AN1702

Brushless DC Motor Control Using the MC68HC705MC4

By John Deatherage and Jeff Hunsinger
CSIC Systems Technology
Austin, Texas

Introduction

This application note details the design and analysis of a brushless DC motor control system using the MC68HC705MC4 with two evaluation boards available from Motorola. For many years, brushed DC motors have been popular partly because of minimal requirements for electronic control. The trade-off between electromechanical commutation and efficiency has traditionally leaned in favor of the former (electromechanical commutation). Today, however, the popularity of reasonably priced, electrically commutated, brushless DC motors is rising as is the need for electronic motor control. In particular, brushless DC motors (also called permanent magnet motors) are found in computers (disk drives), households (appliances), and automobiles (fans and body controllers), among other applications. Designers of these types of motor control systems are confronted with requirements of electronic commutation, variable speed control for energy efficiency, communication to outside nodes (distributed control, diagnostics, etc.), and flexibility at little or no extra cost. The MC68HC705MC4, from Motorola's HC05 Family of microcontrollers, provides a low-cost, highly integrated, flexible platform for brushless DC motor control.

Brushless DC Motor Tutorial

Reviewing a few basic points about brushless DC motors will assist in understanding them. First, the rotor (rotating part) of a brushless DC motor is fitted with cylindrical magnets and its stator (stationary part, usually bolted down) consists of several (typically four, six, or eight) poles which project out from the stator perpendicular to the rotor. For a 3-phase motor, the poles are wrapped with windings which are symmetrically grouped in three sets around the stator and then connected in a delta or wye configuration. Thus, when current is injected through two of the motor's phase windings an electromagnetic force will cause the magnetic rotor to partially rotate (remember the "right hand rule"). In addition, most brushless DC motors have sensors (Hall Effect or optical) built into the housing of the motor to sense the position of the motor shaft. In this fashion, the motor can be rotated by sensing the position of the rotor and feeding signals into two of the three phases of the motor's (see **Figure 1** and **Figure 2**) stator coils. This will cause the rotor to rotate (30 degrees for a motor with three sensors) to its next position, which in turn will be sensed.

NOTE: *The control sequence which generates the output waveform is called electronic commutation (hence the term "brushless" motor) and is typically implemented by a state machine or a microcontroller unit (MCU).*

NOTE: *Heat (wasteful energy) is generated in the stator of a brushless DC motor and can escape easily from the motor compared to motors with rotor windings (which is why brushless DC motors are much more efficient than brushed DC motors).*

Once the motor is rotating, speed and torque control of the motor need to be considered. One method of closed loop speed control under steady-state operating conditions (the motor already is running at a constant speed) involves an interrelationship between the magnitude of voltage across the stator coils and the speed of commutation. For example, a brushless DC motor will rotate faster by increasing the voltage magnitude of the waveform fed into its stator coils. As the motor rotates faster the controller must increase the speed of its commutation

sequence. Thus, the maximum motor speed can be limited both by the maximum voltage rating of the motor and the ability of the controller to commutate at high speeds. On the other hand closed loop torque control is related directly to both the magnitude of magnetic flux (which is constant with a permanent magnet motor) and the magnitude of current fed into the motor's coil windings. One method of torque control can be more easily understood by observing this steady state equation for a motor circuit using Kirchoff's Voltage Law:

$$V = IR + E$$

where **V** is the voltage across the stator coils, **I** is the current through the stator coils, **R** is the resistance of the coils, and **E** is the electromotive force (EMF) generated by the rotating motor. Therefore, the steady state torque of a brushless DC motor can be controlled by sensing its back EMF and thus adjusting the voltage across its coils to control the coil current. From this discussion, it becomes obvious that torque and speed are related, but keep in mind that although speed control will affect the torque of the motor, it does not imply torque control. A pulse width modulation (PWM) method of speed control with interrupt-driven timer feedback is presented in this application note. Torque control will not be discussed any further.

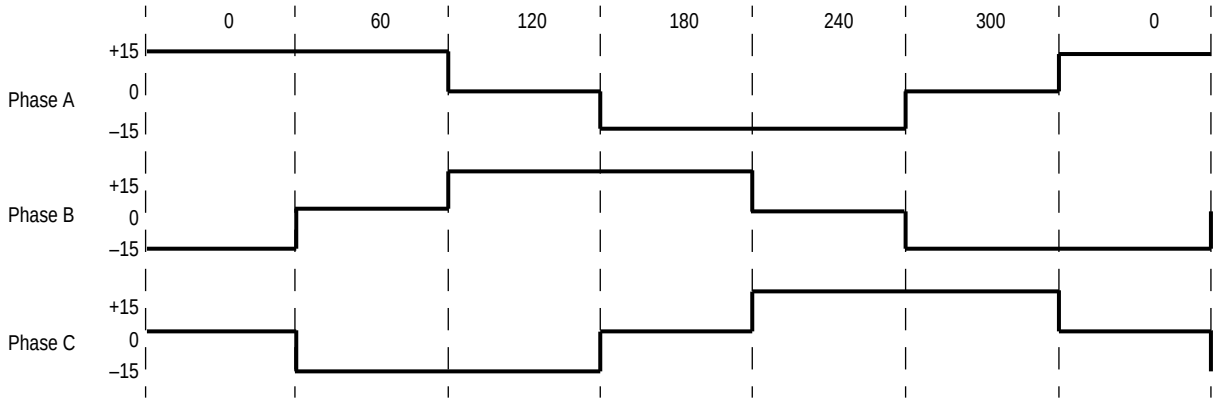

Figure 1. 3-Phase Motor Waveform, 100% Duty Cycle, Full Speed

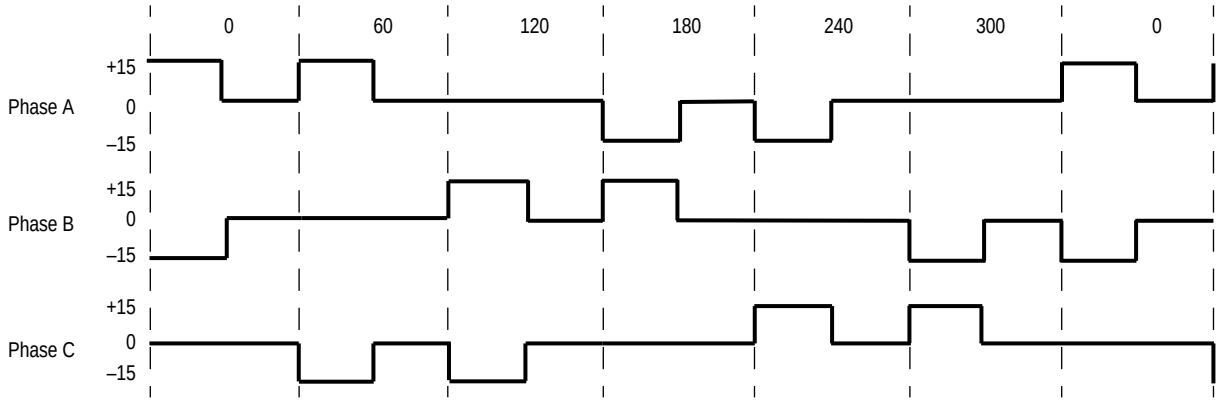

Figure 2. 3-Phase Motor Waveform, 50% Duty Cycle, Half Speed

Several methods can be used to control variable speed, brushless DC motors. Frequently, these motors are driven by an inverter circuit, consisting of complementary pairs of drivers (typically one pair of FETs or IGBTs per phase) which are controlled by PWM signals from an MCU. An inverter is defined as a circuit which is powered by a DC input, and in conjunction with a control (commutation) algorithm acts to create crude AC voltage output signals (Crude voltage signals can actually generate smooth current signals.) to rotate a motor's shaft. The commutation algorithm will assure that the motor's coils are injected alternately with current in a sequential and repetitive fashion. The voltage magnitude across the motor's coils, controlled by the duty cycle of the PWM signals, will control its rotor speed (see **Figure 1** and **Figure 2**). In addition, the sensors built into brushless DC motors feed back the angular position of the motor to the controller (MC68HC705MC4). The position sensors (typically 3 or 4) allow the controller to commutate the motor properly and to monitor the motor's actual speed. The motor chosen for this application note is a 3-phase, 6 pole, brushless DC motor with three Hall effect sensors.

The commutation sequences for clockwise and counterclockwise rotation are shown in **Table 1** and **Table 2**. **Table 1** shows how the 6-step sequence will mechanically rotate the motor 180 degrees in a clockwise direction. The sequence must be repeated twice in succession to rotate the motor completely around. It is helpful to note that the left columns of **Table 1** (labeled Sensor 1, Sensor 2, and Sensor 3) are

inputs to the MCU and the right columns (labeled Phase A, Phase B, and Phase C) are controlled by a mixture of PWM and logic level outputs from the MCU. The rotation speed of the motor is controlled by adjusting the duty cycle of the PWM signals (see **Figure 1** and **Figure 2**) which are fed into the bottom side (low side) FETs (field effect transistor) of the inverter. The period of the PWM signals does not directly affect the speed of the motor, but is important when choosing a control algorithm. For example, a higher speed PWM can increase the effective resolution of speed control and can reduce the amount of audible noise from the motor with the trade-off of higher switching losses in the inverter. Weighing these trade-offs, choosing a PWM control frequency just above the audio range is often desirable.

Table 1. Commutation Sequence for Clockwise Rotation

Rotation Angle in Degrees	Sensor 1	Sensor 2	Sensor 3	Phase A	Phase B	Phase C
0 & 180	1	0	0	+15	–15	NC
30 & 210	1	1	0	+15	NC	–15
60 & 240	0	1	0	NC	+15	–15
90 & 270	0	1	1	–15	+15	NC
120 & 300	0	0	1	–15	NC	+15
150 & 330	1	0	1	NC	–15	+15

Table 2. Commutation Sequence for Counterclockwise Rotation

Rotation Angle in Degrees	Sensor 1	Sensor 2	Sensor 3	Phase A	Phase B	Phase C
0 & 180	1	0	0	–15	+15	NC
–30 & –210	1	0	1	NC	+15	–15
–60 & –240	0	0	1	+15	NC	–15
–90 & –270	0	1	1	+15	–15	NC
–120 & –300	0	1	0	NC	–15	+15
–150 & –330	1	1	0	–15	NC	+15

Under “no load” conditions, motor speed control simply amounts to following the commutation sequence at a constant PWM duty cycle. However, a few conditions occur that will complicate the motor control algorithm. First, the presence of a load usually will produce a different motor speed than desired. Thus, a closed loop algorithm such as a PID (proportional, integral, derivative) control loop is required to maintain constant or predictable speeds under acceptable load conditions for the motor. Such an algorithm will continuously compare the desired motor speed with the actual motor speed, derived by an MCU timer using the sensor input interrupts from the motor, and will gradually correct the system output (PWM duty cycle) accordingly. The PID control loop used in this experiment is detailed in a later section.

Stall is another condition that will complicate a motor control algorithm. Stall occurs when the motor is starting up or when a sudden heavy load is placed on the motor and it completely stops. The control algorithm must sense and correct for stall by creating a steady increase of current (increase the PWM duty cycle) over a period of time to the motor's coils or alternatively shut the motor off under extreme conditions. Additional conditions such as shoot through prevention, dead time generation, and current feedback for torque control also should be considered, but are outside the scope of this application note.

System Overview

A simplified version of the hardware system used to demonstrate brushless DC motor control is shown in **Figure 3**. The system includes a KITITC127 MC68HC705MC4 motion control development board (available from Motorola), a KITITC122 low-voltage MCU to motor interface module, a 3-phase brushless DC motor, and a split DC power supply. The MC68HC705MC4 integrates several features for motor control including a high-speed (up to 23.4 kHz), 2-channel, double-buffered PWM module (eight bits of resolution) with a commutation multiplexer (three pins per PWM channel), which allows for a flexible interface to motors, and an interlock mechanism for coherent updates. Other key features include a 6-channel analog-to-digital (A/D) converter (eight bits of resolution) which can be used for measuring the speed,

position, or back EMF of the motor; an asynchronous serial port (SCI) for communications outside of the motor control system; and a 3-MHz bus (333 ns instruction cycle) which will allow the MC68HC705MC4 to efficiently control the motor at higher speeds. (See **Motor Control Software Analysis**.)

Following a brief hardware overview, the remainder of this application note will focus on the commutation scheme and software used by the MC68HC705MC4 to control the brushless DC motor shown in **Figure 3**.

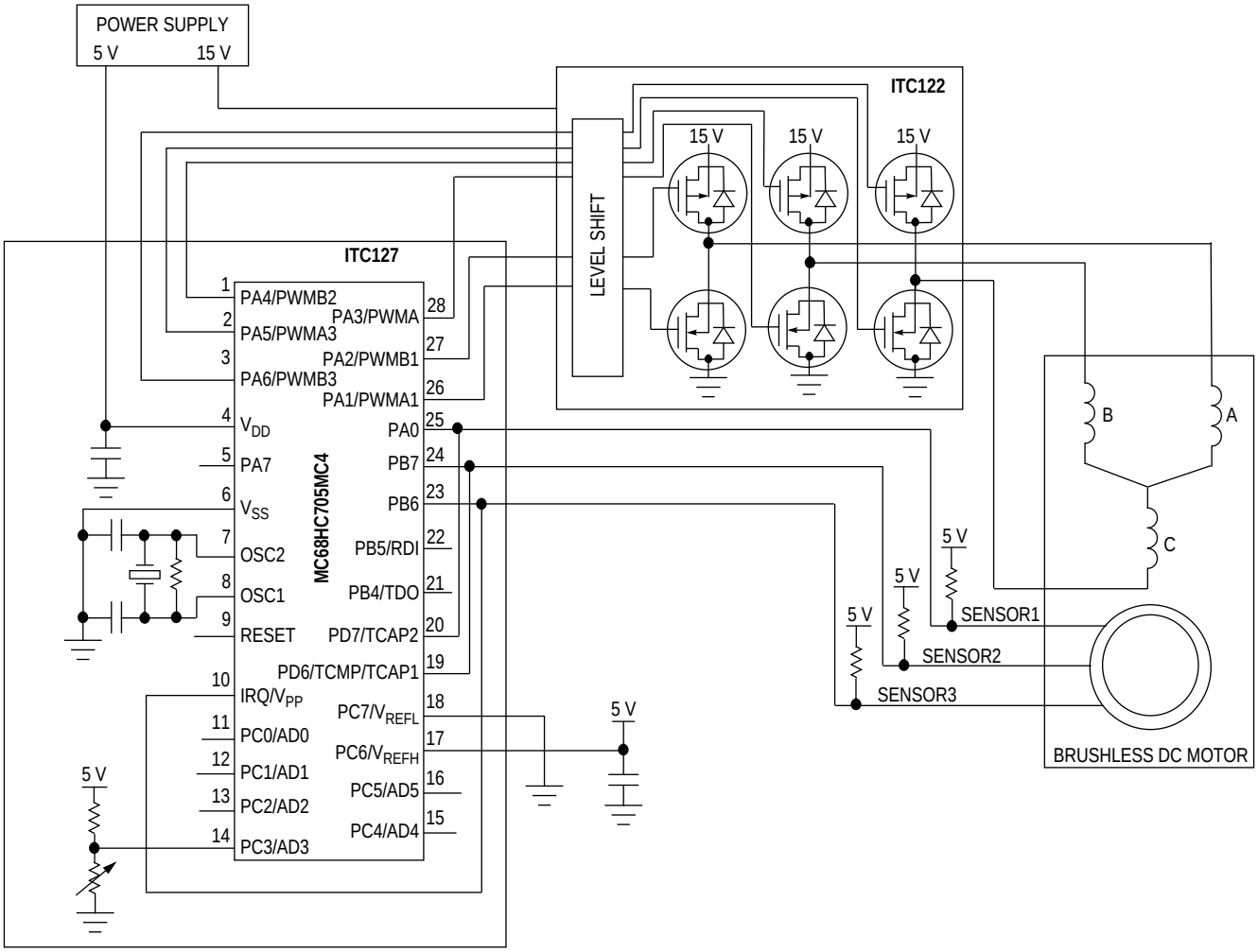

Figure 3. MC68HC705MC4 Motor Control Circuit

Brushless DC Motor Control Hardware

As shown in **Figure 3**, the interface for this experiment is quite simple. On the KITITC127 board, a 6-MHz crystal across the OSC1 and OSC2 pins will allow the MC68HC705MC4 to output a 23.4-kHz PWM signal with eight bits of resolution. The motor's three position sensors are directly connected to the MC68HC705MC4's IRQ, TCAP1, and TCAP2 pins to cause separate, time stamped interrupts (for measuring actual motor speed) to the MC68HC705MC4 as the motor rotates. In addition, pins PA0 and PB6–PB7 are used to read the position sensor inputs which allow the MC68HC705MC4 to commutate the motor properly.

NOTE: *The position sensor pins can be read directly from the TCAP1, TCAP2, and IRQ pins if extra port pins are not available.*

NOTE: *The position sensor inputs toggle in such a manner that only one of the three changes state per MCU interrupt (see **Table 1**), which consequently allows the software to handle one interrupt at a time.*

Another feature on KITITC127 is a potentiometer input into pin PC3/AD3 of the MC68HC705MC4 which is used to interactively control the speed of the motor in software. Additional features of KITITC127 (not discussed in this application note) include amplifiers for overvoltage and over-temperature, an RS232 port from the SCI on the MC68HC705MC4, and run/stop and forward/reverse switches which are input into port C (PC4 and PC5) for control of the motor. The motor drive interface (through ITC122) is controlled by port A (pins PA1–PA6) of the MC68HC705MC4. The KITITC122 is designed to drive three phases of a fractional horsepower motor (12–40 volts) and accepts six logic inputs which control a 3-phase inverter (constructed of MOSFETs).

NOTE: *All of the drivers on KITITC122 use negative logic such that a logic level zero will turn on a driver and a logic level one will turn it off. Additional features of the KITITC122 include level shifters (from CMOS logic to the MOSFET inputs), automatic lockout of invalid inputs (for shoot through prevention), current and temperature sense outputs, and various noise/EMI (electromagnetic interference) considerations.*

Brushless DC Motor Control Software

The motor control software kernel presented in this section will rotate a 3-phase brushless DC motor in a clockwise direction, implementing variable speed control under various load conditions. The MC68HC705MC4 assembly code is relatively simple due in large part to a 3-pin commutation multiplexer on each PWM channel in combination with an interlock mechanism which allows a coherent update to the motor in less than 100 instruction cycles (33.3 μseconds). In detail, **Figure 4** describes the control register for channel A of the PWM module (CTLA), which controls the polarity of PWM A and the commutation multiplexer output signals to pins PWMA1/PA1, PWMA2/PA3, and PWMA3/PA5 (the lowside drivers on the inverter). The commutation multiplexer will allow individual control of pins PA1, PA3, and PA5 to be forced to a logic level or to receive the channel A PWM signal output. Similarly, the CTLB register controls the polarity of PWM B and the multiplexer signals to pins PWMB1/PA2, PWMB2/PA4, and PWMB3/PA6 (the highside drivers on the inverter). The six output pins (PA1–PA6) are coherently updated via a hardware interlock mechanism which requires writes to CTLB and then CTLA for a commutation sequence to take effect (other interlock sequences exist for updates to the PWM duty cycle and period). Observing the flow chart in **Figure 5**, the software in this application is responsible for three major tasks. First, and simplest, is the rotation of the motor. Second, the software runs a PID control algorithm during each motor rotation to maintain desired speed under load conditions. And third, the software will detect and correct for a stall condition. In total, the motor control software kernel consumes less than 300 bytes of program memory.

Address: $0014

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	MEA	POLA	MSK3A	MSK2A	MSK1A	CS3A	CS2A	CS1A
Reset:	0	0	0	0	0	0	0	0

MEA — Mask Enable for PWM Channel A

0 = Mask bits for commutation multiplexer disabled; MSKxA bits disabled

1 = Mask bits for communication multiplexer enabled; apply MSKxA bits

POLA — Polarity for PWM Channel A

0 = Negative polarity; PWM output(s) toggled to one at data match

1 = Positive polarity; PWM output(s) toggled to zero at data match

MSKxA — Mask Bit for PWMAx Pin

0 = PWMAx pin forced to a logic level low

1 = PWMAx pin forced to a logic level high

CSxA — Channel Select Bit for PWMAx Pin

0 = PWMAx pin does not get a PWM signal from channel A

1 = PWMAx pin gets a PWM signal from channel A (overrides mask bit)

Figure 4. PWM Control Register for Channel A (CTL-A)

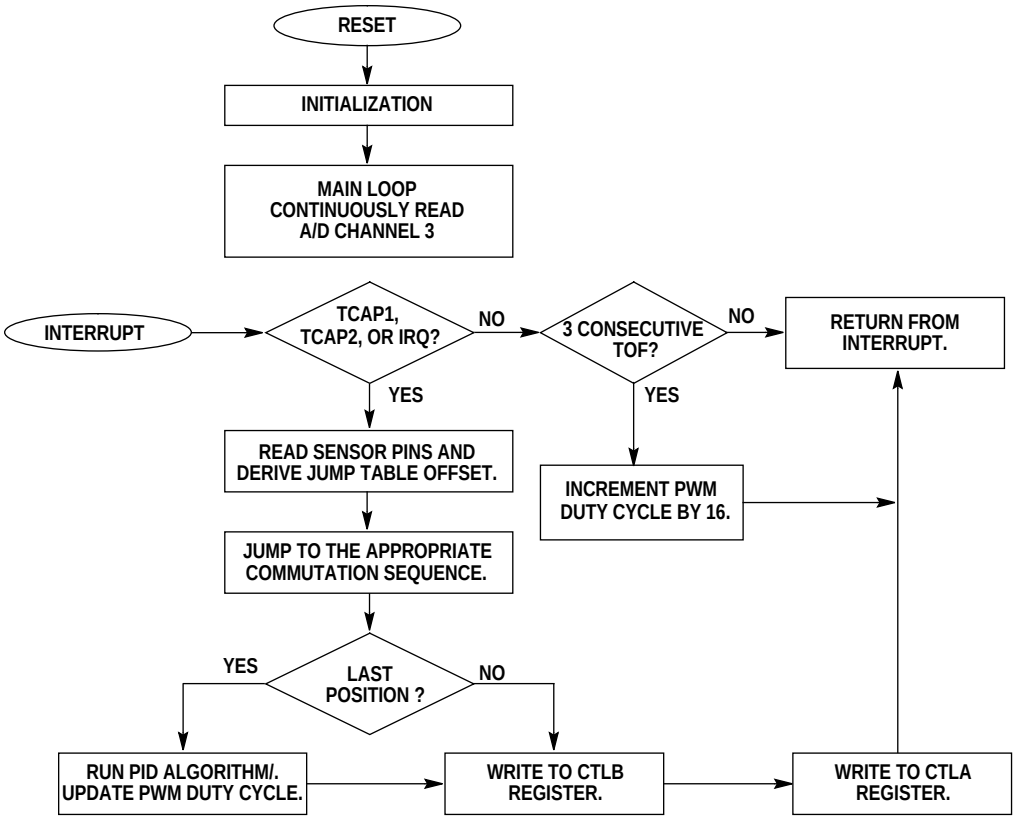

Figure 5. Motor Control Software Flowchart

Initialization

Immediately after each reset, the software will initialize several variables and registers before jumping to the main loop. In detail, the variable “delta,” used in the PID routine, and the variable “timeout,” used to count timer overflow interrupts are both cleared to zero. Also, the timer control register is set to enable input capture interrupts for the TCAP1 and TCAP2 pins, as well as timer overflow interrupts. For the PWM outputs, the PWM control registers for channels A and B are initialized to disable the PWM outputs, the rate register is initialized for a PWM period of 23.4 kHz, and the duty cycle is initialized to 6% via the PWM data register. The A/D converter is enabled and initialized to measure conversions on channel 3 for motor speed selection. Finally, the global interrupt bit in the condition code register is cleared to enable hardware interrupts.

Stall Condition

Stall conditions occur when the motor is initially started or when excessive loads are placed on the motor shaft. A stall condition is detected by the MC68HC705MC4 when three or more successive timer overflows occur — an indication that the motor is not rotating. The stall interrupt routine (triggered by timer overflow interrupts) will slowly increase the PWM duty cycle (by 16 counts per overflow), thus increasing the amount of current to the motor's coils (see the equation on page 3), until the motor rotates. Once the motor begins rotating, the PID algorithm will retain control to commutate the motor at its desired speed.

Rotation of the Shaft (Commutation)

One of the three sensor interrupts (IRQ, TCAP1, or TCAP2) will occur to begin an iteration of commutation. The interrupt service routine will clear the interrupt, check the polarity of the interrupt input pin, toggle the polarity of the pin for its next interrupt, read the position sensor port, and branch to a location which will commutate the motor to its next position in accordance with **Table 1**. Note that the actual commutation sequence involves only writing the two PWM control registers (CTL-B and CTL-A) and the MC68HC705MC4's double buffered PWM output is coherently updated at the end of each PWM period. The desired speed of the motor's shaft (PWM duty cycle) is set by an external potentiometer which is input into channel 3 (PC3/AD3) of the MC68HC705MC4's A/D converter. The main loop continually reads the AD3 pin and implements a PID loop to maintain an updated desired motor speed.

PID Control Algorithm

PID (proportional, integral, derivative) control algorithms often are used in closed-loop systems which need to correct for varying conditions. The underlying concept of PID is to smoothly (over time) correct for an error in the output of a system by comparing its known output to a desired output. First, an error term is calculated and amplified via the proportional operation. And then the integral operation acts to correct for the error, but is damped by the derivative operation which allows for a smooth correction. In this application, a closed-loop PID control algorithm using speed feedback is implemented twice per rotation to compensate for load conditions which require an adjustment to the PWM duty cycle (motor speed). First, actual motor speed is calculated by reading the MC68HC705MC4's 16-bit timer at two different rotation

angles of the motor and calculating the difference. Next, the actual motor speed is compared to the desired motor speed which is obtained from the scaled potentiometer value on pin PC3/AD3. The delta between actual and desired speeds is used to calculate a PID error term using the control algorithm shown in **Figure 6**. The PID error term represents a gradual correction from actual to desired motor speed (for instance, as the actual speed approaches the desired speed, the error term gets smaller). In this experiment, the constants for each PID term (K_P, K_D, and K_I) were determined by trial and error. Note that for ease of programming, all scale factors were limited to fractional powers of two which allows shifts for multiplication and division calculations. Finally, the PID error term is used to “smoothly” correct the rotation speed of the motor in an iterative fashion. A negative error indicates that the motor is rotating too fast, thus the PID error term is subtracted from the PWM duty cycle. Conversely, a positive error term indicates the motor is rotating too slowly and the PID error term is added to the PWM duty cycle.

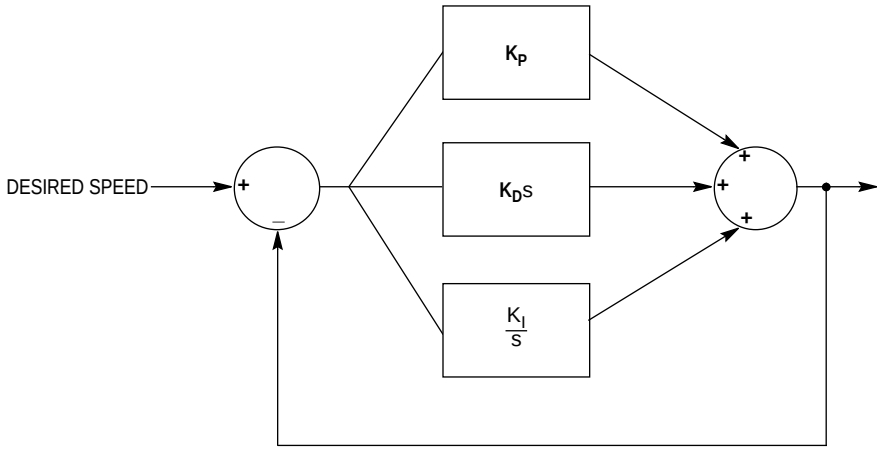

Figure 6. PID Control Loop Equation

Motor Control Software Analysis

This section analyzes the motor kernel shown in **Software Listing**. Three important software characteristics are investigated: code size, code speed, and CPU loading.

Code Size

The motor control algorithm in **Software Listing** consumes 298 bytes of user ROM (8.3% of total) and 10 bytes of user RAM (5.7% of total). This leaves a large amount of user memory for other important routines such as communications protocols, diagnostics analysis, and A/D calculations.

Speed of Execution

The execution speed of the motor software is related directly to the MC68HC705MC4's ability to rotate the motor fast. The motor's speed will be limited by the time it takes the MC68HC705MC4 to read the motor's position and commutate it to its next position. **Table 3** shows the cycle counts and total time required for each commutation sequence in one rotation of the motor. Studying **Table 3** reveals that the software is limited by the sequences at rotation angles 150° and 330° which perform the PID control loop. Since the motor's position sensors are equally spaced, all sequences must be assumed to take 272 cycles (90.67µ seconds). Therefore, the minimum rotation time is:

(12)(90.67 µmicroseconds) = 1.088 ms

And the maximum rotation speed of the motor is:

(1 rotation/1.088 ms)(1000 ms/1 second)
(60 seconds/1min) = 55147 rpm

which is quite adequate for most applications. For higher motor speed requirements, the PID routine could be broken up into equal segments over the commutation sequence of the motor or a simpler control algorithm (such as PI or PD) could be implemented.

Table 3. Cycle Count for Each Commutation Sequence

Rotation Angle in Degrees	Cycles	Time (µs) @ 3 MHz Bus	Interrupt Source
0 & 180	98	32.67	IRQ
30 & 210	100	33.33	TCAP1
60 & 240	98	32.67	TCAP2
90 & 270	99	33	IRQ
120 & 300	95	31.67	TCAP1
150 & 330	272 (Worst Case)	90.67	TCAP2

CPU Bandwidth

CPU loading involves the amount of CPU bandwidth consumed by the motor control algorithm and will indicate the amount of free time for other algorithms. Totalling the number of cycles in **Table 3** and dividing by the rotation time will yield the CPU loading. **Table 4** shows a worst case loading of 46.7% CPU bandwidth at maximum speed. Keeping in mind that most brushless DC motors operate at less than 10 k rpm, the MC68HC705MC4 has a high level of efficiency for these types of applications and will allow designers to add extra features to their system.

Table 4. CPU Bandwidth for Various Motor Speeds

Motor Speed (rpm)	CPU Bandwidth (%)
55.1 k	46.7
40 k	33.9
20 k	16.9
10 k	8.5
5 k	4.2
3500	3.0

Conclusion

The MC68HC705MC4 solution provides a good balance of hardware and software for low-cost 3-phase, brushless DC motor control. Variable speed motor control with a PID loop can be achieved using less than 300 bytes of code and minimal CPU bandwidth for most applications. In addition to motor control, the MC68HC705MC4's general-purpose features (such as A/D, SCI, and flexible PWM in a 28-pin package) make it a useful MCU for other applications such as power supply control, smart sensor controller, and battery chargers.

References

MC68HC705MC4 General Release Specification; HC05MC4GRS/D

Electric Motors and Drives: Fundamental, Types, and Applications, 2nd ed.; Austin Hughes; 1993; Newnes

Software Listing

```
 1  ************************************************************
 2  * MC68HC705MC4 3-phase brushless DC variable speed motor    *
 3  * controller for clockwise rotation.                        *
 4  *                                                           *
 5  * Interrupt driven                                          *
 6  * Version 4/15/96 FOR MC4 APP NOTE                          *
 7  *                                                           *
 8  * Assumed Commutation Sequence for clockwise rotation:      *
 9  *  Angle   Hall1  Hall2  Hall3  PhaseA  PhaseB  PhaseC      *
10     0&180     1      0      0     +15V    -15V    NC        *
11  *  30&210    1      1      0     +15V     NC    -15V       *
12  *  60&240    0      1      0      NC     +15V   -15V       *
13  *  90&270    0      1      1     -15V    +15V    NC        *
14  * 120&300    0      0      1     -15V     NC    +15V       *
15  * 150&330    1      0      1      NC     -15V   +15V       *
16  *                                                          *
17  * ITC127 Wirelist:                                         *
18  * IRQ,PB6   Phase A sensor (Hall 1)                        *
19  * TCAP1,PB7 Phase B sensor (Hall 2)                        *
20  * TCAP2,PA0 Phase C sensor (Hall 3)                        *
21  * PWMA1/PA1 Phase A Bottom (LOW SIDE)                      *
22  * PWMA2/PA3 Phase B Bottom (LOW SIDE)                      *
23  * PWMA3/PA5 Phase C Bottom (LOW SIDE)                      *
24  * PWMB1/PA2 Phase A Top (HIGH SIDE)                        *
25  * PWMB2/PA4 Phase B Top (HIGH SIDE)                        *
26  * PWMB3/PA6 Phase C Top (HIGH SIDE)                        *
27  * PA7       Not used (Pull up to Vdd)                      *
28  * PC0/AD0   Power board buffered B+ feedback               *
29  * PC1/AD1   Power board current feedback                   *
30  * PC2/AD2   Power board temperature diode feedback         *
31  * PC3/AD3   Speed control pot                              *
32  * PC4/AD4   Direction control                              *
33  * PC5/AD5   Run/Stop control                               *
34  ************************************************************
35
```

```
                    36  ************************************************************
                    37  *                          I/O REGISTERS                   *
                    38  ************************************************************
0000                39  PORTA   EQU     $00             ;DATA REGISTER FOR PORT A
0000                40  PORTB   EQU     $01             ;DATA REGISTER FOR PORT B
0000                41  PORTC   EQU     $02             ;DATA REGISTER FOR PORT C
0000                42  PORTD   EQU     $03             ;DATA REGISTER FOR PORT D
0000                43  DDRA    EQU     $04             ;DATA DIRECTION REGISTER FOR PORT A
0000                44  DDRB    EQU     $05             ;DATA DIRECTION REGISTER FOR PORT B
0000                45  DDRC    EQU     $06             ;DATA DIRECTION REGISTER FOR PORT C
0000                46  DDRD    EQU     $07             ;DATA DIRECTION REGISTER FOR PORT D
0000                47  CTCSR   EQU     $08             ;CORE TIMER CONTROL AND STATUS REGISTER
0000                48  CTCR    EQU     $09             ;CORE TIMER COUNTER REGISTER
                    49
0000                50  PWMAD   EQU     $10             ;PWM A DATA REGISTER
0000                51  PWMAI   EQU     $11             ;PWM A INTERLOCK REGISTER
0000                52  PWMBD   EQU     $12             ;PWM B DATA REGISTER
0000                53  PWMBI   EQU     $13             ;PWM B INTERLOCK REGISTER
0000                54  CTLA    EQU     $14             ;PWM A CONTROL REGISTER
0000                55  CTLB    EQU     $15             ;PWM B CONTROL REGISTER
0000                56  RATE    EQU     $16             ;PWM RATE REGISTER
0000                57  UPDATE  EQU     $27             ;PWM UPDATE REGISTER
                    58
0000                59  TCR     EQU     $17             ;TIMER CONTROL REGISTER
0000                60  TSR     EQU     $18             ;TIMER STATUS REGISTER
0000                61  ICRH2   EQU     $19             ;INPUT CAPTURE 2 REGISTER - HIGH BYTE
0000                62  ICRH1   EQU     $1b             ;INPUT CAPTURE 1 REGISTER - HIGH BYTE
0000                63  OCRH    EQU     $1d             ;OUPUT COMPARE REGISTER - HIGH BYTE
0000                64  TMRH    EQU     $20             ;TIMER REGISTER - HIGH BYTE
0000                65  ACRH    EQU     $22             ;ALTERNATE TIMER REGISTER - HIGH BYTE
                    66
0000                67  ADDR    EQU     $24             ;A/D CONVERTER DATA REGISTER
0000                68  ADSCR   EQU     $25             ;A/D CONVERTER STATUS & CNTRL REGISTER
                    69
0000                70  ISCR    EQU     $0f             ;IRQ STATUS AND CONTROL REGISTER
                    71
```

```
                    72  ************************************************************
                    73  *                              CONSTANTS                   *
                    74  ************************************************************
 0000               75  MIN     EQU     $10             ;MINIMUM ALLOWED PWM DUTY CYCLE
 0000               76  MAX     EQU     $FF             ;MAXIMUM ALLOWED PWM DUTY CYCLE
                    77
 0000               78  HALL1   EQU     0               ;HALL 1 SENSOR CONNECTED TO BIT 0
 0000               79  HALL2   EQU     7               ;HALL 2 SENSOR CONNECTED TO BIT 7
0000                80  HALL3   EQU     6               ;HALL 3 SENSOR CONNECTED TO BIT 6
                    81
                    82  * Note: the following control constants are valid for ITC122 which uses
                    83  * negative logic (0 is on, 1 is off) for all 6 drivers. These constants
                    84  * should be changed if positive logic is used for any of the drivers.
                    85
 0000               86  ABOT    EQU     $09             ;CONTROL FOR PWM TO PHASE A BOTTOM
 0000               87  BBOT    EQU     $12             ;CONTROL FOR PWM TO PHASE B BOTTOM
 0000               88  CBOT    EQU     $24             ;CONTROL FOR PWM TO PHASE C BOTTOM
 0000               89  ATOP    EQU     $08             ;CONTROL FOR PHASE A TOP POSITIVE
 0000               90  BTOP    EQU     $10             ;CONTROL FOR PHASE B TOP POSITIVE
 0000               91  CTOP    EQU     $20             ;CONTROL FOR PHASE C TOP POSITIVE
 0000               92 CTLMSK  EQU     $B8            ;CONTROL MASK FOR CTLA AND CTLB REGISTERS
                    93
                    94  ************************************************************
                    95  *                          RAM VARIABLES                   *
                    96  ************************************************************
 0050               97          ORG     $50
 0050               98  REF     RMB     1               ;DESIRED PWM RATE
 0051               99  FIRST   RMB     1               ;TIMER VALUE AT MOTOR POSITION 0
 0052              100  SECOND  RMB     1               ;TIMER VALUE AT MOTOR POSITION 5
 0053              101  PERIOD  RMB     1               ;CALCULATED PWM PERIOD
 0054              102  DELTA   RMB     1               ;DIFF. BETWEEN DESIRED & ACTUAL SPEED
 0055              103  DIFF    RMB     1               ;DIFFERENTIAL TERM FOR PID ALGORITHM
 0056              104  INT     RMB     1               ;INTEGRAL TERM FOR PID ALGORITHM
 0057              105  TMP     RMB     1               ;TEMPORARY STORAGE VARIABLE #1
 0058              106  TMP2    RMB     1               ;TEMPORARY STORAGE VARIABLE #2
 0059              107  TIMEOUT RMB     1               ;COUNTER FOR TIMER OVERFLOW TIMEOUTS
                   108
```

```
                    109  ************************************************************************
                    110  * PROGRAM CODE - Start with initialization of variables and registers   *
                    111  ************************************************************************
                    112
0100                113          ORG     $100
0100                114  START   EQU     *
0100 3F14           115          CLR     CTLA            ;NEGATIVE PWM POLARITY, MASK DISABLED
0102 3F15           116          CLR     CTLB
0104 A6F0           117          LDA     #$F0            ;INITIALIZE REFERENCE SPEED
0106 B750           118          STA     REF
0108 A6E0           119          LDA     #$E0            ;ENABLE TOF,TCAP1, AND TCAP2 INTERRUPTS
010A B717           120          STA     TCR
010C 3F16           121          CLR     RATE            ;INITIALIZE PWM PERIOD TO 23.4 KHZ
010E 3F54           122          CLR     DELTA           ;INITIALIZE VARIABLES
0110 3F59           123          CLR     TIMEOUT
0112 A6F0           124          LDA     #$F0            ;START AT SLOWEST SPEED
0114 B710           125          STA     PWMAD           ;PWM DUTY CYCLE TO 6%
0116 A623           126          LDA     #$23            ;ENABLE CHANNEL 3 OF A/D CONVERTER
0118 B725           127          STA     ADSCR           ;TURN ON A/D CONVERTER
011A 9A             128          CLI                     ;ENABLE HARDWARE INTERRUPTS
                    129
                    130  ************************************************************************
                    131  * The MAIN loop will continuously monitor PC0/AD0 to determine       *
                    132  * the desired speed of the motor.  MAIN can be interrupted from      *
                    133  * four sources (IRQ,TCAP1,TCAP2, and TOF) as the motor is commutated *
                    134  ************************************************************************
                    135
011B 0F25FD         136  MAIN    BRCLR   7,ADSCR,MAIN    ;LOOP ON THE A/D READY BIT
011E B624           137          LDA     ADDR            ;GET THE POT VALUE
0120 44             138          LSRA                    ;SCALE THE POT VALUE
0121 AB50           139          ADD     #$50
0123 43             140          COMA
0124 B750           141          STA     REF             ;UPDATE THE REFERENCE SPEED
0126 20F3           142          BRA     MAIN
                    143
                    144  ********************************************************************
                    145  * IRQ interrupts will occur when the sensor on the IRQ pin toggles.*
                    146  * This routine will clear the int., toggle the sensitivity of the  *
                    147  * next triggered interrupt, and then branch to a routine which will *
                    148  * sense the position of the motor.                                  *
                    149  ********************************************************************
                    150
0128 A620           151  IRQ     LDA     #$20            ;TOGGLE IRQ SENSITIVITY
012A B80F           152          EOR     ISCR
012C B70F           153          STA     ISCR
012E 140F           154          BSET    2,ISCR          ;ACKNOWLEDGE INTERRUPT
0130 2016           155          BRA     POS
                    156
```

```
                    157  ***********************************************************************
                    158  * Input capture interrupts will occur when either TCAP1 or TCAP2      *
                    159  * sensor pins toggle. This routine will first check the source of     *
                    160  * int. - TCAP1 or TCAP2, then clear the interrupt, toggle the         *
                    161  * sensitivity of the next triggered interrupt, & then branch to a     *
                    162  * routine which will sense the position of the motor.                 *
                    163  ***********************************************************************
                    164
0132 B618           165  ICISR   LDA     TSR             ;FIRST PART OF CLEARING FLAG
0134 2A0A           166          BPL     TC1
0136 B61A           167          LDA     ICRH2+1
0138 A602           168          LDA     #$02            ;TOGGLE EDGE SENSITIVITY
013A B817           169          EOR     TCR
013C B717           170          STA     TCR
013E 2008           171          BRA     POS
0140 B61C           172  TC1     LDA     ICRH1+1         ;AND CLEAR ANY FLAGS
0142 A604           173          LDA     #$04            ;Toggle edge sensitivity
0144 B817           174          EOR     TCR
0146 B717           175          STA     TCR
                    176
                    177  ***********************************************************************
                    178  * This routine will sense the pos. of the motor shaft by reading     *
                    179  * port pins PA0,PB6,and PB7.  The routine will then use the value to *
                    180  * generate an index into a jump table which is used to jump to the   *
                    181  * proper commutation sequence.                                       *
                    182  ***********************************************************************
                    183
0148 3F57           184  POS     CLR     TMP
014A 010002         185          BRCLR   HALL1,PORTA,NOT1   ;HALL 1 GRAY WIRE NOT = 1
014D 1057           186          BSET    0,TMP
014F 0F0102         187  NOT1    BRCLR   HALL2,PORTB,NOT2   ;HALL 2 BLUE WIRE NOT = 1
0152 1257           188          BSET    1,TMP
0154 0D0102         189  NOT2    BRCLR   HALL3,PORTB,NOT3   ;HALL 3 WHITE WIRE NOT = 1
0157 1457           190          BSET    2,TMP
0159                191  NOT3    EQU     *
                    192
0159 B657           193          LDA     TMP             ;GET THE PATTERN
015B 4A             194          DECA                    ;CHANGE TO NUMBER BETWEEN 0 AND 5
015C B758           195          STA     TMP2
015E 48             196          LSLA                    ;MULTIPLY BY 3
015F BB58           197          ADD     TMP2
0161 97             198          TAX                     ;MOVE ACCUMLATOR TO INDEX REGISTER
0162 DC01FC         199          JMP     JMPTABF,X       ;USE THE CLOCKWISE JUMP TABLE
                    200  * Note: use     JUMTABR,X  for counter clockwise rotation
                    201
```

```
                    202  ********************************************************************
                    203  * Commutation pos. 0 degrees for the motor. Read/store the value of *
                    204  * the 16-bit timer and commutate the motor to its next position.    *
                    205  ********************************************************************
                    206
0165 B622           207  A_TO_B  LDA     ACRH            ;SAVE TIMER VALUE AT POSITION 0
0167 B751           208          STA     FIRST
0169 A6AA           209          LDA     #CTLMSK^BBOT    ;PA3=PWM (PHASE B BOTTOM)
016B AEB0           210          LDX     #CTLMSK^ATOP    ;PA6=0 (CTOP),PA4=0 (BTOP),PA2=1 (ATOP)
016D BF15           211  POSX    STX     CTLB            ;UPDATE PA2, PA4, AND PA6
016F B714           212          STA     CTLA            ;UPDATE PA1, PA3, AND PA5
0171 B623           213          LDA     ACRH+1          ;COMPLETE READ SEQUENCE
0173 80             214  POSX2   RTI
                    215
                    216  ********************************************************************
                    217  * Commutation pos. 30 degrees for the motor. Commutate the motor to *
                    218  * its next position.                                               *
                    219  * Note: if rotating the motor CCW, the PID control routine         *
                    220  *  must be run during this commutation sequence                    *
                    221  ********************************************************************
                    222
0174 A69C           223  A_TO_C  LDA     #CTLMSK^CBOT    ;PA5=PWM (PHASE C BOTTOM)
0176 AEB0           224          LDX     #CTLMSK^ATOP    ;PA6=0 (CTOP),PA4=0 (BTOP),PA2=1 (ATOP)
0178 20F3           225          BRA     POSX
                    226
                    227  ********************************************************************
                    228  * Commutation pos. 60 degrees for the motor. Commutate the motor to *
                    229  * its next position.                                               *
                    230  ********************************************************************
                    231
017A A69C           232  B_TO_C  LDA     #CTLMSK^CBOT    ;PA5=PWM (PHASE C BOTTOM)
017C AEA8           233          LDX     #CTLMSK^BTOP    ;PA6=0 (CTOP),PA4=1 (BTOP),PA2=0 (ATOP)
017E 20ED           234          BRA     POSX
                    235
                    236  ********************************************************************
                    237  * Commutation pos. 90 degrees for the motor. Commutate the motor to *
                    238  * its next position.                                               *
                    239  ********************************************************************
                    240
0180 A6B1           241  B_TO_A  LDA     #CTLMSK^ABOT    ;PA1=PWM (PHASE A BOTTOM)
0182 AEA8           242          LDX     #CTLMSK^BTOP    ;PA6=0 (CTOP),PA4=1 (BTOP),PA2=0 (ATOP)
0184 20E7           243          BRA     POSX
                    244
                    245  ********************************************************************
                    246  * Commutation pos. 120 degrees for the motor. Commutate the motor   *
                    247  * to its next position.                                            *
                    248  ********************************************************************
                    249
0186 A6B1           250  C_TO_A  LDA     #CTLMSK^ABOT    ;PA1=PWM (PHASE A BOTTOM)
0188 AE98           251          LDX     #CTLMSK^CTOP    ;PA6=1 (CTOP),PA4=0 (BTOP),PA2=0 (ATOP)
018A 20E1           252          BRA     POSX
```

```
                        253
                        254  ********************************************************************
                        255  * Commutation pos. 150 degress for the motor. Read the timer,       *
                        256  * excute PID algorithm, &commutate the motor to its next position.  *
                        257  * Note:  the PID algo. should NOT be run during this commutation    *
                        258  *  sequence if the motor rotation is counterclockwise.             *
                        259  ********************************************************************
                        260
018C AD08               261  C_TO_B  BSR     PID             ;BRANCH TO PID ROUTINE
018E A6AA               262          LDA     #CTLMSK^BBOT    ;PA3=PWM (PHASE B BOTTOM)
0190 AE98               263          LDX     #CTLMSK^CTOP    ;PA6=1 (CTOP),PA4=0 (BTOP),PA2=0 (ATOP)
0192 3F59               264          CLR     TIMEOUT         ;CLEAR TIMEOUT COUNTER
0194 20D7               265          BRA     POSX
                        266
                        267  ********************************************************************
                        268  * This routine will implement a PID algo. to correct for conditions   *
                        269  * which will cause the motor to rotate slower or faster than desired. *
                        270  * The PID routine will smoothly correct for speed  by incrementally   *
                        271  * closing in on the desired motor speed.                              *
                        272  ********************************************************************
                        273
0196 B622               274  PID     LDA     ACRH            ;SAVE SECOND TIME
0198 B752               275          STA     SECOND
                        276
019A B051               277          SUB     FIRST           ;SUBTRACT FIRST
019C B753               278          STA     PERIOD          ;SAVE AS PERIOD
                        279
019E B050               280          SUB     REF             ;CALCULATE DELTA TERM
                                                                  (MEASURED PERIOD - ACTUAL)
01A0 B757               281          STA     TMP
01A2 B054               282          SUB     DELTA           ;CALCULATE DIFFERENTIAL TERM
01A4 B755               283          STA     DIFF
01A6 2A05               284          BPL     ABS             ;TAKE ABSOLUTE VALUE
01A8 4F                 285          CLRA
01A9 B055               286          SUB     DIFF
01AB B755               287          STA     DIFF
01AD B657               288  ABS     LDA     TMP             ;CALCULATE INTEGRAL TERM
01AF BB54               289          ADD     DELTA
01B1 B756               290          STA     INT
01B3 2A05               291          BPL     AB20            ;ABSOLUTE VALUE
01B5 4F                 292          CLRA
01B6 B056               293          SUB     INT
01B8 B756               294          STA     INT
01BA B657               295  AB20    LDA     TMP
01BC B754               296          STA     DELTA
01BE 2A05               297          BPL     S2              ;ABSOLUTE VALUE OF DELTA
01C0 4F                 298          CLRA
01C1 B054               299          SUB     DELTA
01C3 B754               300          STA     DELTA
01C5 44                 301  S2      LSRA                    ;SCALE IT
01C6 44                 302          LSRA
```

```
01C7 44          303           LSRA
01C8 44          304           LSRA
01C9 B757        305           STA     TMP
01CB B655        306           LDA     DIFF            ;READ DIFFERENTIAL TERM
01CD 44          307           LSRA                    ;SCALE IT
01CE 44          308           LSRA
01CF 44          309           LSRA
01D0 44          310           LSRA
01D1 BB57        311           ADD     TMP             ;ADD IT ON
01D3 B757        312           STA     TMP
01D5 B656        313           LDA     INT             ;READ INTEGRAL TERM
01D7 44          314           LSRA                    ;SCALE IT
01D8 44          315           LSRA
01D9 44          316           LSRA
01DA 44          317           LSRA
01DB BB57        318           ADD     TMP             ;ADD IT ON
01DD B757        319           STA     TMP
                 320
01DF B653        321           LDA     PERIOD          ;COMPARE ACTUAL SPEED TO DESIRED SPEED
01E1 B150        322           CMP     REF
01E3 240C        323           BCC     FASTER          ;LESS THAN ZERO? -> FASTER
                 324
01E5             325  SLOWER   EQU     *               ;DECREASE THE MOTOR SPEED
01E5 B610        326           LDA     PWMAD
01E7 B057        327           SUB     TMP             ;DECREASE PWM DUTY CYCLE
01E9 A110        328           CMP     #MIN            ;CHECK LOW SPEED LIMIT
01EB 240C        329           BCC     DONE
01ED A610        330           LDA     #MIN
01EF 2008        331           BRA     DONE
                 332
01F1             333  FASTER   EQU     *               ;INCREASE THE MOTOR SPEED
01F1 B657        334           LDA     TMP
01F3 BB10        335           ADD     PWMAD           ;INCREASE PWM DUTY CYCLE
01F5 2402        336           BCC     DONE            ;CHECK THE HIGH SPEED LIMIT
01F7 A6FF        337           LDA     #MAX
01F9 B710        338  DONE     STA     PWMAD
01FB 81          339           RTS
                 340
                 341  * Jump table for clockwise rotation
01FC             342  JMPTABF EQU     *
01FC CC0165      343           JMP     A_TO_B           ;TURN ON Atop AND Bbot DRIVERS
01FF CC017A      344           JMP     B_TO_C           ;TURN ON Btop AND Cbot DRIVERS
0202 CC0174      345           JMP     A_TO_C           ;TURN ON Atop AND Cbot DRIVERS
0205 CC0186      346           JMP     C_TO_A           ;TURN ON Ctop AND Abot DRIVERS
0208 CC018C      347           JMP     C_TO_B           ;TURN ON Ctop AND Bbot DRIVERS
020B CC0180      348           JMP     B_TO_A           ;TURN ON Btop AND Abot DRIVERS
                 349
                 350  * Jump table for counterclockwise rotation
                 351  * Note: if using counterclockwise rotation the PID routine needs
                 352  *  to be run during the A_TO_C commutation sequence
```

```
020E             353  JMPTABR EQU     *
020E CC0180      354          JMP     B_TO_A           ;TURN ON Btop AND Abot DRIVERS
0211 CC018C      355          JMP     C_TO_B           ;TURN ON Ctop AND Bbot DRIVERS
0214 CC0186      356          JMP     C_TO_A           ;TURN ON Ctop AND Abot DRIVERS
0217 CC0174      357          JMP     A_TO_C           ;TURN ON Atop AND Cbot DRIVERS
021A CC017A      358          JMP     B_TO_C           ;TURN ON Btop AND Cbot DRIVERS
021D CC0165      359          JMP     A_TO_B           ;TURN ON Atop AND Bbot DRIVERS
                 360
                 361  ********************************************************************
                 362  * Timer overflow int. service routine. Checks for prolonged stall    *
                 363  * conditions. Increases PWM duty cycle until normal operation occurs *
                 364  ********************************************************************
                 365
0220 B618        366  TOFISR  LDA     TSR              ;CLEAR FLAG
0222 B621        367          LDA     TMRH+1
0224 B659        368          LDA     TIMEOUT          ;TIMEOUT?
0226 A103        369          CMP     #$03
0228 230D        370          BLS     TOFX             ;NO -> EXIT
022A B610        371          LDA     PWMAD            ;YES -> INCREASE POWER TO COILS
022C AB10        372          ADD     #$10
022E 2402        373          BCC     TOF2             ;OVERFLOW  -> SET TO MAX
0230 A6FF        374          LDA     #$FF
0232 B710        375  TOF2    STA     PWMAD
0234 CC0148      376          JMP     POS              ;MOVE TO NEXT POSITION
0237 3C59        377  TOFX    INC     TIMEOUT          ;INCREMENT TIMEOUT COUNTER
0239 80          378          RTI
                 379
                 380  ************************************************************
                 381  * Reset/Interrupt Vectors.                                  *
                 382  * Note that the core timer and SCI vectors are not used in  *
                 383  * this application.                                         *
                 384  ************************************************************
                 385
0FF0             386          ORG     $0FF0
                 387
0FF0 0100        388          FDB     START            ;CORE TIMER VECTOR - NOT USED
0FF2 0100        389          FDB     START            ;SCI VECTOR - NOT USED
0FF4 0220        390          FDB     TOFISR           ;TIMER VECTOR 1 - TIMER OVERFLOW
0FF6 0132        391          FDB     ICISR            ;TIMER VECTOR 2 - TCAP1
0FF8 0132        392          FDB     ICISR            ;TIMER VECTOR 3 - TCAP2
0FFA 0128        393          FDB     IRQ              ;IRQ VECTOR
0FFC 0100        394          FDB     START            ;SWI VECTOR
0FFE 0100        395          FDB     START            ;RESET VECTOR
```

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