Freescale Semiconductor
Application Note

Document Number: AN4796
Rev. 1, 11/2013

Sensorless BLDC Motor Control Using FRDM-KE02Z Based on Tower Board

by: Zhen Liu, Howard Liu, and Binbin Zhang

1 Introduction

This application note describes the design of a three-phase sensorless BLDC motor drive with Back-EMF zero-crossing. It is based on Freescale's FRDM-KE02Z that can be effectively used for motor-control applications.

The application uses Back-EMF zero-crossing technique for position detection, speed design, and current-loop regulation. It serves as an example of a sensorless BLDC motor control system using Freescale's MCU and three-phase BLDC/PMSM Low-Voltage Motor Control Drive. It also illustrates the usage of general on-chip peripherals for motor-control applications, controller features, basic BLDC motor theory, system design concept, hardware implementation, software design including the FreeMASTER software visualization tool, application setup, and demo operation.

Contents

2 KE02 advantages and features

On-chip modules available within the family include the following features:

- ARM® Cortex®-M0+ core
- Up to 20 MHz CPU at 2.7 to 5.5 V
- Up to 64 KB flash, 256 B EEPROM, 4 KB RAM
- 12-bit ADC with 16 channels
- Analog comparator
- Periodic interrupt timer with two channels and FlexTimer module with 10 channels
- Up to two 8-bit SPI modules , three SCI/UART modules, and one I^{2}C module
- 57 GPIOs, including 8-pin, 20 mA drive, and 2-pin true open-drain

3 BLDC motor control theory

The brushless DC motor (BLDC Motor) is a rotating electric machine with a classic three-phase stator of an induction motor; the rotor has surface-mounted permanent magnets. It is also referred to as an electronically-commuted motor. There are no brushes on the rotor and the commutation is performed electronically at certain rotor positions. The stator is usually made of magnetic steel sheets. A typical cross-section of a BLDC Motor is shown in
Figure 1. The stator-phase windings are inserted in the slots (distributed winding) or they can be wound as one coil onto the magnetic pole. The rotor magnetic field is constant, because the air-gap magnetic field is produced by permanent magnets.

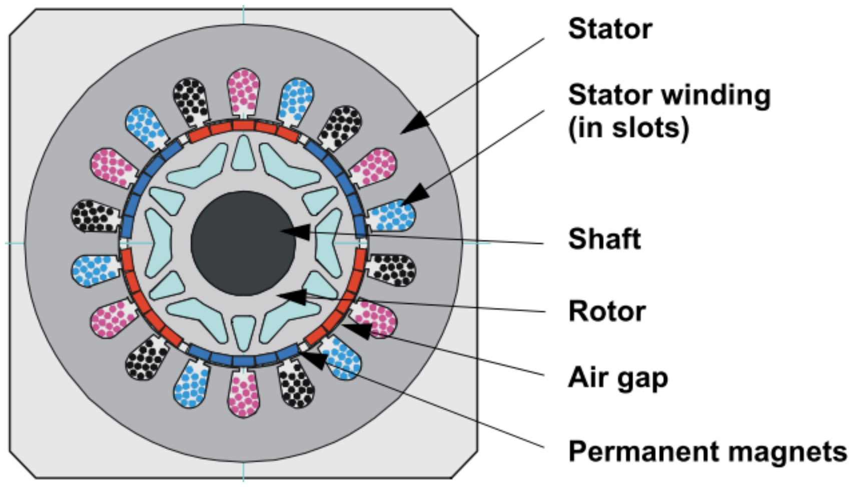


Figure 1. BLDC motor/cross section

The magnetism of the permanent magnets and their displacement on the rotor is chosen so that the Back-EMF (the voltage induced on the stator winding due to rotor movement) shape is trapezoidal. This allows the DC voltage (see Figure 2) with a rectangular shape to be used to create a rotational field with low-torque ripples.

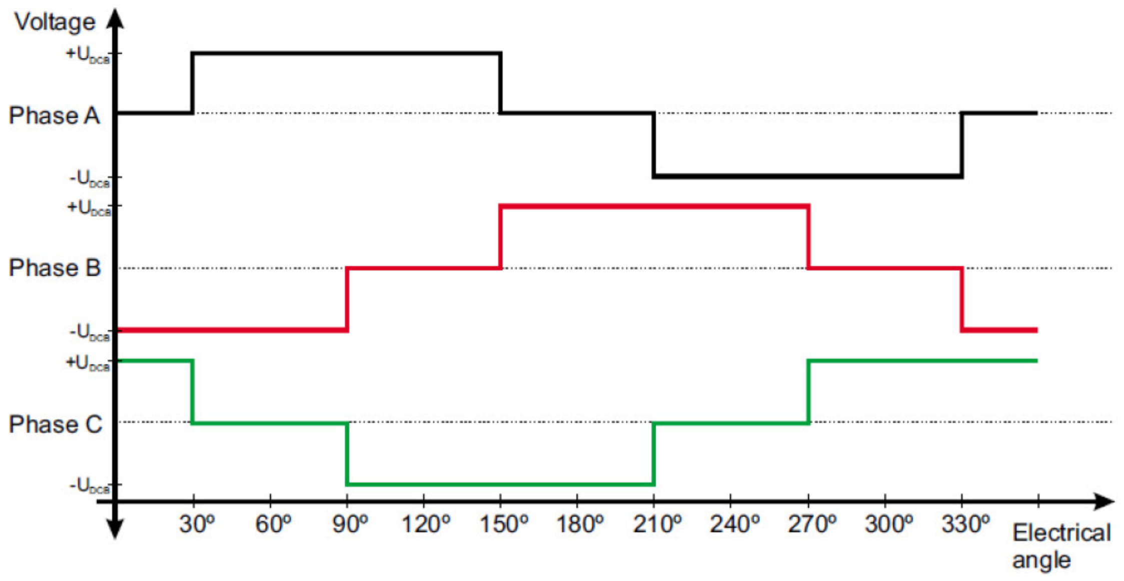


Figure 2. Three-phase voltage system of BLDC motor

4 System design concept

4.1 System specification

The motor-control system is designed to drive a three-phase, brushless-DC (BLDC) motor in a speed and torque-closed loop. The application meets these performance specifications:

- It has a sensorless brushless DC motor control using Back-EMF zero-crossing sensing.
- Control technique incorporates:
  - — Low-voltage sensorless control with speed and torque-closed loop.
  - — ADC converter for zero-crossing sensing.
  - — Overvoltage, Undervoltage, Overcurrent, and Fault protection.
  - — Start from any motor position with rotor alignment.
  - — Minimum speed of 500 rpm and maximum speed of 4500 rpm.
  - — FreeMASTER software-control interface (motor START/STOP and speed/torque setup).
  - — FreeMASTER software remote monitor.

4.2 Sensorless drive concept

As shown in Figure 3, the system incorporates the following hardware:

- FRDM – KE02Z board
- TWR-MC-FRDMKE02Z board
- TWR – MC – LV3PH board
- TWR – SER board
- Tower Elevator

- LINIX45ZWN24-40 BLDC motor
- Power Supply 24 V DC, 3.75 A

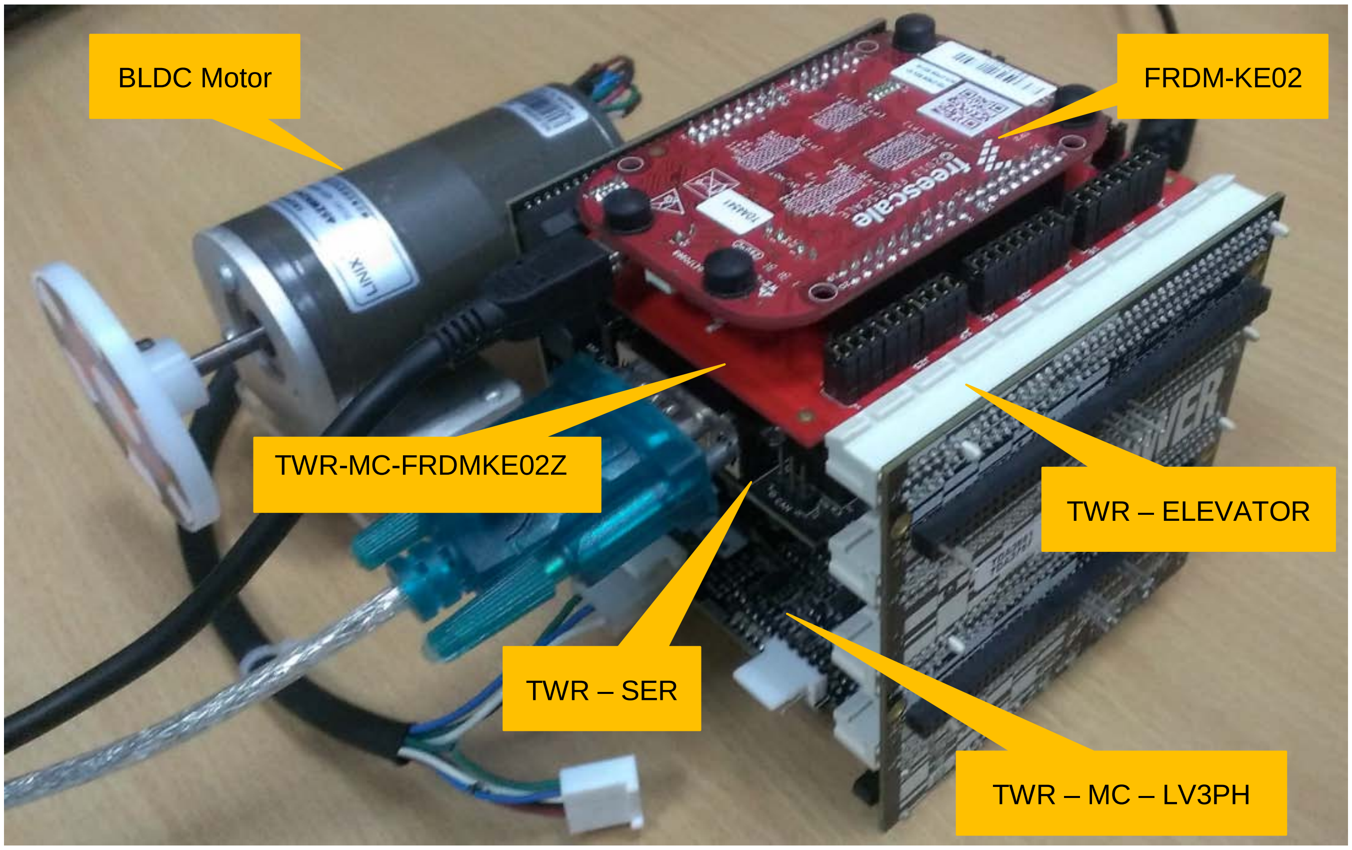


Figure 3. Application concept

The application concept is shown in Figure 4. The zero-crossing points of Back-EMF induced in the motor windings reflect the rotor position of Sensorless BLDC motor. While one of the three phase windings is not powered, the zero-crossing points of the Back-EMF are sensed. The interval time between two zero-crossing points is used to control commutation, and Pulse Width Modulation is used to control the phase voltage.

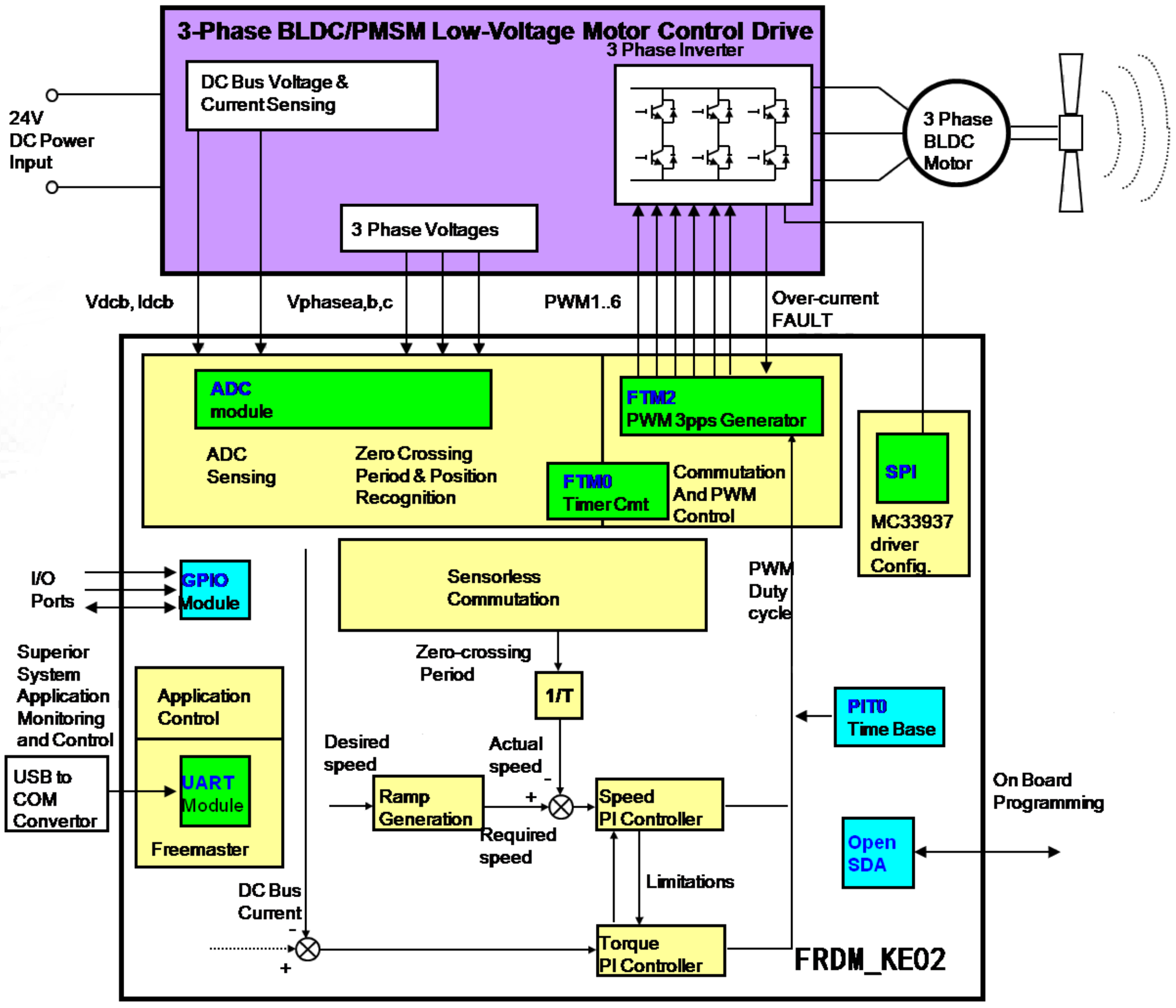


Figure 4. System configuration

5 Hardware

5.1 Component description

5.1.1 Three-phase bridge inverter and DC_Bus overvoltage protection

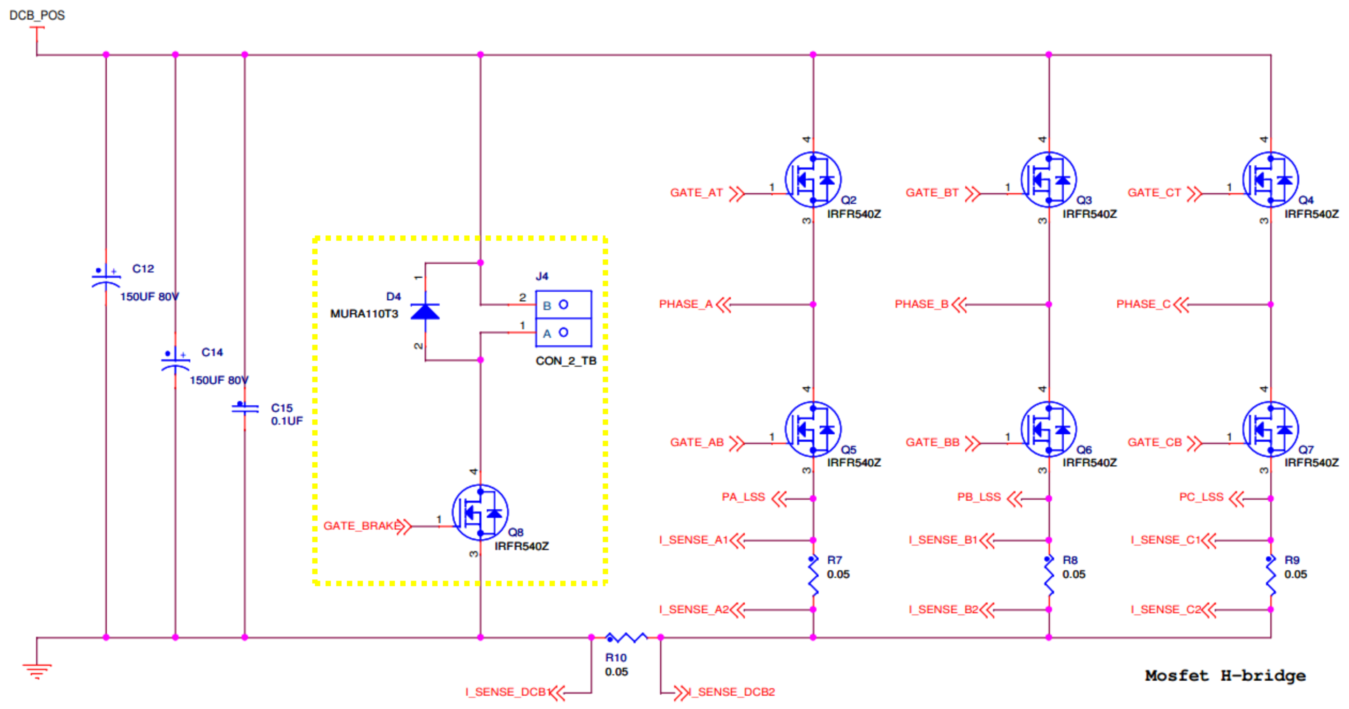


Figure 5. Three-phase bridge inverter and overvoltage protection

The three-phase bridge inverter contains six MOSFETs, three-phase current sample resistors, and one bus current sample resistor, as shown in Figure 5.

In this demonstration, a braking resistor and a MOSFET is installed in series between DCB_POS and GND (the yellow line range in Figure 5). The MOSFET is implemented if DC_Bus voltage is larger than preset maximum voltage and energy is lost through the braking resistor, which results in a decrease in the DC_Bus voltage. The hardware plays the role of over voltage protection.

5.1.2 Predriver (MC33937)

MC33937 is a Field Effect Transistor (FET) predriver designed for three-phase motor control (as described in Figure 6). Three external bootstrap capacitors provide gate charge to the high-side FETs. The IC interfaces to an MCU via six direct input control signals, an SPI port for device setup and asynchronous reset, enable and interrupt signals. MC33937 integrates overcurrent detect circuit, if overcurrent occurs, DRV_OC port level changes to a high-level.

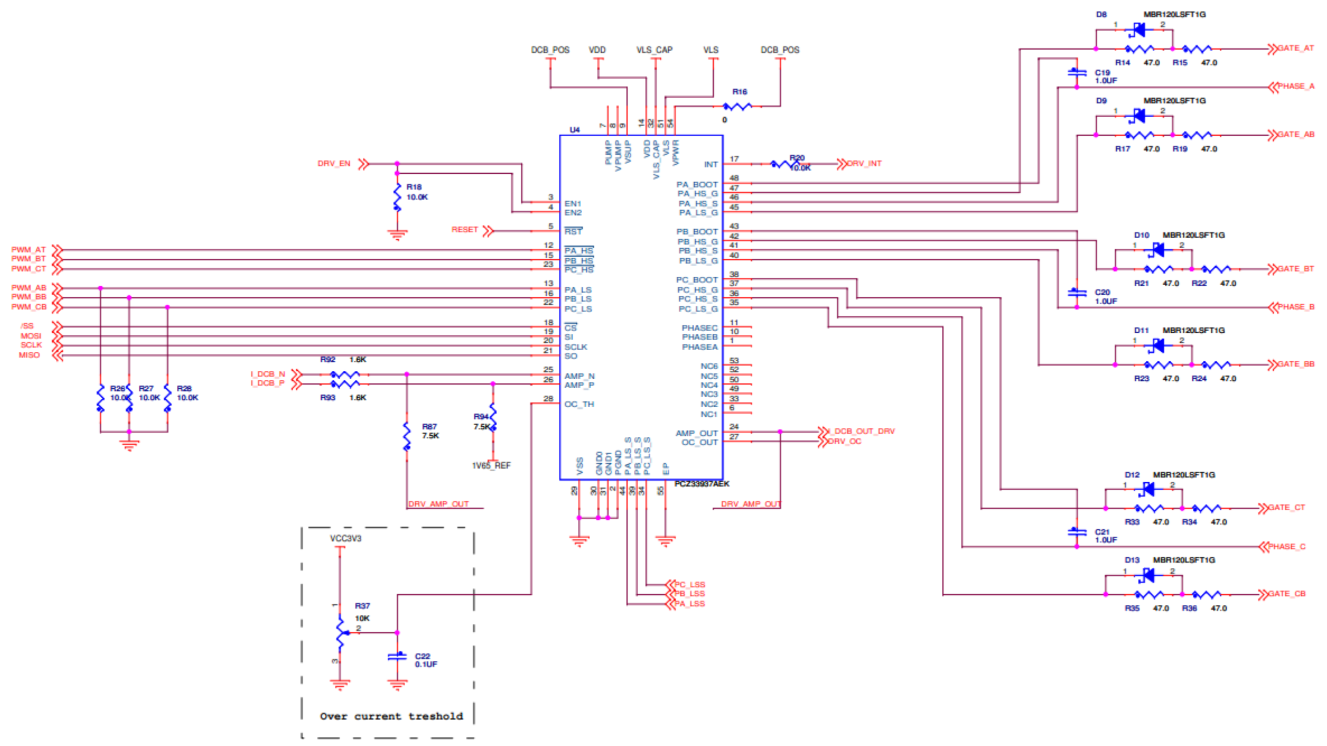


Figure 6. Predriver (MC33937)

5.1.3 BEMF and DC_Bus voltage sensing circuit

Figure 7 is the DC_Bus voltage sensing circuit and the BEMF sensing circuit for phase A, while phase B and phase C are same. In this project, if DC_Bus voltage is preset to 36.3 V, and the power supply voltage of MCU is 3.3 V, the divide resistance value is 30 kΩ and 3 kΩ.

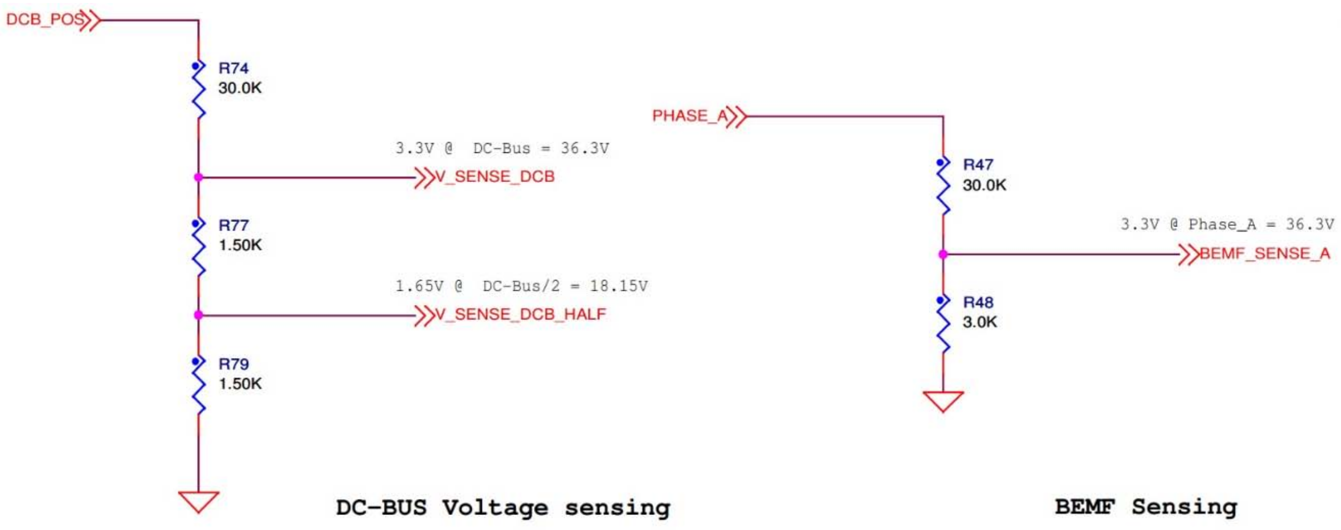


Figure 7. BEMF and DC_Bus voltage sensing circuit

5.1.4 DC_Bus current sensing circuit

The following figure shows the DC_Bus current sensing circuit:

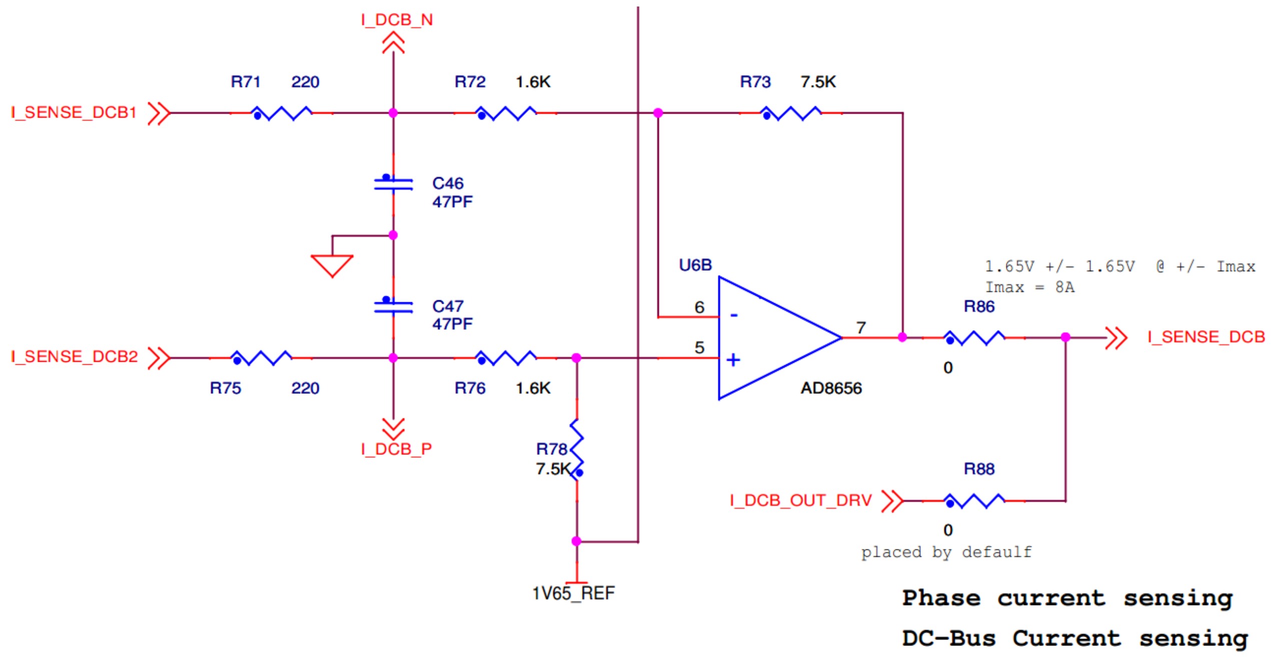


Figure 8. DC_Bus current sensing circuit

DC_Bus current sample is needed to conduct current loop PI regulator and over current software protection. In this demonstration board, the sample resistor is 0.05 Ω, due to current’s positive and negative conduction 1.65 V bias voltage is required. The differential amplifier equation is:

$$I\_DCBUS = 1.65 + \frac{R73}{R71 + R72}(0.05 * i)$$

Equation 1

Thus, the maximum current cannot exceed 8 A.

5.1.5 Brake switch circuit

Figure 9 represents the brake switch circuit. MIC4127YME is equivalent to a power amplifier, whose function is to drive MOSFET of overvoltage protection circuit. If the DC_Bus voltage is higher than preset maximum DC_Bus voltage value, the BRAKE_CONTROL and GATE_BRAKE are set to high level and the MOSFET conducts, thereby reducing the DC_Bus voltage.

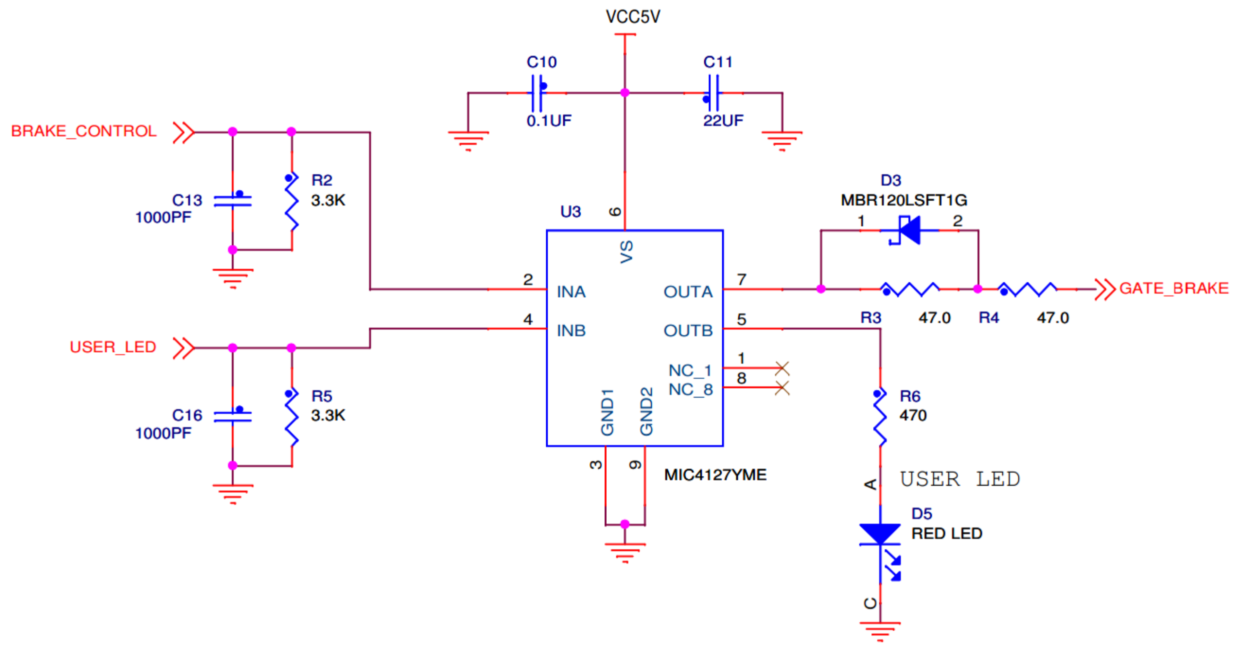


Figure 9. Brake switch circuit

5.2 Motor 45ZWN24-40 (Produced by Linix)

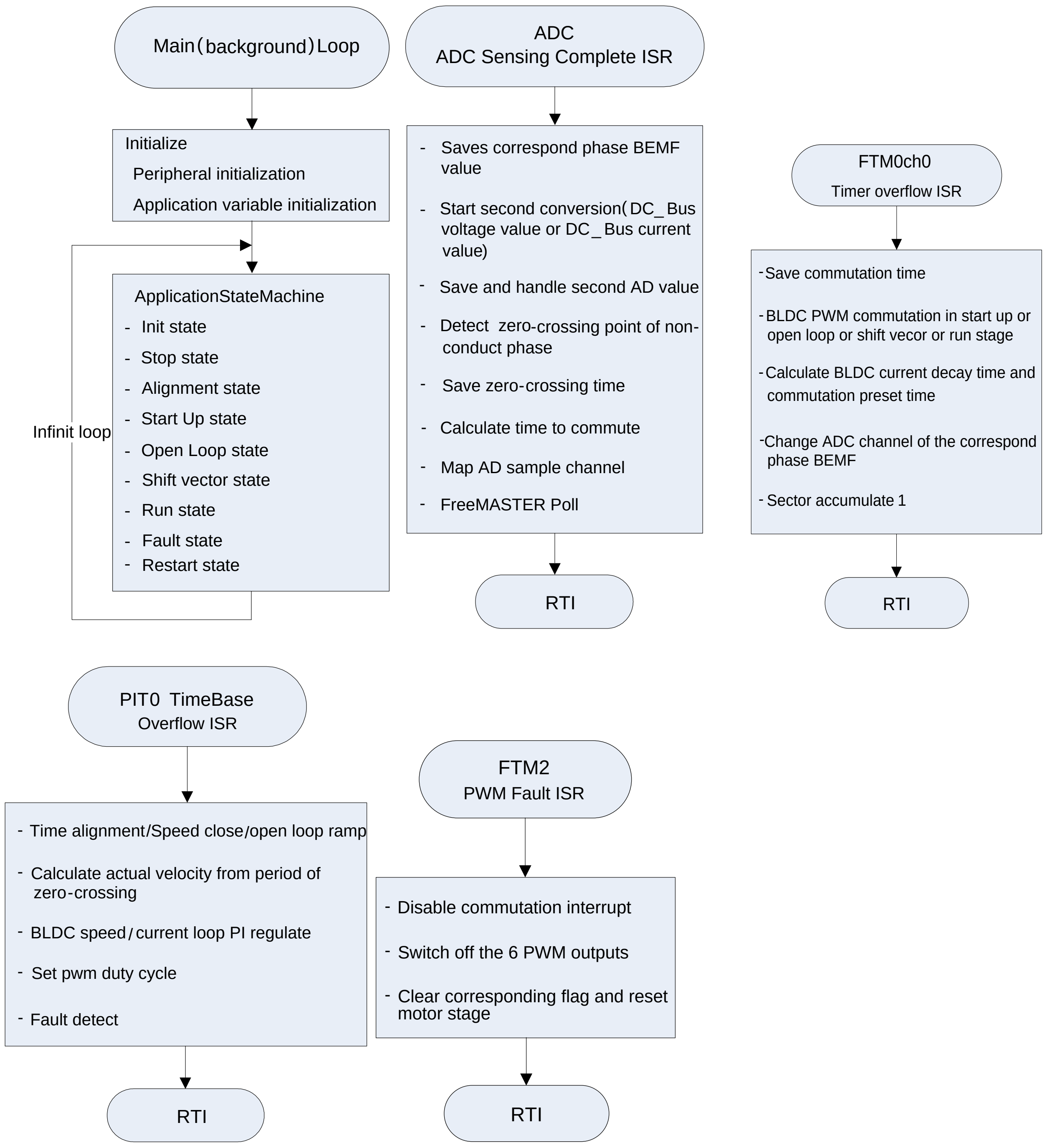
The following motor is used for the BLDC sensorless application. Other motors can also be adapted to the application, just by defining and changing the motor-related parameters. A detailed motor specification is described in the following table:

Table 1. Electrical characteristics of Linix 45ZWN24-40 motor

Characteristic	Symbol	Min	Type	Max	Unit
Reference Winding Voltage	Vt	Jm		24	V
Speed (supply voltage=Vt)		—	—	4000	rpm
Torque Constant	Kt	—	—	—	Nm/A
Voltage Constant	Ke	—	—	—	V/RPM
Terminal Resistance	Rt	—	—	—	Ω
Winding Inductance	L	—	—	—	mH
Continuous Current	Ics	—	—	—	A
Number of Pole Pairs		—	2	—	—
Temperature Rating		—	—	—	°C

6 Software

The main flowchart of the control system is shown in the following figure:



Main(background)Loop
Initialize
Peripheral initialization
Application variable initialization
ApplicationStateMachine
- Init state
- Stop state
- Alignment state
- Start Up state
- Open Loop state
- Shift vector state
- Run state
- Fault state
- Restart state
Infinit loop
ADC
ADC Sensing Complete ISR
- Saves correspond phase BEMF value
- Start second conversion(DC_Bus voltage value or DC_Bus current value)
- Save and handle second AD value
- Detect zero-crossing point of non-conduct phase
- Save zero-crossing time
- Calculate time to commute
- Map AD sample channel
- FreeMASTER Poll
RTI
FTM0ch0
Timer overflow ISR
-Save commutation time
-BLDC PWM commutation in start up or open loop or shift vecor or run stage
-Calculate BLDC current decay time and commutation preset time
-Change ADC channel of the correspond phase BEMF
-Sector accumulate 1
RTI
PIT0 TimeBase
Overflow ISR
- Time alignment/Speed close/open loop ramp
- Calculate actual velocity from period of zero-crossing
- BLDC speed/current loop PI regulate
- Set pwm duty cycle
- Fault detect
RTI
FTM2
PWM Fault ISR
- Disable commutation interrupt
- Switch off the 6 PWM outputs
- Clear corresponding flag and reset motor stage
RTI

Figure 10. Main software flowchart

6.1 Data flow

The data flow of the control algorithm is shown in the following figure:

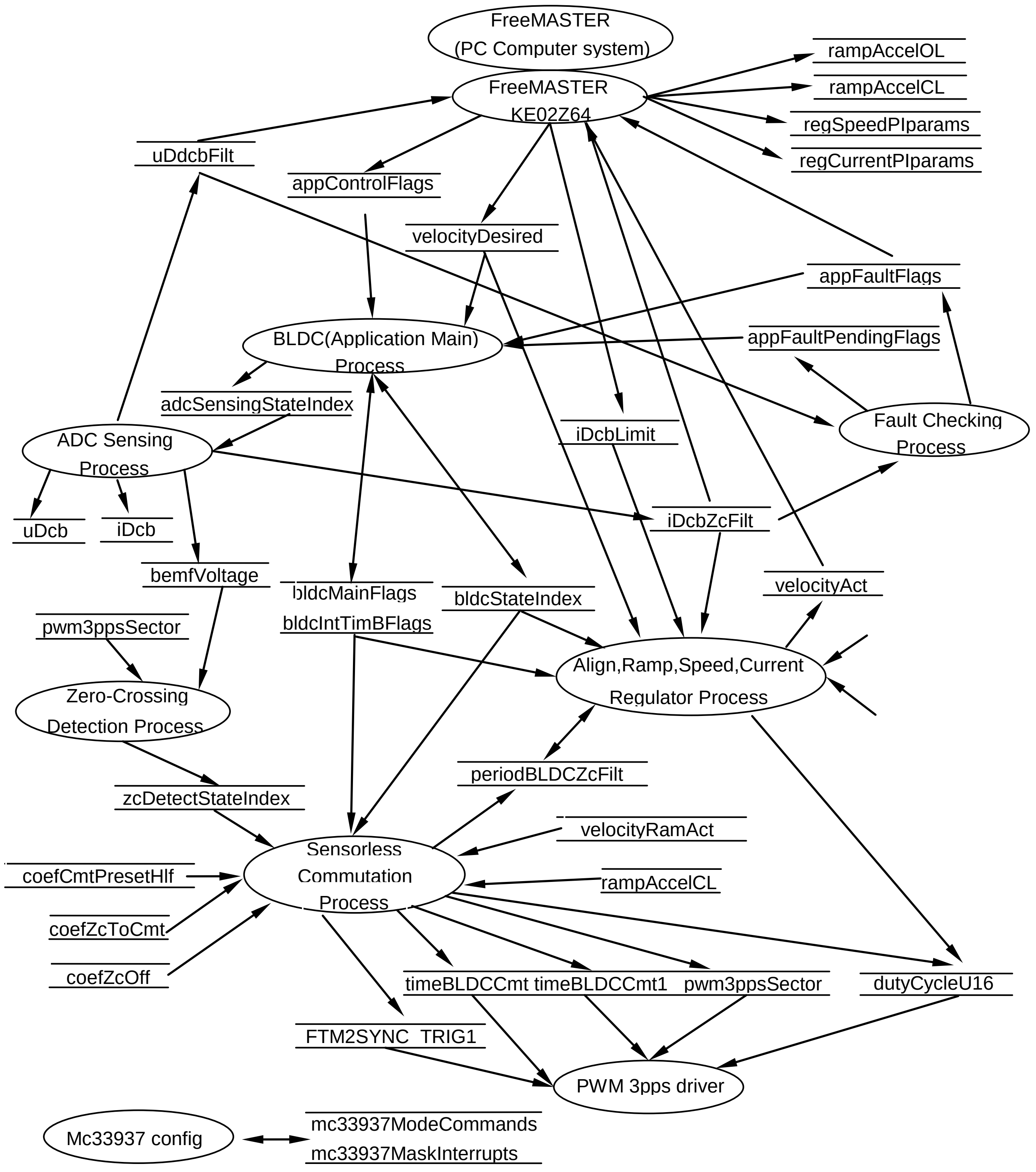


Figure 11. Software data flow

6.1.1 BLDC (application main) process

Based on the nonzero variable **velocityDesired** set from the FreeMASTER, the BLDC starts. Motor's stage changes according to the **bldcStateIndex**. The Running stage requests calculation of the current PI controller or speed PI controller and detection of the zero-crossing point.

6.1.2 ADC sensing process

This process converts DC_Bus voltage, DC_Bus current, and Back-EMF(Running stage). The results, **uDcbFilt,** and **iDcbZcFilt**, indicate voltage and current. The uDcbFilt variable is used for over or under voltage checking in Fault Checking Process, and the **iDcbZcFilt** variable is used for Current Regulator Process. The output **bemfVoltage** is, calculated from the conversion of BEMF, delivered to Zero-Crossing Detection Process.

6.1.3 Zero-Crossing detection process

The zero-crossing detection is based on the ADC conversion of BEMF. When the non-conducting phase branch voltage subtracts the half of DC_Bus, voltage changes the sign from negative to positive or from positive to negative. Then the zero crossing point is detected and zcDetectStateIndex is set to ZCDETECT_STG3ZCDETECTED.

6.1.4 Sensorless commutation process

This process controls sensorless BLDC motor commutations by changing the variable **pwm3ppsSector** from 0 to 5. The process outputs, the **timeBLDCCmt** and the **timeBLDCCmt1**, are used to set the time for next commutation. The output **periodBLDCZcFilt** is used to calculate the actual velocity for the Speed Regulator Process.

6.1.5 Fault checking process

This process is used as protection is important for motor control. The main faults occur during motor control includes overvoltage, undervoltage, and overcurrent. In this application, overvoltage protection and undervoltage protection are implemented by the software using polling mode. The PWM is disabled when fault happens.

6.2 State diagram

This application contains two main state diagrams, described in subsequent subsections.

6.2.1 BLDC motor control state diagram

The motor control state diagram is displayed in Figure 12. The application is controlled by FreeMASTER the Start/Stop is controlled based on a non-zero or zero velocity set from FreeMASTER. In addition, the required speed is set using the FreeMASTER software. The motor is stopped whenever the reset button is pushed or velocity is set to zero. All the software processes are controlled according to this control state diagram.

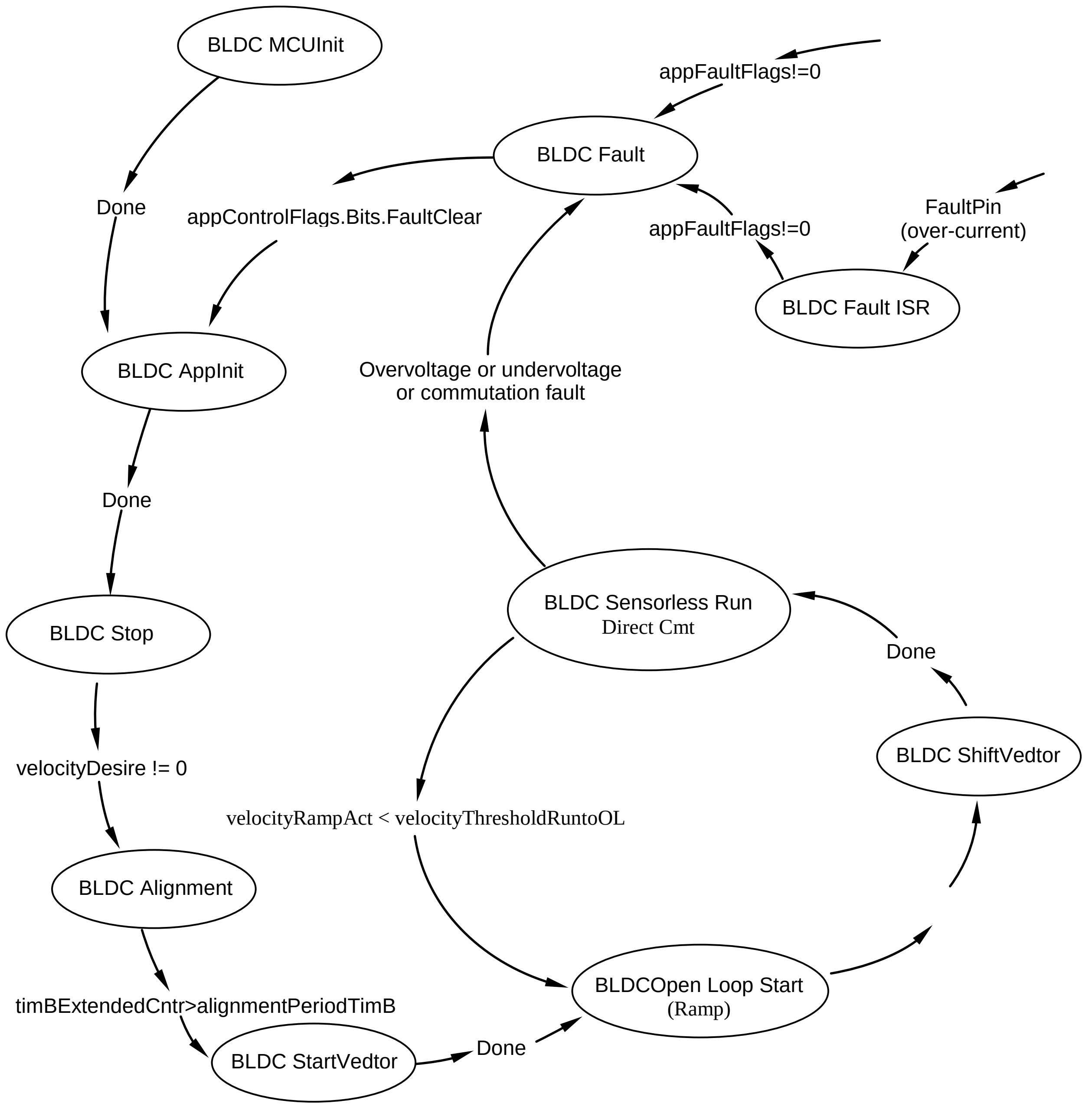


Figure 12. BLDC motor control state diagram

6.2.2 State diagram - commutation with BEMF zero crossing sensing

The state diagram of the Commutation with BEMF Zero Crossing Sensing is shown in Figure 13. The selection of state after the motor commutation depends on the detection of the Back-EMF Zero Crossing during the previous commutation period. In the beginning of run, the commutation time is preset. If zero crossing point is detected after the periodBLDCZcToff time period expired, the commutation period and commutation register is reset using the calculation as follows:

```
timeBLDCZcPrev = timeBLDCZc;
timeBLDCZc = timeBackEmf;
periodBLDCZc = timeBLDCZc - timeBLDCZcPrev;
periodBLDCZcFlt = (periodBLDCZc0 + periodBLDCZc)>>1;
periodBLDCZc0 = periodBLDCZc;
periodBLDCZcToCmt = F16Mul(periodBLDCZcFlt, coefZcToCmt);
timeBLDCCmt = timeBLDCzC + periodBLDCZctoCmt;
FTM0_MOD = timeBLDCCmt – timeBLDCCmt1;
```

where, the `timeBackEmf` is calculated in Sensorless Commutation Process and corrective commutation will be performed after commutation time expires.

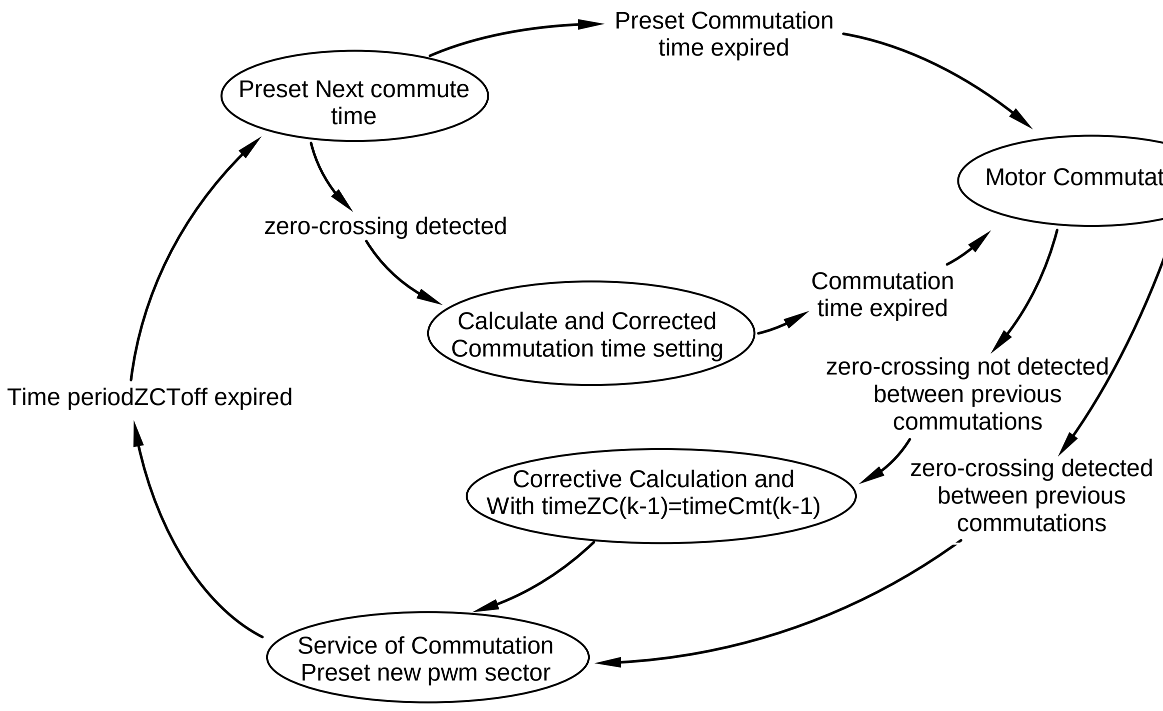

Preset Commutation time expired
Preset Next commute time
Motor Commutation
zero-crossing detected
Commutation time expired
Calculate and Corrected Commutation time setting
Time periodZCToff expired
zero-crossing not detected between previous commutations
Corrective Calculation and With timeZC(k-1)=timeCmt(k-1)
zero-crossing detected between previous commutations
Service of Commutation Preset new pwm sector

Figure 13. Commutation with BEMF zero crossing sensing

If the preset commutation time expires indicating no BEMF Zero Crossing detected, the commutation is performed immediately and the commutation period is corrected with Corrective Calculation as follows:

```
timeBackEmf = timeBLDCCmt + FTM0_CNT;
timeBLDCZcPrev = timeBLDCZc;
timeBLDCZc = timeBackEmf;
periodBLDCZc = timeBLDCZc - timeBLDCZcPrev;
periodBLDCZcFlt = (periodBLDCZc0 + periodBLDCZc)>>1;
periodBLDCZc0 = periodBLDCZc;
periodBLDCZcToCmt = F16Mul(periodBLDCZcFlt, coefZcToCmt);
```

6.3 Configurations

6.3.1 MOSFET driver configuration

For the correct operation of the MC33937, the predriver should be configured. This driver is able to configure only through SPI communication. There are two more files, providing SPI communication between the MCU and the driver, and configuring the MOSFET driver.

- The spi_comm.h header file contains configuration and status constants defined for the MC33937 driver.
- The spi_comm.c file contains SPI communication functions and configuration function for the MC33937 driver.

The SPI communication is not used only for driver configuration, but also for diagnosing this driver.

6.3.2 PWM generation and timers

The KE02Z64VQH2’s FTM module has three submodules. Only FTM0 and FTM2 are used to commutate and generate six PWM signals connected via MC33937 to three-phase inverter bridge. The FTM used are configured as follows:

FTM0

- System clock source divided by 32

```
FTM0->SC |= FTM_SC_CLKS(1) | FTM_SC_PS(5);
```

- Channel 0 enabled to serve as edge pwm mode
- Select high-true polarity of pwm signal

```
FTM0->CONTROLS[0].CnSC |= FTM_CnSC_MSB_MASK | FTM_CnSC_ELSB_MASK;
```

- Overflow interrupt with variable modulo value for commutation
- Set channel value to 10 when enable commutation interrupt

```
FTM0->CONTROLS[0].CnV = 10;
```

FTM2

- System clock source

```
FTM2->SC |= FTM_SC_CLKS(1);
```

- Generate pwm with running frequency of 16 kHz
- Modulo 1250 with 0.08% resolution

```
FTM2->MOD = PWM_MODULO;        ( #define PWM_MODULO 1250 )
```

- Combine and complement mode with 1μs deadtime

```
FTM2->COMBINE =  FTM_COMBINE_FAULTEN0_MASK
  | FTM_COMBINE_SYNCEN0_MASK | FTM_COMBINE_DTEN0_MASK
  | FTM_COMBINE_COMP0_MASK | FTM_COMBINE_COMBINE0_MASK
     | FTM_COMBINE_FAULTEN1_MASK
  | FTM_COMBINE_SYNCEN1_MASK | FTM_COMBINE_DTEN1_MASK
  | FTM_COMBINE_COMP1_MASK | FTM_COMBINE_COMBINE1_MASK
     | FTM_COMBINE_FAULTEN2_MASK
  | FTM_COMBINE_SYNCEN2_MASK | FTM_COMBINE_DTEN2_MASK
  | FTM_COMBINE_COMP2_MASK |FTM_COMBINE_COMBINE2_MASK;

    FTM2->DEADTIME = FTM_PWM_DEAD_TIME;
```

```
            ( #define FTM_PWM_DEAD_TIME 20 )
```

- External trigger 1 enabled to get synchronization signal from FTM0CH0_Output
- High-side switch PWM_T output in low polarity
- Low-side switch PWM_B output in high polarity

```
            FTM2->POL = FTM2POL_INIT ;
            ( #define FTM2POL_INIT FTM_POL_POL0_MASK | FTM_POL_POL2_MASK |
                                                 FTM_POL_POL4_MASK )
```

FTM2 Fault

- High-level on fault input pin1 indicate a fault signal
- High side PWM signal set to high-level when fault signal detected
- Low side PWM signal set to low-level when fault signal detected
- Fault input filter disabled

```
        FTM2->FLTCTRL |= FTM_FLTCTRL_FAULT1EN_MASK;
        FTM2->MODE |= FTM_MODE_FAULTM(2) | FTM_MODE_FAULTIE_MASK;
```

The BLDC motor uses only one PIT module to generate periodic interrupt, which is used as speed loop regulator and current loop regulator timebase.
PIT0

- Runs at frequency 20 MHz

```
        PIT->MCR = 0x00;
```

- Counts until compare and reinitializes

```
        PIT->CHANNEL[0].TCTRL = 0x03;
```

- Generates 3 ms interrupt with modulo value 0xEA60

```
        PIT->CHANNEL[0].LDVAL = (0xEA60-0x01);
```

6.3.3 Bipolar PWM versus Unipolar PWM

Bipolar PWM and Unipolar PWM are two pulse width modulations. The difference between the two modulations is that Unipolar PWM can be used for two quadrants operation. Whereas, Bipolar PWM (top bottom in diagonal on) can be used for four quadrants operation.

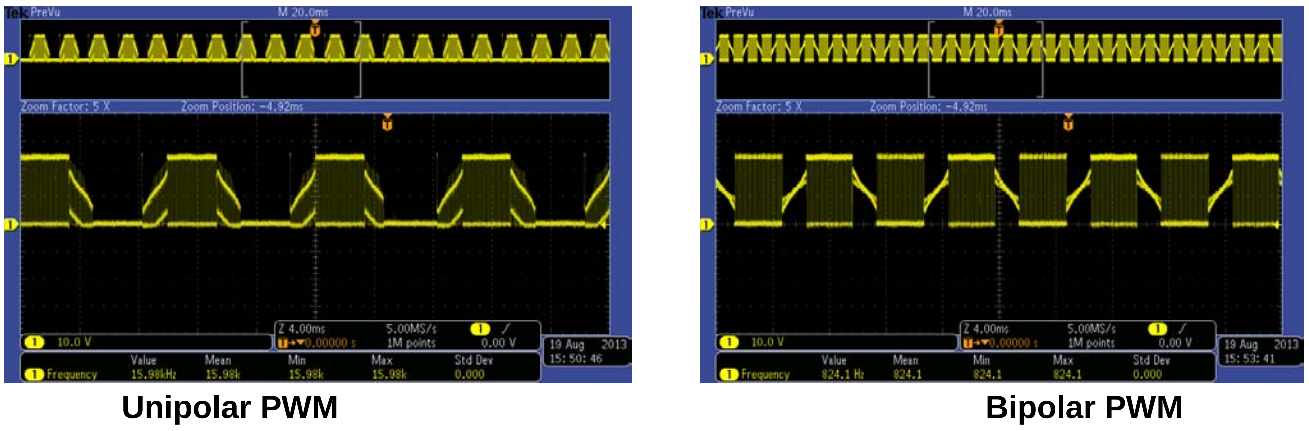


Figure 14. Unipolar PWM vs Bipolar PWM

Figure 14 is the scope for a phase voltage waveform, the left is Unipolar PWM, the right is Bipolar PWM. The source code attached to this application note support this two pulse width modulations, default is Unipolar PWM, you can open the macro *#define PWM_BIPOLAR_SWITCHING* in *State_machine.c* to change it to Bipolar PWM.

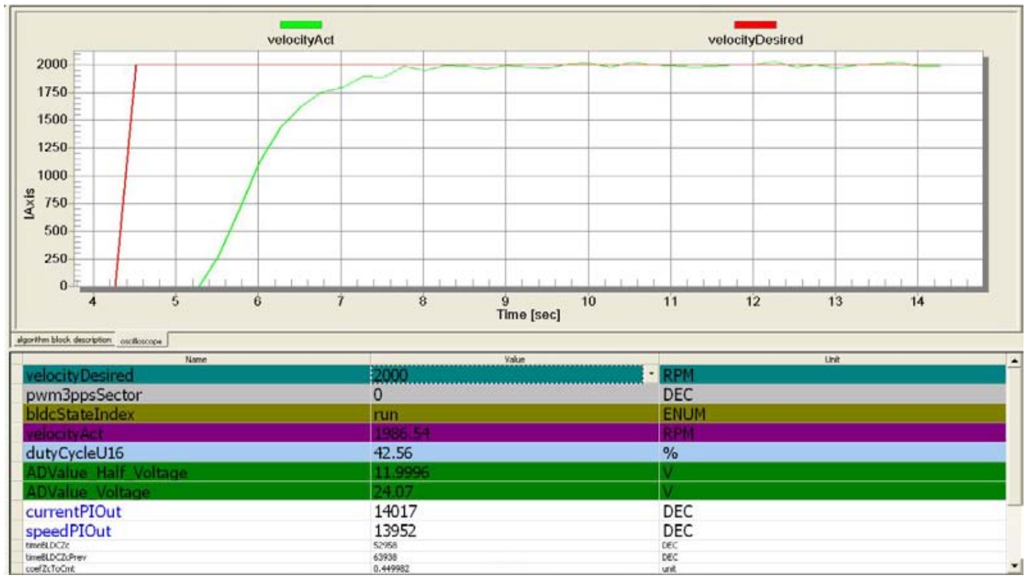
6.3.4 AD conversion

The ADC module is configured for BEMF, DC_Bus voltage, and current sampling and conversion as follows:

- Input clock BusClk
- Single conversion mode
- Right-justified result data with 12-bit resolution
  ```
  ADC->SC3 = ADC_SC3_MODE(2) | ADC_SC3_ADIV(2);
  ```
- External PWM trigger control
  ```
  ADC->SC2 |= ADC_SC2_ADTRG_MASK;
  ```
- Sample channels is set as follows: DC_Bus voltage channel AD11,current channel AD14,phase A BEMF channel AD10, phase B BEMF channel AD3, phase C BEMF channel AD7
  ```
  ADC->APCTL1 = 0xC488;
  ```

AD channel’s select is non-conduct phase according to sector of BLDC. The task of AD interrupt program is to save correspond non-conduct phase voltage value, start second conversion (DC_Bus voltage or DC_Bus current)using polling, zero-crossing point detection, map AD sample channel. The analog sample of BEMF which is to be converted to digital sample must be synchronized with PWM due to mutual inductance of stator windings.

6.3.5 FreeMASTER communication



Name	Value	Unit
velocityDesired	2000	RPM
pwm3ppsSector	0	DEC
bldcStateIndex	run	ENUM
velocityAct	1986.54	RPM
dutyCycleU16	42.56	%
ADValue_Half_Voltage	11.9996	V
ADValue_Voltage	24.07	V
currentPIOut	14017	DEC
speedPIOut	13952	DEC
timeBLDCZc	52958	DEC
timeBLDCZcPrev	63938	DEC
coefZcToCmt	0.449982	unit

Figure 15. FreeMASTER debug interface

FreeMASTER GUI is represented in Figure 15. Serial communication using UART module is implemented for remote control using FreeMASTER. The host computer is connected to the controller via a USB cable. The computer's USB port works as a virtual COM port. Signal conversion from USB form to UART form, and vice versa, is done by the USB/UART bridge.
In project > Options > Comm > Communication, please select Direct RS232 as communication, the baud rate is 9600 bps. In project > Options > MAP Files, please select the suffix for out file as Default symbol file and the "File format" as Binary ELF with DWARF1 or DWARF2 dbf format.

6.3.6 Others

Finally, the motor parameters, alignment, and starting constants are stored in the main.h and hw_config.h file. The motor used in this application is same as used in the reference design of MC9S08PT60, so the motor parameters are same.

7 Demo setup and operation

For demonstrating the operation, this demo is built and available for customers.

7.1 Hardware setup

The hardware is shown in Figure 3 as explained in Sensorless drive concept section.

Follow the following steps to run the sensorless BLDC motor:

1. Plug the power supply jack connector to the low-voltage motor control board connector J1.
2. Connect the USB 2.0 cable to the PC and to the KE0Z central control board connector J6.
3. Check the settings of jumpers J2, J3, J10, J11, J12, J13, and J14 on the TWR-MC-LV3PH board as follows.
   - J3 (pins 2 and 3 shorted) is elevator analog supply.
   - J10, J11, and J12 (pins 2 and 3 shorted) represent BEMF sense phase A, BEMF sense phase B, and BEMF sense phase C, respectively.
   - J13 (pins 2 and 3 shorted) represents DC_Bus current sense.
   - J14 (pins 2 and 3 shorted) represents DC_Bus half voltage sense.
4. Check the settings on the KE02Z central control board and jumpers J31 and J32 on the adapter board (TWR-MC-FRDMKE02Z board).
   - J3, J4, and J5 used for debug convenience must be shorted.
   - Remove the resistor R37, R52, and R53.
   - J31and J32 (pins 2 and 3 shorted) represents 3.3 V and 5 V from elevator connector.

KE02 adapter board is described in Figure 16, KE02 central control board is described in Figure 17, and low-voltage motor control tower board is described in Figure 18.

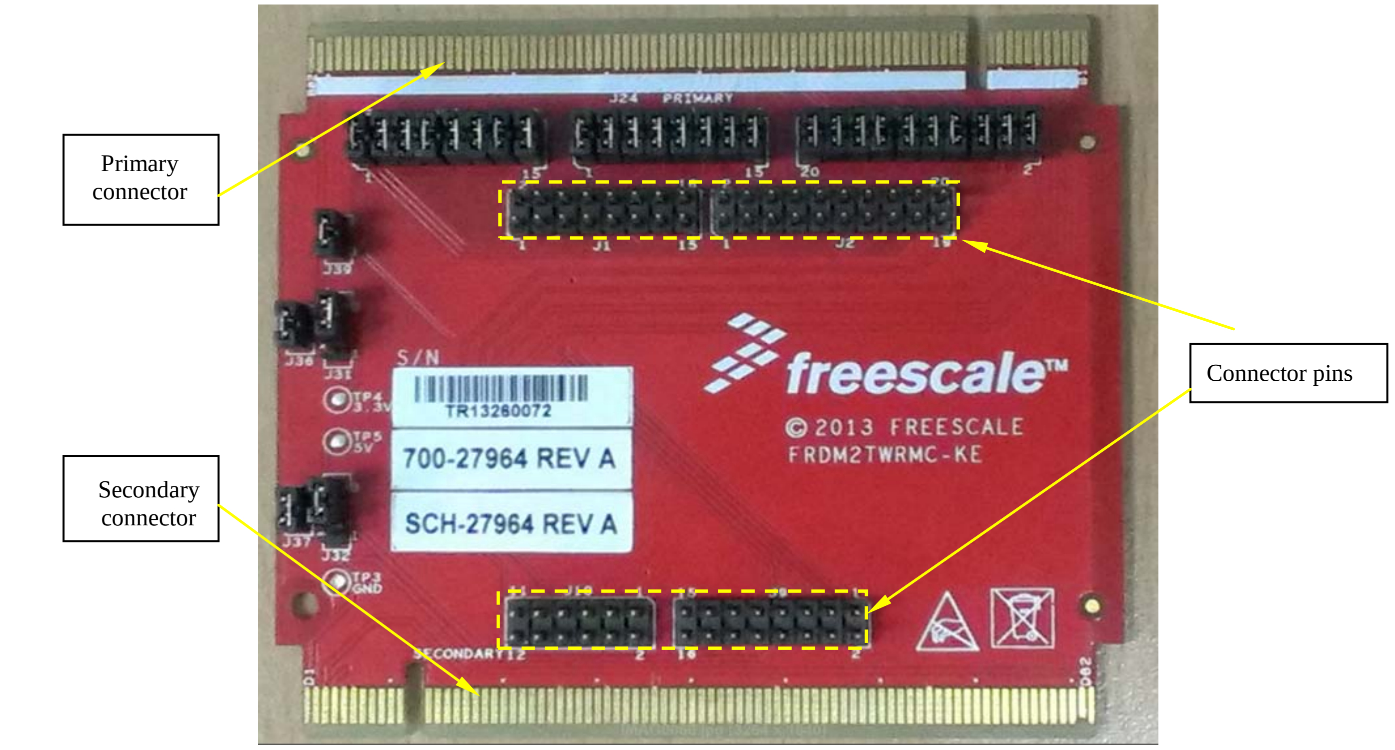


Figure 16. KE02Z adapter board

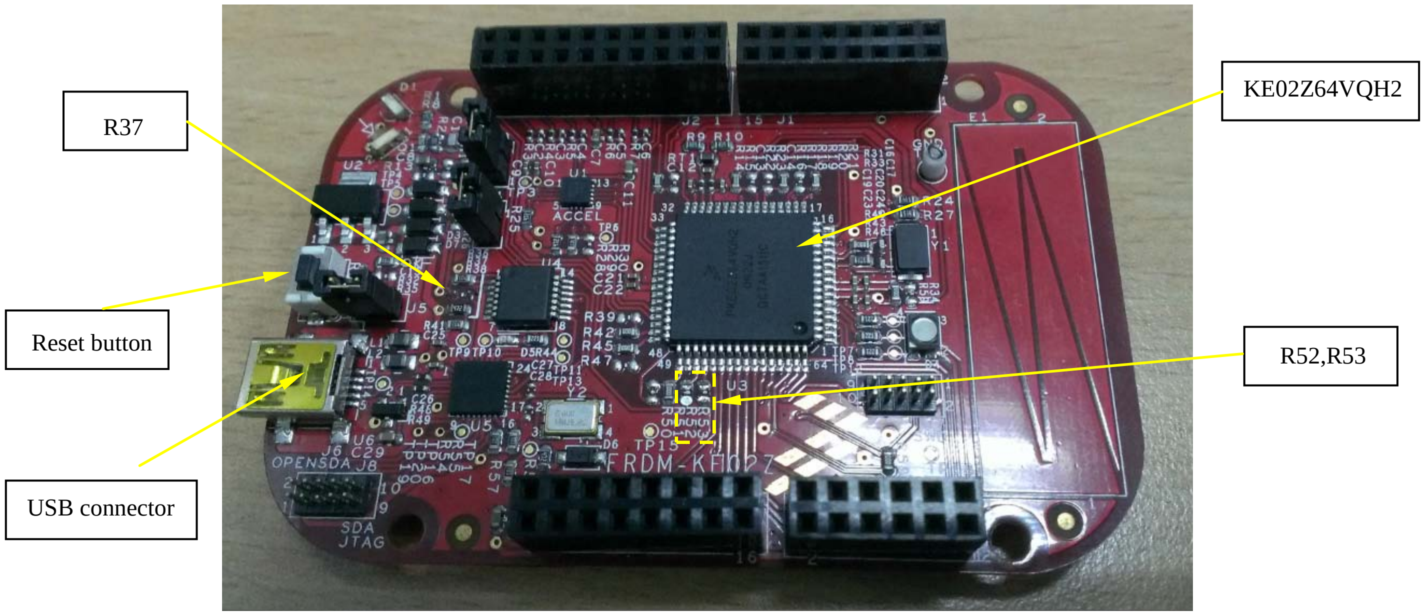


Figure 17. KE02 central control board (FRDM-KE02Z)

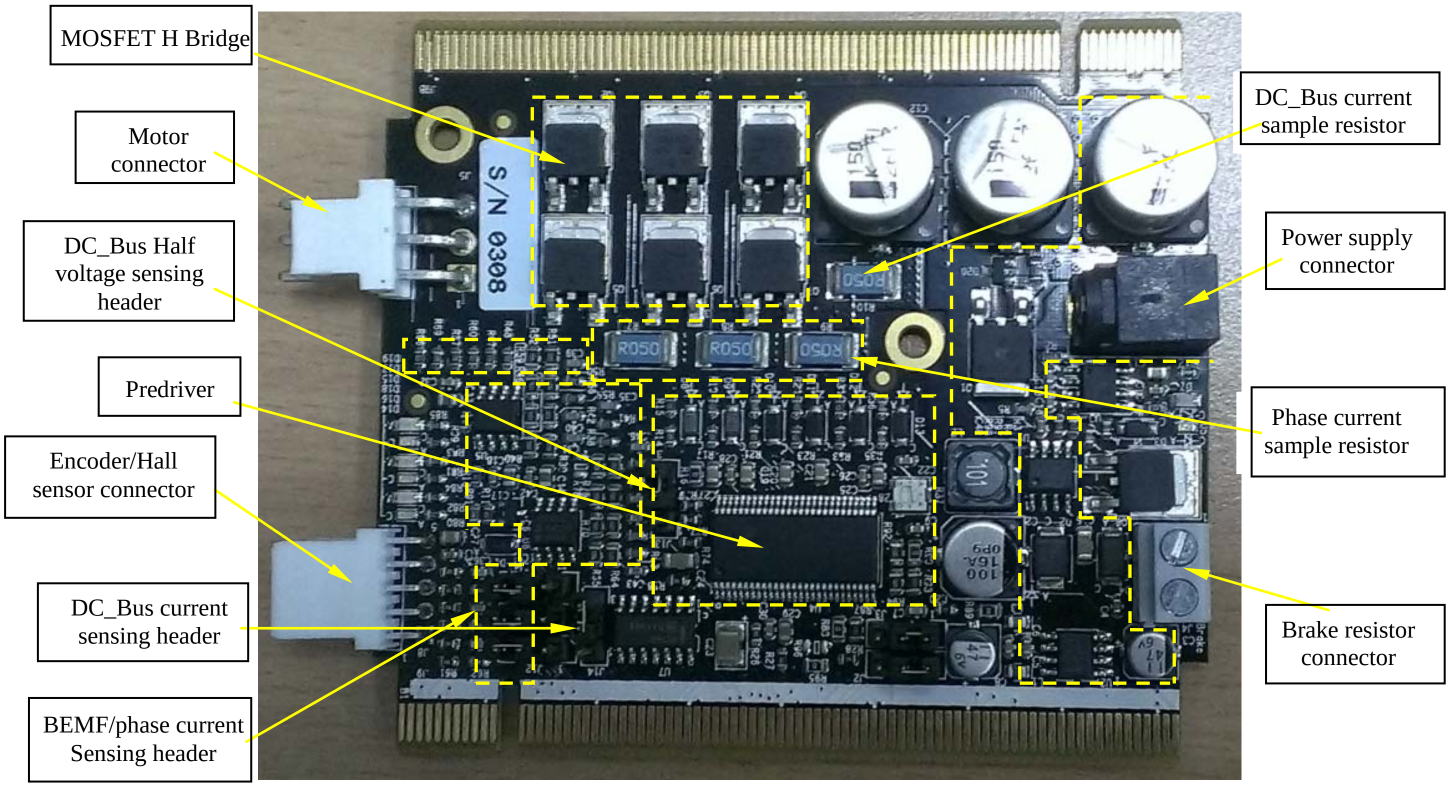

MOSFET H Bridge
Motor connector
DC_Bus Half voltage sensing header
Predriver
Encoder/Hall sensor connector
DC_Bus current sensing header
BEMF/phase current Sensing header
DC_Bus current sample resistor
Power supply connector
Phase current sample resistor
Brake resistor connector

Figure 18. Low-voltage motor control demo board

7.2 Software setup

The software developing environment is IAR Embedded Workbench for ARM V6.5. USB/SCI driver installation is required prior to the first usage of FreeMASTER. Driver installation is described in the MS Word file "Installation USB/SCI Bridge manual". After successfully installing the driver, select a virtual COM port attached to the USB port, and then FreeMASTER is ready to use.

8 References

The following references are available on freescale.com.

- KE02 Sub-Family Reference Manual
- TWRMCLV3PHUG: TWR-MC-LV3PH User's Guide
- TWR-MC-LV3PH Schematic
- TWR-SER-SCH Schematic
- KE02 Series Data Sheet

9 Revision history

Revision number	Date	Substantial changes
0	09/2013	Initial release
1	11/2013	Updated FDRM2TWRMC-KE board name to MC-FRDMKE02Z board.

Document Number: AN4796
Rev. 1
11/2013