltage driving the motor windings instead of being turned on and off abruptly. There are many different microstepping modes available, with step lengths from ¼ full-step down to 1/64 full-step. This means that a stepper motor with 90° step angle will have a new step angle of 22.5° at ¼ microstepping.
and 1.4° at 1/64 microstepping. Aside from the different microstepping modes, microstepping can also be classified according to its torque output. It can be classified as either a high-torque or a constant-torque microstepping. The next section will explain the working principle of the different modes and types of microstepping and how to implement it in PIC microcontrollers.

**High-Torque Microstepping**

As indicated by the name, high-torque microstepping maximizes the torque production on the stepper motor. It creates a microstepping by alternately varying the current in the two windings of a stepper motor. A brief description of what is happening is that one winding is powered while the current in the other winding is gradually dropped to zero, reversed, and then ramped up again. This sequence is then repeated for the other winding. Implementation of high-torque microstepping can be easily understood by comparing the phase diagram and the corresponding torque diagram of a full-step drive and high-torque ¼ microstepping in Figure 18.

**FIGURE 18: PHASE DIAGRAM AND TORQUE CURVE RESPONSE**

![Phase Diagram and Torque Curve Response](image)

Figure 18 (A) and Figure 18 (B) represent the phase current diagram of a Full-Step Drive mode and high-torque ¼ microstepping, respectively. In a phase current diagram, phase currents are analyzed by plotting the current of Winding A (Ia) vs. the current of Winding B (Ib). Both stepping algorithms in the diagram are arranged in counter-clockwise rotation starting from the point on the upper right corner. On the full-step drive (A), the first step is achieved by completely turning ON both Winding A and Winding B as denoted by the coordinates of the point Step 1 which is (100% Ia, 100% Ib). On Step 2, it is achieved by completely turning on both Winding A' and Winding B. Note that the points lie on the Quadrant II of the phase diagram with coordinates of (-100% Ia, 100% Ib) which means that the opposite leads of Winding A (Winding A') need to be energized. Continuing the diagram to the 4th step, it can be noticed that the phase current diagram followed
the same driving algorithm as the full-step drive table discussed in the previous chapter. Now looking on the Figure 18 (B), the resolution from the previous is increased by adding more points or steps on the diagram. The coordinate of the added step points can be identified by creating a plot from the center point (0%IA, 0%IB) extending up to the square path line with a \( \Theta \) equal to the mode step angle. For example, the total step points for \( \frac{1}{4} \) microstepping is 16, this results to the \( \Theta \) or step angle of the \( \frac{1}{4} \) microstepping to be 22.5°. Extending an arm or ray going to the square path will lead to Step 1 coordinates of (-41% IA, 100% IB), Step 2 coordinates of (-100% IA, 100%) and the last Step 16 with coordinates of (0% IA, 100% IB).

The torque output on the diagram is denoted by the length or distance of the center from the step points. Therefore, the maximum torque possible can be achieved on Step 2, 6, 10 and 14, as shown in the torque response in Figure 18. This is true because the Winding A and B on the following steps are completely turned ON. It can also be noticed that the high-torque steps are located on the four corners of the diagram. By connecting all the points together, a square phase diagram can be seen. This is the reason why high-torque microstepping is also referred to as the square path microstepping.

Due to the varying production of torque per steps of the high-torque microstepping, this method tends to be a bit choppy and produces some vibrations. To overcome this problem, another form of microstepping is introduced which is the constant-torque microstepping.

**Constant-Torque Microstepping**

Constant-torque microstepping is used to produce a constant torque all throughout the stepping algorithm. It produces less torque compared to high-torque microstepping but produces a smoother, less noisy and less vibration stepping drive. It creates a microstepping by simultaneously varying the current in both windings of a stepper motor compared to just alternately varying the winding current in the high-torque microstepping.

The torque curve produced by both Winding A and Winding B of the constant current microstepping can be expressed mathematically by Equation 1 and Equation 2, respectively.

### EQUATION 1: WINDING TORQUE

\[
T_A = H \times \sin(\theta) \\
T_B = H \times \cos(\theta)
\]

Where:

- \( T_A \) = Winding A Torque
- \( T_B \) = Winding B Torque
- \( H \) = Motor Holding Torque
- \( \theta \) = Angle in electrical degrees from a full step position

To successfully implement a constant-torque microstepping, a technique referred to as sine-cosine microstepping is used to adjust the current in each winding so the net torque produced will be constant. In general, the torque produced by each winding is proportional with the current in that winding and the torques add linearly. Therefore, if we want to hold or step the motor at any angle \( \theta \) of the microstepping, it can be done by setting the currents through the motor windings to the values given in Equation 2.

### EQUATION 2: WINDING CURRENT

\[
I_A = I_{MAX} \times \sin(\theta) \\
I_B = I_{MAX} \times \cos(\theta)
\]

With the given Equation 2, the resultant stator current is the vector sum of the individual winding currents represented by Equation 3. This only shows that at any angle \( \theta \), the resultant current remains the same and equal to \( I_{MAX} \), producing a constant-torque output for the motor.

### EQUATION 3: RESULTANT STATOR CURRENT

\[
I_{MAX} = \left[ (I_{MAX} \times \sin^2(\theta))^2 + (I_{MAX} \times \cos^2(\theta))^2 \right]^{\frac{1}{2}}
\]

\[
= (I_{MAX} \times \sqrt{\sin^2(\theta) + \cos^2(\theta)})
\]

\[
= (I_{MAX} \times \Theta_{electrical\ degree})
\]

To further understand the implementation of the constant current microstepping, Figure 19 shows the phase diagram of both wave drive and the \( \frac{1}{4} \) constant-torque drive algorithm.
Figure 19 (A) and Figure 19 (B) represent the phase current diagram of a Wave Drive mode and constant-torque \( \frac{1}{4} \) microstepping, respectively. On the wave drive (A), the first step is achieved by driving the winding lead A high while winding lead A' is low and winding lead B/B' is turned off, as denoted by the coordinates of the point Step 1 which is (100% Ia, 0% Ib). On the other hand, Step 2 is achieved by driving the winding lead B high while winding lead B' is low and winding lead A/A' is turned OFF. Continuing the diagram until the 4th step, it can be noticed that the windings are energized the same as the Wave Drive mode discussed in the previous chapter. Now looking at Figure 19 (B), the resolution from the previous is increased by adding more points or steps to the diagram. The coordinates of the added step points can be identified by using Equation 4 and Equation 5 for motor current Ia and Ib, respectively. For example, assume that the IMAX value is one and the drive is in step number one. The sin of 360° multiplied by the present step number divided by 16 (\( \frac{1}{4} \) microstepping resolution) will result to .38. This means that the modulation for Winding A at Step 1 should be only 38% of the maximum current. When the calculation is continued up to Step 16 and the result is plotted on the phase diagram, it can be noticed that a circle path is produced by connecting all the step points. This is the reason why constant-torque microstepping is also referred to as the circle path microstepping.

**EQUATION 4: WINDING A CURRENT FORMULA FOR MICROSTEPPING**

\[ I_A = I_{MAX} \times \sin\left(\frac{\text{Step Number} \times 360}{\text{Step Resolution}}\right) \]
EQUATION 5: WINDING B CURRENT FORMULA FOR MICROSTEPPING

\[ I_B = I_{MAX} \times \cos\left(\frac{\text{Step Number} \times 360}{\text{Step Resolution}}\right) \]

Microstepping Implementation

The next question is how to drive calculated variable currents through the coil and how to implement it using PIC microcontrollers. There are different ways to achieve this, but the best way is:

1. Use the Core independent COG peripheral available on PIC microcontrollers to easily drive the H-Bridge circuit.

2. Apply the PWM peripheral as the input source for the COG to be used for the current modulation on the motor windings.

3. Create a timer interrupt that will be triggered for every step on the algorithm.

Figure 20 and Figure 21 show the COG drive signal for high- and constant-torque microstepping, respectively.
FIGURE 21: CONSTANT-TORQUE MICROSTEPPING COG DRIVE SIGNAL

Microstepping Performance

Figure 22 and Figure 23 shows the actual motor performance when the high-torque and constant-torque microstepping is applied.

FIGURE 22: HIGH-TORQUE ¼ MICROSTEPPING
FIGURE 23: CONSTANT-TORQUE ¼ MICROSTEPING

Stepper motors are often run at voltages higher than their rated voltage, specifically on applications that use a high-torque stepper motor. Although this is not necessarily the case for very small stepper motors, high-torque stepper motors need to run at higher voltages in order for the motor to reach its full potential. The higher the voltage applied, the greater the torque a motor can produce but the more it will affect the behavior of the current on the windings. Figure 24 shows the behavior of the current on a stepper motor.

CURRENT LIMITING

FIGURE 24: CURRENT WAVEFORM ON AN INDUCTIVE-REACTIVE CIRCUIT
A stepper motor can be modeled as an inductive reactive circuit. Due to its composition of several coils of copper wire, its two inherited physical properties are resistance and inductance. The resistance is the one responsible for some of the motor's power loss while the inductance makes the motor winding oppose current changes, and therefore limits high-speed operation. Referring to Figure 24, when a supply voltage is connected to the winding, the current rises according to Equation 6. Initially the current increases at a rate according to Equation 7. The rise rate will decrease as the current approaches its maximum level as denoted by Equation 8. On the other hand, when a supply voltage is disconnected at $t = t_1$, the current will start to decrease according to Equation 9.

**EQUATION 6:** INSTANTANEOUS CURRENT IN RL CIRCUIT

$$I(t) = \frac{V}{R} \cdot (1 - e^{-\frac{R}{L} \cdot t})$$

**EQUATION 7:** RL CIRCUIT CURRENT RISE RATE

$$\frac{\delta I}{\delta t}(t) = \frac{V}{L}$$

**EQUATION 8:** CURRENT RISE RATE AT MAXIMUM CURRENT LEVEL

$$I_{MAX} = \frac{V}{R}$$

**EQUATION 9:** INSTANTANEOUS CURRENT DECAY

$$I(t) = \left( \frac{V}{R} \right) e^{-\left(\frac{t-t_1}{\tau}\right) \cdot \frac{R}{L}}$$

This behavior of the current is acceptable on low speed and low stepping rate motor application. The problem comes when a higher motor speed operation is necessary. When a square wave voltage is applied to the winding, which is the case in Wave, Full and Half-Step Drive mode, the current will be decreased and smoothed. Figure 25 shows current waveform behavior at three different motor stepping rates.

**FIGURE 25:** CURRENT WAVEFORM AT DIFFERENT STEPPING FREQUENCY

Above a certain stepping rate frequency (B) (C) the total motor current never reaches its maximum value. As the torque of the motor is approximately proportional to the current, the maximum produced torque will be reduced as the stepping rate frequency increases. Also, when the stepping rate is further increased, the motor has now a tendency to be stalled and missed steps due to the lack of current that will drive the motor windings.

To overcome the effect of stepping rate and inductance to the motor current and also to gain a high-torque response at a higher stepping rate, one solution will be driving the motor at the maximum voltage possible.
Driving the motor with a higher voltage will force more current on the windings as well as increase the rise rate of the current, as denoted when increasing the value of voltage (V) in Equation 6. Although it is a good solution, it is important to take into consideration that exceeding the maximum specified voltage and current will result in a decrease in motor lifetime and damage the drive circuitry. A solution to avoid this scenario is to implement a chopper driver circuit. The chopper drive will be used to limit and control the current while feeding the motor with a higher voltage. The next section will discuss how a chopper drive can be implemented using the PIC® microcontroller peripherals.

**Chopper Drive**

The chopper driver provides an optimal solution for current control on motor winding. The basic idea behind the chopper control is to use a high-voltage source to bring the current in the winding of a stepping motor up to \( IMAX \) very quickly then when \( IMAX \) is reached, the voltage is chopped or switched off to maintain its rated voltage and current rating. **Figure 26** illustrates the implementation of the chopper drive using a PIC microcontroller.

**FIGURE 26: PIC MICROCONTROLLER IMPLEMENTATION OF CHOPPER DRIVE**

Chopper drive is done by sensing the peak current on the motor windings via a shunt resistor (RSHUNT) connected in series with the motor. As the current increases, a voltage develops across the RSHUNT, which is used as an input to the comparator. When the predetermined reference level defined by the FVR and DAC voltage output (VDAC_Reference) is reached, the comparator triggers an Auto-shutdown command on the COG peripheral. As the COG completely turns off, the current will then decay until the RSHUNT voltage is below the VDAC_Reference and then triggers an Auto-restart command on the COG, which turns back ON the COG peripheral.

The comparator used is provided with hysteresis so that the voltage is not reapplied until the VRSHUNT = VDAC_Reference – VCHYSTERESIS. Hysteresis is necessary to prevent limiting the frequency with which the comparator and MOSFET chop the supply voltage. **Figure 27** and **Figure 28** show the drive waveform and the actual resulting waveform in a chopper driver implementation, respectively.
FIGURE 27: CHOPPER DRIVE CURRENT WAVEFORM

FIGURE 28: CHOPPER DRIVE CURRENT AND VOLTAGE RESPONSE

(A). Motor Voltage and Current Response at 100RPM
(Without the Chopper Drive Implementation)

(B). Motor Voltage and Current Response at 150RPM
(Without the Chopper Drive Implementation)

(C). Motor Voltage and Current Response at 150RPM
(With Chopper Drive Implementation)
Chopper Drive Microstepping

Aside from its current limiting function, the chopper drive can also be used to implement a microstepping technique on a stepper motor. From the previous discussion, microstepping varies the current sinusoidally between 0 and IMAX rather than bringing the current up to IMAX as quickly as possible, resulting in a reduction of motor resonance and generating higher resolution steps. By combining the characteristics of microstepping drive and high-voltage chopper drive, a motor driver that allows a high-power/torque motor to operate at the highest speed possible while gaining an increase in the overall motor performance can be created.

Figure 29 shows the implementation circuit for chopper drive microstepping. In chopper drive microstepping, the winding current is being monitored and controlled instead of the winding voltage. The winding current is directly proportional and in phase with the produced torque while the winding voltage is out of phase with the produced torque. As a result, controlling the current provides the best performance in driving the motor. Chopper drive microstepping can be implemented by creating a dynamic VDAC_Reference voltage set point that resembles a digitized sine and cosine waveform. The available high resolution DACs on PIC16F1776/9 are used as a reference for Comparator 1 and Comparator 2. Using a digitized sine wave reference on the RSHUNT1 and a digitized cosine wave reference on the RSHUNT2 allows the driver to produce a voltage level that shows that the motor is microstepped. Refer to Figure 30 for the resulting waveform for the chopper drive microstepping.

**FIGURE 29: CHOPPER DRIVE MICROSTEPPING IMPLEMENTATION**

**FIGURE 30: CHOPPER DRIVE MICROSTEPPING WAVEFORM**
Motor Fault Detection Feature

In order to avoid system failure or motor driver performance degradation, appropriate early Fault detection strategies are implemented in this application.

Over-Current Detection

A good reason to run a stepper motor at a higher supply voltage is to push the maximum rated current through the motor windings. Running at a higher voltage leads to a faster current rise time that leads to a high-torque response at a higher speed when added with a chopper drive. Although stepper motor can be driven at a higher voltage level, conditions such as winding isolation breakdown and overheating can still be a risk when voltage and current goes too high. To avoid excessive winding current and to limit the current through the windings, over-current detection can be implemented.

To implement over-current detection, a RSHUNT is added to the drive circuitry, giving a voltage corresponding to the current flowing in the motor winding. The voltage drop across this resistor varies linearly with respect to the motor current. The voltage is fed to the inverting input of the Comparator and compared with a certain reference voltage. This reference voltage is based on the product of RSHUNT resistance and the maximum allowable stall current of the motor. The reference voltage can be provided by the FVR which can be narrowed down further by the DAC. In this manner, very small reference voltage can be used, allowing the RSHUNT resistance to be kept low. Keeping the resistance low reduces the RSHUNT power dissipation. If the RSHUNT voltage exceeds the reference, the comparator output will trigger the Auto shutdown feature of the COG.

Ambient Temperature Detection

Ambient Temperature can be detected using the device on-chip temperature indicator peripheral present on the PIC16F177X family. The indicator measures device temperature, corresponding to the temperature in its environment with some delay. It allows the drive system to lower the current limit based on the ambient temperature, letting the motor operate at a desirable temperature.

The indicator is used to measure the device temperature between -40°C and +85°C. The internal circuit of the temperature indicator produces a variable voltage relative to temperature using internal transistor junction threshold voltage. This voltage is converted to digital form by the Analog-to-Digital Converter (ADC). The ADC result will be used to determine the actual temperature reading defined by Equation 10. For a more accurate temperature indicator reading, a single-point calibration is implemented. Refer to Application Note AN1333, “Use and Calibration of the Internal Temperature Indicator” for more details regarding the calibration process.

EQUATION 10: TEMPERATURE READING CALCULATION

\[
\text{Temperature Reading} = \frac{0.659 \left[ \frac{V_{DD, \text{mode}}}{l - \frac{ADC_{RESULT}}{2^N - 1}} \right]}{0.00132} - 40
\]

Where:
- High-Range mode = 4
- Low-Range mode = 2
- \(N\) = number of bits of ADC Resolution
- \(ADC_{RESULT}\) = ADRES Register Value

The implementation of the overtemperature detection uses the ADC internal channel input selection (CHS) bit. The temperature indicator module is used as the channel input for the ADC. For every Timer interrupt, the completed ADC conversion result will be compared with the desired maximum temperature limit. When the ADC result exceeds the maximum temperature limit, the output of the COG is disabled.
Motor Stall Detection

Under normal conditions, when the motor is spinning, it gives periodic Back-EMF signals on both windings. In the case of a stalled motor, there is a little to no BEMF signal produced. Hence, by monitoring when the motor stops producing these signals, the motor stall condition can also be detected. To implement motor stall detection, Level-Triggered Hardware Limit Timer (HLT) mode in Timer2/4/6 peripheral is used. For more details regarding the HLT, refer to TB3122, “Hardware Limit Timer on PIC® Microcontrollers”. Its main function is to monitor any changes in the periodic Back EMF signal. Refer to Figure 31 for HLT implementation on motor control design.

**FIGURE 31: HLT STALL DETECTION**

In this application, the outputs of Comparator 1 and Comparator 2, whose inputs are connected to motor Winding A+ and Winding B+, respectively, are used as the external signal source for the Timer2/4 HLT. The motor produces Back EMF signals ranging from 50 kHz (20μS) to 400Hz (2.5mS) depending on the motor speed. The period (PR) register of the Timer2/4 is then set to a value that is sufficiently larger than the minimum input frequency (400Hz). By doing this, the motor stall condition can be detected at a much wider speed range. Refer to Equation 11 for the PR value calculation.

**EQUATION 11: PR2 CALCULATION**

\[
PR > \frac{TMR2/4 \text{ Clock Source}}{\text{Input Signal (MIN)} \times \text{Prescaler} \times \text{Postscaler}}
\]

\[
PR > \frac{LFINTOSC}{400Hz} > \frac{31000}{400 \times 1 \times 8} > 9
\]

The reason for a larger PR value is for the input signal to occur first and reset the Timer2/4 count before the PR period match occurs. In the case of a rotating motor, Back EMF pulses are always present to continuously reset the Timer2/4 count, preventing the period match from occurring. Otherwise, in a case of stalled motor, Back EMF pulses are not present to reset the Timer2/4 count, hence the timer continues to increment until the period match occurs. An interrupt event is triggered every time period match occurs. This can be used to shut down the COG output and indicate that the motor is not spinning or in a stall condition. Refer to Figure 32 for the implementation of HLT mode as stall detection.
FIGURE 32: HLT MODE IMPLEMENTATION

Motor Normal Running Condition

BEMF A/B C1/C2OUT

PR Register Value

Timer Clock

Timer Count

Timer2/4 Interrupt

Motor Stall Condition

BEMF A/B C1/C2OUT

PR Value

Timer Clock

Timer Count

Timer Interrupt

FIGURE 33: HLT STALL DETECTION
CONCLUSION

In a stepper motor application where precise measurement, current control, fault detection and high-torque output is needed, an efficient and flexible microcontroller that can accommodate all these features can provide an advantage and significant impact. This application note describes how the PIC16F1776/9 microcontroller meets these requirements.

The microcontroller is able to drive all the different types of stepping motors. Full-step, wave, and half-step drive techniques are well within the capability of the microcontroller. The step resolution, accuracy, and motor resonance reduction can also be improved through microstepping and current limiting techniques. The available Core Independent Peripherals such as COG, 16-bit PWM, 10-bit DAC and high-speed comparator allow for the implementation of more advanced stepper motor control techniques, such as chopper drive microstepping. For added application flexibility, an option for constant-torque or high-torque microstepping can also be implemented. Also, different motor fault detection methods can be employed to ensure a safe and proper drive control. In addition, the use of CIP’s to accomplish these features will give the user a lot of spare capacity in the microcontroller to implement any product-specific features.
APPENDIX A: CIRCUIT SCHEMATIC

FIGURE A-1: HIGH-TORQUE STEPPER MOTOR DRIVER USING PIC16F1776/9

[Diagram of the circuit schematic with labeled components and connections.]
APPENDIX B: CIP PERFORMANCE EVALUATION

TABLE B-1: COG IN DRIVING H-BRIDGE CIRCUIT

<table>
<thead>
<tr>
<th>Operation</th>
<th>H-Bridge Drive Using COG</th>
<th>H-Bridge Drive Using Conventional Method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Availability</td>
<td>Execution Time</td>
</tr>
<tr>
<td>Shoot-Through Current Protection</td>
<td>Yes</td>
<td>1 instruction cycle or FOSC/4</td>
</tr>
<tr>
<td>Phase Delay</td>
<td>Yes</td>
<td>1 instruction cycle or FOSC/4</td>
</tr>
<tr>
<td>Blanking Delay</td>
<td>Yes</td>
<td>1 instruction cycle or FOSC/4</td>
</tr>
</tbody>
</table>

Input Sources Available: CCP, Comparator, PWM, CLC, MD, Input Pin

No. of Implementing Register: 20 Register

Auto-Shutdown/Restart Sources: Timer2/4/6/8, CLC, Comparator, Input Pin

TABLE B-2: RESOURCE COMPARISON

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Proposed Stepper Motor Driver Solution</th>
<th>Conventional Stepper Motor Driver Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Microstepping mode</td>
<td>Microstepping mode</td>
</tr>
<tr>
<td>Flash Memory (Words)</td>
<td>359</td>
<td>523</td>
</tr>
<tr>
<td>RAM (Bytes)</td>
<td>15</td>
<td>19</td>
</tr>
<tr>
<td>Peripherals Used</td>
<td>Comparator, DAC, FVR, COG, PWM, Timer</td>
<td>ADC, ECCP, Timer, PWM</td>
</tr>
</tbody>
</table>
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, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, Helda, JukeBloks, KEELOO logo, KEELOO logo, Kleer, LANCheck, LINK MD, maXSylus, maXTouch, Medialink, megaAVR, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, Prochip, Designer, OTouch, RightTouch, SAM-BA, SpyNIC, SST, SST Logo, SuperFlash, tinyAVR, UNI/O, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

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

© 2017 Microchip Technology Inc.
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:
http://www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Austin, TX
Tel: 512-257-3370

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453
Tel: 317-536-2380

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608
Tel: 951-273-7800

Raleigh, NC
Tel: 919-844-7510

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110
Tel: 408-436-4270

Canada - Toronto
Tel: 905-695-1980
Fax: 905-695-2078

ASIA/PACIFIC
Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon

Hong Kong
Tel: 852-2943-5100
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Chongqing
Tel: 86-23-8960-9588
Fax: 86-23-8980-9500

China - Dongguan
Tel: 86-769-8702-9880

China - Guangzhou
Tel: 86-20-8755-8029

China - Hangzhou
Tel: 86-571-8792-8115
Fax: 86-571-8792-8116

China - Hong Kong SAR
Tel: 852-2943-5100
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-3326-8000
Fax: 86-21-3326-8021

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8664-2200
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

ASIA/PACIFIC
China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-3019-1500

Japan - Osaka
Tel: 81-6-6152-7160
Fax: 81-6-6152-9310

Japan - Tokyo
Tel: 81-3-6880-3770
Fax: 81-3-6880-3771

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or 82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-5778-366
Fax: 886-3-5770-955

Taiwan - Kaohsiung
Tel: 886-7-213-7830

Taiwan - Taipei
Tel: 886-2-2508-8600
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

ASIA/PACIFIC

EUROPE
Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

Finland - Espoo
Tel: 358-9-4520-820

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

France - Saint Cloud
Tel: 33-1-30-60-70-00

Germany - Garching
Tel: 49-8931-9700

Germany - Haan
Tel: 49-2129-3766400

Germany - Heilbronn
Tel: 49-7131-67-3636

Germany - Karlsruhe
Tel: 49-721-625370

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Germany - Rosenheim
Tel: 49-8031-354-560

Israel - Ra'anana
Tel: 972-9-774-7705

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Italy - Padova
Tel: 39-049-7625286

Netherlands - Drenthe
Tel: 31-416-690399
Fax: 31-416-690340

Norway - Trondheim
Tel: 47-7289-7561

Poland - Warsaw
Tel: 48-22-3325737

Romania - Bucharest
Tel: 40-21-407-87-50

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Gothenburg
Tel: 46-31-704-60-40

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820

Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Gothenburg
Tel: 46-31-704-60-40

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820