Application Report
SLAA503–July 2011

Sensored 3-Phase BLDC Motor Control Using MSP430

Bhargavi Nisarga, Daniel Torres — *MSP430 Applications*

ABSTRACT

Brushless DC (BLDC) motors are electronically commutated motors that offer many advantages over brushed DC motors and, therefore, are becoming very popular industrially and commercially. This application report discusses a sensored 3-phase BLDC motor control solution using MSP430™ as the motor controller. Hall sensors are used to detect the rotor position and close the commutation loop. Both open loop and closed-loop control implementations are discussed.

Project collateral and source code discussed in this application report can be downloaded from the following URL: http://www.ti.com/lit/zip/slaa503.

Contents

List of Figures

List of Tables

1 BLDC Motor-Control Introduction

Brushless DC motors have gained increasing popularity in the recent years. BLDC motors have permanent magnets that rotate (rotor) and a fixed armature (stator). An electronic controller is used for motor commutation instead of the brushed commutation used in the brushed DC motors. BLDC motors offer many advantages over the brushed DC motors, which include increased speed vs torque efficiency, longer life (as no brushes are used), noiseless operation, and increased efficiency in converting electrical power to mechanical power (especially because there are no electrical and frictional losses due to brushed commutation).

The motor commutation in BLDC motors is implemented by an electronic controller and, to determine the rotor position and to know when to commutate, either Hall sensors (sensored commutation) or the back EMF generated in the stator windings of the motor (sensorless commutation) are used. Sensorless controllers have challenges during motor start-up, as no back EMF is present when the motor is stationary; this is worked around by starting the motor from an arbitrary position; however, this can cause the motor to briefly jerk or even rotate backward during start-up. Hall sensor based controllers are simpler to implement compared to the sensorless control and are used in applications that require good starting torque and that require the motors to run at lower speeds. The sensored motor solutions are especially critical for applications that operate in noisy electrical systems. This application report discusses a Hall sensored commutation control that uses an MSP430 microcontroller as the motor controller.

Depending on the number of windings on the stator, BLDC motors are available in 1-phase, 2-phase, and 3-phase configurations. This application report discusses the 3-phase BLDC motor control in both open loop and closed-loop control configurations.

A typical Hall sensored 3-phase BLDC motor control is discussed in Section 2.

2 Open-Loop Control

Open-loop control is a simple control system in which the system does not track the output of the process it is controlling. In other words, open-loop control does not use feedback to determine if the output has achieved the intended goal of the input. This type of control is used in systems in which the relationship between input and the resultant state is well-defined and the feedback is not critical. The key advantages of this control system are:

- Simplicity in designing the control system
- Lower cost as the feedback signal chain is not required

2.1 Basic Block Diagram

In motor control applications, open-loop control is used to control the speed of the motor by directly controlling the duty cycle of the PWM signal that directs the motor-drive circuitry. The duty cycle of the PWM signal controls the ON time of the power FETs in the half bridges of the motor-drive circuit and this in turn controls the average voltage supplied across the motor windings.

Figure 1 shows a typical open loop motor control system.

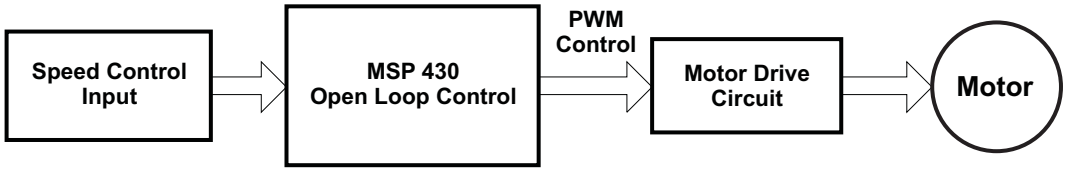

Figure 1. Open-Loop Control – Basic Block Diagram

The *Speed Control Input* unit provides the motor-speed input to the control system. This input can either be analog or digital. Depending on the speed-control input, the open-loop control system implemented in the MSP430 either increases or decreases the PWM duty cycle, which in turn increases or decreases the average voltage or current applied to the motor via the motor-drive circuitry and controls the motor speed accordingly.

2.2 Principle of Operation

Figure 2 shows the block diagram of the open-loop control implementation.

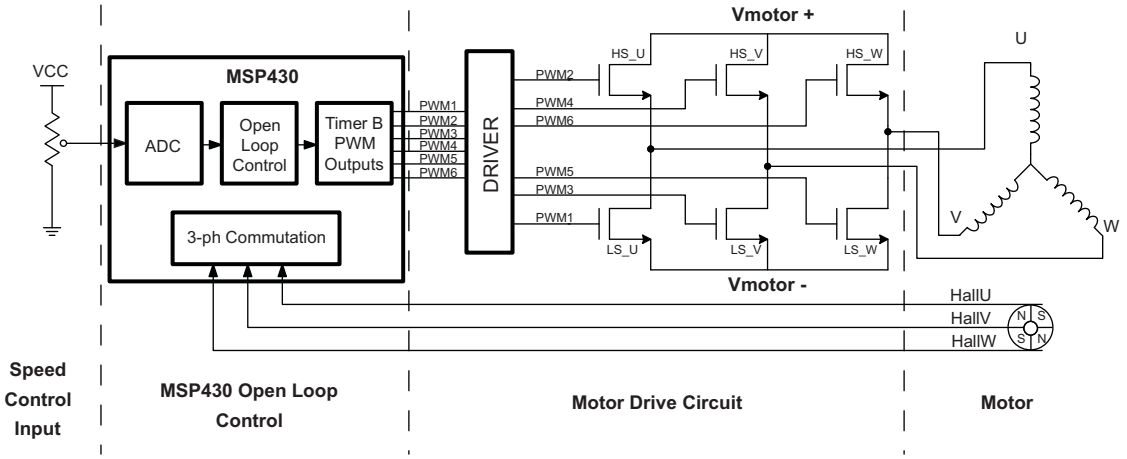

Figure 2. Open-Loop Control – MSP430-Based Implementation

2.2.1 Speed-Input Control

The speed-input control is provided by an analog potentiometer. The potentiometer value is measured by the analog-to-digital (A/D) converter integrated in the MSP430. A linear relationship between the ADC input value measured and motor speed is assumed in this implementation, such that ADC output ranging from 0 to maximum counts corresponds to motor speed ranging from 0% to 100%.

2.2.2 Open-Loop Control

MSP430 implements the open-loop control plus the 3-phase motor commutation. The open-loop control interprets the speed input measured in terms of timer PWM duty cycle counts and updates the output PWM duty cycles accordingly to control the motor speed. The 3-phase commutation is discussed in the next section.

2.2.3 Motor-Drive Circuit

The logic levels of the timer PWM output from the MSP430 are 0 to V_{CC}. To boost the voltage to drive motors at higher voltage levels (V_{motor}), predrivers are used. The motor drivers also help protect the logic chips and isolate electrical noise. The outputs of the predrivers are fed to three half bridges as a part of the commutation loop. The power FETs used in the three half bridges are all NFETs, and the PWM signal with variable duty cycle is applied to only the high side FETs. The low side FETs are controlled by the ON/OFF signals.

2.2.4 3-Phase BLDC Motor Commutation

The PWM outputs of the MSP430 are used to control the 3-phase motor commutation, and its duty cycle is used to control the current through the power FETs and motor windings and, in turn, the speed of the motor.

In the Hall sensored solution, Hall sensors mounted on the motor reflect the motor position. These Hall sensor signals from the motor are input to the MSP430 to close the commutation loop. The Hall sensor signals are connected to the GPIO input pins with the respective interrupts enabled. On capturing a Hall sensor state change event, the PWM outputs that control the motor-drive circuit are updated according to the commutation sequence. Table 1 and Figure 3 show the 3-phase BLDC motor commutation sequence in accordance with the standard 120˚ commutation using Hall sensors.

Table 1. Hall Sensor Based Motor Commutation Sequence

	Hall Inputs (U|V|W) = (P1.3|P1.2|P1.1)			
State	DIR = CW	DIR = CCW	Active PWMs and Power FETs	
1	001	110	PWM4, PWM5	HS_V, LS_W
2	011	100	PWM2, PWM5	HS_U, LS_W
3	010	101	PWM2, PWM3	HS_U, LS_V
4	110	001	PWM6, PWM3	HS_W, LS_V
5	100	011	PWM6, PWM1	HS_W, LS_U
6	101	010	PWM4, PWM1	HS_V, LS_U

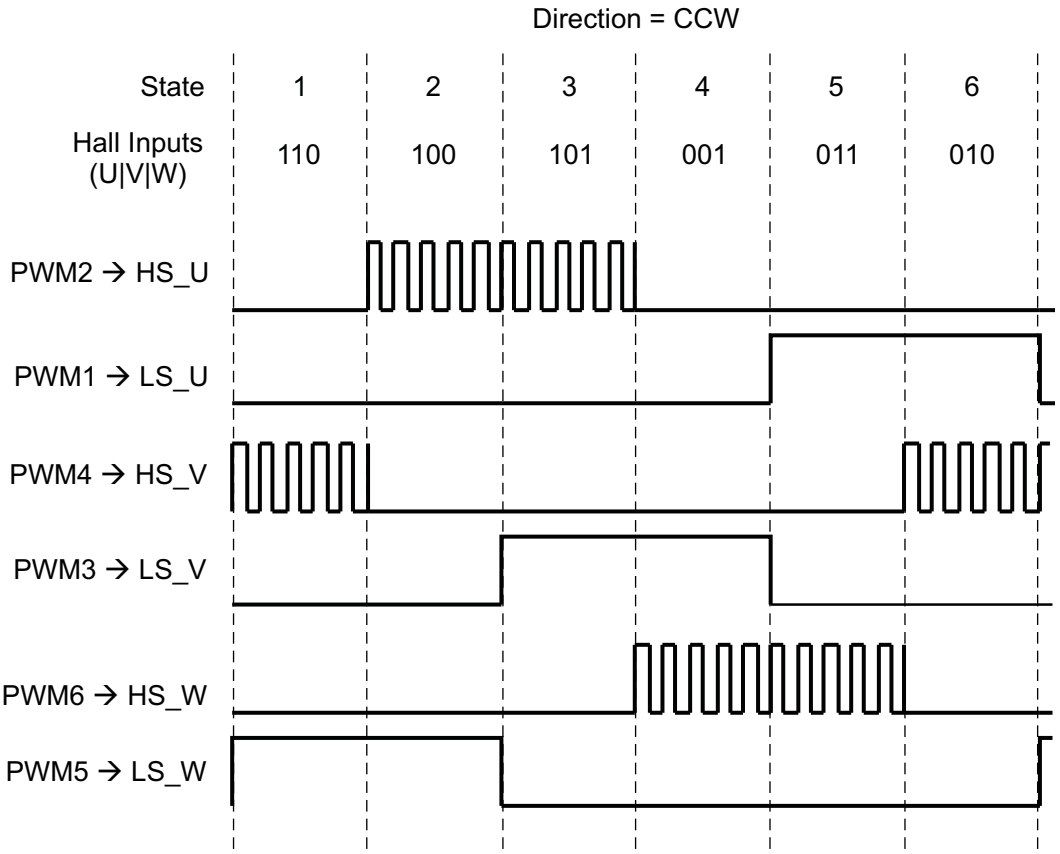

Figure 3. Hall Sensor Based Motor Commutation Sequence (One Electrical Cycle)

2.3 Open-Loop Control Firmware

This section discusses the open-loop motor control implementation in the demo firmware that is provided with this application report.

Table 2 lists all the user configurable parameters available for the open-loop control implementation. The clock timing definitions listed apply to the closed-loop solution as well.

Table 2. Open-Loop Control – User Configurable Parameter Definitions

Configurable Parameter (Defined in main_open_loop Header File)	Comments	Demo Firmware Values
Clock Timing Definitions		
SYSTEM_FREQ	Defines MCLK = SMCLK = Timer PWM base frequency (in kHz)	16000 kHz

Table 2. Open-Loop Control – User Configurable Parameter Definitions (continued)

Configurable Parameter (Defined in main_open_loop Header File)	Comments	Demo Firmware Values
PWM_FREQ	Defines timer PWM switching frequency (in Hz) (see Note(1))	15640 Hz
TIMER_PWM_PERIOD	Represents the number of duty cycle steps or the number of motor speed steps = $\left(\frac{SYSTEM\_FREQ}{PWM\_FREQ}\right)$	$\frac{16\ MHz}{15.64\ kHz}$ = 1023 Steps
SPEEDIN_PWM_FACTOR	= $\left(\frac{2^{12}}{Timer\_PWM\_Period}\right)$ (see Note (2))	$\frac{2^{12}}{1023} = 4$
DUTYCYCLE_MIN	Defines minimum PWM duty cycle or minimum motor speed (in percentage)	5%
MIN_PWM_DUTYCYCLE	Defines minimum motor speed (in number of duty cycle counts) = $\left(Timer\_PWM\_Period \times \frac{Dutycycle\_Min}{100}\right)$	1023 × 0.05 = 51 Counts
Motor Start-Up and Open-Loop Control Definitions		
MOTOR_STARTUP_TIME	Defines motor start-up time (in ms)	100 ms
DUTYCYCLE_CHANGE_PERIODS	Defines the number of timer PWM periods after which PWM duty cycle update is expected during motor start-up routine (see Note(3))	10 PWM Periods
STARTUP_STEPS	Defines the number of PWM duty cycle update steps during motor start-up $= \left(\frac{PWM\_FREQ \times MOTOR\_STARTUP\_TIME}{DUTYCYCLE\_CHANGE\_PERIODS}\right)$	$\left(\frac{15.64\ kHz \times 100\ ms}{10}\right) = 15$
ADC_SAMPLING_PWM_PERIODS	Defines the number of timer PWM periods after which PWM duty cycle update is expected during motor start-up routine (see Note(3))	1000 PWM Periods
MAIN_PWM_BUCKET_PERCENT	Defines the PWM duty cycle resolution (in percentage), also called the bucket step, that is used to either increase or decrease PWM duty cycle values and, in turn, the motor speed during open-loop control.	0.20%
MAIN_PWM_BUCKET_DC	Represents the bucket step resolution in PWM duty cycle counts (see Note(4)) = TIMER_PWM_PERIOD × MAIN_PWM_BUCKET%	$\left(1023 \times \frac{0.2}{100}\right)$ = 2 Counts

(1) The PWM switching frequency should be selected such that it is beyond the audible frequency range. In the demo firmware, PWM switching frequency of 15.64 kHz is chosen. This particular PWM frequency was chosen so that TIMER_PWM_PERIOD or the number of duty cycle steps is a multiple of 2, which generates a whole number for the PWM_SPEEDIN_FACTOR (which is a ratio of the maximum ADC counts to maximum duty cycle counts). This makes normalization and other computations in firmware simpler – shift left/right instructions instead of multiply and divide functions.

(2) MSP430F5529 with ADC12 is used in the demo firmware and, therefore, an ADC resolution of 2^{12} is used in this application.

(3) The Timer PWM period interval is used as a heart beat signal to keep track of the PWM duty cycle update and ADC sampling intervals.

(4) Ensure that the MAIN_PWM_BUCKET_PERCENT selected yields a bucket step resolution of at least 1.

The timer PWM signals control the motor-drive circuitry, and the PWM duty cycle is used to control the motor speed. The PWM duty cycle is directly proportional to the motor speed; therefore, these terms are used interchangeably in this application report.

2.3.1 Motor Start-Up Routine

The motor start-up routine implemented in the demo firmware increases the motor speed from minimum motor speed (pre-defined minimum PWM duty cycle value - MIN_PWM_DUTYCYCLE) to the desired speed level (based on the speed input) in a pre-defined period of time (MOTOR_STARTUP_TIME) and pre-defined number of steps (STARTUP_STEPS). With the desired motor speed and pre-defined MOTOR_STARTUP_TIME and STARTUP_STEPS, the bucket step size or the duty cycle increment value is computed runtime in the Start_Motor function. The PWM duty cycle value and, in turn, the motor speed is incremented by the bucket step value inside the Timer PWM ISR. If the desired speed input level is less than the minimum motor speed level, then the motor speed is latched to the minimum level.

2.3.2 Open-Loop Control

The control input from the analog pot in the system is used to control the motor speed in the open-loop control implementation. The integrated A/D converter samples the speed control input in system to detect the desired PWM duty cycle or the desired motor speed. Depending on if the current PWM duty cycle is higher or lower than the desired PWM duty cycle value, the PWM duty cycle is decremented or incremented, respectively, in pre-defined PWM duty cycle bucket steps (MAIN_PWM_BUCKET_DC) inside the Timer PWM ISR and this, in turn, decrements or increments the motor speed.

The firmware implementation of the open-loop control is as shown in Figure 4 and Figure 5.

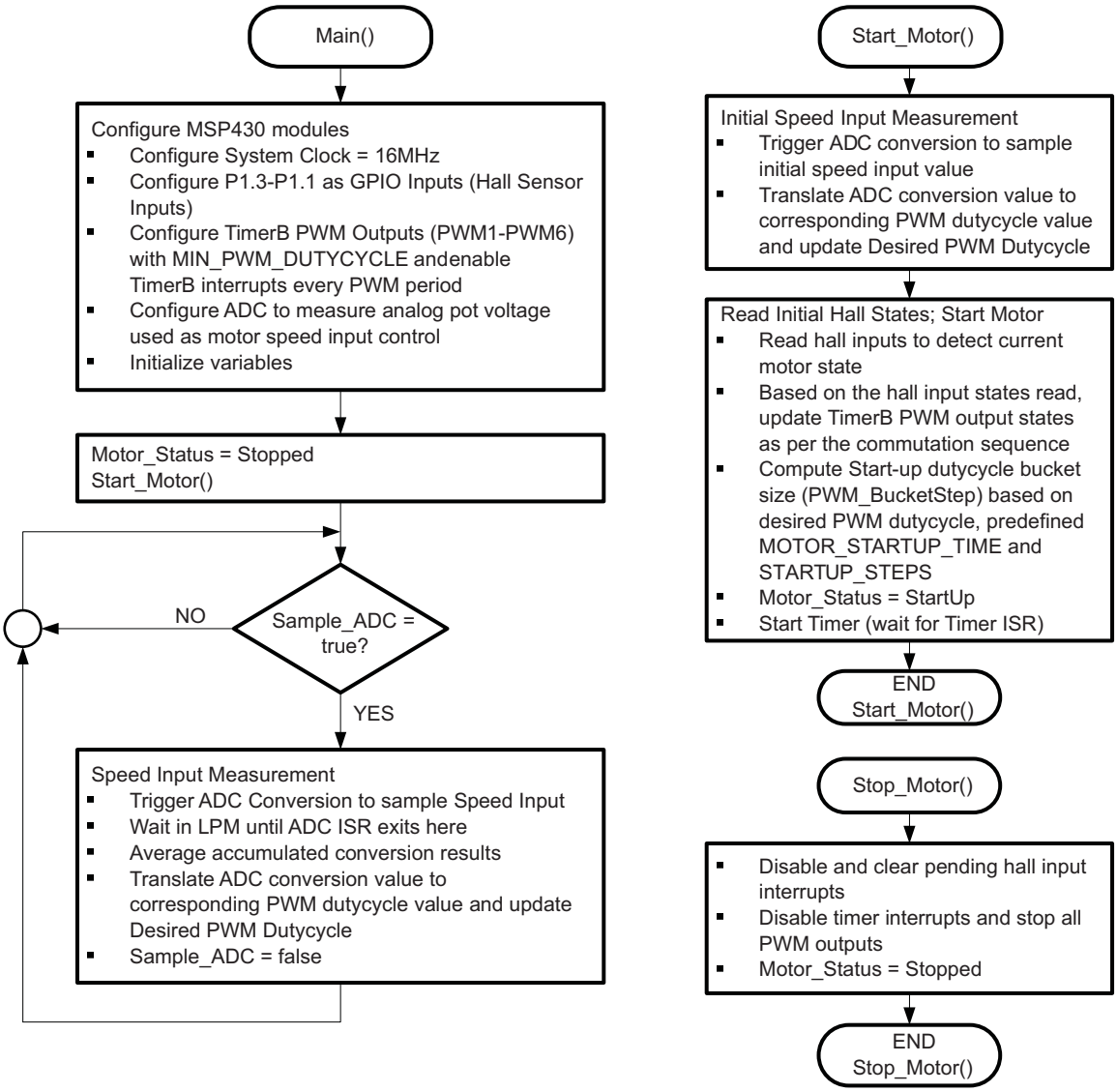

Figure 4. Flowchart – Open-Loop Control Functions

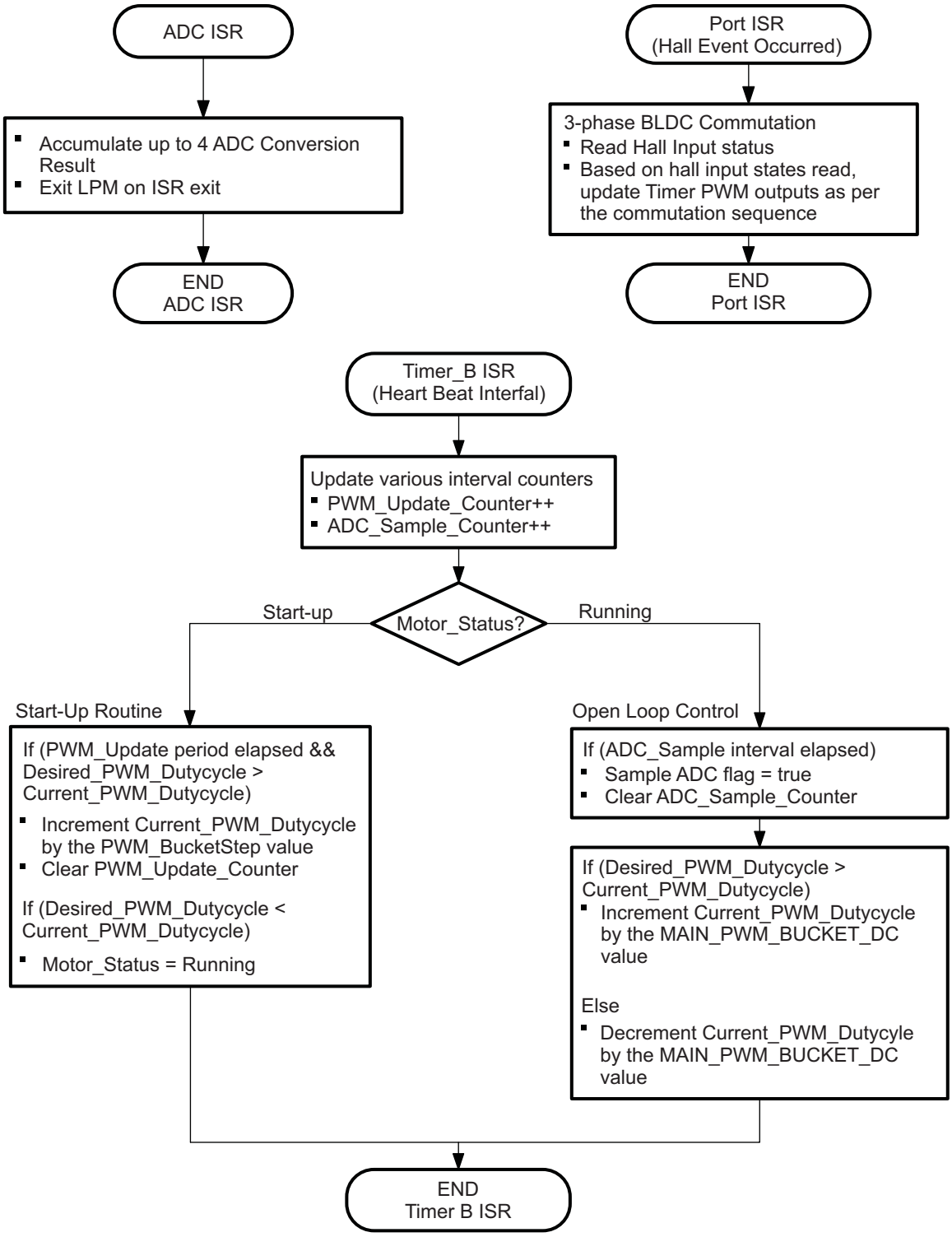

Figure 5. Flowchart – Open-Loop Control ISRs

While designing Hall sensored motor control solutions, there have been concerns regarding the time taken for the electronic controller to detect Hall state change and commutate accordingly by updating the timer PWM output. In this implementation, the MSP430 microcontroller that is used as the electronic controller is configured with a system clock of 16 MHz and, therefore, the Hall events are almost instantly detected. The time taken to enter the Hall input interrupt service routine (port ISR) is approximately 6 MCLK cycles = 6/16 MHz = 375 ns[1]. The total time taken to detect the Hall input state change and update the timer PWM output states as per the commutation sequence is approximately 130 MCLK cycles = 130 / 16 MHz = 8.125 µs [(1)] (this value is obtained from the demo firmware implementation).

The maximum motor speed is limited by this number. In this particular case, with a Hall event response time of 8.125 µs, the maximum motor speed that can be achieved is approximately 41000 rpm . [(2)]

3 Closed-Loop Control

Closed-loop controls are used in applications that require more accurate and adaptive control of the system. These controls use feedback to direct the output states of a dynamic system. Closed-loop controls overcome the drawbacks of open-loop control to provide compensation for disturbances in the system, stability in unstable processes, and reduced sensitivity to parameter variations (dynamic load variation).

A PID controller is a closed-loop control implementation that is widely used and is most commonly used as a feedback controller. This application report implements a PI controller to provide closed-loop control for the 3-phase BLDC motor control.

3.1 Basic Block Diagram

Similar to the open-loop control, closed-loop control regulates the speed of the motor by directly controlling the duty cycle of the PWM signals that direct the motor-drive circuitry. The major difference between the two control systems is that the open-loop control considers only the speed control input to update the PWM duty cycle, whereas, the closed-loop control considers both speed-input control and actual motor speed (feedback to controller) for updating the PWM duty cycle and, in turn, the motor speed.

Figure 6 shows a typical closed loop motor control system.

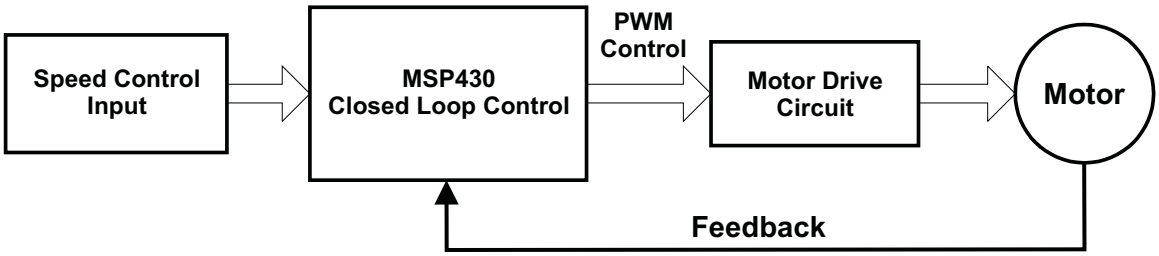

Figure 6. Closed-Loop Control – Basic Block Diagram

The Speed Control Input unit provides motor speed input to the control system. This input can either be analog or digital. The actual motor speed is fed back to the closed-loop controller, which is implemented on an MSP430 microcontroller. The PI controller is used as the closed-loop control algorithm to track the actual motor speed and also apply the speed control input. Based on speed control input and present and past errors (proportional and integral values), the closed-loop control either increases or decreases the PWM duty cycle, which in turn controls the speed of the motor.

3.2 Principle of Operation

Figure 7 shows the block diagram of the closed-loop control implementation.

(1) Assuming the device is in LPM0 or active mode (where DCO clock not turned off) during Hall state change detection

(2) Maximum number of Hall ISRs per second is converted to respective electrical and mechanical cycles using formulas in Table 3 and applied to the Hurst Motor DMA0102024C (used in this demo application) pole pair specification (1 / 3 × 8.125 µs ≈ 41000 rpm).

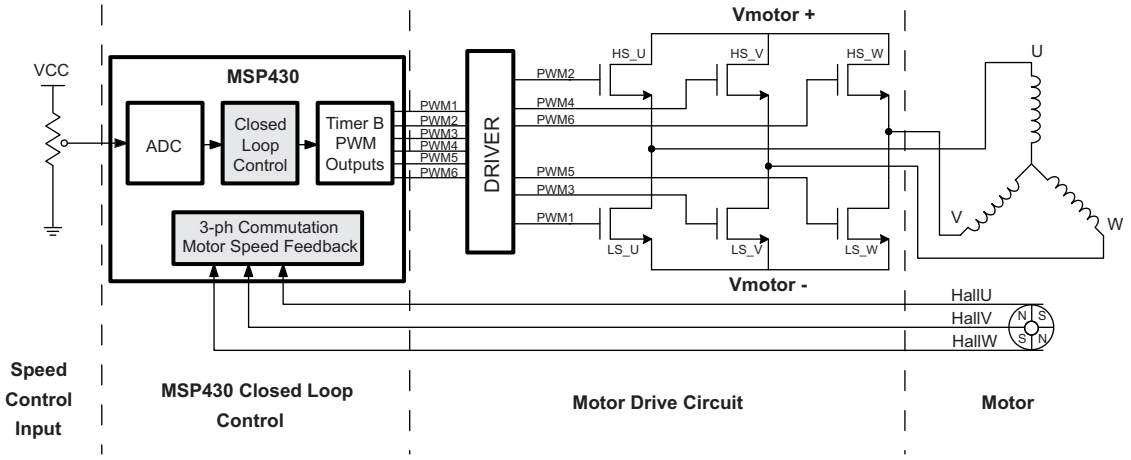

Figure 7. Closed-Loop Control – MSP430 Based Implementation

The speed-control input, motor-drive circuit, and 3-phase BLDC commutation implemented in the closed-loop solution is the same as the open-loop solution discussed previously.

3.2.1 Closed-Loop Control

The MSP430 implements the closed-loop control plus the 3-phase motor commutation. A PI controller is used to implement the closed-loop control that uses both the speed-control input and the actual motor-speed feedback to update the timer PWM duty cycle that, in turn, controls the motor speed.

3.2.2 Motor-Speed Feedback (Using Hall Sensors)

The actual motor speed is calculated by tracking the time period between successive Hall events, which represents a part of the mechanical cycle of the motor. In a 3-phase BLDC motor control, one electrical cycle has six Hall states and, depending on the number of poles pairs in the motor, the electrical angle measured between successive Hall state changes can be translated to a respective mechanical angle. For example, for a 4-pole 3-phase BLDC motor with three Hall sensors, one mechanical revolution is equal to two electrical cycles; for an 8-pole 3-phase BLDC motor with three Hall sensors, one mechanical revolution is equal to four electrical cycles.

For closed-loop control implementation, it is not required to compute the actual speed of the motor in rpm. A timer counter (TimerA0) with the same time base as the PWM timer (TimerB in this case) is used to track the time period between two successive Hall events and is interpreted as PWM period counts. Based on the speed-control input, the expected Hall events per second and, in turn, the expected PWM period counts between successive Hall events are computed and this represents the expected motor speed. This is compared against the actual speed measured (or the actual PWM counts measured between two Hall events) and the difference is input to the PI controller.

3.3 Closed-Loop Controller (PI Controller)

PID controllers are the most commonly used closed-loop controllers in the industry. This application report implements a PI controller with the derivative gain parameter set to zero. For more details regarding PID controllers and why a PI controller is implemented, see Appendix A.

Figure 8 shows the closed-loop control implementation in the demo firmware.

The inputs of the closed-loop control are normalized to PWM period counts, and the output of the PI controller is translated to PWM duty cycle counts that, in turn, represent the motor speed.

The speed-input control is provided by an analog potentiometer, and the MSP430 uses the integrated A/D converter to measure the potentiometer value. In this case, a 12-bit ADC is used and, therefore, the speed input range is 0 to 2^{12} ADC counts. 0 to 2^{12} ADC counts represents 0% to 100% of the motor speed, and this is interpreted in PWM duty cycle counts as Desired_PWM_Dutycycle, which can vary from 0 to TIMER_PWM_PERIOD counts.

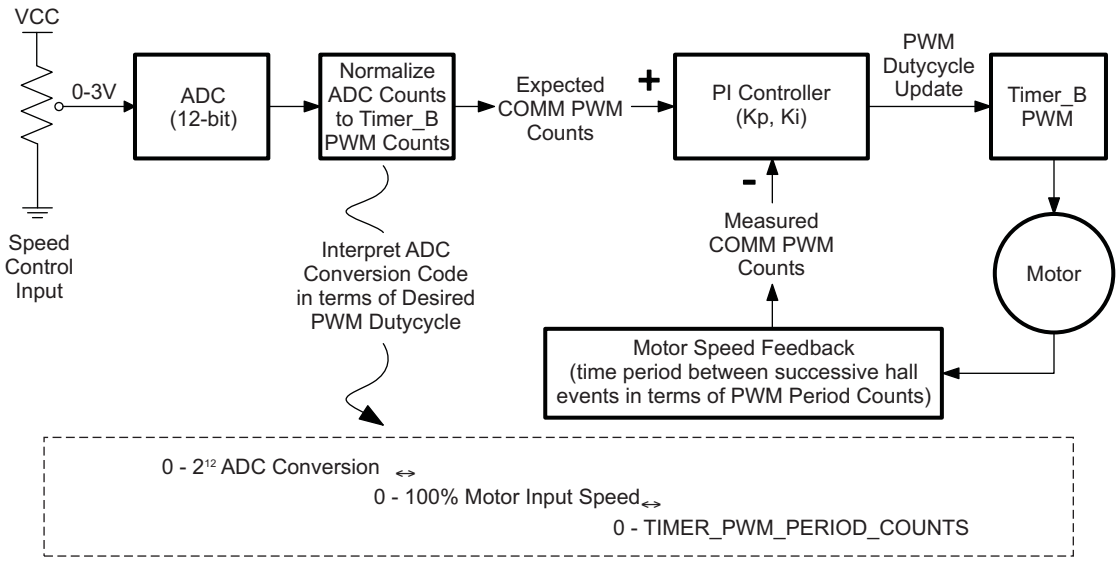

Figure 8. Closed-Loop Control Implementation (PI Controller)

The Desired_PWM_Dutycycle is also translated as Expected_COMM_PWM_Counts, and this represents the number of PWM period counts that are expected between successive commutation steps or Hall events when running at the desired speed.

$$\text{Expected\_COMM\_PWM\_Counts} = \left(\frac{\text{Motor\_Speed\_Input(\%)}}{100}\right) \times \text{MAX\_HALL\_ISRs\_1SEC}$$

Where MAX_HALL_ISRs_1SEC is a predefined value that represents the maximum number of Hall events expected in one second when the motor is running at its maximum speed and is calculated as:

$$\text{MAX\_HALL\_ISRs\_1SEC} = \left(\frac{\text{MAX\_MECH\_SPEED\_IN\_RPM}}{60}\right) \times \left(\frac{\text{NUM\_MOTOR\_POLES}}{2}\right) \times 6$$

The motor-speed feedback is measured as the time period between successive Hall events or Hall input interrupts and is represented in terms of PWM period counts as Measured_COMM_PWM_Counts. The difference between the expected and measured COMM PWM counts gives the proportional error and this is accumulated every time to generate the integral error. With the proportional and the integral error, the PI control output is calculated as:

$$\text{PI\_Control\_Output} = (\text{Kp} \times \text{Proportional\_Error}) + (\text{Ki} \times \text{Integral\_Error})$$

Where Kp and Ki are the proportional and integral gain parameters, respectively.

The PI-controller output is translated in terms of PWM duty cycle as PID_Applied_PWM_Dutycycle, which controls the motor speed by using this equation:

$$\text{PID\_Applied\_PWM\_Dutycycle} = \text{Desired\_PWM\_Dutycycle} + \left(\frac{\text{PI\_Control\_Output}}{\text{Divide\_Factor}}\right)$$

In the previous equations, Kp, Ki, and Divide_Factor are selected during control parameter tuning and do not change during runtime. During PI parameter tuning, depending on the motor-input speed, two different sets of Kp and Ki are applied to achieve closed-loop operation for wider motor-speed range.

3.4 Closed-Loop Control Firmware

This section discusses the closed-loop motor control implementation in the demo firmware that is provided with this application report.

The framework of the closed-loop control is very similar to the open-loop control solution discussed in Section 2.3. An additional timer (Timer_A) is used to measure the motor-speed feedback via the Hall inputs events, and a PI controller is used to provide the closed-loop control. With the closed-loop control implementation, there is no need to have a separate start-up routine like in the open-loop control; the PI controller takes care of the start-up scenario as well. Table 3 summarizes the differences in the two control loop implementations.

Table 3. Differences Between Open-Loop and Closed-Loop Implementations

	Open-Loop Control	Closed-Loop Control
Feedback	No feedback	Motor-speed feedback (using Hall inputs) – measured by the motor-speed feedback timer (Timer_A)
Control Implementation	Based on speed-input control, increments or decrements PWM duty cycle by bucket step value	Based on speed-input control and motor-speed feedback, the PI controller updates the PWM duty cycle values.
Motor Start-up	Requires a separate start-up routine as open-loop control does not provide a controlled start-up	PI controller takes care of the motor start-up
Control Parameter Tuning	Not required	PI (Kp and Ki) gain parameter tuning required

Table 4 lists all the user configurable parameters available for the closed-loop motor control implementation. The clock timing definitions listed in Table 2 for the open-loop control solution apply to the closed-loop control as well.

Table 4. Closed-Loop Control – User Configurable Parameter Definitions

Configurable Parameter (Defined inmain_closed_loop Header File)	Comments	Demo Firmware Values
Closed-Loop Control Definitions		
NUM_MOTOR_POLES	Defines number of motor poles (see Note (1))	6 poles
MAX_MECH_SPEED_IN_RPM	Defines maximum motor speed in rpm (see Note (1))	6000 rpm
MAX_ELEC_SPEED_IN_RPS	Maximum number of electrical rotations per second $\left(\frac{\text{MAX\_MECH\_SPEED\_IN\_RPM}}{60} \times \frac{\text{NUM\_MOTOR\_POLES}}{2}\right)$	$\left(\frac{6000}{60} \times \frac{6}{2}\right) = 300$
MAX_HALL_ISRs_1SEC	(MAX_ELEC_SPEED_IN_RPS × 6) Because one electrical cycle has six Hall states	300 x 6 =1800
PID_EXECUTE_PWM_PERIODS	Defines the number of timer PWM periods after which the PID control loop is required to execute (see Note (2)).	10
LOW_DUTYCYCLE_PERCENTAGE	Represents the percentage speed input below which a different set of Kp and Ki gain parameters is used (see Note (3)).	50%
LOW_KP_KI_DUTYCYCLE	Represents the percentage speed input below which a different set of Kp and Ki in terms of PWM duty cycle counts is used. $= \left(\text{TIMER\_PWM\_PERIOD} \times \left(\frac{\text{LOW\_DUTYCYCLE\_PERCENTAGE}}{100}\right)\right)$	$\left(1023 \times \frac{50}{100}\right) = 511$

(1) These specs refer to the Hurst Motor DMA0102024C that is used in this demo application.
(2) Timer PWM period interval is used as a heart beat signal to keep track of the PID execute counter.
(3) This value is dependent on outcome of the PI gain parameter tuning. In this case, it's used to increase the motor-speed control range while implementing the PI controller.

The firmware implementation of the closed-loop control is shown in Figure 9 and Figure 10.

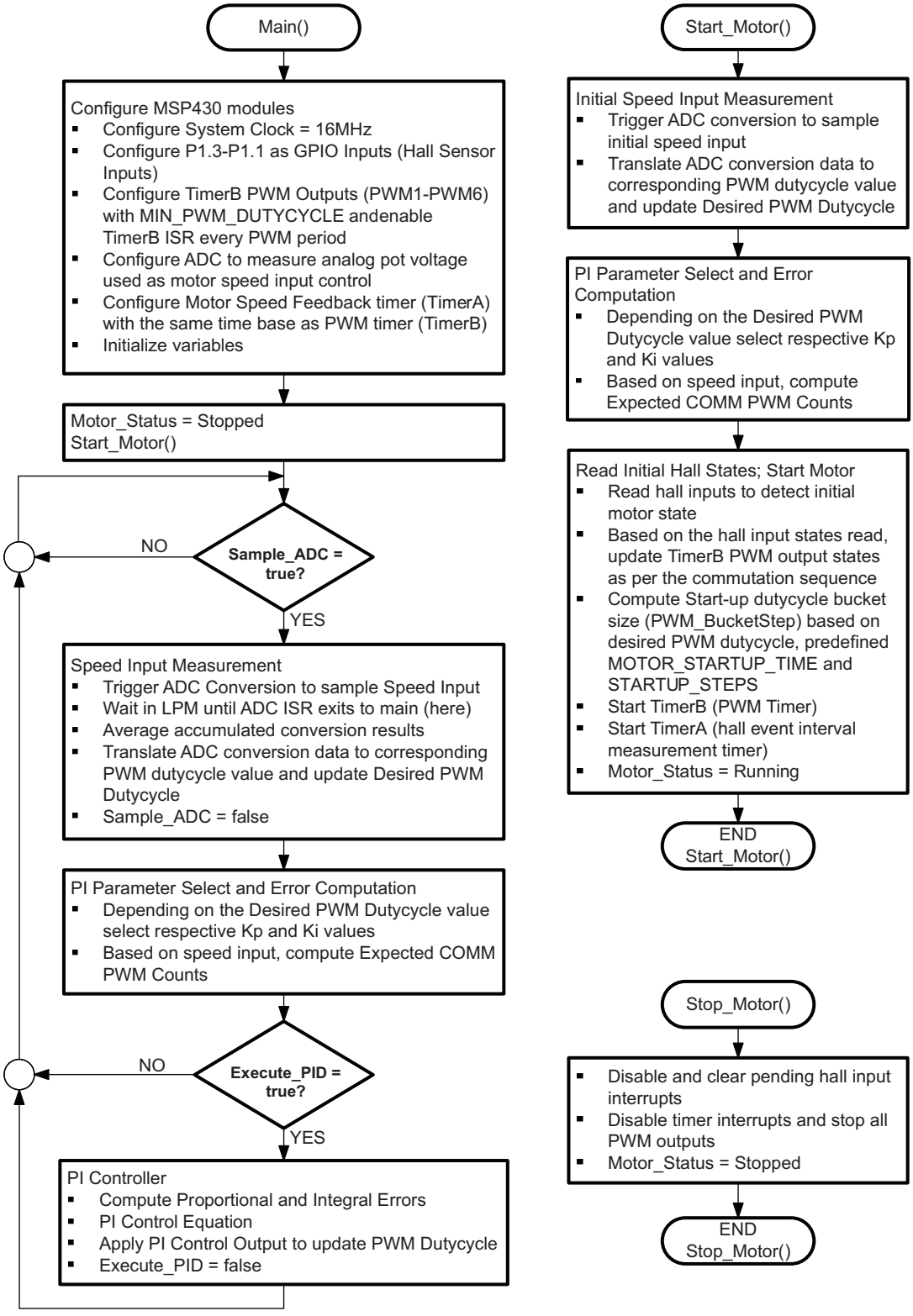

Figure 9. Flowchart – Closed-Loop Control Functions

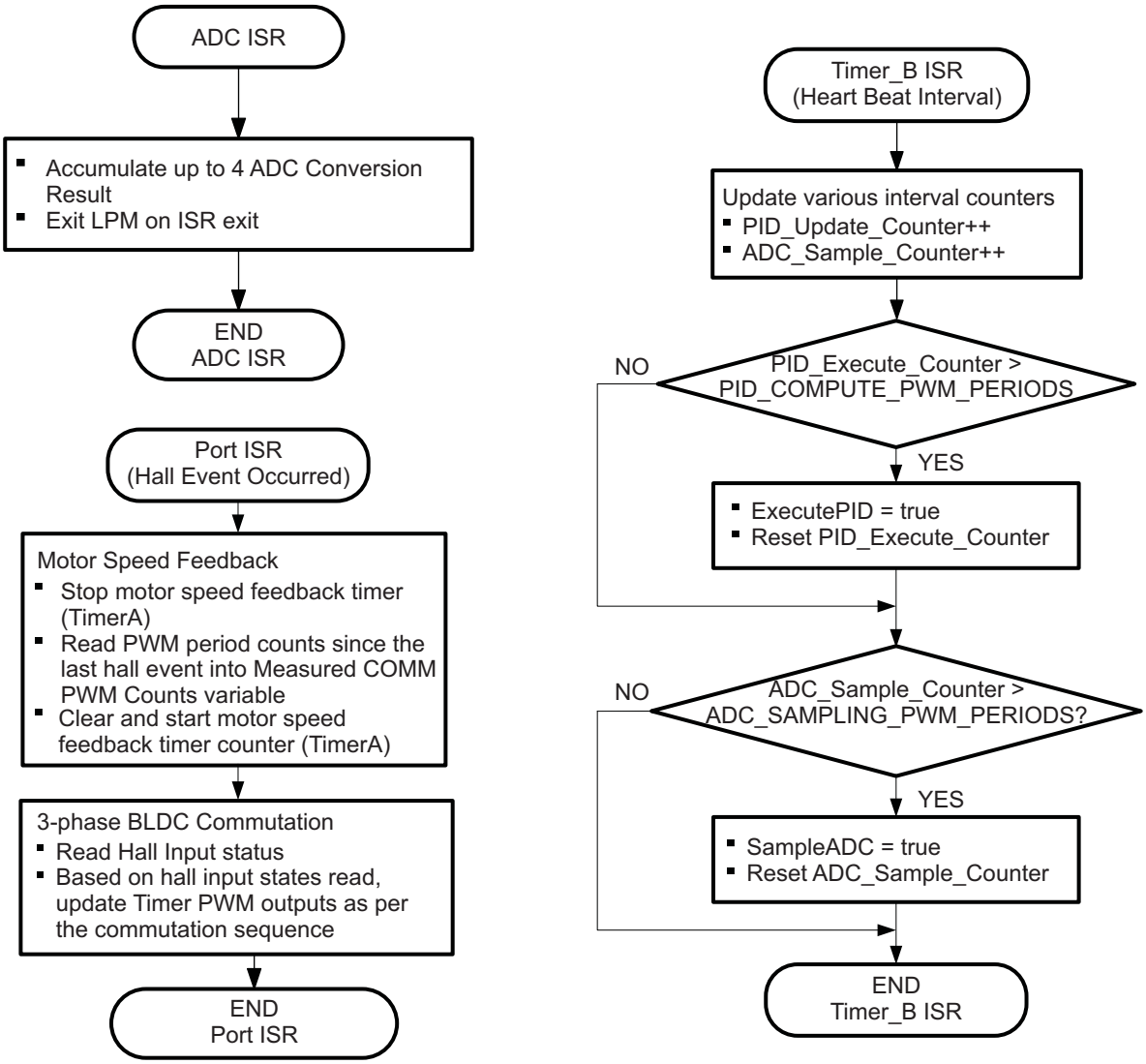

Figure 10. Flowchart – Closed-Loop Control ISRs

In the demo firmware, the PI controller is implemented in C. To reduce the number of instructions required to implement the PI control loop, assembly coding can be used.

4 Current Monitor Limit and Overcurrent Protection

The current through the motor and half-bridge circuitry can be monitored or tracked by converting the current to equivalent voltage via an R_{sense} resistor. Integrated ADCs in the MSP430 can be used to sample this R_{sense} voltage to provide current and torque control or just limit the motor current by comparing with software thresholds. The ADC can be configured to sample the R_{sense} voltage every fixed interval time, and with this firmware framework, the fixed intervals for R_{sense} voltage sampling can be generated by including another interval counter in the timer PWM ISR or the heart beat function.

To provide immediate overcurrent protection to ensure the exceeded motor current does not damage the power FETs in the motor-drive circuit or other external circuitry in the system, the motor current can be compared against a known reference using the integrated comparator in the MSP430. The comparator can be configured to trigger an interrupt on its output state change. When the motor current exceeds the current threshold, the PWM output can either be turned off as long as the comparator output is high or other necessary steps can be taken within the ISR to handle the overcurrent condition.

Overcurrent protection can also be handled automatically by using the comparator output to enable and disable external circuitry. MSP430F5xx devices that have the integrated Comparator_B module have in-built hysteresis generation capability that aids automatic enable and disable of external circuitry during overcurrent conditions.

Example code demonstrating overcurrent interrupt and automatic enable and disable of external circuitry implementations is provided with this application report. Note that the overcurrent examples are standalone code and are currently not integrated with the open-loop or closed-loop control code.

Figure 11 shows example of typical current monitor and overcurrent protection circuitry.

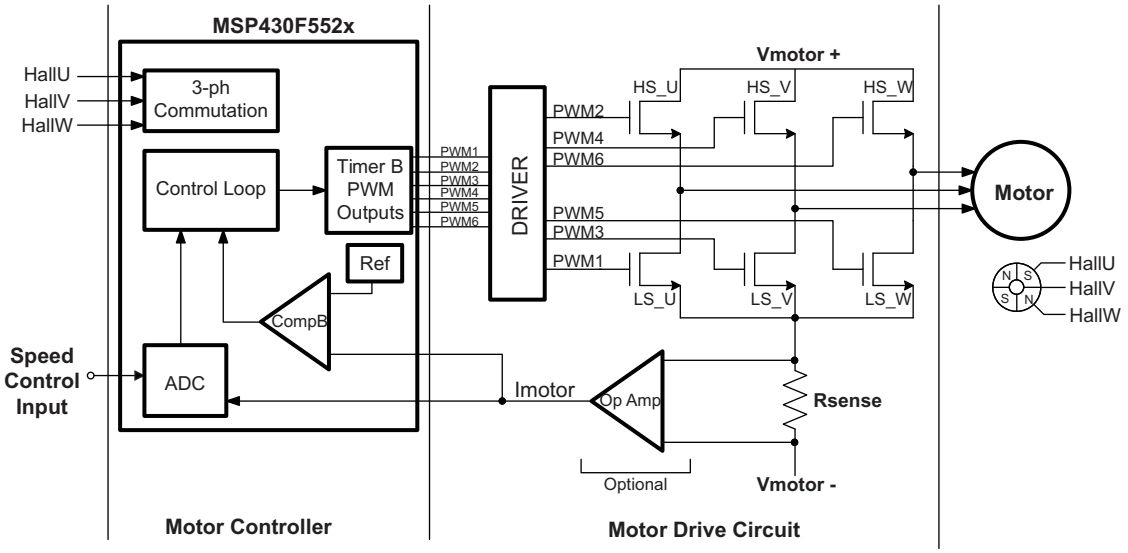

Figure 11. Current Monitor and Overcurrent Protection Circuitry

The motor current can vary from hundreds of milliamps to amps and, therefore, a small value of R_{sense} resistor should be selected to reduce the power dissipation across the resistor. Depending on the value of R_{sense} resistor selected, the circuit may require an operational amplifier to amplify the voltage across R_{sense} before feeding it as an input to the ADC or comparator module.

The voltage across R_{sense} resistor or the output of the operational amplifier represents the equivalent motor current and is fed into the integrated ADC for current monitoring or tracking. For overcurrent protection, this voltage is fed into the integrated comparator and compared against a reference threshold. In the case of MSP430F5xx and MSP430F6xx devices, the shared reference module can provide up to 32 voltage reference levels, giving more flexibility in selecting an internal reference threshold. The comparator interrupt can be used to control the timer PWM output controlling the motor-drive circuit.

Comparator_Config_for_Overcurrent_Limit.c shows how to configure the comparator module along with the internal shared reference (which is used to provide the current limit threshold) is included in the associated code files. Note that the current limit threshold represented by the internal shared reference is dependent on the motor that is used and the power FETs that are used in the motor-drive circuitry and should be changed accordingly.

5 References

1. *MSP430x5xx/MSP430x6xx Family User's Guide* (SLAU208)
2. *MSP430F551x, MSP430F552x Mixed Signal Microcontroller Data Sheet* (SLAS590)
3. http://en.wikipedia.org/wiki/PID_controller
4. *MSP430F5xx and MSP430F6xx Core Libraries* (SLAA448)

Appendix A PID Controller

Proportional integral derivative (PID) controllers are the most commonly used closed-loop controllers used in the industry. PID parameters affect the following system dynamics with respect to closed-loop step response.

- Rise time - time taken for the output to rise beyond 90% of the desired level
- Overshoot - how much higher the peak level is compared to the steady-state level
- Settling time - time taken by the system to converge to its steady state
- Steady-state error - the difference between the steady-state output and the desired output

The effects of the PID-control parameters Kp, Ki, Kd can be summarized as:

- Kp decreases the rise time
- Ki significantly decreases the steady-state error
- Kd reduces the overshoot and settling time

The integral parameter (Ki) almost eliminates steady-state error in the output. Lower Ki values slowly push the motor speed to the expected set point, and higher Ki values can cause hunting around the set point speed.

The proportional parameter (Kp) provides fast response to sudden changes in load, controlling the rise time. This parameter is typically much higher than Ki, so that relatively small deviations in speed are corrected while Ki slowly moves the speed to the set point.

The derivative parameter (Kd) provides very fast response to sudden changes in motor speed. However, with simple PID controllers, it can be difficult to generate a derivative term in the output that has any significant effect on motor speed. Therefore, PI controllers (with derivative parameter Kd = 0) are used in most closed-loop processes to provide a balance of complexity and capability. Also, these controllers are simpler to tune compared to the PID controllers.

The common PID tuning methods involve manual tuning: Ziegler-Nichols tuning and Cohen-Coon tuning. In this application implementation, the PI controller was tuned using the manual tuning method. It involved subjecting the system to a step change in input, measuring the output as a function of time, and using this response to determine the control parameters. In this case, an Microsoft® Excel® spreadsheet was used to formulate the step input, monitor the output timing, and determine the tuning parameters for the system.

For more details regarding PID controllers and different PID tuning methods, see reference [3].

Appendix B Associated Files

Table 5 lists the associated source files for the sensored 3-phase BLDC motor control solution.

Table 5. Associated Files – C Source

C Source Files/Folders	Comments
Open-Loop Control	Main_open_loop.c main_open_loop.h and HallSensor.h
	main_open_loop.h header file included in all user configurable definitionsfor open loop motor control. These values are listed in Table 2.
	HallSensor.h headerfile includes the Hall commutation states with respect to timer PWM output (see Table 1).
Closed-Loop Control	Main_closed_loop.c main_closed_loop.h and HallSensor.h
	main_open_loop.h header file included in all user configurable definitions for closed loop motor control.These values are listed in Table 3.
	HallSensor.h headerfile includes the Hall commutation states with respect to timer PWM output (see Table 1).
F5xxCore Library	Hal_PMM.c and Hall_UCS.c Hal_PMM.h and Hall_UCS.h
	The F5xx/F6xx core library isused to configure the clocks and the PMMsettings of the MSP430F552x device used in the demo firmware.
Overcurrent Protection	OverCurrent_Interrupt.c OverCurrent_Protection_AutoEnable.c
	Example code that demonstrates overcurrent protection using the integrated Comparator_B module.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Audio	www.ti.com/audio	Communications and Telecom	www.ti.com/communications
Amplifiers	amplifier.ti.com	Computers and Peripherals	www.ti.com/computers
Data Converters	dataconverter.ti.com	Consumer Electronics	www.ti.com/consumer-apps
DLP® Products	www.dlp.com	Energy and Lighting	www.ti.com/energy
DSP	dsp.ti.com	Industrial	www.ti.com/industrial
Clocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
Interface	interface.ti.com	Security	www.ti.com/security
Logic	logic.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Power Mgmt	power.ti.com	Transportation and Automotive	www.ti.com/automotive
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Mobile Processors	www.ti.com/omap		
Wireless Connctivity	www.ti.com/wirelessconnectivity		
	TI E2E Community Home Page		e2e.ti.com

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265