AN1858

Sensorless Brushless dc Motor Using the MC68HC908MR32 Embedded Motion Control Development System

By: Ken Berringer and Bill Lucas
Freescale Microcontroller Division
Austin, Texas

Leos Chalupa and Libor Prokop
Roznov System Application Laboratory
Roznov, Czech Republic

Overview

Motorola's MC68HC908MR32 (MR32) microcontroller (MCU) is specifically designed for motor control applications. The MR32 is suitable for ac induction motors as well as brushless dc (BLDC) motors. The pulse-width modulator motor control (PWMMC) module can be configured to control brushless dc (BLDC) motors using several different PWM strategies.

BLDC motors are being designed into many applications that require fractional horsepower, highly efficient, variable speed motors. Brushless dc motors have several advantages over other motor types. For instance:

- Because BLDC motors do not have brushes to wear out, they have a longer life and greater reliability than normal brushed dc motors.
- BLDC motors offer performance similar to dc motors with a high starting torque and a high no-load speed.

- BLDC motors typically have a higher efficiency than ac induction or switched reluctance motors. The permanent magnets used in the rotor means that less energy is wasted in rotor losses.

BLDC motors require electronic control. Some BLDC motors use Hall effect sensors to provide absolute position sensing. This results in more wires and higher cost. Sensorless control eliminates the need for Hall effect sensors, using the back-EMF (electromotive force) of the motor to estimate the rotor position. Sensorless control is essential for low-cost variable speed applications such as fans and pumps. Refrigerator and air conditioning compressors also require sensorless control when using BLDC motors.

Using Hall effect sensors inside a compressor is not practical because the environment is not suitable and the number of electrical connections would be problematic. Also, compressors are difficult to start and are perhaps the most demanding sensorless BLDC application. The motor must achieve high torque on the first rotation to overcome the piston compression.

MC68HC908MR32 Features

The MR32 is a new member of the low-cost, high-performance M68HC08 (HC08) Family of 8-bit microcontrollers, designed specifically for midrange motor control applications.

The Freescale HC08 Family of MCUs is an enhanced, fully upward object code compatible architecture that evolved from the M68HC05 (HC05) Family. The HC08 Family is a performance extension to the HC05 Family of low-cost MCUs. All MCUs in the family use the enhanced M68HC08 central processor unit (CPU08) that includes new addressing modes, many new instructions, and the performance improvements to existing instructions that result from the introduction of instruction pipelining. All MCUs in the HC08 Family are available with a variety of package types, input/output (I/O) modules, and various memory sizes and types.

MR32 motor control MCU features include:

- High-performance M68HC08 (CPU08) architecture
- Fully upward-compatible object code with M68HC05 Family
- 8-MHz internal bus frequency
- 32 Kbytes of on-chip FLASH memory
- FLASH data security
- 768 bytes of on-chip random-access memory (RAM)
- 12-bit, 6-channel pulse-width modulator motor control (PWMMC) module
- Serial peripheral interface module (SPI)
- Serial communications interface module (SCI)
- 16-bit, 4-channel timer interface module (TIMA)
- 16-bit, 2-channel timer interface module (TIMB)
- Clock generator module (CGM)
- Digitally filtered low-voltage inhibit (LVI)
- 10-bit, 10-channel analog-to-digital converter (ADC)
- Optional computer operating properly (COP) reset
- Low-voltage detection with optional reset
- Fault detection with optional PWM disabling
- Low-power design (fully static with wait mode)
- Master reset pin (RST) and power-on reset (POR)
- 64-pin plastic quad flat pack package (QFP)

Some of the features of the CPU08 include:

- Enhanced M68HC05 programming model
- Extensive loop control functions
- 16 addressing modes (eight more than the HC05)
- 16-bit index register and stack pointer
- Memory-to-memory data transfers
- Fast 8 by 8 bit multiply instruction
- Fast 16/8 bit divide instruction
- Optimization for controller applications
- Improved C language support

This application note does not discuss in great detail each of the modules resident on the MR32. **Figure 1** shows a block diagram of the MR32. For a detailed description of the MR32, refer to the *68HC908MR32, 68HC908MR16 Technical Data: Advance Information*, Freescale document order number MC68HC908MR32/D.

The MR32's PWMMC module makes the device an excellent choice for use in an embedded motor control system. A review of the PWMMC module and its features is included here.

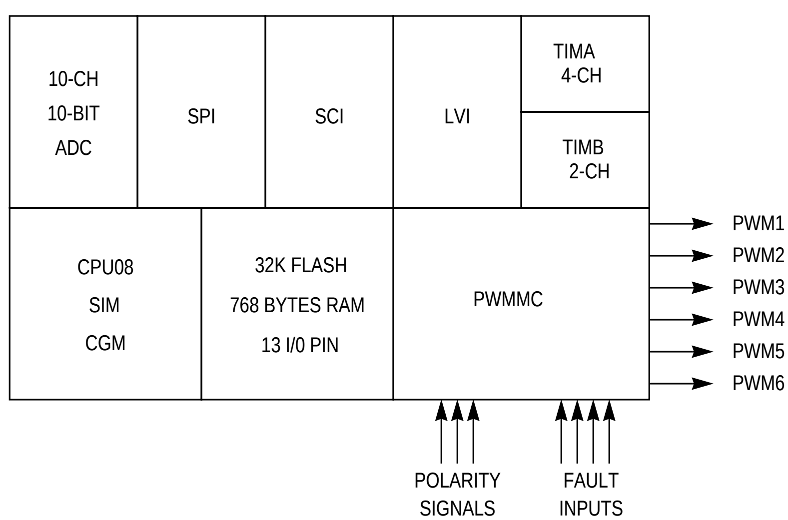

Figure 1. MC68HC908MR32 Block Diagram

MC68HC908MR32 Pulse-Width Modulator

The pulse-width modulator (PWMMC) module resident on the MR32 is specifically designed to provide pulse-width modulated outputs to drive a power stage connected to a dc servo, brushless dc, or 3-phase ac motor system. The PWMMC module can be partitioned and configured in several ways, depending on the specific control motor application. **Figure 2** shows a block diagram of the PWMMC and is referenced throughout this explanation of the PWM generator.

Features of the MR32's PWMMC include:

- Three complementary PWM pairs or six independent PWM signals
- Complementary mode features include:
  - Dead-time insertion
  - Separate top/bottom pulse-width correction via current sensing or programmable software bits
- Edge-aligned PWM signals or center-aligned PWM signals
- PWM signal polarity
- 20-mA current sink capability on all PWM outputs
- Manual PWM output control through software
- Programmable fault protection

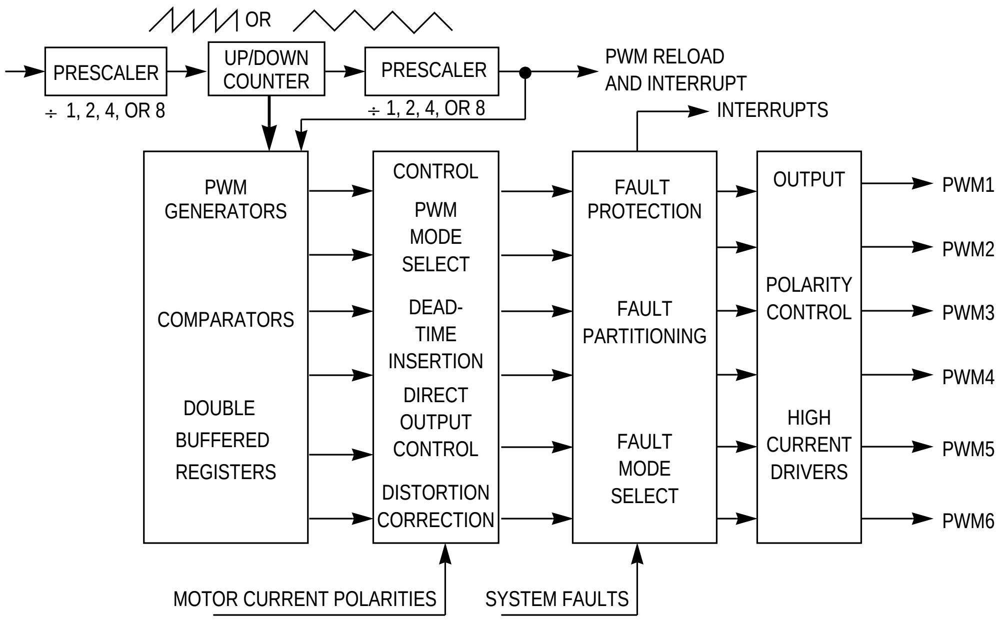

Figure 2. PWMMC Module Block Diagram

The six outputs of the PWM generator can be configured as individual pulse-width modulated signals where each output can be controlled as an independent output. Another option is to configure the output in pairs, with the outputs complementary so that driving complementary top and bottom transistors on a power stage becomes an easy task. The outputs of the PWM are capable of sinking up to 20 mA. That drive capability allows for direct drive of optocouplers without the need for additional drivers.

To prevent erroneous signals being output from the PWMMC module while loading new values, the bulk of the registers are double buffered and new output is inhibited until a bit in a PWM control register (LDOK) is set, indicating it is OK to output the new values.

Fault Protection

Conditions can arise in the external drive circuitry, requiring that the PWM signals become inactive immediately. These conditions include overcurrent, overvoltage, overtemperature, or other error conditions. The four fault input pins on the MR32's PWMMC module can be configured to react in a number of different ways, upon the detection of a fault. Each fault input has its own interrupt vector. In all fault conditions, the output of the PWM generator is forced to a known inactive state.

A number of fault control and recovery options is available to the systems architect. In some cases, it may be desirable to selectively disable PWM(s) solely with software. Manual and automatic recovery mechanisms are available that allow certain acceptable fault situations to occur, such as starting a motor and using a fault input to limit the maximum startup current. The fault inputs may be partitioned if the MR32 is used to control multiple motors.

Two of the fault inputs are used for the brushless dc motor control software. Hardware overvoltage and overcurrent faults use these same two fault pins. Both of these faults are configured to turn off all six of the power transistors and jump to an error-handling routine.

Another possible use for the fault inputs is to provide a cycle-by-cycle current limit. This type of hardware current limit is especially useful in dc, brushless dc, and switched reluctance motors. Due to the requirement for isolation, a hardware cycle-by-cycle current limit is implemented on the power board. In addition, the software developed for this application note uses a software current controller during startup.

PWM Output Alignment

Depending on the system design, there is a choice between edge- or center-aligned PWM signals that are output from the MR32's PWMMC module. The PWM counter uses the value in the timer modulus register to determine its maximum count.

In center-aligned mode, a 12-bit up/down counter is used to create the PWM period. The PWM resolution in center-aligned mode is two clock periods. The highest resolution is 250 ns at a processor speed of 8 MHz.

In edge-aligned mode, a 12-bit up-only counter is used to create the PWM period. Therefore, the PWM resolution in edge-aligned mode is one clock and the highest resolution is125 ns at a processor speed of 8 MHz.

BLDC motors might use edge-aligned or center-aligned PWM signals depending on the application requirements. Most applications, that only require variable speed operation, can use six independent edge-aligned PWM signals. This provides the highest resolution. If the application requires servo-positioning, dynamic braking, or dynamic reversal, it is recommended that complementary center-aligned PWM signals be used. The sensorless control method implemented in this application note uses six independent center-aligned PWM signals to minimize the effect of sampling noise.

PWM Load Operations

When generating sine waves to a motor, an interrupt routine typically is used to step through a sine table in memory, scale that sine value, and output the result from the PWM generator. The rate at which the sine table is scanned can be derived from an interrupt from the PWM generator. The PWMMC module can be programmed to provide an interrupt rate of every one, two, four, or eight PWM reload cycles.

Direct Output Control

In some cases, the user may desire to bypass the PWM generator and directly control the PWM outputs. A mechanism exists to disconnect PWM generator from its outputs and directly control the six PWM outputs. When this mode is used, the PWM generator continues to run; however, its output is disabled, overridden by direct output.

The PWMOUT register can also be used for BLDC motor control to selectively disable the PWM channels. This method permits a single 8-bit pattern to control the BLDC motor commutation.

Dead-Time Insertion and Dead-time Compensation

When the PWM generator is used in complementary mode, the PWMMC module provides dead-time insertion. This provides a short delay time between turning off the top power transistor and turning on the lower transistor in one phase leg. Dead-time is essential to safe switching and low-power losses. Dead-time can be specified in the dead-time write-once register. This 8-bit value specifies the number of CPU clock cycles to use for the dead-time.

During the dead-time, the current might be either positive or negative. The voltage on the motor phase will depend on the direction of the motor current. This leads to distortion from the desired voltage waveform. The MR32 provides three current sense inputs which can be used for dead-time compensation. Dead-time compensation provides smooth sine wave currents for ac induction motors even at low-speed operation.

Most BLDC motors do not require complementary PWMs, dead-time insertion, or dead-time compensation. The software presented here does not use these features. The only BLDC applications that might require these features are high-performance BLDC servomotors, sine wave excited BLDC motors, brushless ac, or ac synchronous motors. These types of motors account for less than 1 percent of BLDC motor drives today, but might find applications in electric vehicles, electric power steering, and robotics in the near future.

Brushless dc Motor Control Algorithm

Many different control algorithms have been used to provide sensorless control of BLDC motors. Most hard disk drive controllers use a linear control IC (integrated circuit) to provide sensorless control of the spindle drive. Typically, the motor voltage is controlled using a power transistor operating as a linear voltage regulator. This is not practical when driving higher-power motors. High-power motors must use PWM control and require a microcontroller to provide starting and control functions.

The control algorithm must provide three things:

- PWM voltage which controls the motor speed
- Mechanism to commutate the motor
- Some method to estimate the rotor position using the back-EMF (electromotive force) of the motor

Pulse-width modulation is used to apply a variable voltage to the motor windings. The effective voltage is proportional to the PWM duty cycle. When properly commutated, the torque-speed characteristics of the BLDC motor are identical to a dc motor. The variable voltage can be used to control the speed of the motor and the available torque.

The commutation of the power transistors energizes the appropriate windings in the stator to provide optimum torque generation depending on the rotor position. In a conventional dc motor, this task is performed by the brushes and commutator. In a BLDC motor, the MCU must know the position of the rotor and commutate at the appropriate time.

When the rotor is turning, it acts as a generator. The moving magnets induce an ac voltage into the stator windings. This ac voltage is always present when the motor is turning. The applied voltage must be greater than this induced voltage to provide torque-generating current. In effect, the motor is pushing back with its own voltage. The induced voltage is called the back-EMF. EMF stands for electromotive force, a physics term for voltage. BLDC motor drive circuits are designed so that one of the three phases is always open, with both upper and lower transistors in the off state. The open phase is used to sense the back-EMF of the motor. This provides useful information on the position of the rotor.

Commutation

A 3-phase brushless dc motor has six power transistors arranged in a 3-phase bridge (see **Figure 3**). The power transistors are switched according to a predefined commutation pattern. Several different patterns are used for BLDC motors. To use sensorless control, one phase must always be open. The motor has three windings, typically in a wye (Y) configuration. The center point is floating. So the open phase voltage is relative to the center point. The center point is affected by the voltage applied to the other two phases.

Many BLDC motor drives modulate only the bottom transistors. This causes the center point to shift upward and the PWM voltage is actively coupled to the open phase. This commutation pattern, therefore, is not suitable for sensorless BLDC motors using PWM. A delta winding is electrically equivalent to a wye connection and behaves in a similar fashion relative to a virtual center point.

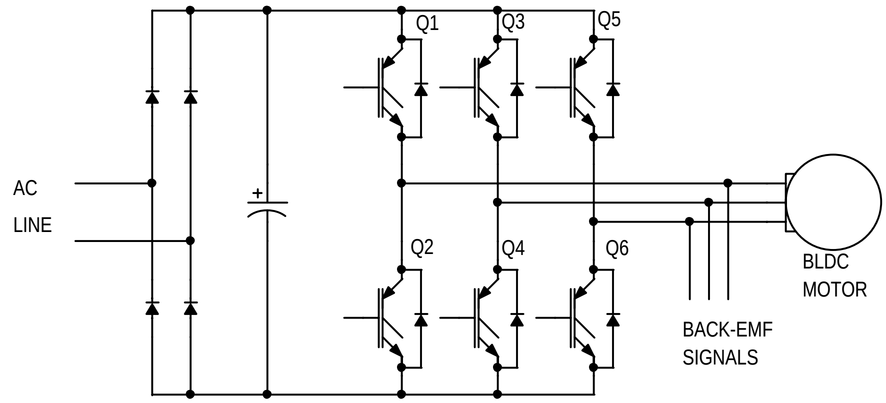

Figure 3. BLDC Motor System

The sensorless control algorithm developed for the MR32 uses a symmetrical PWM commutation pattern (see **Table 1**). The PWM signal always is applied to diagonally opposite transistors. This keeps the center point of the wye in the center of the dc supply voltage. The two active windings balance so that the PWM voltage is not coupled directly to the open winding. There still is some noise due to winding mismatch and capacitive coupling which is filtered out easily.

Table 1. BLDC Motor Commutation

	Top			Bottom		
	A	**B**	**C**	**A**	**B**	**C**
0	—	PWM	—	PWM	—	—
1	—	PWM	—	—	—	PWM
2	PWM	—	—	—	—	PWM
3	PWM	—	—	—	PWM	—
4	—	—	PWM	—	PWM	—
5	—	—	PWM	PWM	—	—
	Q1	Q3	Q5	Q2	Q4	Q6

This commutation pattern requires only a single PWM signal. Identical PWM signals are applied to two of the six transistors. The software uses a trick to force the unused PWM channels to the off state. The unused PWM channels are disabled by setting the most significant bit (bit 15) of the PWM value register. This also corresponds to the sign bit for a 16-bit signed integer. The PWMMC module is designed so that a number equal to or less than 0 will force the channel off. This feature is useful for ac induction motors and provides saturation for data values outside the normal operating range. Alternatively, the unused channels could be disabled by writing a 0 to the PWM value register.

This commutation pattern also could be accomplished using a single PWM with a hardware multiplexer. However, this is not the only acceptable PWM strategy for BLDC motors. Applications which require servo-positioning or regenerative braking might require two PWMs with dead-time. Large motors greater than 750W might require three PWMs to minimize torque ripple. The MR32 PWMMC module is flexible enough to be used with most BLDC PWM strategies.

Back-EMF zero-crossing

The back-EMF of the three motor phases is measured using a resistive divider. The dc bus voltage is also measured using an identical divider circuit. Three comparators are used to provide a zero-crossing signal for each phase. Each phase voltage is compared to one-half the dc bus voltage. The comparators also have some filtering to minimize the effect of the PWM noise on the zero-crossing signal.

The microcontroller is in command of the commutation and knows exactly which channel is open at any time. A multiplexer is used to combine the three zero-crossing signals into a single combined zero-crossing signal for the open phase. The microcontroller provides the three signals used to select the open phase. A single input capture function is then used to measure the zero-crossing time.

Each motor phase has significant inductance. After the motor is commutated, the winding inductance will force the open phase voltage to a diode drop above the dc bus voltage or below ground. This results in a prominent notch at the start of the open phase voltage. This notch must be ignored by the software.

Under normal running conditions, the back-EMF zero-crossing ideally should be located right in the center of the commutation period. When a good zero-crossing is detected, the time is measured from the last commutation, doubled, and added to the last commutation time. Thus, the commutation period is twice the zero-crossing time and the zero-crossing is right in the center of the period.

Unfortunately, just measuring the zero-crossing and doubling does not provide robust control of the BLDC motor. Sometimes the zero-crossing does not occur when expected. Sometimes the zero-crossing happens early and is missed. The inductive notch, PWM noise, and changing motor dynamics all contribute to occasional errors.

A more robust system requires *predicting* when to commutate the motor. The prediction should be based on the known motor speed. The predicted commutation time is used to calculate a time window for the zero-crossing signal. This time window is used to evaluate the validity of the back-EMF voltage zero-crossing. If the zero-crossing occurs within the valid window, the zero-crossing is used to calculate the commutation normally. If the zero-crossing does not occur when expected, the motor is commutated using the predicted commutation time.

The first order prediction is based on the last period. This works fine if only one zero-crossing is missed. The predicted time can be improved by providing correction based on if the zero-crossing is early or late. The software uses different correction equations to predict the next commutation time depending on the validity of the zero-crossing.

Startup Operation

When the motor is stationary, there is no back-EMF and the motor position is unknown. The motor position might be at any one of the six possible commutation states. A special procedure is required to start the motor. The software uses a different control method for starting and acquisition than the normal running mode. There is, of course, also a stop mode. The motor control modes of operation are illustrated in **Figure 4**.

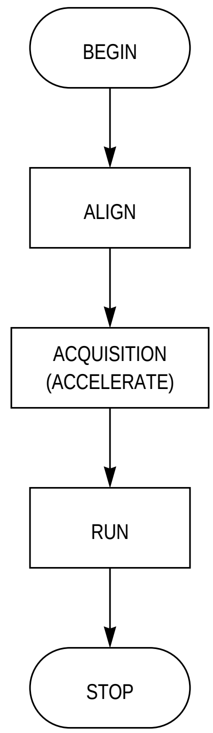

Figure 4. BLDC Simplified Startup Algorithm

The rotor of the BLDC motor is aligned by setting the commutation to a known state and applying a small voltage to align the motor. If a large voltage were applied to the motor, the current might easily exceed the ratings of the power transistors. If the voltage is too small, the motor will not align. A current mode PI controller has been implemented in software to provide maximum alignment torque. The alignment mode is a fixed duration of 300 ms, although this value can be modified according to different applications.

In most applications, it is acceptable if the motor turns slightly in either direction before starting. In piston compressor applications, it is desired to align the piston at top dead center. This will provide one full revolution for acceleration. The starting position is not usually important for fan or centrifugal pump application. Some applications such as disk drives require a different algorithm to prevent counter-rotation.

Following the alignment stage, the motor is accelerated according to a predefined acceleration curve. The acceleration curve is preset to provide the desired performance for the application. The software uses a simple S-shaped acceleration profile. During this phase, the back-EMF zero-crossing is evaluated according to the predicted commutation time. The correction factors are modified during the acceleration phase. The goal is to provide fast acquisition of the rotor position and achieve a lock. Once a positive lock is achieved, the software uses the normal running algorithm and correction factors.

Software

The software provides two modes of operation:

- Manual mode
- PC-Master mode

Manual and PC-Master Modes

The manual mode provides start/stop operation using switches and the potentiometer can be used to vary the speed of the motor. Start/Stop switch SW3 on the MR32 control board provides both start and stop functions. Once started, the potentiometer provides speed control over the full range of the motor. When the potentiometer is turned up all the way, full voltage will be applied to the motor and the motor will run at maximum speed. If the speed control is turned down too low, the motor will lose synchronization and stop. An error will be indicated with a blinking red LED. The motor can be restarted by turning up the potentiometer and toggling the start/stop switch off and on.

The software checks for PC-Master mode at startup. The start/stop switch must be in the stop position at power-up. PC-Master Communication Software is intended to be used as an aid in developing motor control software. All required actions of the motor control software are manipulated by the operator when using the PC-Master software. The PC-Master software executes on a PC that is connected to the isolated RS-232 serial port on the control board. The PC-Master software executing on a PC uses Microsoft Internet Explorer as a user interface to the PC. A small program is resident in the MR32 that communicates with the PC-Master software to parse commands, return status information to the PC, and process control information from the PC.

For the latest information regarding the PC-Master software, refer to the Freescale, Semiconductor Products Sector, Motor Control web page: http://freescale.com/semiconductors/motor.

For the latest application note software, refer to the following web links:
http://www.mcu.freescale.com/dev_tools/appsw.html
http://freescale.com/semiconductors/motor
http://freescale.com/semiconductors

Program Flow and State Diagram

The software uses both a linear process flow and interrupt service routines which occur at variable time intervals. This makes the development, debugging, and documentation more complex than most software.

The program flow for the main function is largely linear procedural and is shown in flowchart form in **Figure 5**. First the PLL (phase-locked loop), ports, ADC (analog-to-digital converter), and timers are initialized. After initialization, the interrupts are enabled. The interrupt service routines for the PWMMC module and output compare function might occur at any time after enabling the interrupts.

Next, the start function is called, which controls the motor operation during the startup phase. Once startup is complete, the start function will return to the main loop and the run function will be called. The run function controls the motor operation during the normal running mode.

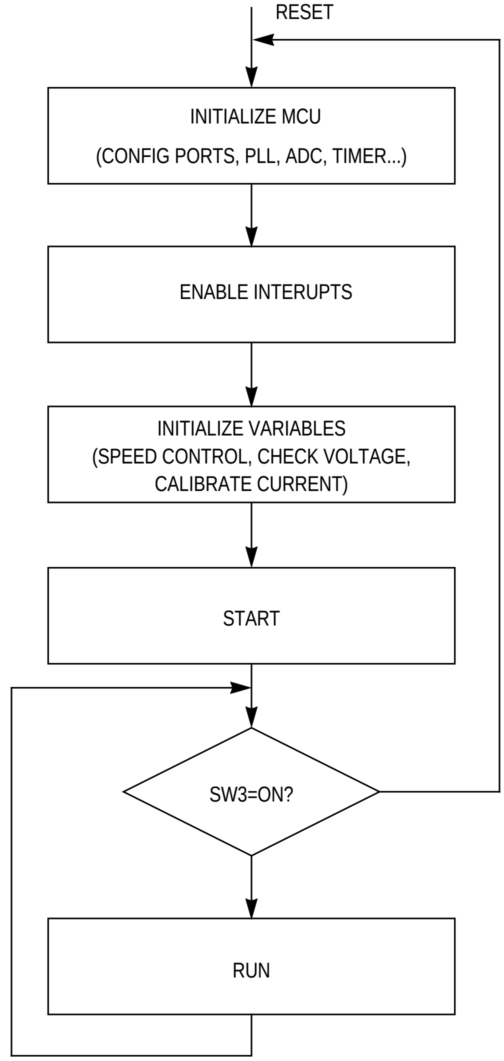

Figure 5. Main Program Flowchart

The program flow is made more complex by the addition of error handling capabilities. A complete state diagram including the error handling capabilities is shown in **Figure 6**. A commutation error might occur during the acquisition or running state. An overvoltage or gross overcurrent fault might occur at any time.

If a fault occurs, the PWM outputs will be disabled. Turning the switch off and on again will cause the software to restart from the beginning.

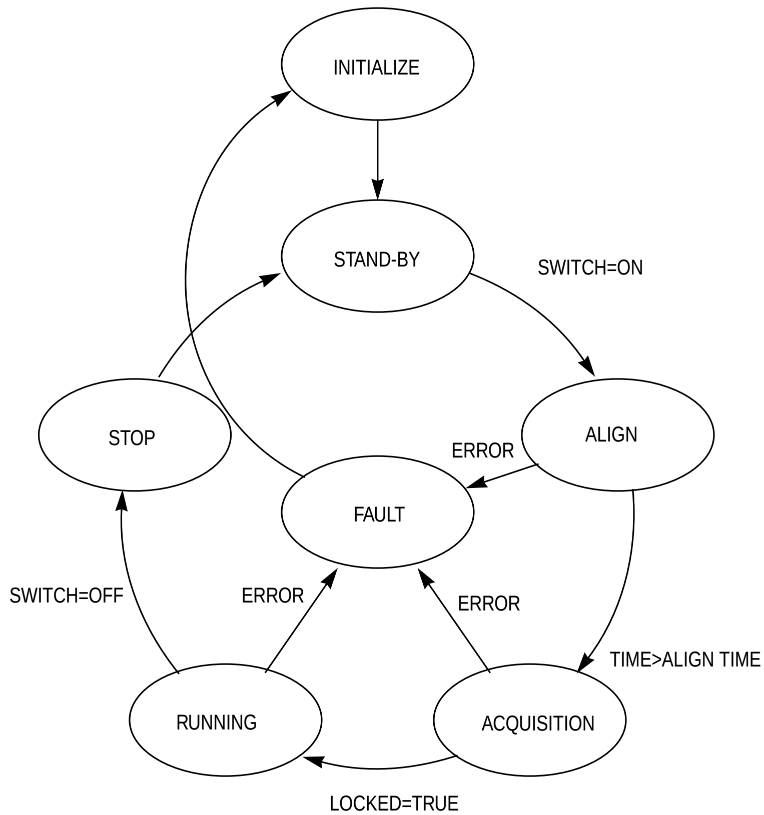

Figure 6. Main Program State Diagram

Interrupt Service Routines

Many of the motor control functions are synchronized to the PWMMC interrupt. During the PWMMC interrupt service routine, the back-EMF signals are analyzed. The back-EMF sampling is performed at the center of the center-aligned PWM signal. This helps to minimize noise on the back-EMF signal which is coupled from the active PWM phases.

A single output compare function is used to schedule the output commutation. Once the motor has been commutated, a flag is set to provide handshaking with the procedural start and run functions.

How to Modify System Parameters

Most of the motor-specific constants are defined in a file called const_cust.h. Some of the constants which might need to be changed are startup current (Align_Curr), startup commutation period (St1_T2P), and the ramp constant (C_rmp). Any of the variables stored in RAM may also be modified using the PC-Master software.

System Hardware

The system used for sensorless brushless dc motors consists of three boards:

- MR32 control board
- Optoisolation board
- 3-phase ac/BLDC high-voltage power board (power board)

Each of these boards comes complete with a user's manual. Refer to the respective user's manual for a detailed description of each board. This application note includes information on the overall system, a basic description of each board, and highlights of some of the circuitry used for brushless dc motors.

These modular development tools were developed to provide motor control solutions for many different kinds of motors. All of the different boards use a common, unified, 40-pin connector. This is a third-generation connector and provides interface signals for sensorless BLDC motor control as well as sensorless vector control of ac induction motors.

Precautions

These boards are intended to be used in a laboratory or suitcase demonstrations. When used in the laboratory, most users will want to connect a host PC to the system. The PC should always be earth grounded as a safety precaution. A battery-powered notebook is often used in the laboratory and provides an additional measure of safety and robustness.

In the end application, a high-voltage BLCD motor operates from the rectified ac power line. This presents safety issues for developers.

WARNING: ***The high-voltage motor control boards should be used only by qualified engineers and technicians with experience in high-voltage power systems. The motor control system described in this application note is capable of operating at dangerous voltages and is capable of supplying high amounts of power to rotating machines. The high-voltage board should be powered down before changing probe connections. Wearing safety glasses, avoiding ties and jewelry, and using shields are also advisable.***

Common practices for ESD (electrostatic discharge) protection can present potential safety hazards when working with high-voltage circuits. Some precautions are:

- The power board should be placed on a wood or plastic table.
- Do not use a conductive table or conductive ESD mat.
- Do not use grounded chairs.
- Beware of grounded floors and wear rubber shoes.
- In general, eliminate all ESD provisions from the high-voltage work area.

The recommended laboratory setup is shown in **Figure 7**. Recommendations for the laboratory setup include:

- An isolation transformer should be used for the power board. This is essential when taking measurements of the actual voltages and currents.
- The variac is used to vary the ac line voltage.
- The isolation transformer should be properly fused for the particular load.
- The power circuit should be allowed to float with respect to earth ground.
- A digital oscilloscope should normally be earth grounded.
- Portable oscilloscopes and digital multimeters provide an additional measure of safety.

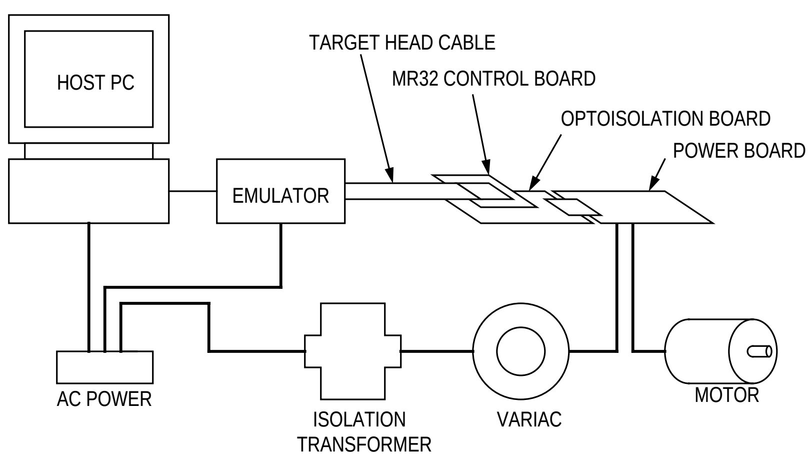

Figure 7. Laboratory Setup

If available, a high-voltage lab supply also can be used to power the power board. A dc supply can be connected directly to the input rectifier of the power board. The polarity of the supply does not matter. While expensive, a lab supply provides better protection against overcurrent faults. Make sure the lab supply provides an isolated output voltage.

The isolation board allows the PC to be connected directly to the MR32 control board. This is necessary when connecting a target head adapter cable from the emulator to the MR32 control board. The isolation board provides safe isolation for all of the 40-pin connector signals.

The software code should be developed and debugged using the emulator connected to the MR32 control board. This may be done using just the MR32 control board. The power board may be disconnected or not powered up. Once the software has been tested using the MR32 board, it can then be tested with a motor using the laboratory setup shown in **Figure 7**.

After the developer is satisfied with the performance of the software, a FLASH MR32 can be programmed. The FLASH device is inserted into a daughter board and connected to the MR32 control board in place of the target head cable.

The isolation board is intended for development purposes. More cost-effective solutions are available for providing isolation in the end user application. The isolation board is necessarily more complex due to the need to support many motor types. Many low-power applications will not require optical isolation in the final end system. Non-isolated systems must provide safe insulation by mechanical means, enclosing the whole system in a plastic housing and providing safe, insulated user controls.

MR32 Control Board

The MC68HC908MR32 control board is designed as an aid for hardware and software design of 3-phase ac, brushless dc, and switched reluctance motor drive applications.

The control board does not contain the MR32 MCU. The control board is designed to be connected directly to an MR32 emulation module (EM) board, which is part of a Freescale Modular Evaluation System (MMEVS) or Freescale Modular Development System (MMDS). A daughter board is designed to house the MR32 MCU and plugs into the control board in place of the emulator cable. With the daughter board plugged into the control board, standalone operation of the system is possible.

Since this application note is intended for use with a brushless dc motor, only the circuitry resident on the MR32 control board pertaining to the 3-phase brushless dc motor is discussed. Applications of the control board with other types of motors are covered in additional application notes. **Figure 8** shows a complete block diagram of the control board.

Control board features include:

- Six motor control PWM outputs with LED indicators
- Speed control potentiometer
- Optoisolated half-duplex RS-232 interface
- Start/Stop and forward/reverse switches
- Hall effect inputs for brushless dc motor control
- Back-EMF inputs for brushless dc motor control
- Tachometer input configuration jumpers
- 2-position DIP (dual in-line package) switch for user option control
- Emulator/Daughter board connector
- Processor reset switch
- Two system fault inputs
- Nine analog inputs
- Three software-controlled LEDs
- On-board regulated power supply
- Motor input/output (I/O) interface via 40-pin ribbon cable

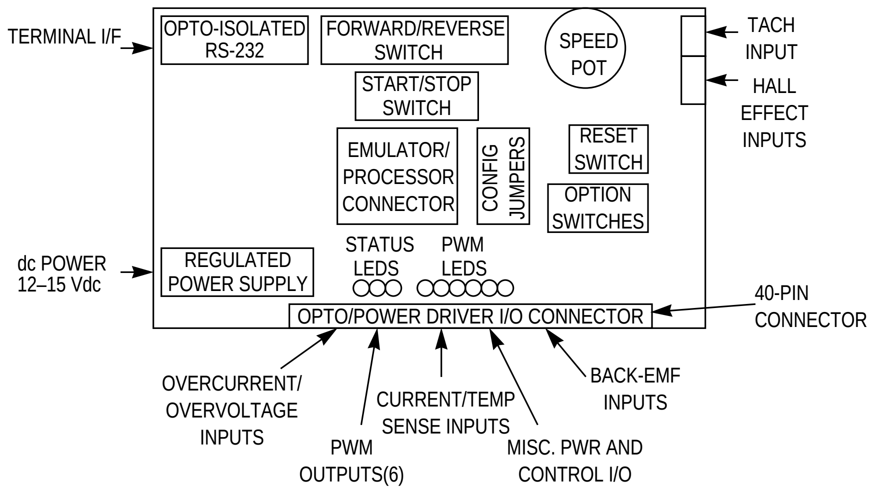

Figure 8. MR32 Control Board Block Diagram

Fault Control

Two system fault inputs to the control board are designed to protect the power board. The faults are system bus overvoltage and system bus overcurrent. The input signals for these fault comparators originate from signals on the power board. If an optoisolation board is used in the system, these signals are optocoupled and transparently passed to the control board as an analog signal. The comparator circuits provide digital signals to the MR32's fault 1 and fault 2 inputs, respectively. These faults, should one or both occur, will force the PWM generator into a known inactive state, protecting the power board outputs. **Figure 9** is a schematic of the circuit used for both of the fault inputs. The potentiometer, connected to the inverting (–) input of the comparator, sets its threshold. When the input from the power board or optoisolation board exceeds the comparator threshold (voltage at the inverting input to the comparator), the respective fault input to the MR32 is driven to a logic 1, triggering a fault input to the PWM generator. Adjusting the set point of the potentiometer allows the user to vary the acceptable system bus current and voltage thresholds for fault generation. Approximately 20 mV of hysteresis is included in the circuit to aid with noise immunity.

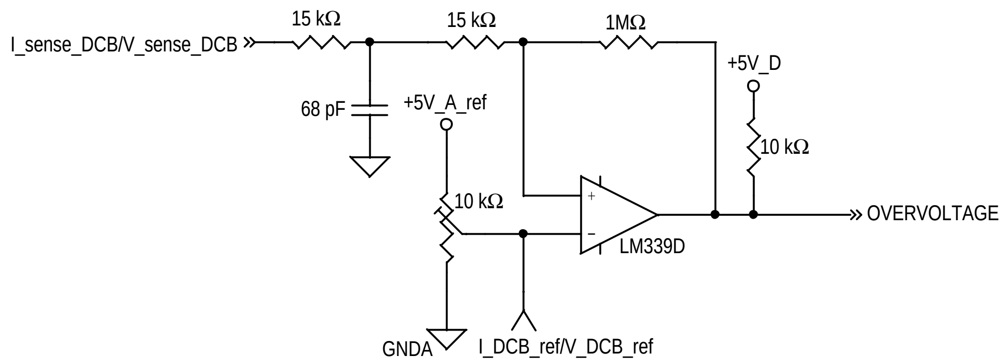

Figure 9. Fault Generation Circuit

Isolated Serial Port

The MR32 features an asynchronous serial port that can be used for a standard RS-232 interface. The MR32 can be used as a smart motor controller with all user commands and feedback communicated over a single serial link. An isolated serial port has been implemented on the MR32 control board to demonstrate this function in a low-cost system.

The circuit in **Figure 10** is the schematic of a half-duplex optoisolated RS-232 interface used on the MR32 control board. The EIA RS-232 specification states the signal levels can range from ±3 volts to ±25 volts. A mark is defined by the EIA RS-232 specification as a signal that ranges from –3 volts to –25 volts. A space is defined as a signal that ranges from +3 volts to +25 volts. The left half of the circuit provides signal inversion and level shifting for the PC serial port. This section uses a clever diode circuit to provide the voltage levels needed for RXD returning to the PC. The right half of the circuit provides signal inversion and level shifting from the MR32's SCI serial port. An RS-232 line driver, such as an MC1488, serves the same purpose without the optoisolation function.

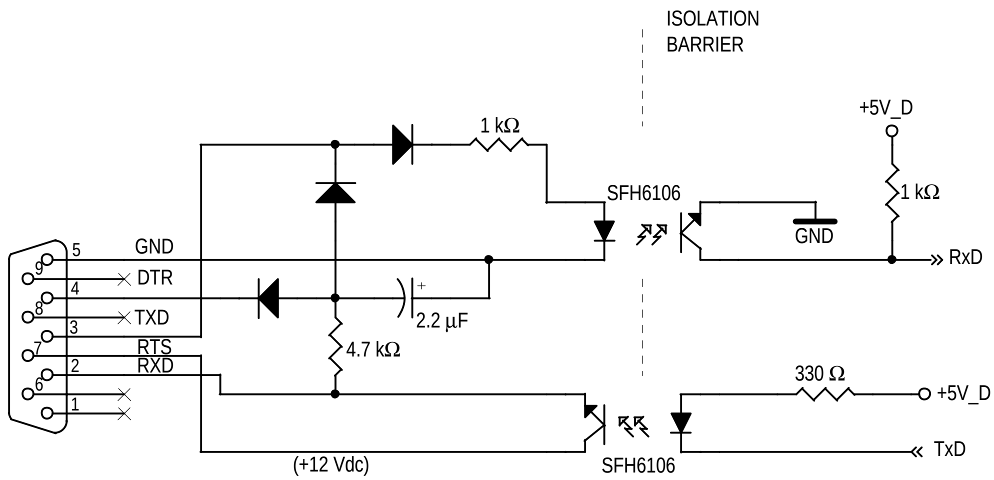

Figure 10. Optoisolated RS-232 Circuit

Back-EMF Multiplexer

The MR32 control board contains a multiplexer for the back-EMF signals. The three zero-crossing signals originate from the power board and are discussed later. These zero-crossing signals are passed through the optoisolation board, when isolation is required.

The microcontroller software is in command of the motor commutation. It always knows what motor phase is open. This information is used to select the proper zero-crossing signal for each commutation state. The three multiplexer signals are connected to output port pins on the MR32. The multiplexer signals are changed each time the motor is commutated.

The three zero-crossing signals from phases A, B and C are routed into the back-EMF selection logic as shown in **Figure 11**. The back-EMF selection logic is designed to combine the three zero-crossing signals into a single signal that is fed into an input capture timer channel. The three open collector NAND gates shown in **Figure 11** are wire ORed such that any one of these outputs changing to a logic 0 will provide an interrupt to the MR32's timer A channel 2. The system software uses the MUXA, MUXB, and MUXC inputs to the NAND gates to enable a particular phase to have an ability to interrupt the processor.

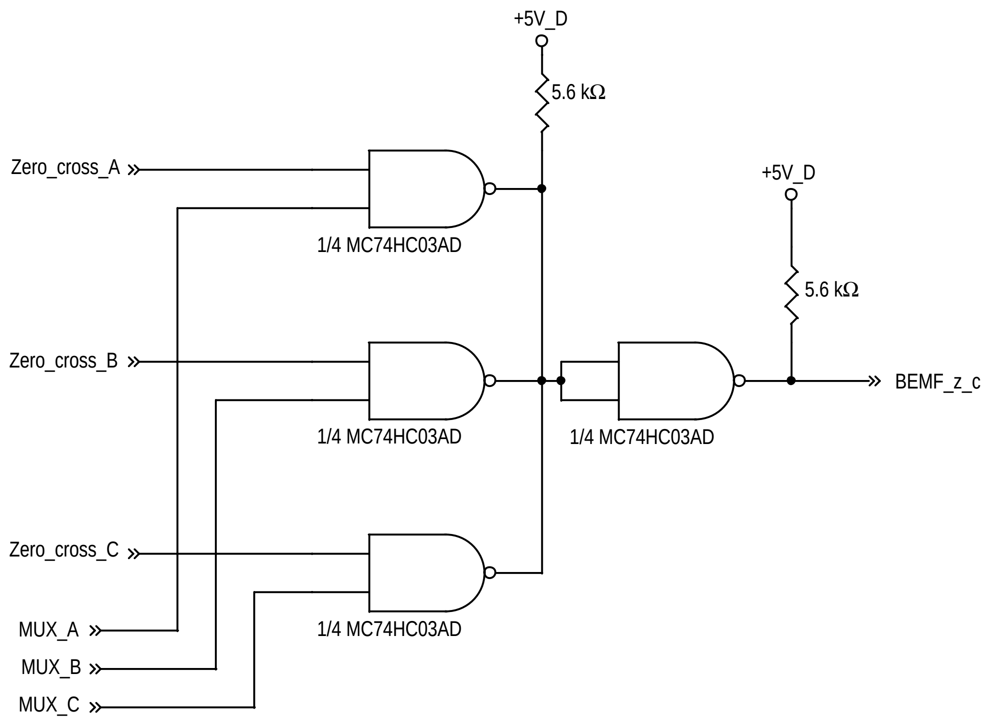

Figure 11. Zero Cross and Back EMF Circuit

Optoisolation Board

The function of the optoisolation board is to provide a galvanic isolation barrier between the control board's I/O, both analog and digital, and the high-voltage system power board's I/O. These isolated signals, to and from the optoisolation board, are connected by two 40-pin connectors. Pin assignments for both connectors are the same. Signal flow through the optoisolation board, in both directions, is a one-to-one relation of its source. For a more detailed description of the optoisolation board, refer to *Motorola's Embedded Motion Control Series Optoisolation Board User's Manual, Freescale* document order number MEMCOBUM/D. **Figure 12** shows a block diagram of the optoisolation board.

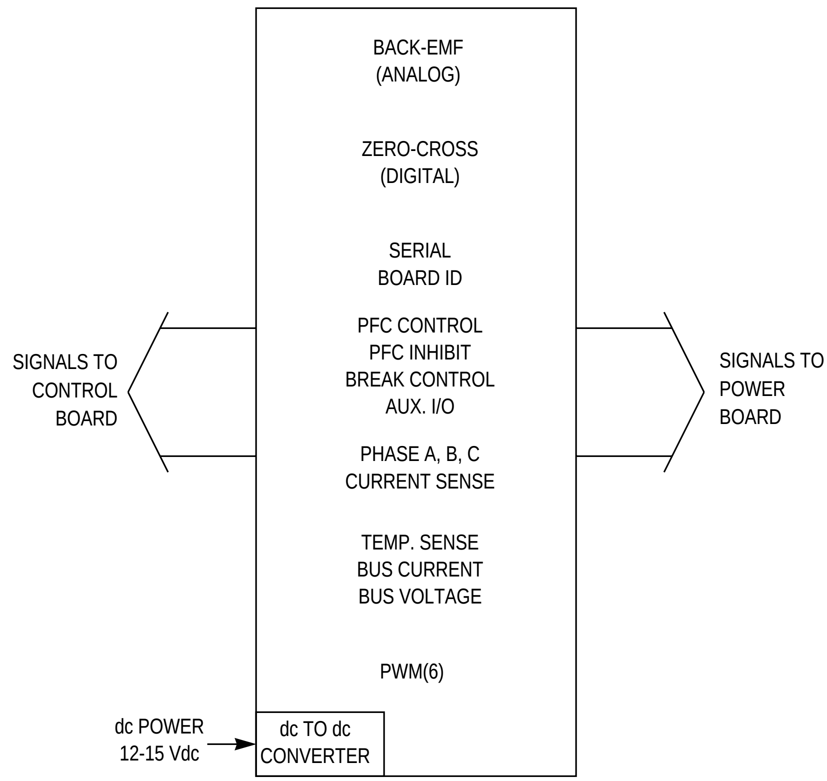

Figure 12. Optoisolation Board Block Diagram

Gate drive signals, from the control board to the power board, are passed from the controller to the power board through high-speed digital optocouplers. Analog feedback signals from the power board to the control board are passed through HCNR201 high-linearity analog optocouplers. Ground signals between the control board and power board are separated by the optocouplers' galvanic isolation barrier.

Power requirements for the control board's circuitry are satisfied with a single external 12-Vdc power supply, connected to the optoisolation board and fed to the control board through the 40-pin ribbon cable. Excitation for the power board side circuitry is supplied to the optoisolation board from the power board.

In addition to the usual motor control signals, an MC68HC705JJ7 serves as a serial link, which allows the control board's software to identify the configuration of the optoisolation board and power board.

3-Phase ac/BLDC High-Voltage Power Board

For a more detailed description of the MR32 power board, refer to *Motorola's Embedded Motion Control Series 3-Phase BLDC High-Voltage Power Stage User's Manual Freescale,* document order number MEMC3PBLDCPSUM/D.

The function of the power board is to provide the high-power drive circuitry for various types of motors. The power board is suitable for driving a wide variety of ac induction and brushless dc motors. A different board is available for switched reluctance motors.

The power board consists of a set of two separate modules. A printed circuit board contains the IGBT (insulated gate bipolar transceiver) gate drive circuits, analog signal conditioning, low-voltage power supplies, power factor control circuitry, and some of the large passive power components. This board also has an MC68HC705JJ7 microcontroller used for board configuration and identification.

A block diagram of the power board is shown in Figure 13.

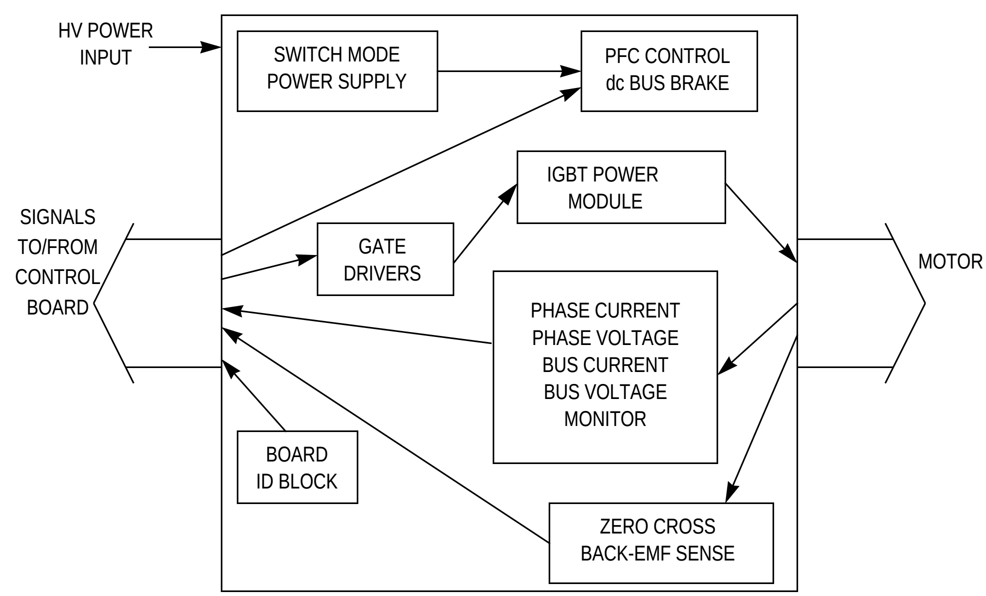

Figure 13. Power Board Block Diagram

All of the power electronics which need to dissipate heat are mounted on a separate power substrate. This substrate includes the power IGBTs, a brake IGBT, a power factor corrector MOS field effect transistor (MOSFET) and temperature sensing diodes. **Figure 14** shows a complete block diagram of the power module.

Power board features are:

- 1-phase bridge rectifier
- Power factor switch and diode
- dc bus brake IGBT and brake current limiting resistors
- 3-phase bridge inverter (six IGBTs)
- Individual phase and dc bus current sensing shunt resistors with Kelvin connections
- Power substrate temperature sensing diodes
- IGBT gate drivers
- Current and temperature signal conditioning
- 3-phase back-EMF voltage sensing and zero cross detection circuitry
- Board identification processor (MC68HC705JJ7)
- Low-voltage on-board power supplies
- Cooling fans

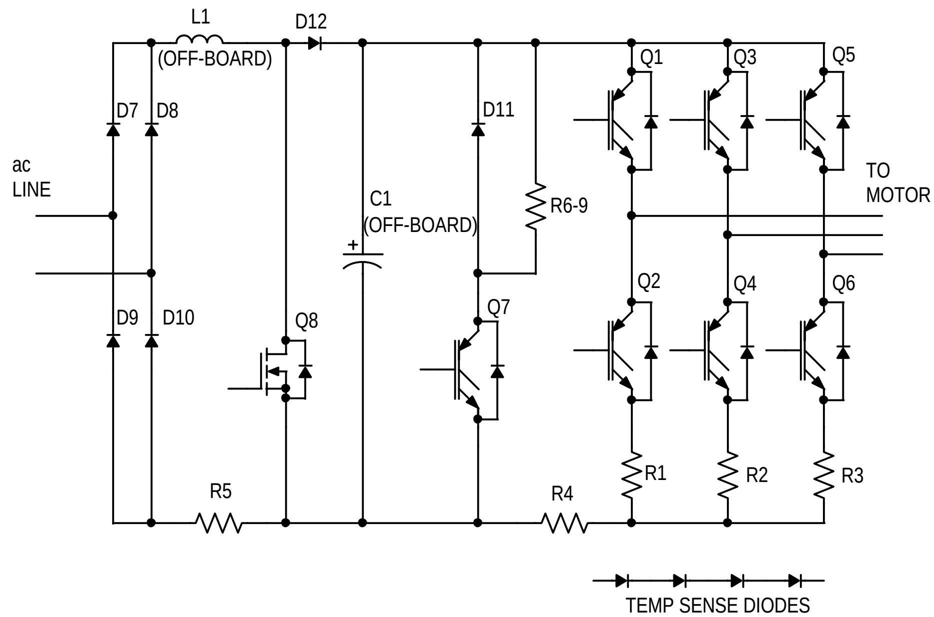

Figure 14. Power Module Simplified Schematic

Power Electronics

The power module provides a single-phase rectifier, optional boost converter, brake circuit, and 3-phase inverter. The single-phase rectifier permits the power board to operate directly off the ac line. An isolation transformer is recommended during development. The power board can also operate off a high-voltage dc lab supply. The optional boost converter can be used for power factor correction. The power factor correction circuit is not used for the brushless dc motor software, and the circuitry should be bypassed using jumpers as described in the power board user's manual. The power board is initially configured with the power factor correction circuitry bypassed. The 3-phase inverter uses IGBTs with integrated free-wheeling diodes. IGBTs are the preferred power device for high-voltage motor control. Power MOSFETs are used for low-voltage motor control and some very low-power high-voltage applications.

Current Sensing

Phase currents and dc bus are measured by sensing the voltage drop across sensing shunt resistors, located on the power module. **Figure 15** is a schematic diagram of the design of the dc bus analog signal conditioning circuit. The output voltage of the amplifier is proportional to the sensed currents. The input to the circuit in **Figure 15** is derived across a 0.075-Ω shunt resistor connected in series with the particular current being measured. The input signal to the amplifier is multiplied times 7.5 and then the output of the amplifier is shifted up by a 1.65-volt reference. The final output of the circuit is ±1.65 volts with an input current of 2.93 amps passing through the 0.075Ω shunt resistor.

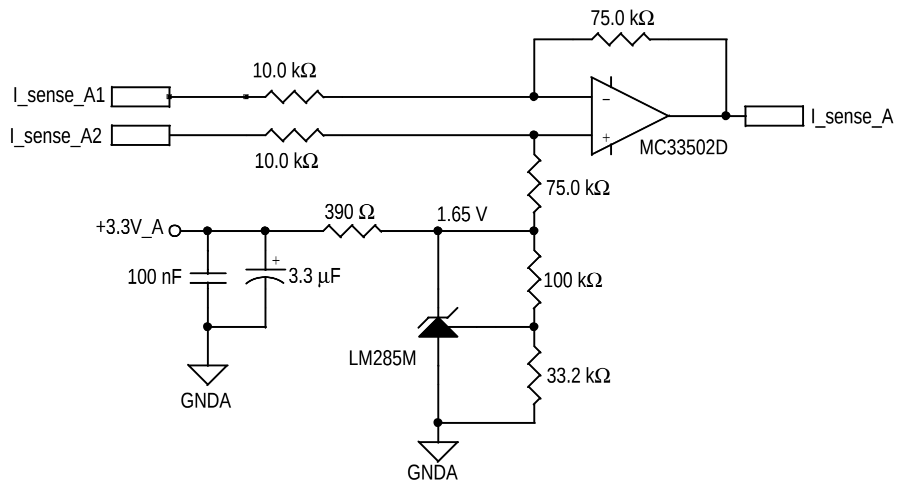

Figure 15. dc Bus Current Analog Signal Conditioning Circuit

NOTE: *Excessive dc bus current can destroy the power transistors on the power board. Including overcurrent protection circuitry in a design is a wise choice, if not an absolute necessity.* ***Figure 16*** *is the schematic of the dc bus overcurrent detection circuit. The input signal to this circuit is a 0.075-Ω resistor, placed in series with the dc bus and amplified by the same type circuit as shown in* ***Figure 16****. The output of this circuit shown in* ***Figure 16*** *drives the shut down input to the gate drivers.*

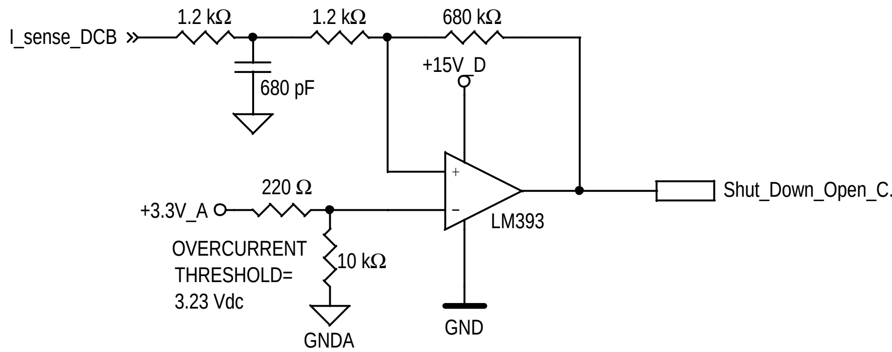

Figure 16. dc Bus Overcurrent Detection Circuit Schematic

Brake Circuit

Under certain operating conditions, a motor can act as a generator, delivering high voltage back into the dc bus through the inverter's power switches and/or the power switch source-drain recovery diodes. That is a very undesirable condition and can damage the power transistors and other components in the inverter. The excess energy must be dissipated, otherwise the dc bus voltage will rise above a safe limit. The power module contains an IGBT and current limiting resistors, which are placed across the dc bus to act as a dc bus brake and dissipate the excess energy. When using the brake, be careful to not exceed the power dissipation of the brake transistor and its current limit resistors. Provisions are made on the power board for the user to install an additional brake resistor across the one composed of R6-R9, allowing for additional bus brake current to be imposed on the system. Again, be careful not to exceed the ratings of the IGBT brake transistor when an additional brake resistor is installed in the system. Typically, the system software will pulse-width modulate the brake to dissipate the excess voltage until it is brought down to an acceptable level.

CAUTION: *Under certain operating conditions, a motor can act as a generator, delivering high voltage back into the dc bus through the inverter's power switches and/or the power switch source-drain recovery diodes. This can damage the power transistors and other components in the inverter.*

dc Bus Sensing

The system software must monitor a number of analog parameters when the motor is running. Those parameters include the dc system bus voltage and the three individual phase voltages. In all four cases, the high voltage is divided down to a level within the measurable range of the MR32's analog-to-digital converter (A/D). **Figure 17** is a schematic of the voltage divider used for monitoring the dc bus voltage. The signal labeled V_sense_DCB_5 is the divided down dc bus voltage that is fed to A/D of the microcontroller either directly or via the optoisolator board, if one is utilized in the system. The signal labeled V_sense_DCB_half_15 is used as a reference for the individual phase A, B, and C zero cross detection circuits.

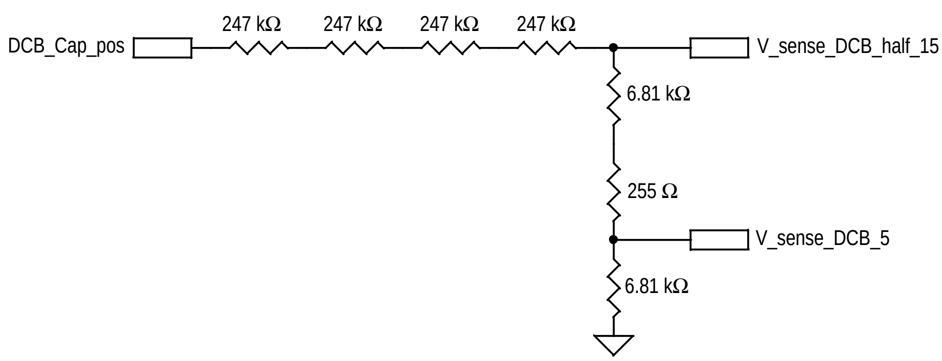

Figure 17. dc Bus Voltage Sensing Circuit

Back-EMF Comparators

In a similar fashion to the dc bus sensing circuit shown in **Figure 17**, the individual phase A, B, and C voltages are divided down to match the input level of the A/D. **Figure 18** shows a schematic of the voltage divider for an individual phase A monitoring circuit. There are additional circuits for voltage monitoring and zero cross detection of phases A and B. This technique allows sensing of the back-EMF from the motor. The individual phase voltage signals are fed to separate A/D inputs of the microcontroller either directly or via the optoisolator board, if it is utilized in the system. The signal shown is labeled BEMF_sense_A (BEMF_sense_B or BEMF_sense_C, depending on the particular phase signal).

An additional function of the circuit shown in **Figure 18** is to detect zero-crossing of phases A, B, and C. Note that the inverting input of the comparator is set by a reference from the dc bus divider (V_sense_DCB_half_15). Using the dc bus divider as a reference, and when phases A, B, or C phase voltages reach 50 percent of the dc bus signal, corresponding to the phase zero cross point, the output of the comparator will transition from a logic 0 to a logic 1. That transition is used for distortion correction information to the IS1–IS3 inputs of the PWM generator and as an input to the zero cross window logic for zero cross interrupt generation.

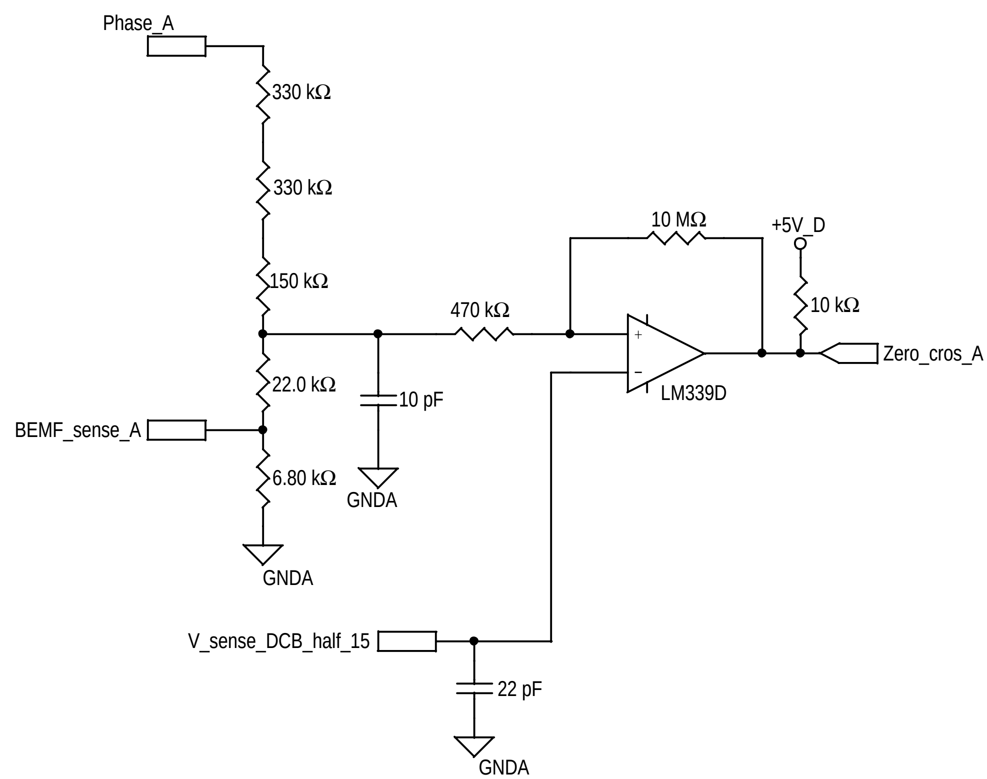

Figure 18. Back-EMF and Zero Cross Detection Schematic

The power factor correction circuitry is not used by this software. Power factor correction control (PFC) circuitry provides control of the PFC switch and handles the necessary feedback to provide a sinusoidal power line current. The capability of PFC can be enabled or disabled by changing a jumper configuration on the power module. The jumper can be found in proximity to the dc bus capacitor. The power board is shipped with the power factor correction circuitry disabled.

Refer to *Motorola's Embedded Motion Control Series 3-Phase BLDC High-Voltage Power Stage User's Manual*, Freescale document order number MEMC3PBLDCPSUM/D, for information on how to reconfigure the power factor correction circuitry.

Conclusion

This application note describes a software and hardware solution for sensorless brushless dc motors. The software algorithm provides starting and variable speed control for a variety of applications. The hardware is based on a modular development system and the key circuits have been highlighted.

The embedded motion control development system can be further used to tailor the software or to create new software for a particular BLDC motor application. This development system provides support for many different kinds of brushless dc motors as well as other motor types.

The MC68HC908MR32 is well suited for all kinds of brushless dc motor applications. It has a rich set of peripherals and features a PWMMC module expressly designed for motor control.

Application Note

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

freescale™ semiconductor

AN1858/D