AN1178

Intelligent Fan Control

Author: *Justin Milks*
Microchip Technology Inc.

INTRODUCTION

This application note describes the creation of an intelligent 4-wire fan. This design incorporates a PIC® microcontroller directly inside of the fan, enabling the fan to provide closed loop speed control, speed feedback, and additional safety features.

Topics covered will include:

- Brushless DC fan basics
- Necessary microcontroller peripherals
- Hardware, and software control techniques.

This application note is targeted toward the PIC16F616 and PIC12F615, however it can be adapted to many PIC® microcontrollers.

BRUSHLESS DC THEORY

Since brushless DC fans are variants of basic brushless DC motors, all of the same brushless DC motor basics apply. There are several Microchip application notes that cover brushless DC basics, and a listing of these application notes can be found in the references section of this document.

In a brushless fan, the armature is stationary and it is the permanent magnet that rotates, as shown in Figure 1. This magnet is in the shape of a ring with fan blades attached. Many fans use 4 magnetic poles in the ring magnet; however, the number of magnetic poles can be increased to create a more powerful fan.

Finally, the pattern in which the armature is winded may vary. A two-phase fan will have two separate windings which can be individually energized. These windings are always energized with current flowing in the same direction. Another option is a single-phase system in which a single winding is constantly energized, but the direction of the current is reversed.

The two-phase method is lower cost, requiring one MOSFET device per phase. However, since this method only utilizes half of the armature windings at any given time, it is not suitable for higher power applications. Since the single phase method energizes every winding on the armature simultaneously, it is better suited for higher power fan applications. However, this method requires a total of 4 MOSFET devices.

The fan determines which phase to energize (or which way to energize the single phase) by using a Hall sensor. These sensors are able to detect the magnetic poles of the ring magnet and provide the necessary information to determine how to energize the armature. These sensors will be discussed in more detail in the following sections.

FIGURE 1: FAN DIAGRAM

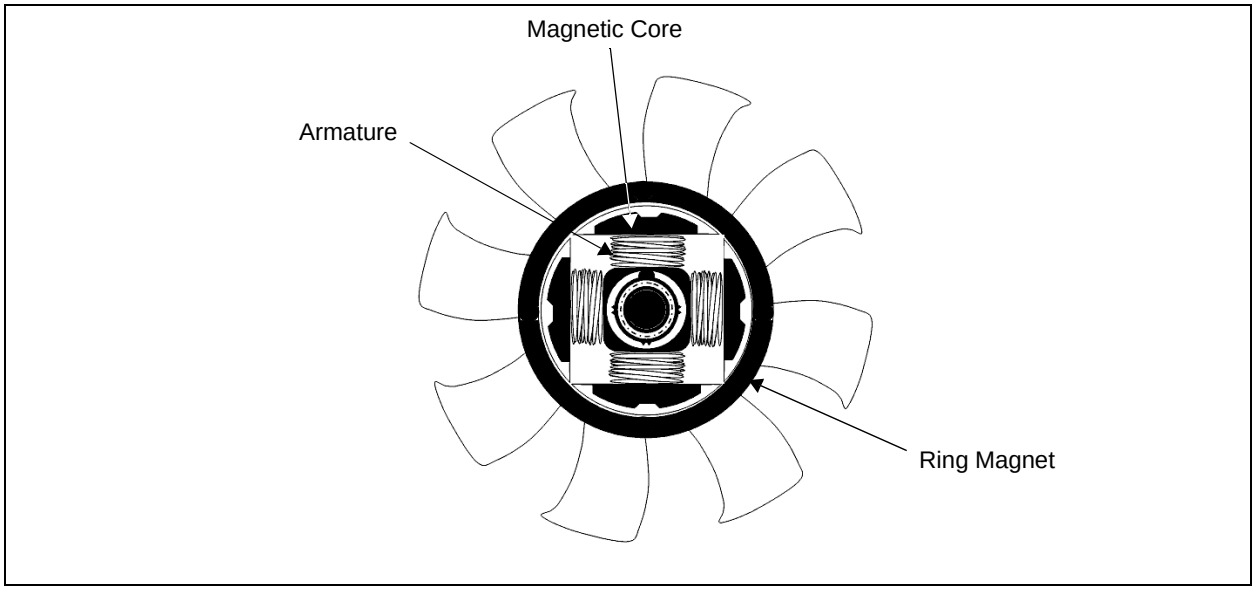


FIGURE 2: FAN CONTROL BLOCK DIAGRAM

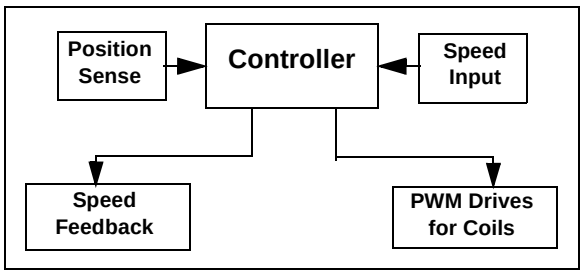


Figure 2 shows a generic fan controller block diagram.

Software and hardware techniques relating to each block in the diagram will be discussed in further detail in the following sections.

POSITION SENSE

Two types of position sensing devices are: Hall elements and Hall effect sensors. Either can be used.

Hall Elements vs. Hall Effect Sensors

The term Hall element is used to describe a device which produces a differential analog voltage corresponding to the strength of the magnetic field.

The output of a Hall element is a relatively small, differential analog voltage. Figure 3 shows the output produced by a Hall element in the presence of the rotating ring magnet.

The main advantage to using a Hall element is low cost. However, for optimum performance, the Hall element requires a constant current power source. Furthermore, the low output voltage levels may require amplification.

A Hall effect sensor removes these disadvantages by incorporating a Hall element, a power source, and an amplifier in a single package.

Hall sensors come in many varieties, including open-collector and TTL level outputs. Many brushless fans utilize latching Hall sensors, in which a north pole will latch the device in one state until a south pole is present and the device is latched in the opposite state.

Figure 3 shows the output of a Hall sensor produced by rotating fan blades.

Interfacing Hall Effect Sensors

Since Hall effect sensors are digital devices, their interface to a microcontroller is simple task. Any available interrupt-on-change (IOC) pin can be used. If an open-collector Hall effect sensor is used, then the internal weak pull-ups may be utilized to reduce the necessary component count.

It is important to ensure that the low-level output from the Hall effect sensor is within the acceptable range for the input pin. If the low-level output is not within the acceptable range, it is possible to use an analog comparator to read the Hall effect sensor.

Interfacing Hall Elements

Interfacing a Hall element to most microcontrollers is not as straightforward as the Hall effect sensor. As mentioned earlier, the Hall element requires a current source. A low-cost solution is to simply use a resistor in series with the Hall element. The disadvantage is lower output levels.

In order to read the low level, differential analog output it is necessary to use both inputs of an analog comparator, as shown in Figure 4. Recalling the output of the Hall element in Figure 3, the typical comparator will suffer problems when the difference between the two outputs is less than comparator input offset voltage.

A microcontroller with a comparator featuring internal hysteresis, allows a Hall element to be used without compromise. The PIC12F615 and PIC16F616 are two microcontrollers that have these features.

FIGURE 3: HALL DEVICE OUTPUTS

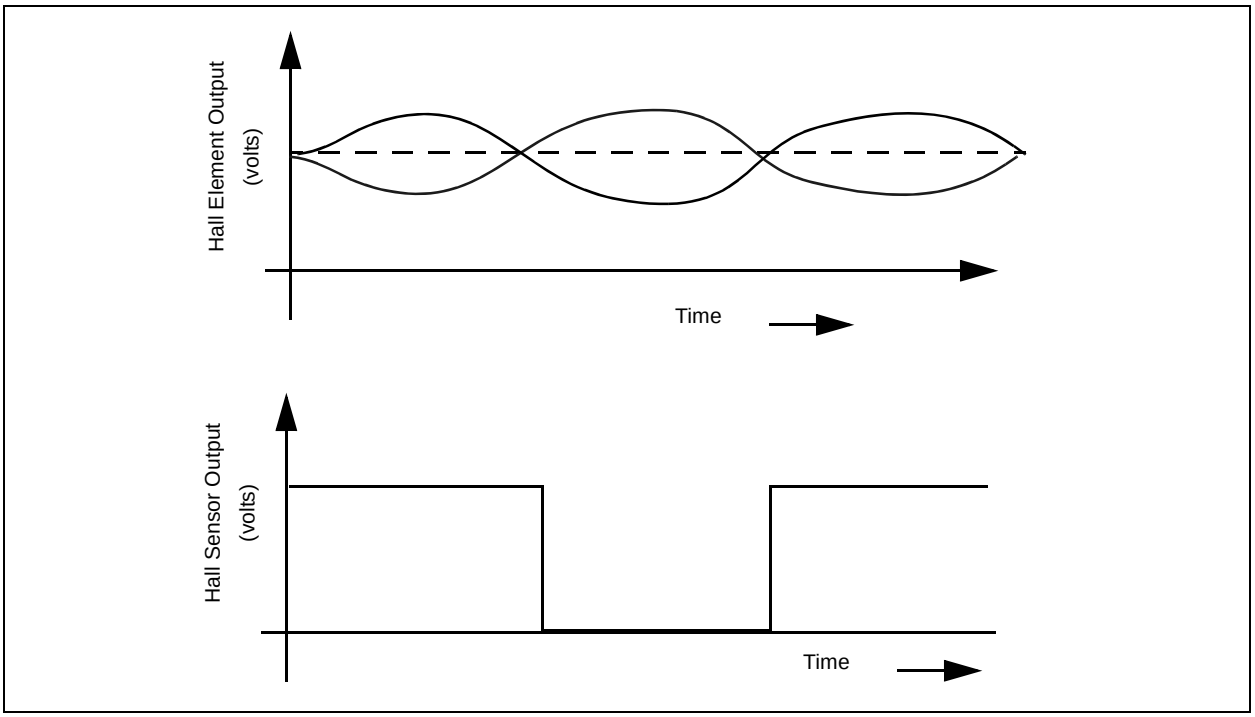
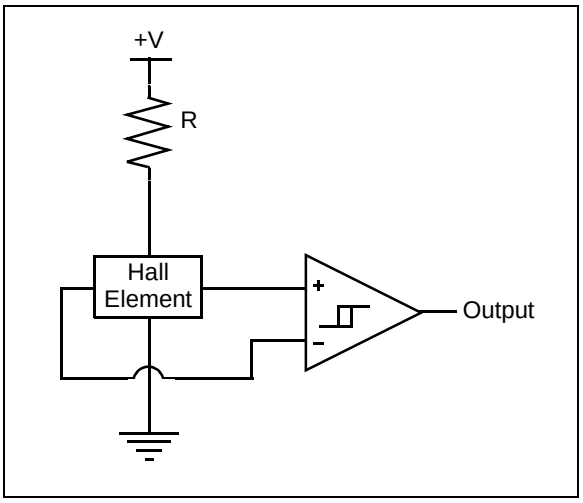


FIGURE 4: HALL SENSOR DIAGRAM



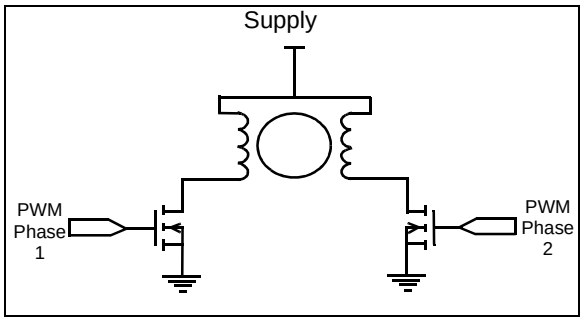
PWM DRIVE FOR FAN WINDINGS

This section will describe several options for generating the necessary PWM drive depending on the available microcontroller peripherals. Methods for increasing PWM resolution and PWM frequency considerations will also be discussed.

Connecting the PWM Outputs

In a two phase fan application, the PWM outputs will be connected to the MOSFET devices as shown in Figure 5.

FIGURE 5: TWO PHASE SCHEMATIC



In this configuration, only one MOSFET device will be active at any moment in time. This will ensure that current is only flowing through one winding at a time. It is important to note that the current always flows in the same direction through the winding.

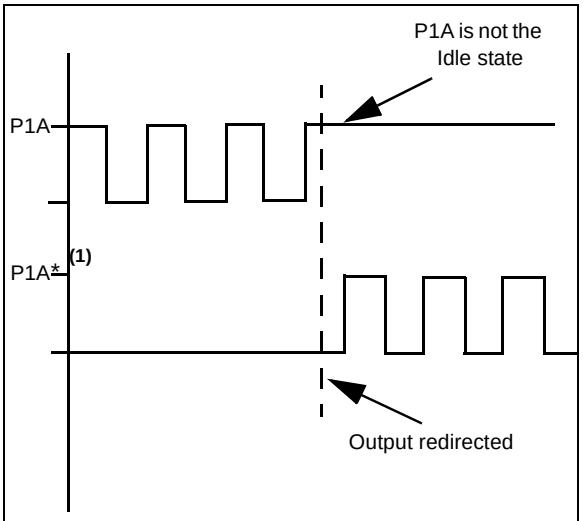
Pin Steering to Generate Multiple Outputs

Select PIC microcontrollers, such as the PIC12F615, have the option of PWM pin steering. This allows the Capture Compare PWM (CCP) module to generate a PWM that can be directed to one or more I/O pins.

This configuration is utilized for fan control in the following manner: with each commutation (determined by the Hall element or Hall effect sensor) the PWM output is re-directed to the appropriate pin.

There are some considerations that need to be taken into account. Consider the condition in Figure 6 below. When the PWM is redirected, the output may remain in its previous state, and if care is not taken, both outputs may be active simultaneously. This will not only cause excess current draw, but it will also affect the speed of the fan blades.

FIGURE 6: PWM USING PIN STEERING



Note 1: P1A* is the alternate PWM output pin for P1A.

To account for this, the following steps can be taken when steering the PWM:

1. Disable the CCP module, removing its control of the outputs
2. Set both PWM outputs to the Idle state again
3. Perform the PWM steering direction change
4. Re-enable the CCP module

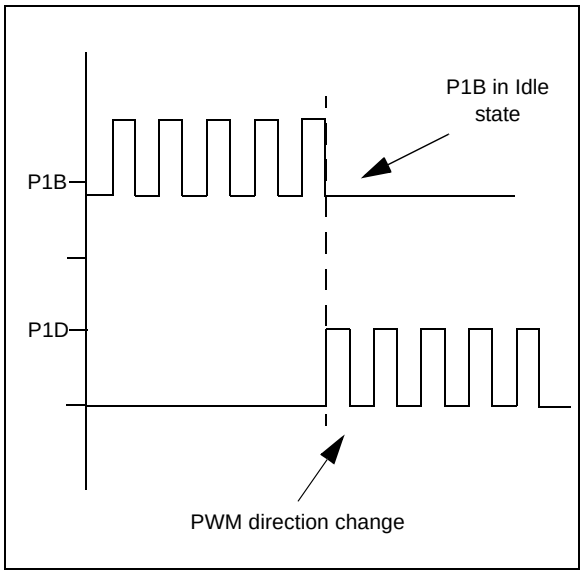
This sequence of steps will ensure that both CCP outputs will not be active simultaneously.

Using the ECCP to Generate Multiple Outputs

Other PIC microcontrollers, such as the PIC16F616, do not have the pin steering capability. However these microcontrollers can still be used in the fan control application by utilizing the different modes of the ECCP modules to generate an appropriate PWM output.

In Full-Bridge mode forward direction, the ECCP module provides a PWM output on P1D while keeping pin P1B in the Idle state, as shown in Figure 7 below. In Full-Bridge mode reverse direction, the ECCP module provides a PWM output on P1B while keeping P1D in the Idle state.

FIGURE 7: PWM USING ECCP



With each commutation it is necessary to change the direction of the ECCP module. If we use the ECCP module in this method, it will ensure that both outputs will never be active simultaneously.

Choosing a PWM frequency

One important constraint in choosing the PWM frequency for the fan windings is to ensure the frequency is above the audible 20 kHz. Given this constraint there are still multiple ways of deciding on a frequency.

The frequency may be chosen such that the resulting PWM resolution is an integer number of bits. For example, with an 8 MHz oscillator, a PWM resolution of 31.25 kHz will provide exactly 8 bits of resolution. Having an integer number of bits of resolution will simplify math routines. In this example, however, the higher switching frequency may cause thermal problems with the fan MOSFETs. In this case it may be better to use a lower frequency.

It is also possible to choose the PWM frequency based on other fan characteristics, such as fan winding characteristics, or to minimize switching losses. If in keeping the PWM resolution at an integer number of bits of resolution is not a constraint. Choosing the lowest frequency possible will minimize switching losses while simultaneously maximizing PWM resolution.

EQUATION 1: PWM PERIOD

$$PWM\ Period = [(PR2) + 1] \bullet 4 \bullet T_{OSC} \bullet (TMR2\ Prescale\ Value)$$

EQUATION 2: PWM RESOLUTION

$$Resolution = \frac{log[4(PR2 + 1)]}{log(2)}\ bits$$

The frequency choice can affect the PWM resolution so some frequencies may not yield an integer number of bits of PWM resolution. For example, a 20 kHz PWM frequency with an 8 MHz oscillator yields a PR2 value of 99 and a PWM resolution of 8.6438 bits (numbers calculated using the equations in the CCP chapter of the data sheet which have been reproduced here as Equation 1 and Equation 2). This means that the possible duty cycle values range from 0 to 399. Depending on the output of the control loop software, the duty cycle may have to be scaled up or down.

Scaling up from a value with less resolution to a value with more resolution is undesirable. Doing this will cause the duty cycle steps to be unequal, and this method does not take full advantage of the available PWM resolution. Therefore a better choice is to scale from a value with more resolution to a value with less resolution.

In this case a range of 0 to 511 (9 bits) is scaled to produce a range of 0 to 399 (8.6438 bits). The scaling is performed by taking the 9-bit control loop output, multiplying by an 8-bit scaling factor, and discarding the low byte of the result. The formula for determining the scaling factor is shown in Equation 3 below.

EQUATION 3: DETERMINING THE SCALING FACTOR

$$Scaling\ Factor = \frac{(maxoutput) \bullet 256}{(maxinput)}$$

In determining the scaling factor, maxoutput is the maximum desired output of the scaling routine and maxinput is the maximum value for the input to the scaling routine.

When scaling from 9 bits to 8.6438 bits the scaling factor will be 200 (decimal). Table 1 shows the input and output of the scaling routine. As you can see, the input value of 511, a full-scale 9-bit input, corresponds to the maximum duty cycle of 399.

TABLE 1: RESULTS OF SCALING ROUTINE

9 bit value (0 - 511)	Multiply by 200	High bytes of result
127	25400	99
255	51000	199
383	76600	299
511	102200	399

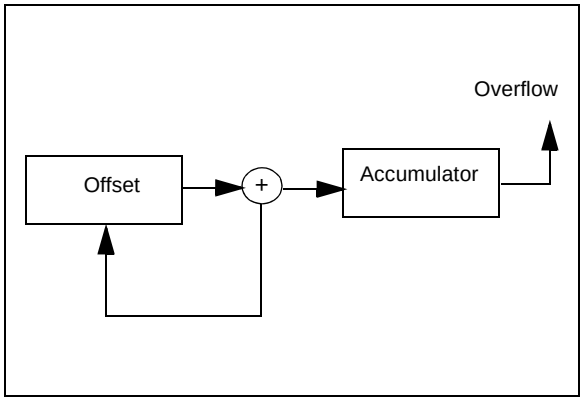
Increasing PWM Resolution

Depending on the particular fan characteristics, the given PWM resolution may not be adequate. One way to increase the PWM resolution is to lower the frequency, but this may not be possible. It is possible, however, to use software techniques to increase the effective PWM resolution.

A technique called PWM dithering can be used. PWM dithering is accomplished by changing between two different duty cycles. For example, by continuously switching between a 55% and 56% duty cycle, the average duty cycle will be 55.5%. It is important that the system connected to the PWM output behave in a low-pass manner. The system (in this case the fan) needs to respond to the average duty cycle and not react quickly enough to show the discrete changes.

The block diagram shown in Figure 8 illustrates the software implementation of the PWM dithering routine.

FIGURE 8: PWM DITHERING BLOCK DIAGRAM



One register is used as an accumulator, and another as the offset. Every n PWM periods, the offset is added to the accumulator. If the accumulator overflows, the PWM period is increased by one for the next n PWM periods. For example, with an 8-bit accumulator and the offset loaded with 128, the accumulator would overflow every other cycle. If the offset was loaded with 1, then the accumulator would overflow 1 out of every 256 cycles, and so on.

The downside to this type of software technique is the required processing time. Every n PWM cycles, the microcontroller must execute the software necessary to determine which PWM duty cycle to use. Furthermore, care must be taken at the boundaries. For example, if the PWM duty cycle is set to 100%, then the dithering can not take place as there is no higher duty cycle. Also, when the duty cycle is set to zero it is important that the dithering routine does not attempt to increase the duty cycle.

MEASURING THE SPEED INPUT

Several methods exist for providing the controller with the desired speed. Two of the most common are using a PWM input or using an analog voltage. Both will be discussed below:

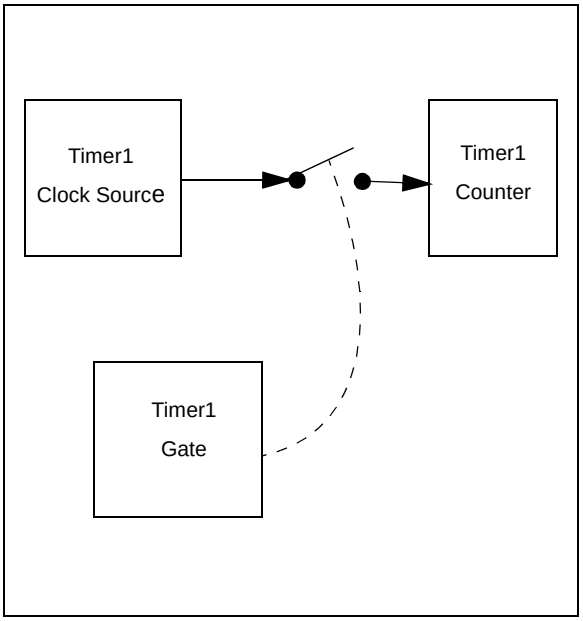
Using a PWM

A PWM input into the fan is a common way of commanding the speed. The duty cycle of PWM input determines the fan operating speed. For example, a duty cycle of 50% commands the fan to 50% of its maximum speed and so on.

An input PWM frequency of 25 kHz is often used (this is done to comply with several existing intelligent fan control specifications). By utilizing several microcontroller peripherals this PWM can be directly measured digitally.

Many PIC microcontrollers have a Timer1 gate feature. As shown in Figure 9 below, the Timer1 gate allows the Timer1 clock source to be disconnected from the counter. The source for the Timer1 gate can be a digital input, or the output of an analog comparator.

FIGURE 9: TIMER1 GATE DIAGRAM



In this application, the Timer1 will be configured to use the T1G pin as its gate source. To perform the measurement, the following steps are taken:

1. The Timer1 gate is configured to allow the timer to increment when the incoming signal is high. Timer0 is cleared and used to time the measurement period.
2. The Timer0 interrupt flag is set, signaling the end of the measurement. The value in Timer1 is stored, and the Timer1 gate is reconfigured to allow Timer1 to increment when the incoming signal is low.
3. The Timer0 interrupt flag is set, signaling the end of the measurement. The value in Timer1 is stored and processed to determine the duty cycle.

This process is illustrated in Figure 10. The output of this portion of the measurement process will be two values, referred to as T_{HIGH} and T_{LOW} which correspond to the high time and low time of the incoming signal during one measurement period.

Based on these two values, Equation 4 can be used to determine the duty cycle.

EQUATION 4: CALCULATING THE DUTY CYCLE

$$DutyCycle = \frac{T_{HIGH}}{T_{HIGH} + T_{LOW}} \bullet 255$$

In Equation 4, T_{HIGH} + T_{LOW} represents the measurement period determined by Timer0. The measurement could be simplified by using a constant for T_{HIGH} + T_{LOW}. However, in doing so, any oscillator tolerances would be reflected in the measurement. By measuring both the high and low time the oscillator tolerance can be removed.

The factor of 255 is present in Equation 4 as a scaling factor. This is done to scale a duty cycle of 100% to 255. This effectively produces an 8-bit value from 0 to 255 that corresponds to the duty cycle of the input.

It is important to ensure that the measurement period is sufficiently longer than the period of the signal being measured. A measurement period 100 times larger than the period of the signal to be measured is a starting point.

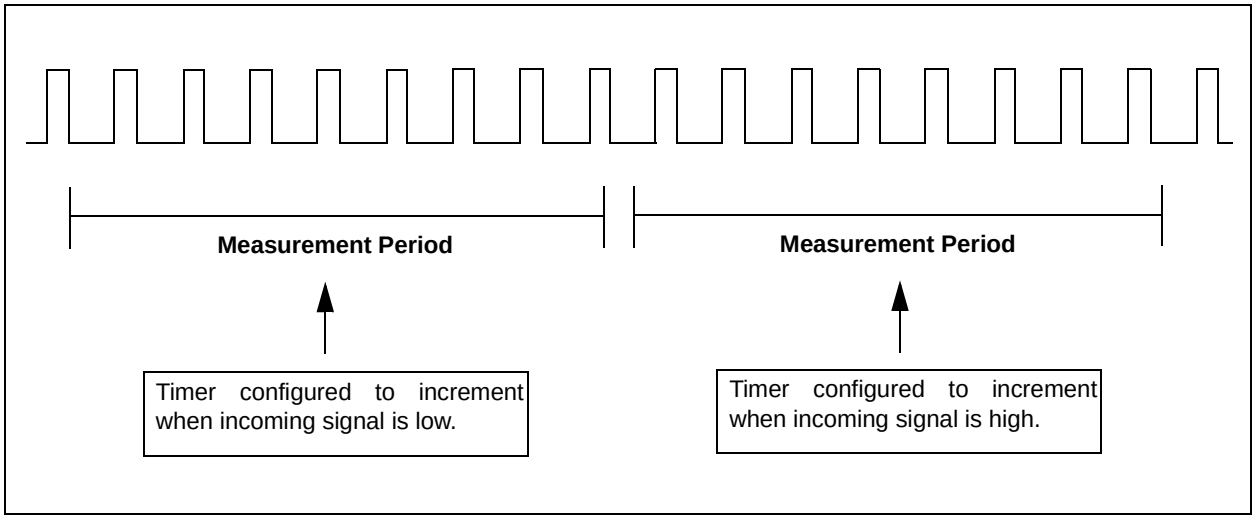
Another consideration is to ensure that the Timer1 peripheral is configured such that it will not roll over during the measurement period.

Once the values for T_{HIGH} and T_{LOW} have been determined it is necessary to use a math routine to perform the division. Since T_{HIGH} can assume a value up to 16 bits, the numerator could require up to 24 bits once the scaling factor is taken into account. The denominator for the fraction will be the result of a 16 bit addition, and therefore up to 16 bits need to be allocated for it.

Microchip application note AN617 "*Fixed Point Routines*", provides many fixed point math routines, including a 24-bit by 16-bit unsigned division routine.

Once the division has been performed, the result will be the 0-255 value corresponding to the duty cycle of the input.

FIGURE 10: PERFORMING DIGITAL DUTY CYCLE MEASUREMENT

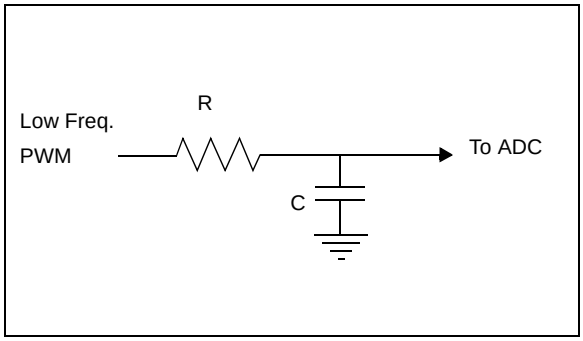


Measuring Speed Input using the ADC

The onboard ADC can also be used to determine the operating set point for the fan controller.

The analog input signal may be provided by a thermistor or possibly the result of a filtered PWM signal through a low pass filter, as shown below in Figure 11.

FIGURE 11: RC LOW-PASS CIRCUIT



This method may be used when measuring the PWM input directly is not possible, such as with very low frequency inputs. In cases where the host system may be operating at a much lower voltage, (i.e., 3.3 volts or 2.5 volts) the analog value can be read by the ADC and the response can be adjusted to match the host system voltage. It is also possible to use the ADC to interface to a thermistor in order to create a temperature controlled fan.

MEASURING THE FAN SPEED

In order to implement the closed loop control, it is necessary to measure the fan's actual speed. This is done by measuring the frequency of the Hall device output. There are several considerations, however, that need to be taken into account.

Creating a Software Timer

In the current application, Timer0 and Timer1 are used for PWM measurement, and Timer2 is used as a PWM time base. This configuration leaves no additional timers for fan speed measurement.

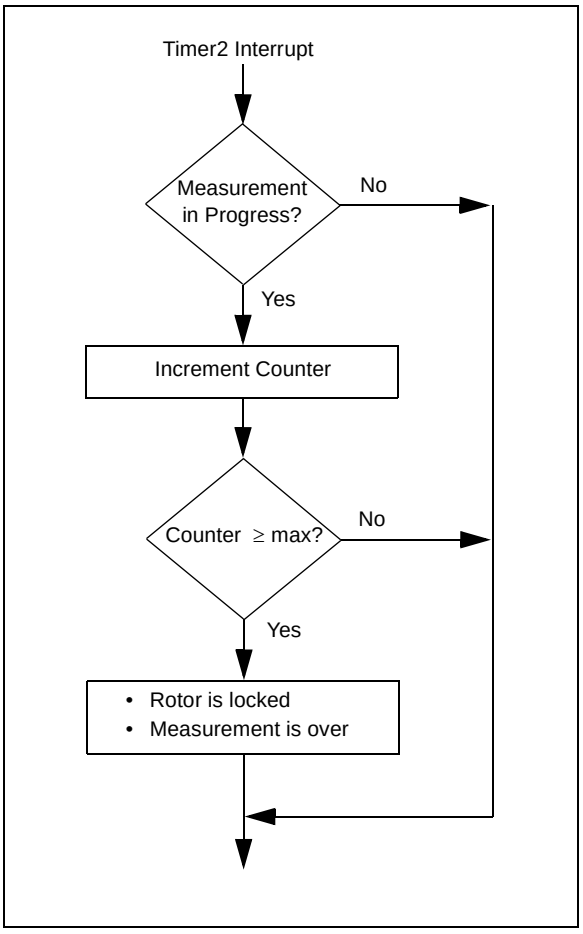
However, since Timer2 will never stop and will always interrupt at defined intervals, we can use it to create a software counter that can be used for fan speed measurement. This will be discussed in further detail below.

Measurement Duration / Locked Rotor

One difficulty in measuring the fan speed occurs during the locked rotor condition. Should the fan's rotor become blocked, the measurement of fan speed would take an infinite amount of time, as the frequency output of the Hall devices is zero.

Figure 12 shows a flowchart of the software measurement technique that can be used to measure the Hall device period.

FIGURE 12: LOCKED ROTOR DETECTION



The locked rotor scenario is taken into account by comparing the software counter against some maximum value. Should the counter ever exceed this value, a flag indicating the locked rotor condition is set. This allows the software to handle this condition appropriately.

However, even after the period measurement is complete, the fan speed has still not been determined, as a math routine is required to translate the Hall device period into RPM. This will be discussed next.

Measurement Math Routine

The goal of the measurement math routine is to produce an 8-bit value (0 to 255) that corresponds to the given fan's RPM. For example, for a fan that has a max RPM of 3300, then 1650RPM would correspond to a value of 127 and 3300 RPM would correspond to a value of 255. This is similar to the PWM input in that the value shows a percentage of full speed.

The relationship between period and RPM is inversely proportional, as shown in Equation 5:

EQUATION 5:

$$frequency = \frac{1}{period}$$

More specifically, the output of the math routine is scaled so that it only assumes values 0 through 255 and these values are specific to the maximum RPM of the given fan. In this case the equation becomes:

EQUATION 6: DETERMINING MAX RPM CONSTANT

$$Speed = \frac{\frac{(F_{COUNT} \cdot 255 \cdot 30)}{RPM_{MAX}}}{TimerCounts}$$

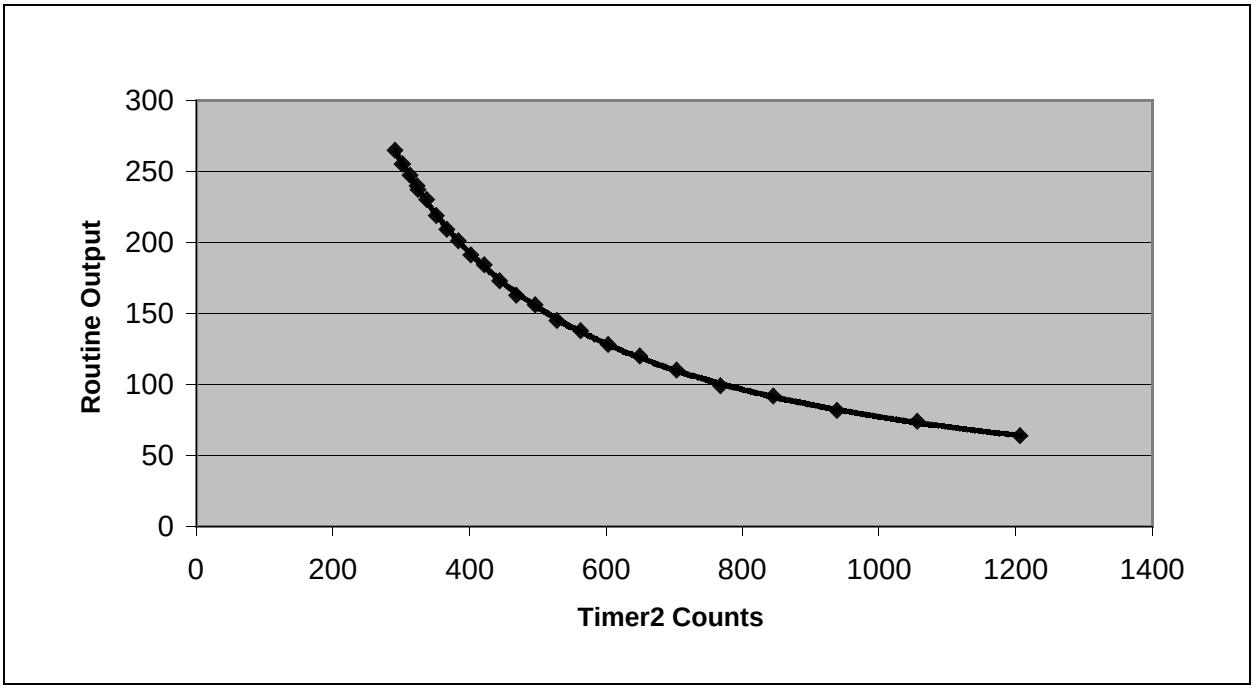
Where F_{COUNT} is the frequency of the Timer2 interrupt (i.e., the PWM frequency), 255 is a scaling factor showing the desired full scale response, and 30 is the conversion from RPM to pulses per second. RPM_{MAX} is the maximum RPM of the given fan.

Note:	The scaling factor of 30 assumes that the Hall device will provide two pulses (4 edges) per revolution. This is true of 4-pole motors and would require adjustment if an 8 pole motor were to be used.

The numerator of Equation 6 is simply a constant that must be chosen for the particular fan range. A better method is to use a tool, such as Excel, to show the output of the equation (considering the rounding) and to optimize the constant to minimize rounding errors.

The graph shown below in Figure 13 shows the relationship (for a given maximum RPM and given PWM frequency) between software measurement counts and the math routine output.

FIGURE 13: SPEED MEASUREMENT ROUTINE



One very important consideration is the condition where the fan may exceed its maximum speed. In Equation 6, the low byte of the result returns the 8-bit (0 to 255) value corresponding to fan speed. However, should the fan exceed its programmed maximum speed the routine would return a 9-bit result and since only the lower 8 bits are considered, it would appear as though the fan were spinning very slowly. An example is shown in Table 2 below, assuming a fan with a max speed of 3300 RPM.

TABLE 2: MEASUREMENT ROUTINE OUTPUT

Fan Speed	Routine Output	Lower Byte
1500	115	115
3300	255	255
3500	270	14

Because of this situation it is necessary to ensure that the result of the measurement math routine is valid. This is simply done by checking the higher bytes of the result, and should they ever assume a non-zero value, to set a flag indicating that the measurement is invalid.

GENERATING A TACHOMETER OUTPUT

Generating the tachometer output is a relatively simple task for the microcontroller. There are still a number of solutions that can be used.

General I/O Pin

Any general purpose I/O pin can be used to generate the tachometer output. The benefit of this method is that it puts the signal under complete software control. In some conditions, such as during a locked rotor condition, it may be desirable to set the tachometer to a certain level (i.e., always high during locked rotor conditions), and this can be done with the software method.

The obvious downside is that it requires software processing for each transition.

Using the Full-Bridge ECCP

If the ECCP module is used in Full-Bridge mode (as discussed earlier in the **Section "PWM Drive For Fan Windings"**), then the P1B and P1D pins are used as the PWMs for the fan windings. The direction change feature is used to transition from P1B to P1D.

A side effect of using the ECCP in this mode is that each direction change, the P1A and P1C pins will also transition. It is possible to use this output as the tachometer signal, and in this case it requires no software processing to generate the tachometer.

Using a Comparator Output

As mentioned earlier in the Position Sense section, a Hall element can be used to determine the rotor position. In this configuration, the comparator is used to interface the Hall element to the microcontroller, and the comparator output provides the position data. The comparator output can also be directed to an output pin, and in this configuration would provide the tachometer output.

GENERATING AN ALARM OUTPUT

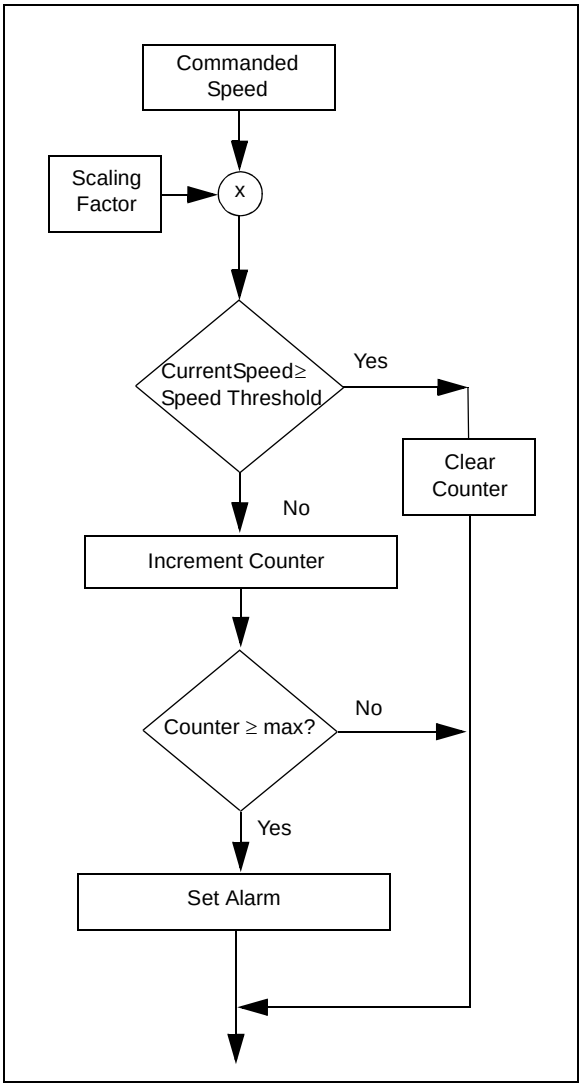
While a tachometer is one very common speed sense output, the alarm signal is another.

The alarm signal is used to signal the host when a problem has occurred. Should the fan rotor become locked or should the fan be unable to reach its desired speed the alarm signal will be asserted to alert the connected system.

One of the difficulties in creating the alarm output lies in the fact that the comparison is not instant. That is, the fan speed needs to be below a threshold for a certain amount of time to assert the alarm. If this were not the case, then speed increases would assert the alarm until the fan reached a steady state, and this is typically not the desired operation.

The flowchart shown below in Figure 14 illustrates the operation of the alarm routine. The first step is to calculate the alarm threshold, and this is done dynamically. For example, the alarm threshold may be set at 65% of the commanded speed. In this case the threshold would be 650 RPM for a commanded speed of 1000 RPM, 1300 RPM for a commanded speed of 2000 RPM, and so on.

FIGURE 14: ALARM OUTPUT DIAGRAM



After the alarm threshold has been calculated, the routine will compare the current speed against the threshold speed. If the current speed is too low, the counter is incremented and compared against a max value to determine when to enter the alarm state. If the current speed is above the threshold then the counter is cleared and the alarm condition is ended.

Note:	There are several algorithms that could be used to detect the alarm condition, and this is just one possibility.

ADDITIONAL FEATURES

The following section will describe several additional features that may be added to the design.

Starting Ramp and Delay

A typical brushless fan will draw a large amount of current when it is first energized. This can place additional stress on power supplies and possibly cause sags in voltage. In order to avoid this, a starting delay and starting ramp can be used.

There are two situations in which the fan will delay its start:

1. The fan has recently been energized
2. The fan was operating at zero speed and is commanded to operate at a non-zero speed.

This delay will significantly lower the amount of inrush current due to the fan.

Furthermore, rather than instantly commanding the fan to operate at full speed, a starting ramp can be used. This ramp will slowly increase the speed of the fan until it reaches its operating speed. The combination of the starting delay and the starting ramp will ensure that the fan will never demand a high inrush current.

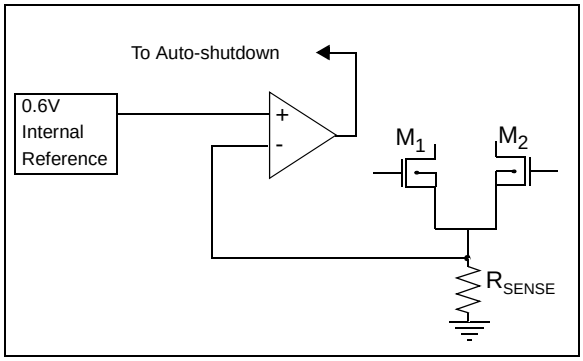
Current Limiting

Depending on the device and configuration being used, it is also possible to limit the current into the fan windings.

The current limiting requires the use of an analog comparator. However, the direct Hall element interface also requires a comparator. Because of this, a device such as the PIC12F615 can not implement both a direct Hall element interface and current limiting.

If an analog comparator is available, then the current limiting can be performed as shown in Figure 15 below.

FIGURE 15: AUTO-SHUTDOWN SCHEMATIC



In the Figure 15, M1 and M2 are the two switching devices for the fan windings. The comparator is used to compare the voltage across the sense resistor, R_{SENSE}, to the internally generated 0.6 volt reference.

In this configuration the comparator is connected to the auto-shutdown logic of the PWM module. Should the comparator output trip, the PWM will automatically be placed in the Idle state. The device can be configured such that once the comparator output is low the PWM is once again allowed to operate.

This hardware feature allows cycle-by-cycle current limiting of the winding current without requiring any software resources or user-intervention. Furthermore, an interrupt can be generated should the current exceed the trip level and software actions can also be taken.

SCHEMATIC OVERVIEW

An example hardware schematic is provided in Appendix A, and will be discussed here in further detail.

Protection Components

One of the most important aspects of the hardware design is the inclusion of necessary protection components.

- Reverse polarity protection – Diode D1 is included in the design in order to implement reverse polarity protection. This will prevent current flow should the power be connected backwards.
- R2, R3, and R4 are chosen to limit the amount of current that flows into the MOSET gates during switching or during a MOSFET failure where excess current would be allowed to flow into the gate.
- Pull-down resistors R5 and R6 are used to prevent any accumulation of charge that may occur on the MOSFET gates. This will ensure when the fan is energized for the first time that excess current will not flow.

- Series resistor R9 is included to prevent microcontroller damage during hot plugging. Hot plugging is the act of plugging the fan into a powered system. When this occurs, some pins may make contact before other pins and cause excess current flow.

  Specifically, should the PWM pin make connection before the ground pin, there exists a path for current to flow into the microcontroller and out of the PWM pin to the host system. This situation could damage both the microcontroller and the host system.
- Capacitors C1 and C2 are used to prevent the drain-source voltage from spiking and damaging the MOSFETs.

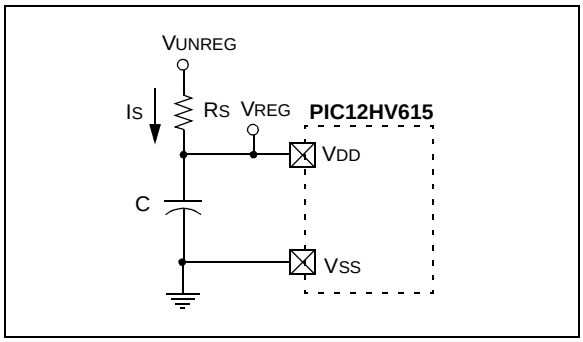
Shunt Regulator

In order to lower system cost, the PIC12F615 and PIC16F616 microcontrollers are available with an HV option as the PIC12HV615 and PIC16HV616.

These HV parts have an integrated shunt regulator that can be used to replace traditional Zener regulators or linear regulators.

A diagram of the shunt regulator is shown below in Figure 16.

FIGURE 16: SHUNT REGULATOR SCHEMATIC



The shunt regulator is cost effective in that it only requires a single resistor to operate, and does not have the supply voltage limitations associated with linear regulators. Furthermore, it can also supply regulated voltage to other components.

FAN RESPONSE CHARACTERISTICS

Two graphs are shown (Figure 17 and Figure 18) to represent different fan response characteristics of two different brushless DC fans. These graphs are obtained by setting the PWM duty cycle to the fan windings at a fixed value and observing the final speed of the fan. Knowing how the fan speed varies with duty cycle will make the implementation of the control routine easier.

The characteristic response of the fan may also vary with supply voltage and with PWM duty cycle. For fan number 1, shown in Figure 17, a very small range of duty cycles (0%-10%) covers a very wide range of speeds. It will be necessary for its control routine to quickly switch between duty cycles in order to achieve a certain speed.

For fan number 2, shown in Figure 18, a much larger range of duty cycles is useful (50%-100%). However, duty cycles under 50% have no affect on the fan speed and will only produce wasted energy. For this fan it is important to design a controller that will attempt to stabilize on the proper duty cycle for the given speed, as any change in duty cycle may be auditable.

FIGURE 17: FAN #1 RESPONSE

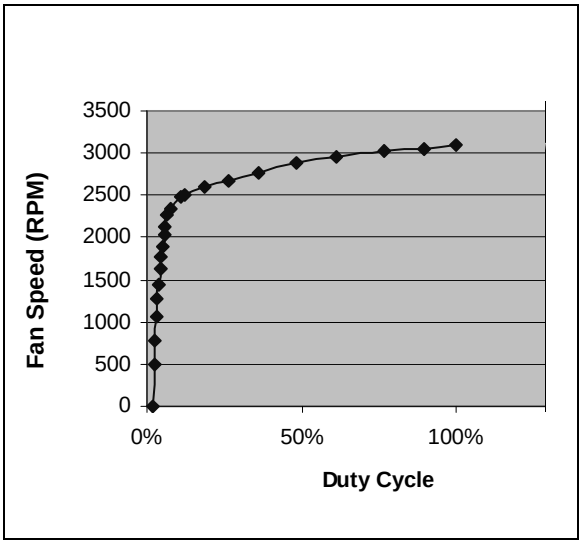
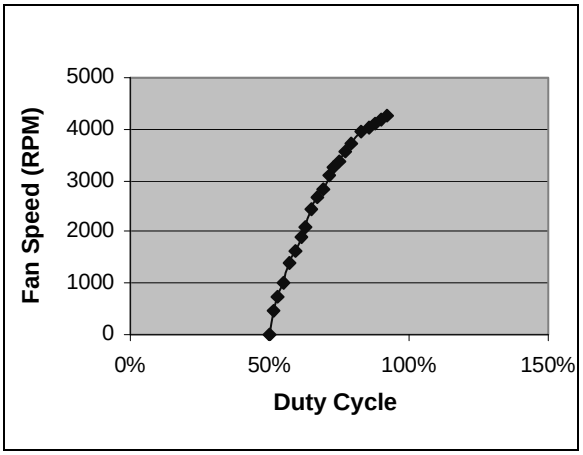


FIGURE 18: FAN #2 RESPONSE



CONTROL LOOP SOFTWARE

The block diagram for our system is shown below in Figure 19. The speed measurement, set-point measurement, and PWM blocks have each been discussed in detail in previous sections. The control loop will now be discussed.

Controlling a fan presents itself as an interesting control problem in several aspects:

- The PIC microcontroller is only able to increase the speed of the fan, but not decrease it.
- The fan has an extremely long settling time and responds very slowly.
- The fan's operating conditions, including voltage and load, may change abruptly at any time.

The control loop software will also depend on the requirements. For example, most fan applications may require that the speed control software not introduce extra noise (i.e., cycling between two duty cycles in an auditable manner). This may come at the cost of a slower response or longer settling time.

However, it is also possible to design a control loop that will reach its set point very quickly, but this may come at the cost of overshoot and added noise.

PID Control (Fan #1)

Proportional, Integral, Derivative (PID) control is a very common control technique for many applications. It has been discussed in numerous application notes, such as AN258, AN964, and AN937. Specifically, the routine taken from AN258 was used as the control loop software for fan #1.

Given the extremely long settling time of the fan, the derivative term from the PID routine has very little effect on the system, and can often be omitted completely.

Increasing the gain (the proportional term) will cause the fan to reach its desired speed faster. However, in some cases it may cause the fan to audibly change between different duty cycles.

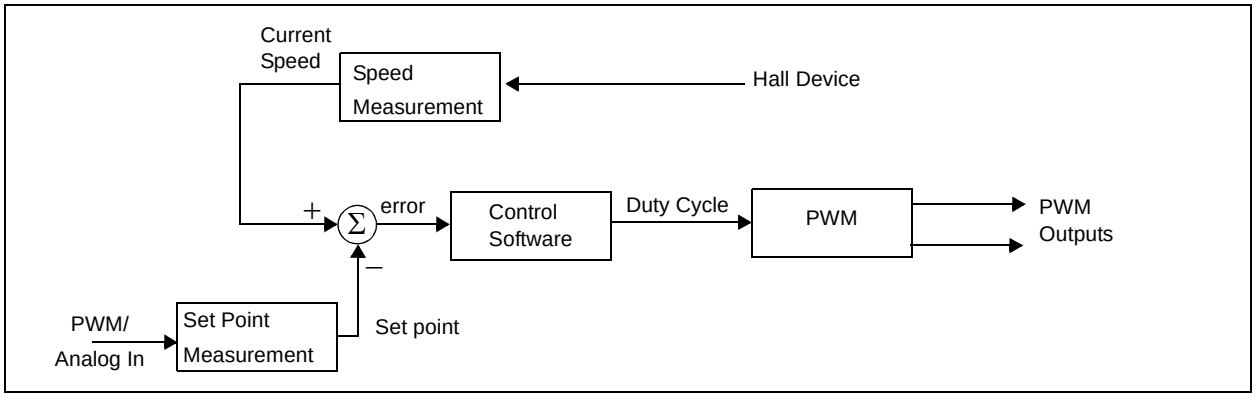
One downside of the PID algorithm is the fact that it is not easy to switch between different PID coefficients. For example, when the fan's speed is increased from a low to a high set point, the controller is able to appropriately control the duty cycle. However, if the speed is suddenly decreased, the control loop has no way to slow the fan down, and the speed may end up falling below the desired value.

For example, if the fan is operating near 100% and the set point is reduced to a minimum value (such as 10%) the fan may undershoot such that it stalls. It is important to test these cases to ensure the controller is operating properly, and to find the PID constants that provide an acceptable response in all conditions.

Integral Control (Fan #2)

Another possible controller is an integral only controller which follows Equation 7. This type of controller was used for fan #2.

FIGURE 19: CONTROL LOOP BLOCK DIAGRAM



EQUATION 7: I CONTROLLER

$$Output = \Sigma K_i \cdot Error$$

One benefit of this type of controller is the constant K_i can be modified without abrupt changes in the output. Depending on the current speed condition, the appropriate K_i can be selected. For example:

- If the current speed is much larger than the desired speed, the fan needs to slow to the desired speed without undershooting. This can be accomplished by reducing the size of K_i.
- If the current speed is close to the desired speed, the K_i can be set to a standard operating value.
- If the current speed is much lower than the desired speed the fan needs to increase speed. The standard K_i can be used, but this may slow the response. The K_i can be increased to allow the fan to respond faster.

The integral controller was chosen for Fan #2 because it allowed the elimination of undershoot when the desired speed was reduced.

This type of controller will often have a longer settling time than the PID controller. However, since one of the main goals of the fan controller is to reduce noise, the gradual changes may be desirable.

Eliminating Steady State Jitter

Under normal operation, the control loop software will cause the actual fan speed to equal the desired fan speed. However, due to limitations in PWM resolution and speed measurement resolution, this may not be possible.

In this case it is possible that the actual fan speed will oscillate around the desired speed. The control loop may choose a duty cycle too small and the fan speed will be less than the desired speed, and consequently the control loop will choose a larger duty cycle that may be too large and cause the fan to spin too quickly.

If this oscillation about the desired speed happens quickly, then it may not be auditable or noticeable. However, if the oscillation happens very slowly, it may be very noticeable and will cause problems.

Suggested here are two ways to eliminate the jitter:

1. Increase the PWM resolution
2. Use a dead band

Of these two methods, increasing the PWM resolution is the most desirable. The PWM drive section of this document describes to maximize the resolution in both hardware and software.

However if maximizing the PWM resolution is not possible, or if it still does not yield desirable results, a dead band can be used.

The dead band works by modifying the error signal. If the magnitude of the error is less than some amount, the error will be set to zero. This is illustrated below:

EQUATION 8: DEAD BAND

$$error = \begin{cases} 0 & if\ |error| < CONSTANT \\ error & otherwise \end{cases}$$

The dead band has the effect of ignoring small errors and considering them to be zero. This will eliminate jitter that is caused by small errors. However there are a few other side effects.

Ignoring small changes in the error will also have the effect of ignoring small changes in the speed input. If the dead band is set to 2, the speed input may need to change by as much as 4 in order for the fan speed to actually change.

Using the dead band may also require changes to the control loop state machine. If the error is very close to the dead band constant, it may still jitter (depending on measurement accuracy), except the jitter will cause the error signal to go between CONSTANT and 0, which may cause erratic behavior.

It is more desirable to wait until the fan has actually reached the desired speed and then enable the dead band. If the fan speed falls out of the range such that the dead band is no longer active, then the dead band can be disabled until the fan speed reaches its desired speed again.

ULTIMATE RESPONSE

The graphs below illustrate the response of the finished software. A transient response graph is shown (Figure 20), illustrating speed versus time. The desired speed is first set to a high value, then to a low value, and then back to a high value. This is done to test and show any over shoot or undershoot that occurs.

The second graph (Figure 21) shows the linearity of the actual speed output (after settling) vs. input duty cycle (desired speed). This is useful in determining how accurate the fan is for a given speed input.

FIGURE 20: TRANSIENT RESPONSE

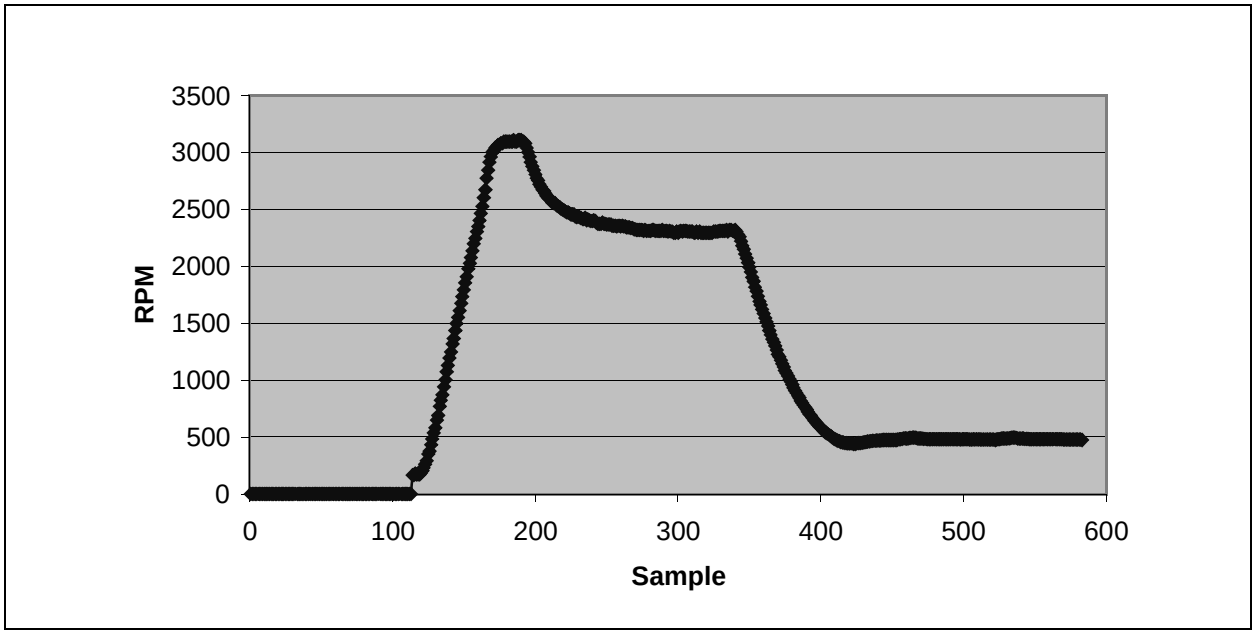
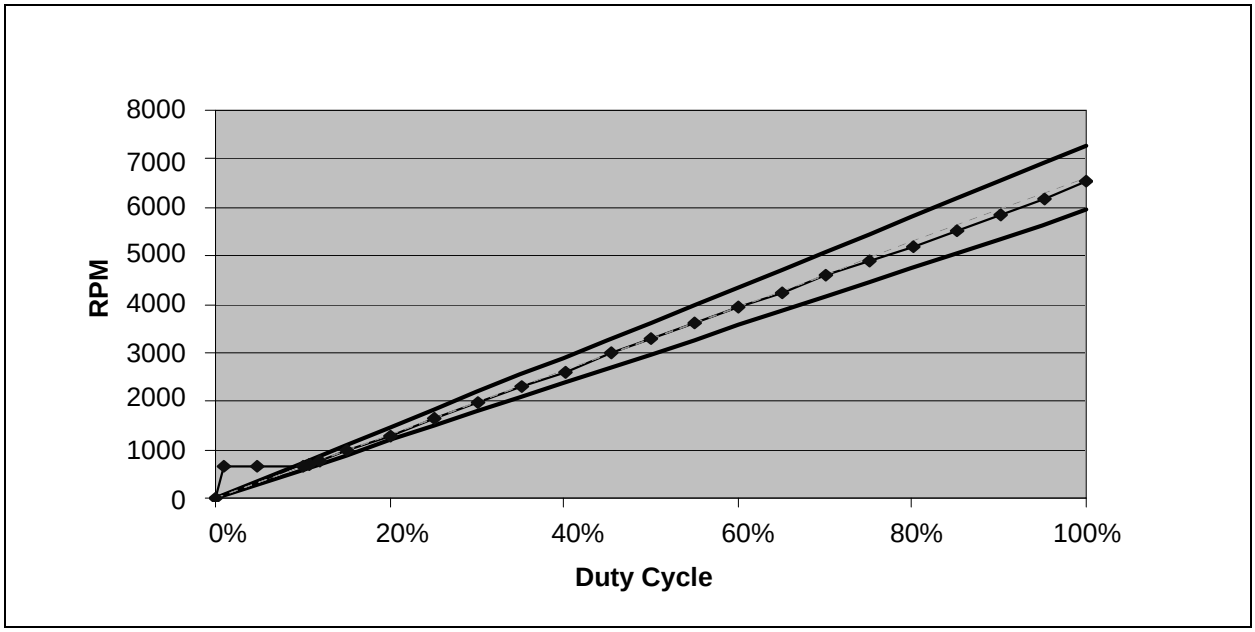


FIGURE 21: LINEARITY



SOFTWARE IMPLEMENTATION

Assembly language software is included for both the PIC12F615 and PIC16F616 that implements all of the features described in this application note.

The software uses a series of **#define** statements to configure various options, such as tachometer or alarm output, active high or active low coil PWM outputs, etc. These options are documented using comments in the source code.

The software uses around 800-900 words of program memory and around 40-50 bytes of data memory. The exact size is affected by the various build options.

REFERENCES

AN847 "*RC Model Aircraft Motor Control*" (DS00847)

AN857 "*Brushless DC Motor Control Made Easy"* (DS00857)

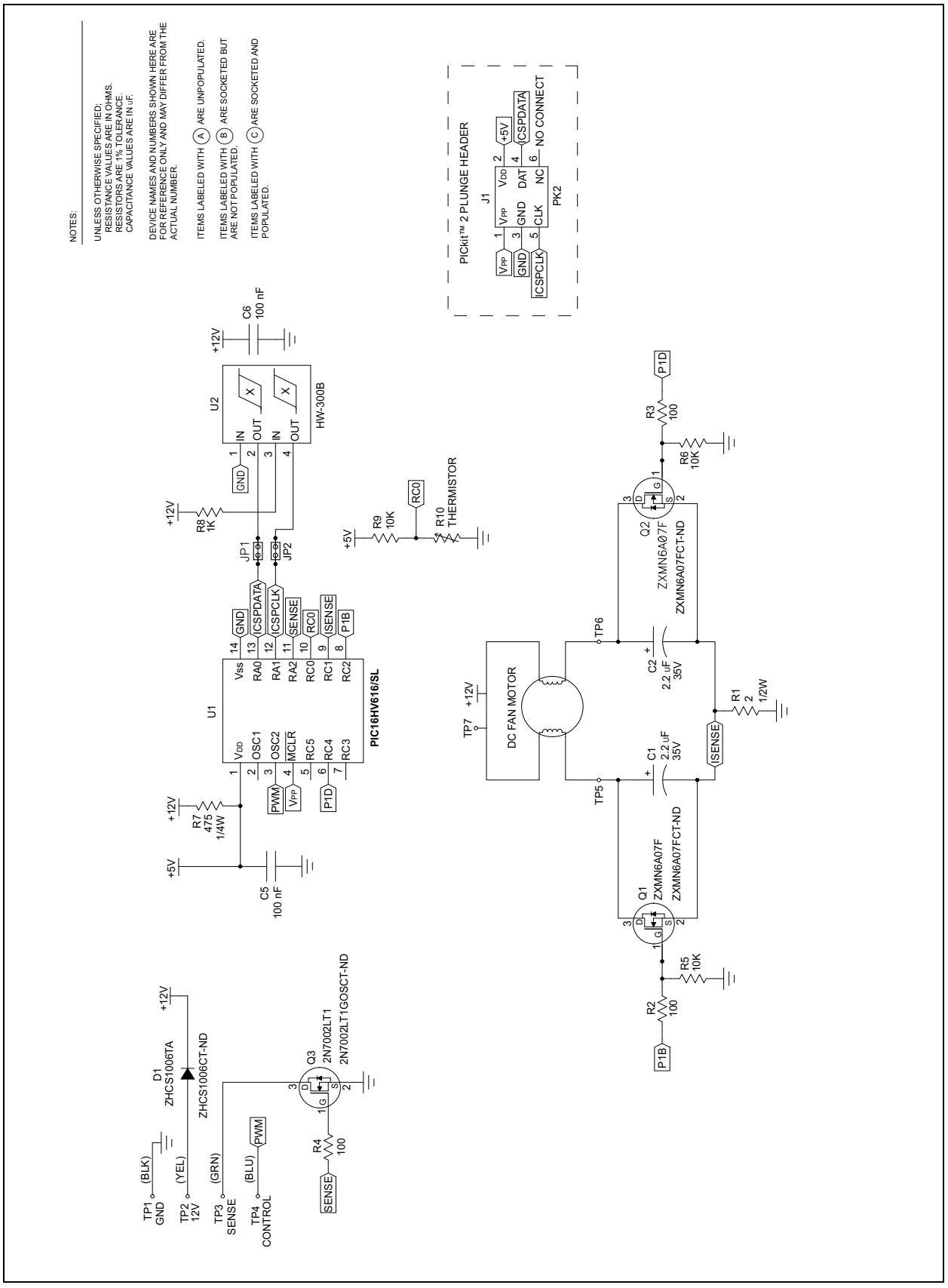
AN893 *"Low-Cost Bidirectional Brushed DC Motor Control Using the PIC16F684"* (DS00893)

AN894 *"Motor Control Sensor Feedback Circuits"* (DS00984)

AN898 *"Determining MOSFET Driver Needs for Motor Drive Applications"* (DS00898)

AN905 *"Brushed DC Motor Fundamentals"* (DS00905)

APPENDIX A: SCHEMATIC



NOTES:

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.

Trademarks

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2002

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.

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://support.microchip.com
Web Address:
www.microchip.com

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

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
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
Kokomo, IN
Tel: 765-864-8360
Fax: 765-864-8387

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

Santa Clara
Santa Clara, CA
Tel: 408-961-6444
Fax: 408-961-6445

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

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

China - Beijing
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

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

China - Hong Kong SAR
Tel: 852-2401-1200
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: 86-21-5407-5533
Fax: 86-21-5407-5066

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

China - Shenzhen
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

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

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

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

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

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-4182-8400
Fax: 91-80-4182-8422

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

India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

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: 886-3-572-9526
Fax: 886-3-572-6459

Taiwan - Kaohsiung
Tel: 886-7-536-4818
Fax: 886-7-536-4803

Taiwan - Taipei
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

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

EUROPE

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

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

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

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

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

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

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

UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

01/02/08