Application Note

Three-Phase Hall Sensor BLDC Driver Using the Z32F128 MCU

AN038101-0816

Abstract

This MultiMotor Series application note investigates the closed- and open-loop control of a 3-phase brushless direct current (BLDC) motor using a Z32F128 ARM Cortex M3 MCU. Zilog's ZNEO32! family of microcontrollers is designed specifically for motor control applications and, with this MultiMotor Series, features an on-chip integrated array of application-specific analog and digital modules using the MultiMotor Development Kit. The result is fast and precise fault control, high system efficiency, on-the-fly speed/torque and direction control, as well as ease of firmware development for customized applications.

This document further discusses ways in which to implement motor control using a sensored feedback control system, and fault protection. Test results are based on using a MultiMotor Development Kit equipped with a Z32F128 MCU, a 3-phase MultiMotor Development Board, and a 3-phase, 24VDC, 30W, 3200RPM BLDC motor.

Note: The source code file associated with this application note, AN0381-SC01, is available free for download from the Zilog website. This source code has been tested with Keil µVision version 5.18.0.0. Subsequent releases of Keil µVision may require you to modify the code supplied with this application note.

Note: The term *MultiMotor* refers to the ability to operate the included BLDC motor with multiple driving scheme program modules. The driving scheme implemented in this application note is the AN0381 program module.

Features

The features of this MultiMotor Series application include:

- Hall sensor commutation
- Motor speed measurement
- Motor protection logic
- Closed-loop or open-loop control for precise speed regulation
- Potentiometer-adjustable motor speed
- Selectable control of motor direction

- Storage of motor condition parameters into an external EEPROM (for more details, refer to the Zilog application note titled Implementing a Data Logger with Spansion SPI Flash (AN0360).
- UART Interface for PC control
- LED to indicate motor operation
- LED to indicate UART control
- LED to indicate a fault condition

Discussion

The Z32F128 series Flash microcontrollers are based on Zilog's advanced 32-bit ARM Cortex M3 CPU core. These Z32F128 devices set a standard of performance and efficiency with an internal clock speed up to 80MHz. Instructions are executed with a single cycle and up to 128 kilobytes of internal Flash memory are accessible by the Z32F128 ARM Cortex MCU, 32-bit at a time, to improve processor throughput. Up to 12KB of internal RAM provides storage of data, variable and stack operations.

The Z32F128 MCU features a flexible16-bit Pulse Width Modulation (PWM) module with three complementary pairs or six independent PWM outputs supporting dead-band operation and fault protection trip input. These features provide multiphase control capability for a variety of motor types, and ensure safe operation of the motor by providing pulse-by-pulse or latched fast shutdown of the PWM pins during a fault condition.

Also featured are up to twelve single-ended channels of 12-bit analog-to-digital conversion (ADC) with a sample and hold circuit. One operational amplifier performs current sampling, and one comparator performs overcurrent limiting or shutdown. A high-speed ADC enables voltage and current sensing, while dual-edge interrupts and a 16-bit timer provide a Hall-effect sensor interface.

A four channel UART with DMA support, 2 I2C and 2 SPI peripherals provide asynchronous communication to external devices.

The Z32F128 ARM Cortex M3 provides timer block consists of 6 channels of 16-bit general-purpose timers to support periodic interval timer, PWM pulse, one-shot timer, and capture mode.

The single-pin debugger and programming interface simplifies code development and allows easy in-circuit programming.

Features

The Z32F128 MCU offers the following features:

- High performance low-power Cortex-M3 core
- 128 KB code Flash memory with cache function
- 12 KB SRAM
- 3-phase motor PWM with ADC triggering function

  - 2 channels
- 1.5 MSPS high-speed ADC with burst conversion function
  - 3 units with 16 channel input
- Built-in Programmable Gain Amplifier (PGA) for ADC inputs
  - 4 channels
- Built-in analog comparator
  - 4 channels
- System fail-safe function by clock monitoring
- XTAL OSC fail monitoring
- Precision internal oscillator clock (20 MHz ±3%)
- Watchdog timer
- Six 16-bit general purpose timers
- Quadrature encoder counter
- External communication ports: 4 UARTs, 2 I2Cs, 2 SPIs
- High current driving port for UART and photo couplers
- Debug and emergency stop function
- Real-time monitoring of peripherals in debug mode
- JTAG and SWD in-circuit debugger
- Various memory size and package options
- LQFP-80, LQFP-64
- Industrial grade operating temperature (–40°~ +85°)

Figure 1 shows a block diagram of the Z32F128 MCU architecture.

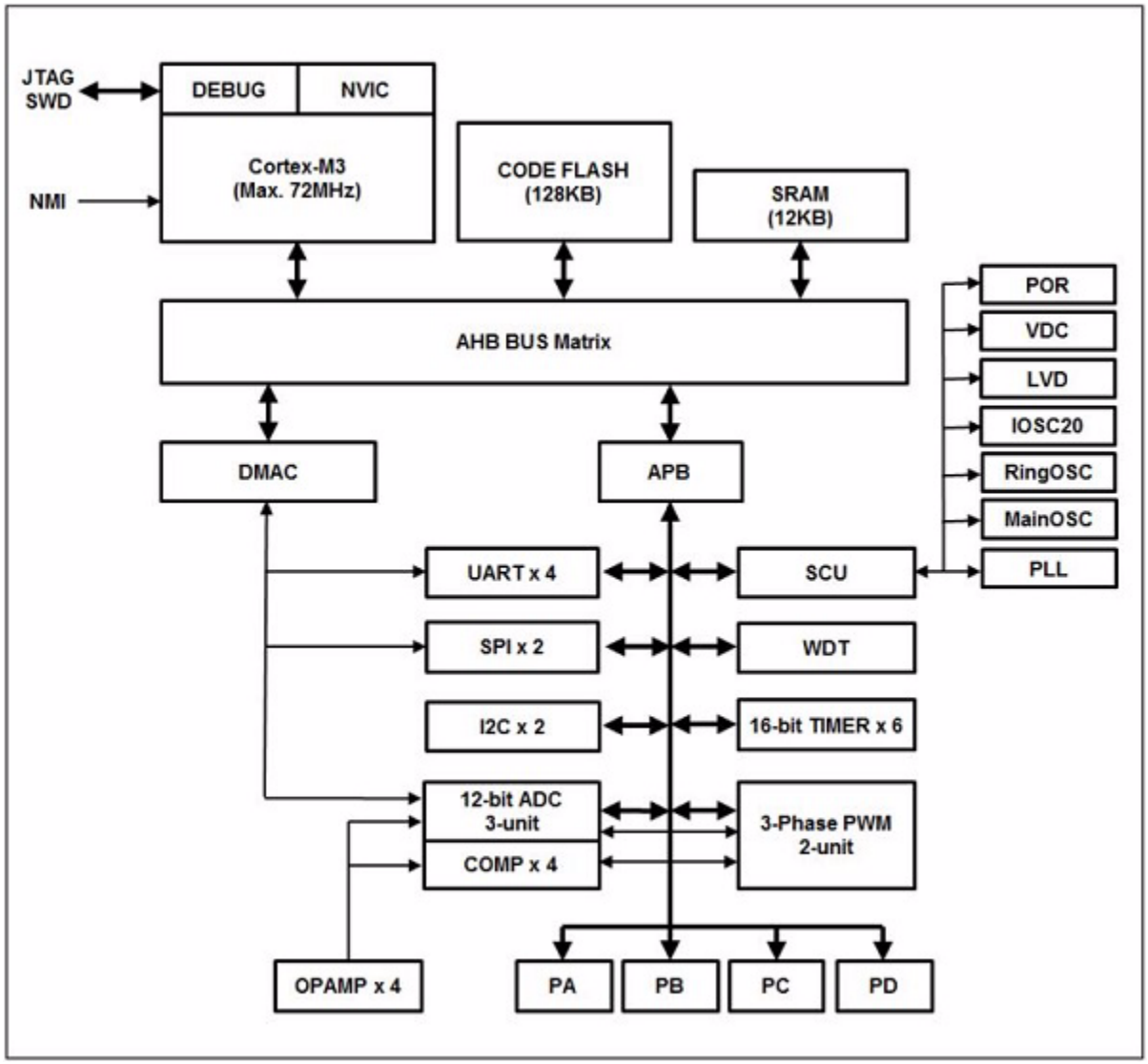

Figure 1. Z32F128 MCU Block Diagram

Hardware Architecture

In a Brushed DC motor, commutation is controlled by brush position. In a BLDC motor, however, commutation is controlled by the supporting circuitry. The rotor's position must therefore be fed back to the supporting circuitry to enable proper commutation.

Two different techniques can be used to determine rotor position:

Hall Sensor-Based Commutation. In the Hall sensor technique, three Hall sensors are placed inside the motor, spaced 120 degrees apart. Each Hall sensor provides either a High or Low output based on the polarity of magnetic pole close to it. Rotor position is deter-

mined by analyzing the outputs of all three Hall sensors. Based on the output from hall sensors, the voltages to the motor's three phases are switched.

The advantage of Hall sensor-based commutation is that the control algorithm is simple and easy to understand. Hall sensor-based commutation can also be used to run the motor at very low speeds. The disadvantages are that its implementation requires both separate Hall sensors inside the motor housing and additional hardware for sensor interface.

Sensorless Commutation. In the sensorless commutation technique, the back-EMF induced in the idle phase is used to determine the moment of commutation. When the induced idle-phase back-EMF equals one-half of the DC bus voltage, commutation is complete.

The advantage of sensorless commutation is that it makes the hardware design simpler. No sensors or associated interface circuitry are required. The disadvantages are that it requires a relatively complex control algorithm and, when the magnitude of induced back-EMF is low, it does not support low motor speeds.

When a BLDC motor application requires high torque, or when the motor is moving from a standstill, the Hall sensor commutation technique is an appropriate choice. A motor used in an electric bicycle application, for example, requires high initial torque and is a perfect application for Hall sensor commutation.

Furthermore, two voltage application techniques can be applied, based on the configuration of the supply-to-motor windings:

Sinusoidal. Sinusoidal voltage is continuously applied to the three phases. Sinusoidal voltage provides a smooth motor rotation and fewer ripples.

Trapezoidal. DC voltage is applied to two phases at a time, and the third phase remains idle. Trapezoidal voltage is less complex to implement. The idle phase is generating the BEMF from the rotating magnet that passes the unenergized idle phase and provides the BEMF zero-crossing data.

How Hall Sensor Commutation Works

To better understand how Hall sensor commutation works, let's look at how it's implemented with a two-pole motor. Six different commutation states are required to turn the rotor one revolution. The motor’s commutation states are shown in Figure 2.

Table 1 indicates the relationship between the Hall sensor output and phase switching operations shown in Figure 2.

Table 1. Relationship Between Hall Sensor Output and Phase Switching

State	Hall A	Hall B	Hall C	Phase B	Phase C	Phase A
1	0	1	1	0	$+V_{DC}$	$-V_{DC}$
2	0	0	1	$+V_{DC}$	0	$-V_{DC}$
3	1	0	1	$+V_{DC}$	$-V_{DC}$	0
4	0	1	0	0	$-V_{DC}$	$+V_{DC}$
5	1	1	0	$-V_{DC}$	0	$+V_{DC}$
6	1	0	0	$-V_{DC}$	$+V_{DC}$	0

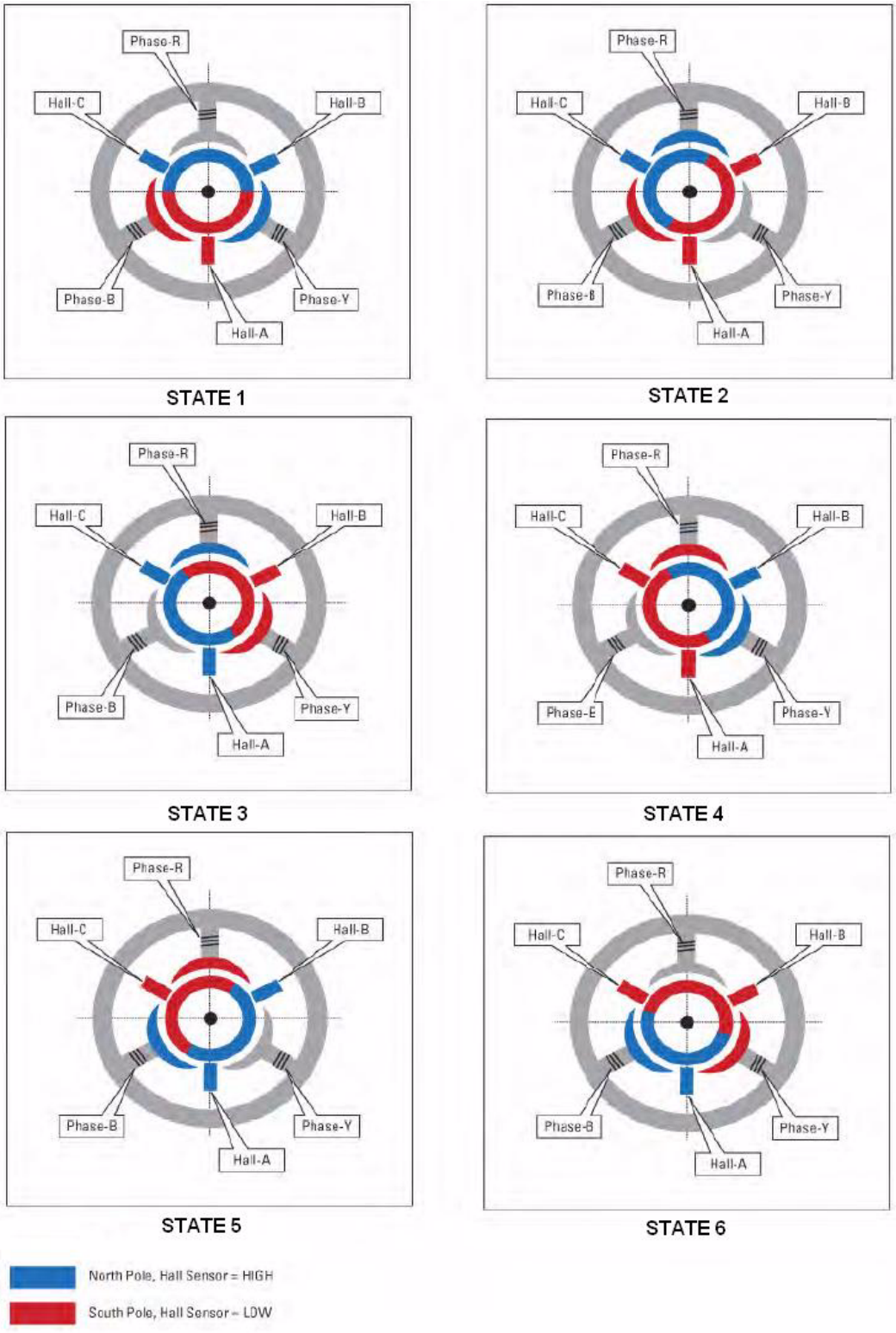

Figure 2. Hall Sensor Commutation States for a 2-Pole Motor

Using the Z32F128 MCU with a 3-Phase Hall Sensor BLDC Motor Controller

Figure 3 offers a visual overview of the 3-phase Hall sensor BLDC motor controller. For more details about hardware connections, see Appendix A. Schematic Diagrams on page 13.

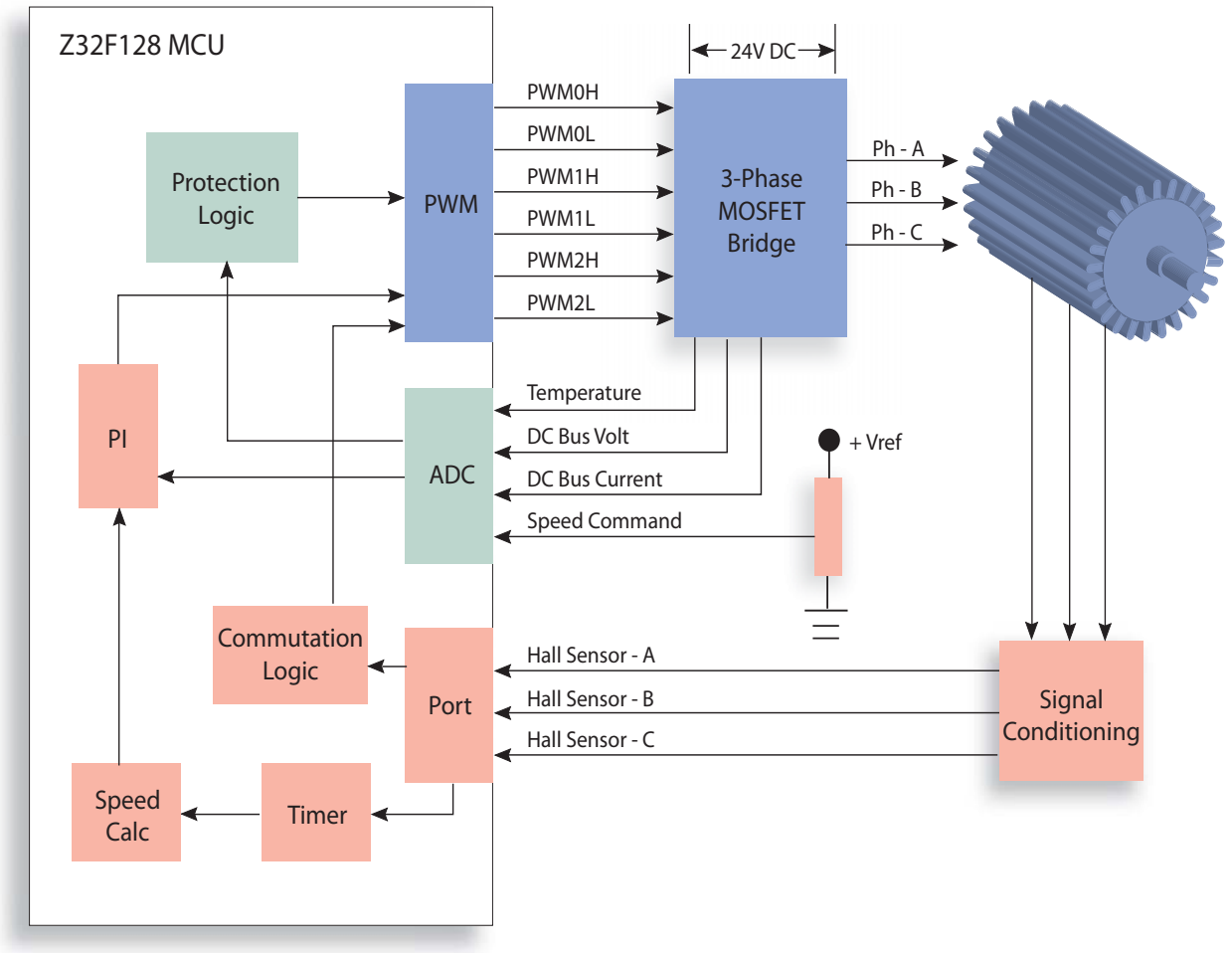

Figure 3. 3-Phase Hall Sensor BLDC Motor Controller Block Diagram

Hardware Design

The design involves running the BLDC motor in a closed loop or an open loop, with speed as set by a potentiometer. As shown in the architecture diagram, the design generates PWM voltage via the Z32F128 MCU's PWM module to run the BLDC motor.

After the motor is running, the states of the three Hall sensors change based on the rotor position. Voltage to each of the three motor phases is switched based on the state of the sensors (commutation). Hall sensor interrupts capture timer ticks every sixty degrees to

measure the rotor speed of the motor. Other peripheral functions can be used to protect the system in case of current overload, under- or overvoltage, and overtemperature.

The hardware is described in the following sections.

Three-Phase Bridge MOSFET

The three-phase bridge MOSFET consists of six MOSFETs connected in bridge fashion used to drive the three phases of the BLDC motor. The DC bus is maintained at 24 V, which is same as the voltage rating of the BLDC motor. A separate Hi-Lo gate driver is used for each high- and low-side MOSFET phase pair, making the hardware design simpler and robust. The high-side MOSFET is driven by charging the bootstrap capacitor.

The DC bus voltage is monitored by reducing it to suitable value using a potential divider. The DC bus current is monitored by putting a shunt in the DC return path. An NTC-type temperature sensor provides an analog voltage output proportional to temperature.

PWM Module

The Z32F128 MCU contains a six-channel, 16-bit PWM module configured in this application to run in Complementary Mode. The switching frequency is set to 20 KHz. The PWM outputs are controlled according to the inputs from the Hall sensors.

The inputs from the Hall sensors determine the sequence in which the three-phase bridge MOSFET is switched. The Duty cycle of the PWM is directly proportional to the accelerator potentiometer input. The change in the duty cycle controls the current through the motor winding, thereby controlling motor torque.

Commutation Logic

The Hall sensors are connected to ports PB8, PB9, and PB10 on the Z32F128 MCU. An interrupt is generated when the input state on any pin changes. An interrupt service routine checks the state of all three pins and accordingly switches the voltage for the three phases of the motor.

Trapezoidal commutation is used for this application to make implementation simple. In this process of commutation, any two phases are connected across the DC bus by switching the top MOSFET of one phase and bottom MOSFET of another phase ON. The third phase is left un-energized (both top and bottom MOSFET of that phase are switched OFF).

Speed Measurement

One out of the three Hall sensors is used to capture the Timer0 ticks, which represent the actual Hall period for closed loop calculations.

Closed Loop Speed Control

Closed-loop speed control is implemented using a PI loop, which works by reducing the error between the speed set by the potentiometer and actual motor speed. The output of this PI loop changes the duty cycle of the PWM module, thereby changing the average

voltage to the motor, and ultimately changing the power input. The PI loop adjusts the speed at the same rate as the Hall frequency from one of three Hall sensors.

Protection Logic

The ADC module periodically checks DC bus voltage, DC bus current, and temperature. If these values go beyond the set limits, the motor is shut down. These checks are timed by Timer0 interrupt.

Over-Current Hardware Protection

The Z32F128 MCU has a built-in comparator that is used to shut down the PWM for over-current protection. When the current exceeds the set threshold, a PWM Comparator Fault is generated to turn OFF the PWM Module.

Software Implementation

During implementation of the software, the following actions are performed:

Initialization. Hardware modules are initialized for the following functions:

- Switch from an internal to an external oscillator for system operation
- Enable alternate functions on the respective pins for the ADC, Comparator, and UART, and to drive the LEDs
- Configure Timer0 to run in Continuous Mode to capture the Hall period timing
- Configure the comparator to shut down the PWM module when an overcurrent results
- Enable the Op Amp to measure the DC bus current flowing to the motor
- Configure the ADC to read analog values such as DC bus voltage, current, temperature, and acceleration potentiometer (only one channel at a time)
- Configure the PWM module for the individual mode of operation with a 20 kHz switching frequency, control output depending on the values in the PWMOUT Register, and drive the PWMOUT as defaulted to a low off state at Power-On Reset and at any Reset

Interrupt. The Port Port B(odd and even) interrupt controls commutation. The Hall sensor output is read on pins PB8:10, the software performs its filtering operation, and the switching sequence of the MOSFET is determined. The PWM timer interrupt is used to time periodically occurring tasks and for the background loop to read analog values from different channels and average these values, update the LED indicator status, and update the read parameters on the UART.

For a visual representation of the application, see Appendix B. Flowcharts on page 17.

Testing

This section describes how to run the code and demonstrate this sensorless brushless motor application including its setup, implementation and configuration, and the results of testing.

Equipment Used

The following equipment is used for the setup; the first four items are contained in the MultiMotor Development Kit (ZMULTIMC100ZCOG).

- MultiMotor Development Board (99C1358-0001G)
- 24V AC/DC power supply
- LINIX 3-phase 24VDC, 30W, 3200RPM BLDC motor (45ZWN24-30)
- Opto-Isolated UART-to-USB adapter (99C1359-001G)
- Z32F128 MultiMotor MCU Module (99C1461-001G) – Order separately
- Opto-Isolated USB SmartCable (99C0968) – Order separately
- Digital Oscilloscope or Logic Analyzer

Hardware Setup

Figure 4 shows the application hardware connections.

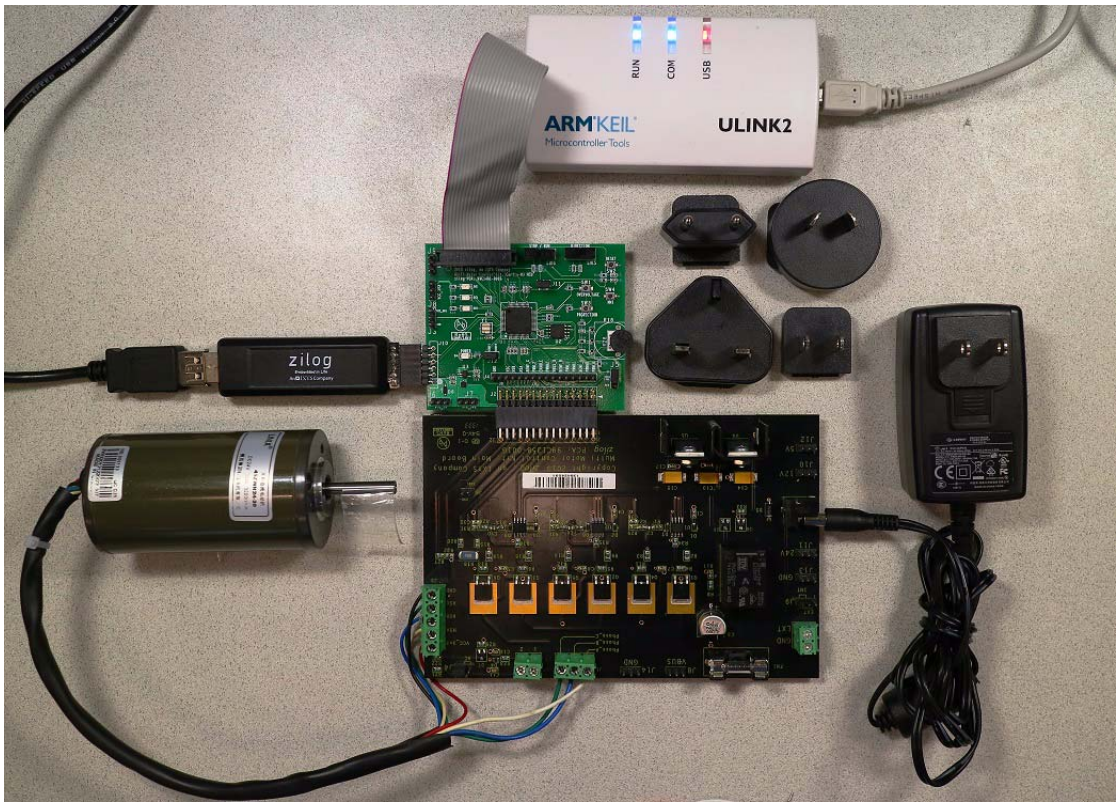

Figure 4. The MultiMotor Development Kit

Procedure

Observe the following procedure to test the motor in sensorless block commutation on the Z32F128 MCU Module.

1. Download and install Keil MDKARM µVision IDE version 5.14.0.0 (or newer) on your PC from the Keil website.
2. Download the AN0381-SC01.zip source code file from the Zilog website and unzip it to an appropriate location on your PC.
3. Connect the hardware as shown in Figure 4:
   - The cables from the Keil ULINK2 debugger and the UART-to-USB adapter must be connected to two of the PC's USB ports.
   - Download and install the drivers for the Keil ULINK2 and the UART-to-USB adapter, if required.
4. Power up the MultiMotor Series Development Board using the 24 V DC adapter that is included in the Kit.
5. Using a serial terminal emulation program such as HyperTerminal, TeraTerm, or RealTerm, configure the serial port to 57600-8-N-1-N. A console screen should appear on the PC which will show the status of the motor and allow changes to the motor's operation.
6. Launch Keil µVision version 5.14.0.0 (or newer), select **Open Project** from the **Project** menu, browse to the directory on your PC to which the AN0381-SC01 source code was downloaded, locate the AN0381_SC01 µVision 5 file, highlight it, and select **Open**.
7. Ensure that the RUN/STOP switch on the Z32F128 MCU Module is in the STOP position.
8. In Keil µVision, compile and flash the firmware to the Z32F128 MCU Module by selecting **Rebuild All** target files from the **Project** menu. Next, select **Debug → Start/Stop Debug Session**, followed by **Debug → Run**.
9. Set the RUN/STOP switch on the Z32F128 MCU Module to RUN. The motor should begin turning.
10. In the GUI terminal console, enter the letter **U** to switch to UART control. As a result, commands can now be entered using the console to change the motor's operation.

You can now add your application software to the main program to experiment with additional functions.

Results

This three phase, sensored, brushless motor control application was tested with a 3-phase BLDC motor connected to Zilog's MultiMotor Development Board. Testing of the Z32F128 MultiMotor MCU Module confirms a seamless start-up of the motor from an

idle mode to full operational speed, plus on-the-fly reversal of the direction of rotation, an extremely fast fault-detection cycle, and a lower total solution cost.

- Maximum motor speed: 3200 RPM
- The motor can be controlled using two methods:
  - Manually using the Stop/Run & Direction switches and the speed pot on the MultiMotor MCU Module
  - Using menu-driven commands on a PC terminal emulator connected to the MultiMotor MCU Module through the UART connections
- The Green LED illuminates when the motor is running
- The Yellow LED illuminates when under UART control
- The Red LED flashes when the motor is stopped or a fault is detected

References

The following documents are associated with the Z32F128 Series of Motor Control MCUs; each is available for download on www.zilog.com.

- MultiMotor Series Development Kit Quick Start Guide (QS0091)
- MultiMotor Series Development Kit User Manual (UM0262)
- Z32F128 MCU Product Specification (PS0345)
- Z32F128 Evaluation Kit User Manual (UM0277)

Appendix A. Schematic Diagrams

Figures 5 and 6 show the schematics for the Z32F128 MCU Module.

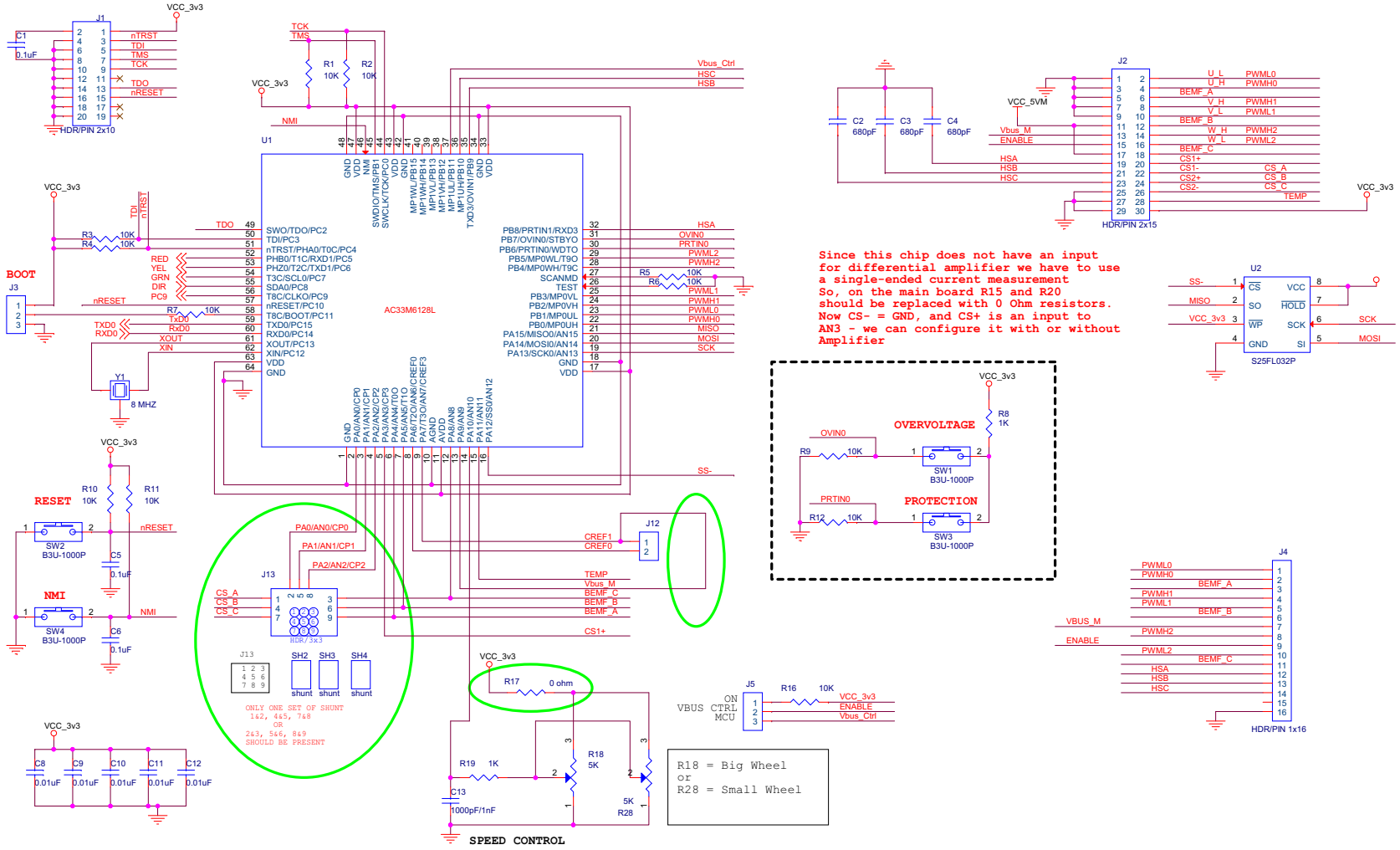

Figure 5. Z32F128 MultiMotor MCU Module, #1 of 2

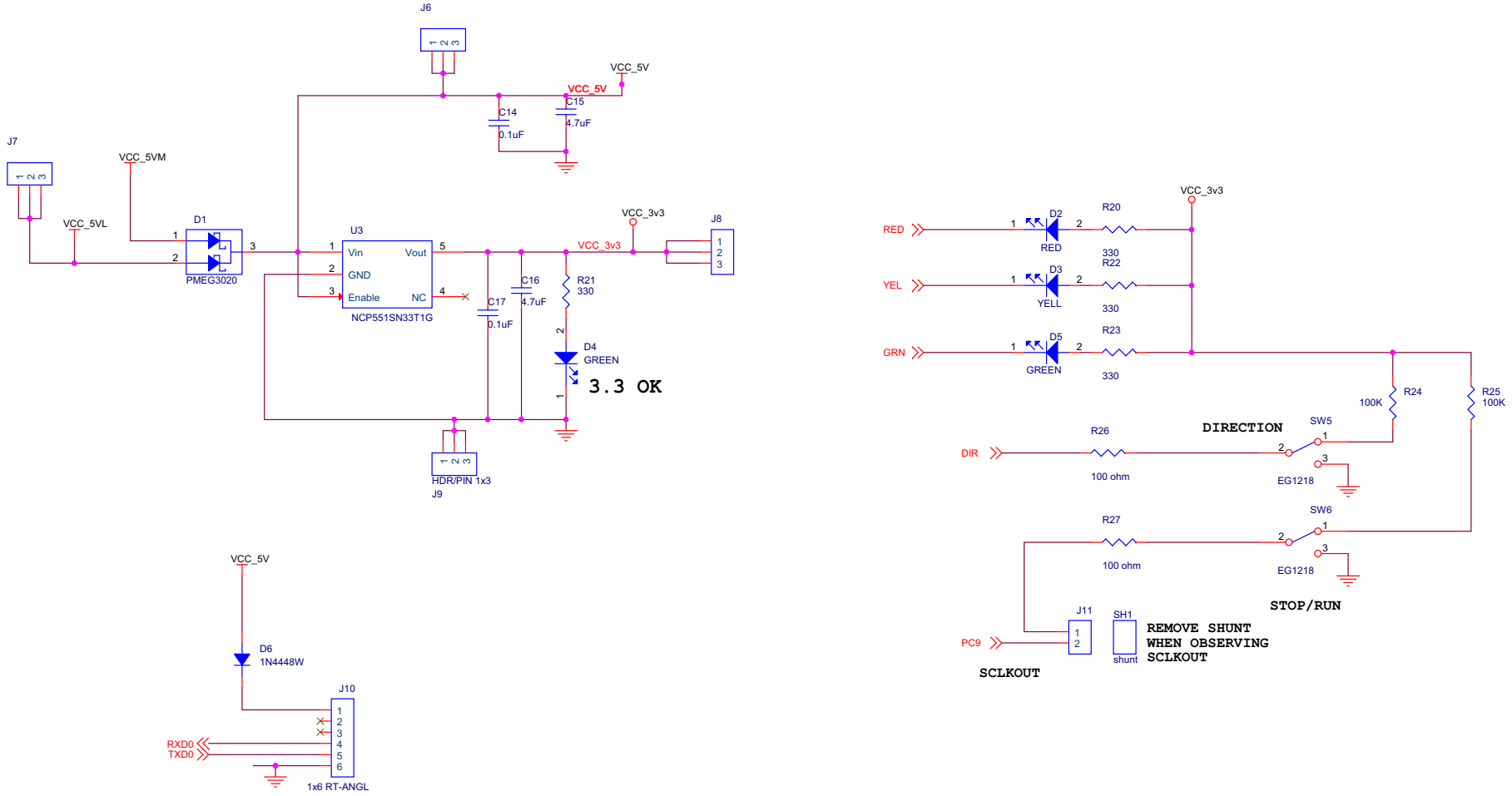

Figure 6. Z32F128 MultiMotor MCU Module, #2 of 2

Figures 7 and 8 show the schematics for the MultiMotor Main Board.

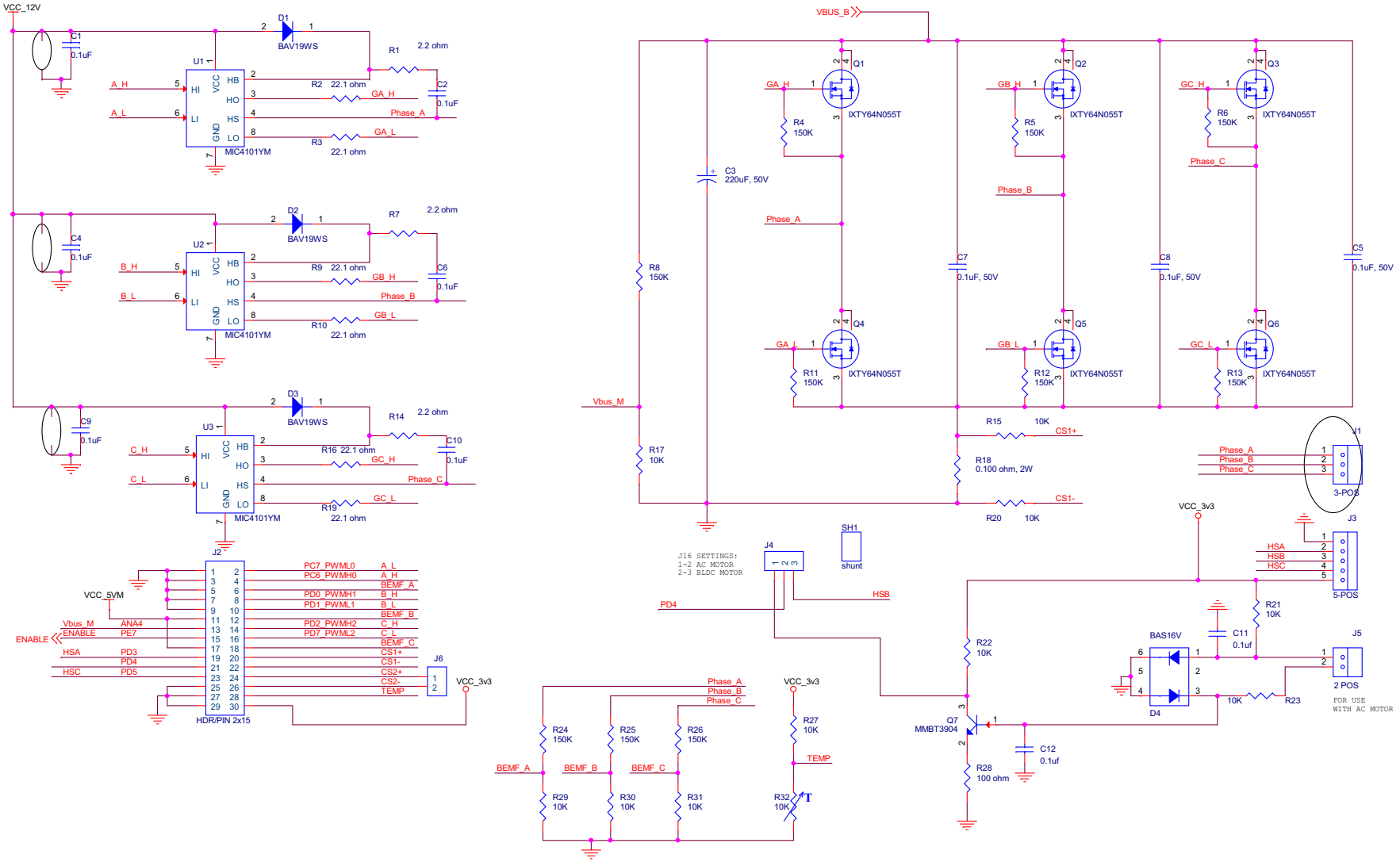

Figure 7. MultiMotor Development Board, #1 of 2

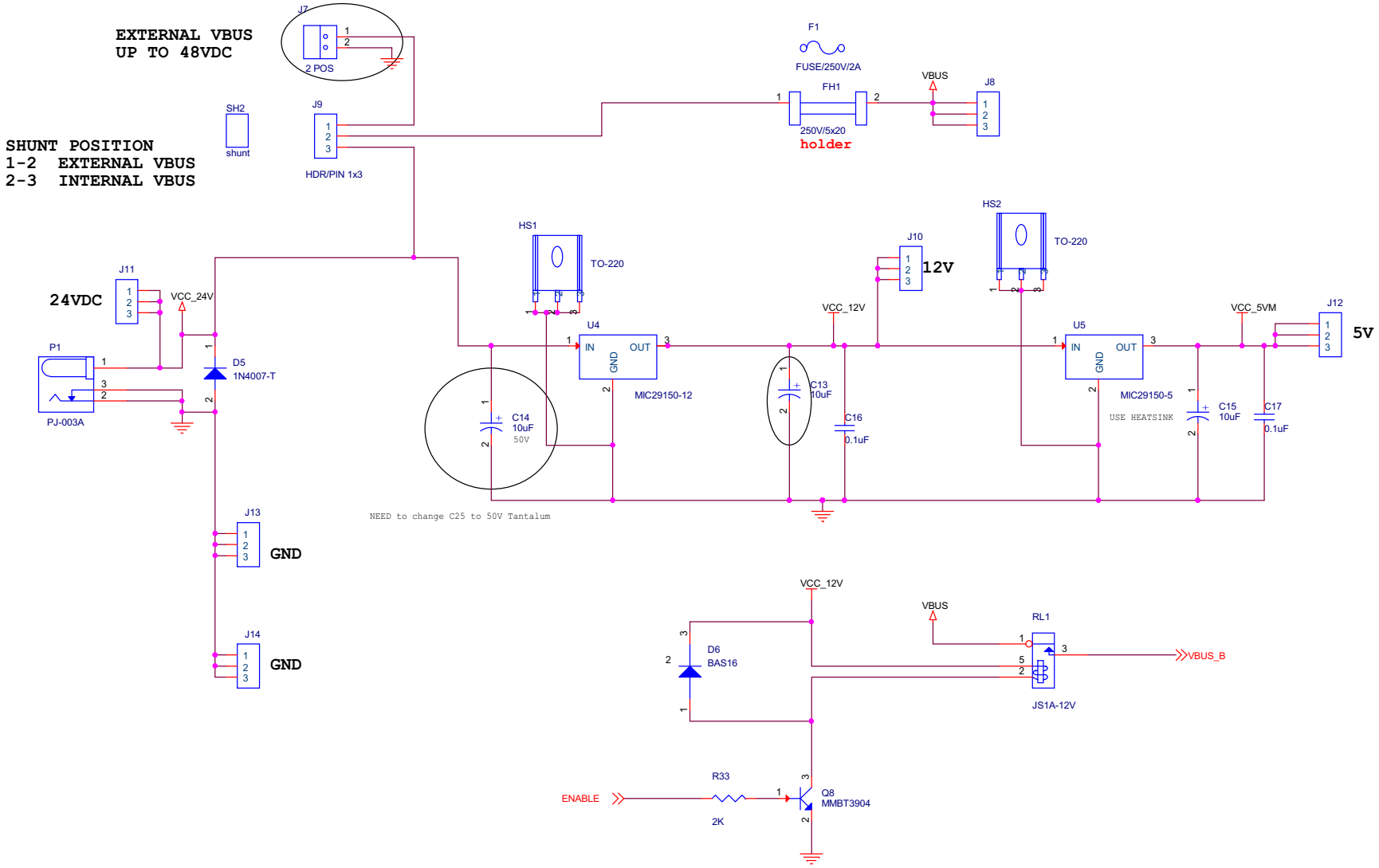

Figure 8. MultiMotor Development Board, #2 of 2

Appendix B. Flowcharts

Figure 9 presents a simple flow chart of the main, timer interrupt and Port D interrupt routines for a 3-phase Hall sensor BLDC motor control application.

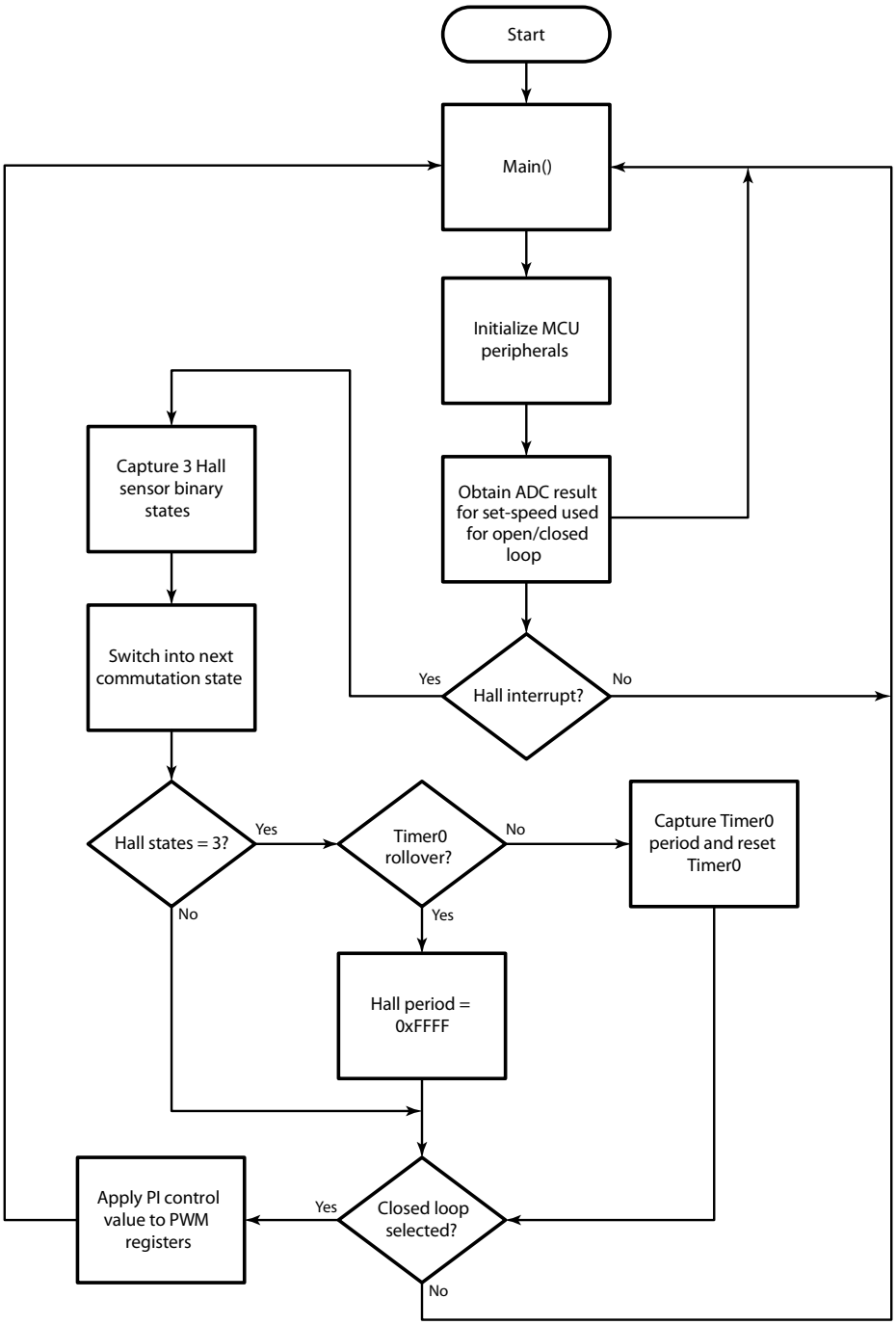

Figure 9. 3-Phase Hall Sensor BLDC Motor Control Application Flowchart

Customer Support

To share comments, get your technical questions answered, or report issues you may be experiencing with our products, please visit Zilog's Technical Support page at http://support.zilog.com.

To learn more about this product, find additional documentation, or to discover other facets about Zilog product offerings, please visit the Zilog Knowledge Base at http://zilog.com/kb or consider participating in the Zilog Forum at http://zilog.com/forum.

This publication is subject to replacement by a later edition. To determine whether a later edition exists, please visit the Zilog website at http://www.zilog.com.

Warning: DO NOT USE THIS PRODUCT IN LIFE SUPPORT SYSTEMS.

LIFE SUPPORT POLICY

ZILOG'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS PRIOR WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF ZILOG CORPORATION.

As used herein

Life support devices or systems are devices which (a) are intended for surgical implant into the body, or (b) support or sustain life and whose failure to perform when properly used in accordance with instructions for use provided in the labeling can be reasonably expected to result in a significant injury to the user. A critical component is any component in a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system or to affect its safety or effectiveness.

Document Disclaimer

 Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZILOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZILOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. The information contained within this document has been verified according to the general principles of electrical and mechanical engineering.

ZNEO32! and Z32F128 are trademarks or registered trademarks of Zilog, Inc. All other product or service names are the property of their respective owners.