3-Phase Sensorless BLDC Motor Control Using MC9S08MP16

Design Reference Manual

Devices Supported:

MC9S08MP16

Document Number: DRM117
Rev. 0
11/2009

freescale™
semiconductor

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 China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

DRM117
Rev. 0
11/2009

Chapter 1

Introduction

Chapter 2

Control Theory

Chapter 3

System Concept

Chapter 4

Hardware

Chapter 5

Software Design

Chapter 1
Introduction

1.1 Introduction

This document describes the cost effective design of a sensorless 3-phase brushless DC (BLDC) trapezoidal motor closed-loop speed control using MC9S08MP16. The design is targeted at the consumer, automotive and industrial applications. This cost-effective solution benefits from the dedicated motor control features in the Freescale Semiconductor MC9S08MP16 device.

1.1.1 Application Features and Components

The system is designed to drive a 3-phase brushless DC BLDC motor with trapezoidal back-EMF. The application features:

- Targeted at the MC9S08MP16 8-bit microcontroller.
- Sensorless 3-phase trapezoidal BLDC motor control with 6-step commutation (60, 120 degree control).
  - — Star or Delta 3-phase connected motor.
- Back-EMF zero-crossing used to synchronize the 6-step commutation with rotor position:
  - — MC9S08MP16 in-built high-speed comparator (HSCMP) detects the back-EMF voltage zero-crossing.
  - — Three sensorless synchronized commutation control algorithms incorporating:
    - – Commutation instant calculated directly from the period between two back-EMF zero-crossings.
    - – Commutation period synchronized with the back-EMF zero-crossing using a closed-loop according to a phase error — Synchronized PLL.
    - – Constant commutation period forced with the motor voltage controlled in a closed-loop according to a phase error — Forced PLL.
- Controlled acceleration and deceleration.
- Bidirectional rotation.
- Both motor and generator modes.
- One of two PWM techniques possible (determined in a code conditional compile):
  - — unipolar PWM commutation.
  - — bipolar PWM commutation.
- PWM frequency of 20 kHz.
- Application controlled from a Personal Computer using the FreeMASTER software tool.

- Software over-voltage and under-voltage protection.
- Hardware over-current protection.
- Three-phase low-voltage (24 V) power board.
- MC9S08MP16 daughter controller board.
- FreeMASTER software control interface (motor start/stop, speed setup).
- FreeMASTER software monitor.
  - FreeMASTER software graphical control page (required speed, actual motor speed, start/stop status, DC-Bus voltage level, motor current, system status).
  - FreeMASTER software speed scope.

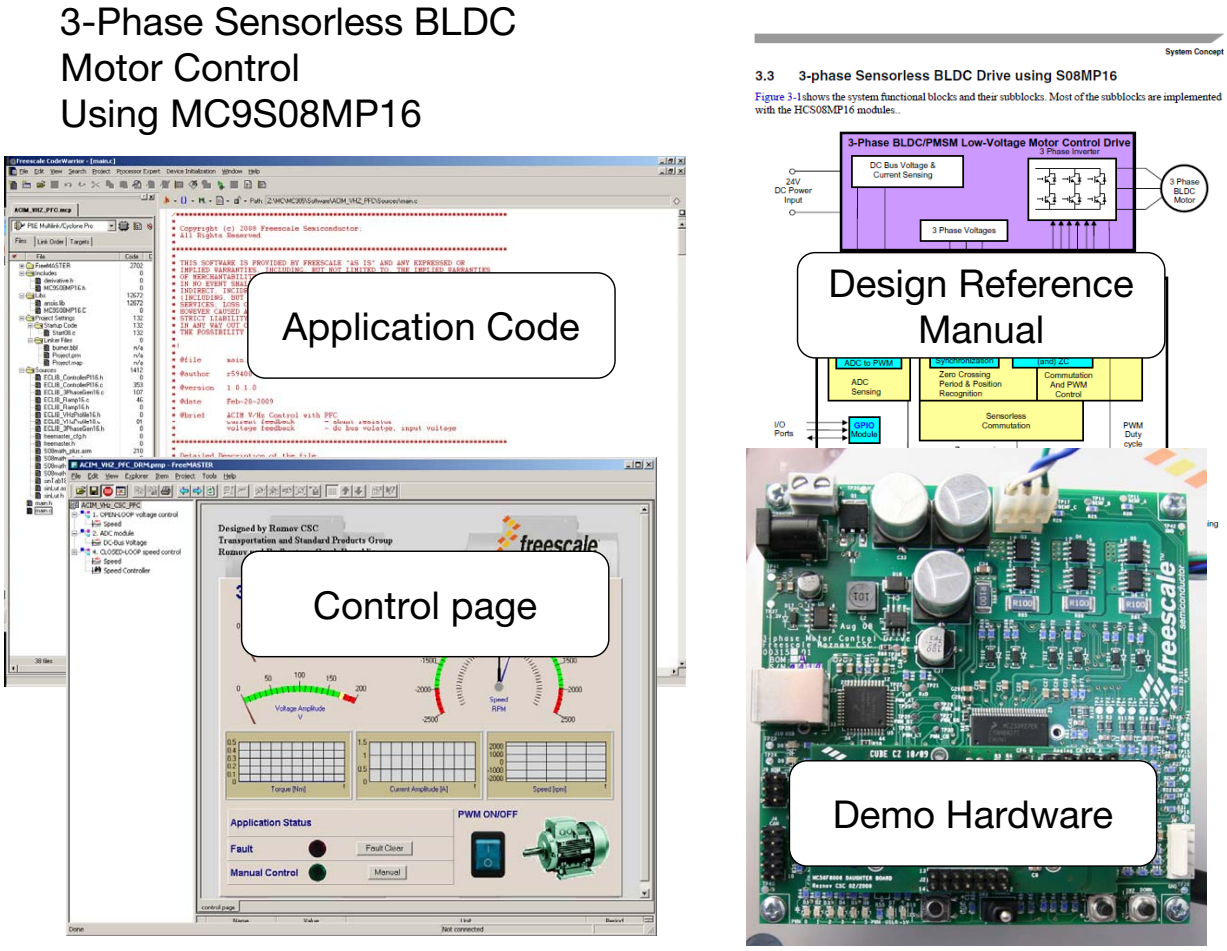

Figure 1-1. 3-Phase Sensorless BLDC Motor Control Using MC9S08MP16

Main application components available for customers are:

- Software — written in C-code using some library algorithms available for the HCS08.
- Hardware — based on Freescale universal motor control hardware modules.
- Documentation — this document.

1.1.2 Overview of Sensorless BLDC Variable Speed Drives

Replacing standard brushed DC motors, or variable speed universal motors, with maintenance-free, brush-less motors are a trend. The design of cost-effective variable-speed 3-phase motor control drives has become a prime focal point for the appliance designers and semiconductor suppliers.

The big push in this direction is driven by several factors:

- low power consumption,
- the flexibility that can be achieved by using electronic commutation,
- the maturity level and affordable price trend of power devices,
- the system efficiency optimization that microprocessor controlled drives can provide,
- the size, weight, and dissipated power reduction of the motors for a given mechanical power.

An advantageous solution for the brushless drives is the synchronous motor with permanent magnet rotor. Those motors achieve high efficiency by generating the rotor magnetic flux with rotor magnets. Therefore, they are used in applications which require high reliability, power, and size efficiency. These may be refrigerators, washing machines, dishwashers, pumps, fans, electrical power steering, and many others in the appliance and automotive applications.

The 3-phase synchronous motors with permanent magnets come in two very popular variants. The Sinusoidal PM synchronous motor and the trapezoidal BLDC motor. The sinusoidal PM synchronous motor is very similar to the trapezoidal BLDC (Electronically Commuted) motor. There are two main differences:

- Motor construction:
  - the shape of the BEMF inducted voltage — sinusoidal (PM synchronous motor) versus trapezoidal (BLDC) motor.
- Control — the shape of the control voltage:
  - 3-phase sinusoidal (all three phases connected at one time) versus rectangular 6-step commutation (one phase is non-conducting at any time).

The drive system cost is a very sensitive parameter for any mass production application. The trapezoidal BLDC motor is a good candidate for the cost-effective drives, thanks to its simple 6-step commutation and rotor position detection. This enables the use of a cost effective 8-bit microcontrollers.

The BLDC motor control system requires the detection of the rotor position in order to create synchronous commutation. The rotor position can be detected with Hall sensors or by using a sensorless technique. Sensorless position detection saves on the system cost. It displaces the Hall sensors, cables, and connectors, which leads to a lower system cost, a higher reliability, and less bushings (important, for example for compressors).

One well-tried method is to use the back-EMF zero-crossing of the non-conductive motor phase. Such a method can be easily implemented on 8-bit microcontrollers. The HCS08MP16 is a very good choice for such application, thanks to its peripheral like the Flextimer module, high speed comparators, and others.

This reference design manual describes the basic motor theory, the system design concept, hardware implementation, and the software design, including the FreeMASTER software visualization tool.

1.2 Freescale Controller Advantages and Features

The MC9S08MP16 is a member of the low-cost, high-performance HCS08 Family of 8-bit microcontroller units (MCUs). All MCUs in this family use the enhanced HCS08 core and are available with a variety of modules, memory sizes, memory types, and package types.

The Freescale MC9S08MP16 microcontroller is well-suited for digital motor control offering many dedicated peripherals, such as Flextimer (FTM) modules, analog-to-digital converters (ADC), timers, communications peripherals (SCI, SPI, I^2C), and on-board flash and RAM.

The MC9S08MP16 device provides the following features:

- Two Flextimer modules with a total of eight channels.
- One analog-to-digital converter (ADC) 13-channel, 12-bit resolution, 2.5 μs conversion time.
- One 8-bit modulo counter with an 8-bit prescaler and overflow interrupt.
- Interrupt Priority Controller (IPC) with four programmable interrupt priority levels.
- One differential Programmable Gain Amplifier (PGA) with a programmable gain (x1, x2, x4, x8, x16, or x32).
- Three fast analog comparators (HSCMP1, HSCMP2, and HSCMP3) with both positive and negative inputs.
- Three 5-bit digital-to-analog converters (DAC) used as a 32-tap voltage reference.
- Two programmable delay blocks (PDB). PDB1 synchronizes the PWM with the samples from the ADC, PDB2 synchronizes the PWM with comparison window of the analog comparators, PWM output synchronizes with the FTM PWM output.
- One serial peripheral interface (SPI).
- One serial communications interface (SCI) with LIN slave functionality.
- One inter-integrated circuit (I^2C) port.
- On-board 3.3 V to 2.5 V voltage regulator for powering internal logic and memories.
- Integrated power-on reset and a low-voltage interrupt module.
- Multiplexing of all pins with the general-purpose input/output (GPIO) pins.
- Computer operating properly (COP) watchdog timer.
- External reset input pin for hardware reset.
- Single-wire background debug interface (BDM).
- Internal Clock Source (ICS) — Containing a frequency-locked-loop (FLL) controlled by an internal or external reference.

1.2.1 Peripheral Application Usage

The sensorless BLDC benefits greatly from the flexible PWM module, high-speed comparators HSCMP, a fast ADC, program delay block PDB and the modulo timer MTIM.

The Flextimer offers flexibility in its configuration, enabling efficient 3-phase motor control and BLDC motor 3-phase commutation. Each FTM module supports an initialization trigger function which is used to trigger both PDBs.

The **FTM1** (Flextimer module) features used in the application:

- Input capture mode for the tacho period signal measurement.
- HSCMP output connected to the FTM1 input for back-EMF zero-crossing detection.

The **FTM2** (Flextimer module) features used in the application:

- Generation of six PWM signals.
- Set combine mode with complementary outputs.
- Dead time inserted.
- Fault protection enabled for external over current fault trigger.
- Synchronized PWM register update on timer overflow event.
- Synchronized PWM output configuration with the FTM1 compare to generate a 6-step BLDC commutation.
- Hardware trigger generation for ADC synchronization with the PWM signal (optional feature, not required for application functionality).

The **ADC** features used in the application:

- Hardware trigger for an ADC start generated by the PDB1.
- Linear successive approximation algorithm with a 12-bit resolution.
- High-speed conversion.
- 5 MHz module clock.

The **PDB1** (Program delay block) features used in the application (optional, not required):

- The controllable delay from the Flextimer module's SYNC output to the sample trigger input of the programmable gain amplifiers and the ADC.

The **PDB2** features used in the application:

- Window mode synchronized with the FTM2 (motor phase PWM).
- The PDB2 generates a window for the comparator HSCMP1 to sense the zero-crossing synchronously with the FTM2 (motor phase PWM).
- Both hardware and software triggering.

The **HSCMP2** (High-speed comparator) features used in the application:

- 3-phase back-EMF voltages zero-crossing detection for rotor positional feedback.
- Generates FTM1 input capture for zero-crossing time instant detection.

The **DACx** features used in the application:

- Sets the compare level for each HSCCMPx.

The **MTIM** (Modulo timer) features used in the application:

- 8-bit modulo limit running.
- Sets the time base for speed control and the long periods timing loop.
- Overflow interrupt generation.

1.3 Front Matter

1.3.1 Bibliography

1. *MC9S08MP16 Reference Manual*, Freescale Semiconductor, 2009
2. *3-Phase BLDC/PMSM Low-Voltage Motor Control Drive*, Freescale Semiconductor, 2009
3. *MC9S08MP16 Controller Daughter Board for BLDC/PMSM Motor Control Drive*, LVBLDCMP16DBUM, Freescale Semiconductor, 2009
4. *3-phase BLDC Motor Control with Sensorless Back-EMF ADC Zero Crossing Detection using DSP 56F80x*, AN1913, by Prokop L., Motorola, 06/2001
5. *Sensorless BLDC Motor Control on MC68HC908MR32 Software Description*, AN2355, by Prokop L., Motorola, 11/2002
6. *Sensorless BLDC Motor Control on MC68HC908MR32 Software Porting to Customer Motor*, AN2256, by Prokop L., Motorola, 11/2002
7. *Code Warrior Development Studio for Freescale Microcontrollers V6.2*, Freescale Semiconductor, 2008
8. *PC Master Software Usage*, Freescale Semiconductor, AN2395, 2002
9. *3-Phase PM Synchronous Motor Vector Control*, AN1931, by Prokop L., Grasblum P., 2005
10. *Embedded Software Library for S08 User's Manual*, Freescale Semiconductor, 2009

For a current list of documentation, refer to www.freescale.com.

1.3.2 Acronyms and Abbreviations

Table 1-1 contains sample acronyms and abbreviations used in this document.

Table 1-1. Acronyms and Abbreviated Terms

Term	Meaning
AC	Alternating current.
ADC	Analog-to-digital converter.
BDM	Background Debug Mode.
back-EMF,	Back Electro-Magnetic Force. In this case, Voltage generated by BLDC motor rotation.
BLDC motor	Brushless DC motor. In this article, it means a synchronous electrically commuted motor with a permanent magnet and a trapezoidal BEMF shape.
COP	Computer operating properly (watchdog timer)
DT	Dead time: a short time that must be inserted between the turning off of one transistor in the inverter half bridge and the turning on of the complementary transistor, due to the limited switching speed of the transistors.
DC-Bus	Direct current bus of the 3-phase power stage.
GPIO	General parallel input/output modul.
FTMx	Flextimer module.

Table 1-1. Acronyms and Abbreviated Terms (continued)

Term	Meaning
GPIO	General-purpose input/output module.
HSCMP2	High-speed comparator module 2.
I/O	Input/output interfaces between a computer system and the external world — a CPU reads an input to sense the level of an external signal and writes to an output to change the level of an external signal.
LED	Light emitting diode.
MC9S08MP16	A Freescale HCS08 family member dedicated to motor control.
MTIM	Modulo timer module.
PDBx	Programmable Delay Block module x (=1 or 2).
PLL	Phase-locked loop: a clock generator circuit in which a voltage-controlled oscillator produces an oscillation that is synchronized to a reference signal.
Synchronized PLL	Synchronized phase closed loop. A closed-loop where the commutation period is synchronized with the rotor position according to a phase error.
Forced PLL	Forced commutation phase-locked loop. A closed-loop where the amplitude of the 3-phase voltage system is controlled according to a phase error.
PM	Permanent Magnet.
PM synchronous motor	Permanent Magnet Synchronous Motor. In this article, it means a synchronous motor with a sinusoidal BEMF shape.
PWM	Pulse-width modulation.
RPM	Revolutions per minute.
SCI	Serial communication interface module: a module that supports asynchronous communication.
SPI	Serial peripheral module.
six-step commutation	Commutation technique used to create rotational field for trapezoidal BLDC motors. Sometimes also called 60, 120 degree control.

1.3.3 Glossary of Symbols

Table 1-2 shows a glossary of symbols used in this document.

Table 1-2. Glossary of Symbols

Term	Definition
$c_{ZCtoCMT}$	A constant for calculation of the period from the back-EMF zero-crossing to commutation.
f^*_{estim}	Commutation estimated frequency.
p_p	Stator pole-pair number.
p	Stator pole number.
periodCmtPreset	Period commutation preset.
periodZCFlt	Filtered period between two consecutive back-EMF zero-crossing occurrence instants.
periodZCtoCmt	Period from the back-EMF zero-crossing to commutation.

Table 1-2. Glossary of Symbols (continued)

Term	Definition
t_{CMT}, *timeCmt,*	BLDC (six-step) commutation instant.
t_{ZC}	Back-EMF zero-crossing occurrence instant.
$t_{ZCForced}$	Back-EMF zero-crossing occurrence instant.
$t^*_{ZCestim}$	Estimated back-EMF zero-crossing occurrence instant.
T_{estim}	Commutation period estimated.
T_{Forced}	Forced back-EMF zero-crossing and commutation period.
T_{ZC}, *periodZC*	Period between two consecutive back-EMF zero-crossing occurrence instants.
periodZC0	Previous period between two consecutive back-EMF zero-crossing occurrence instants.
$T_{ZCerror}$	Error between the estimated and real back-EMF zero-crossing occurrence instants.
V_{BLDC}	Voltage applied to the BLDC motor; 3-phase voltage system amplitude.

Chapter 2
Control Theory

2.1 3-Phase BLDC Motor

The brushless DC motor (BLDC motor) is also referred to as an electronically commuted motor. There are no brushes on the rotor, and commutation is performed electronically at certain rotor positions. The stator magnetic circuit is usually made from magnetic steel sheets. Stator phase windings are inserted in the slots (distributed winding) as shown in Figure 2-1, or it can be wound as one coil on the magnetic pole. Magnetization of the permanent magnets and their displacement on the rotor are chosen in such a way that the back-EMF (the voltage induced into the stator winding due to rotor movement) shape is trapezoidal. This allows a rectangular-shaped 3-phase voltage system (see Figure 2-2) to be used to create a rotational field with low torque ripples.

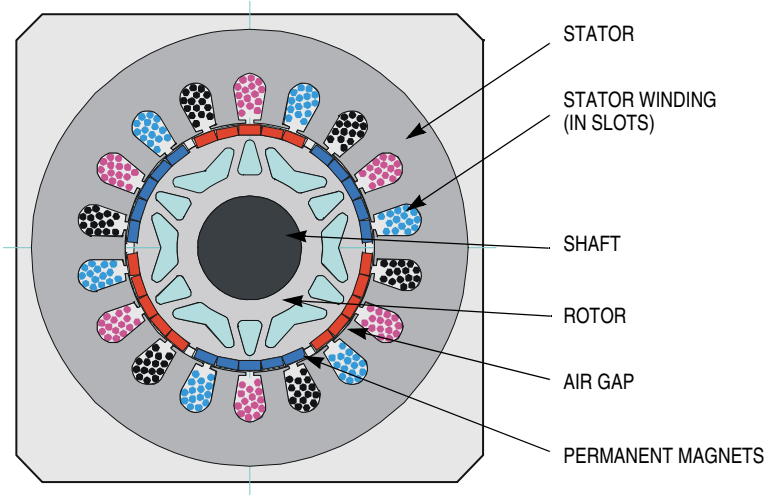

Figure 2-1. . BLDC Motor Cross Section

The motor can have more than just one pole-pair per phase. This defines the ratio between the electrical revolution and the mechanical revolution. The BLDC motor shown has three pole-pairs per phase, which represent three electrical revolutions per one mechanical revolution.

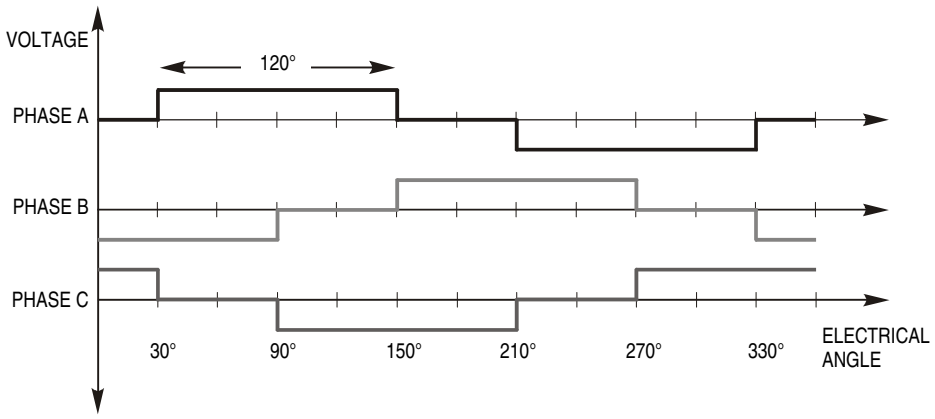

Figure 2-2. . 3-Phase Voltage System

2.2 6-Step Commutation for a 3-Phase BLDC Motor Control

The easy-to-create rectangular shape of applied voltage ensures the simplicity of control and drive. The BLDC motor rotation is controlled by a 6-step commutation technique (sometimes called 60, 120 degree control). A simplified principle is shown in Figure 2-3.

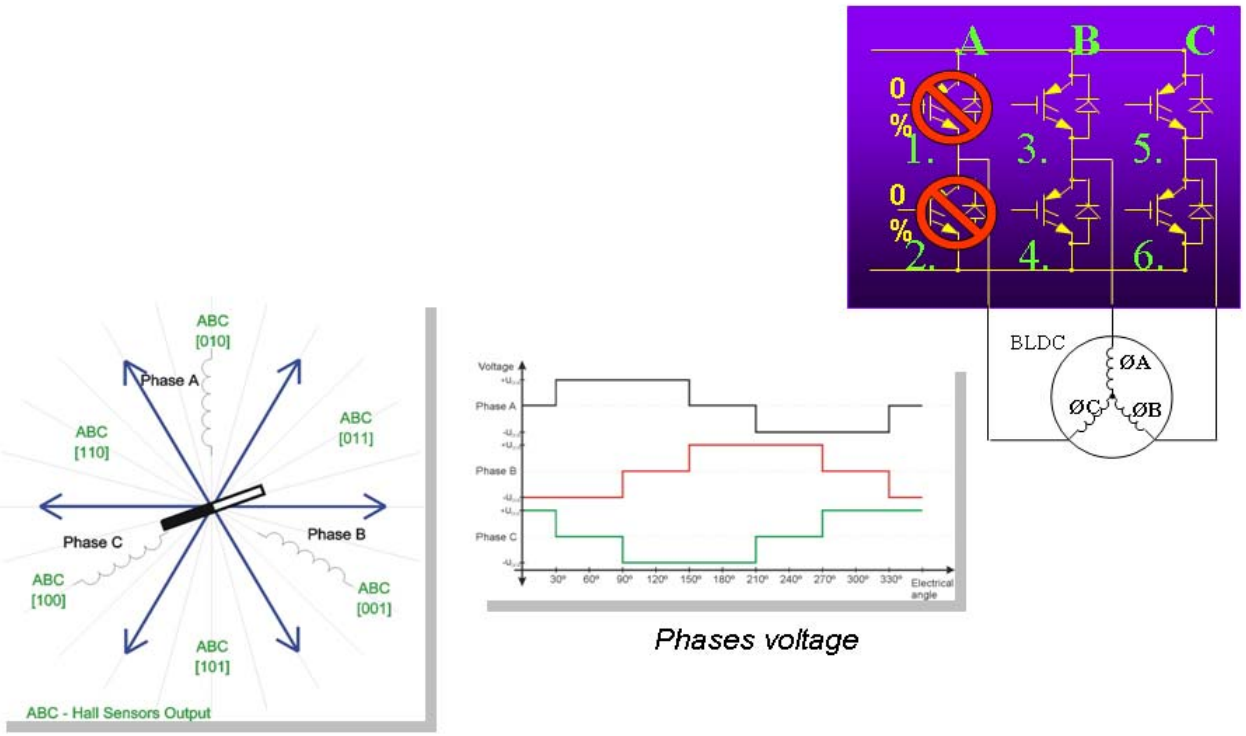

Figure 2-3. 6-Step BLDC Commutation for a 3-Phase Voltage System

The 6-step technique which creates the voltage system according to Figure 2-2, with six vectors over one electronic rotation. One of the most important characteristics of the 6-step BLDC motor control is that one

of the phases is switched off at a time. This is important for the rotor position sensing, as will be explained below.

The applied voltage needs to have amplitude and phase aligned with the back-EMF. So the BLDC motor controller needs to:

- Control the applied 3-phase amplitude.

The applied voltage can be controlled by the DC-Bus voltage level regulation or with a Pulse Width Modulation (PWM) technique. This Sensorless BLDC Control application uses the PWM technique as described in Section 2.2.2, "Voltage Amplitude Controlled by PWM."

- Synchronize the 6-step commutation with the rotor position.

The rotor position must be known at certain angles in order to align the applied voltage with the back-EMF (voltage induced due to movement of the PM). The alignment between back-EMF and commutation events is very important. In this condition, the motor behaves as a DC motor and runs at the best working point. Thus, simplicity of control and good performance make this motor a natural choice for low-cost and high-efficiency applications.

When the back-EMF voltage is aligned with the applied voltage, the stator and rotor field relation is maintained according to Figure 2-4. Under that condition, the BLDC motor is controlled with minimal torque ripple and maximal power efficiency.

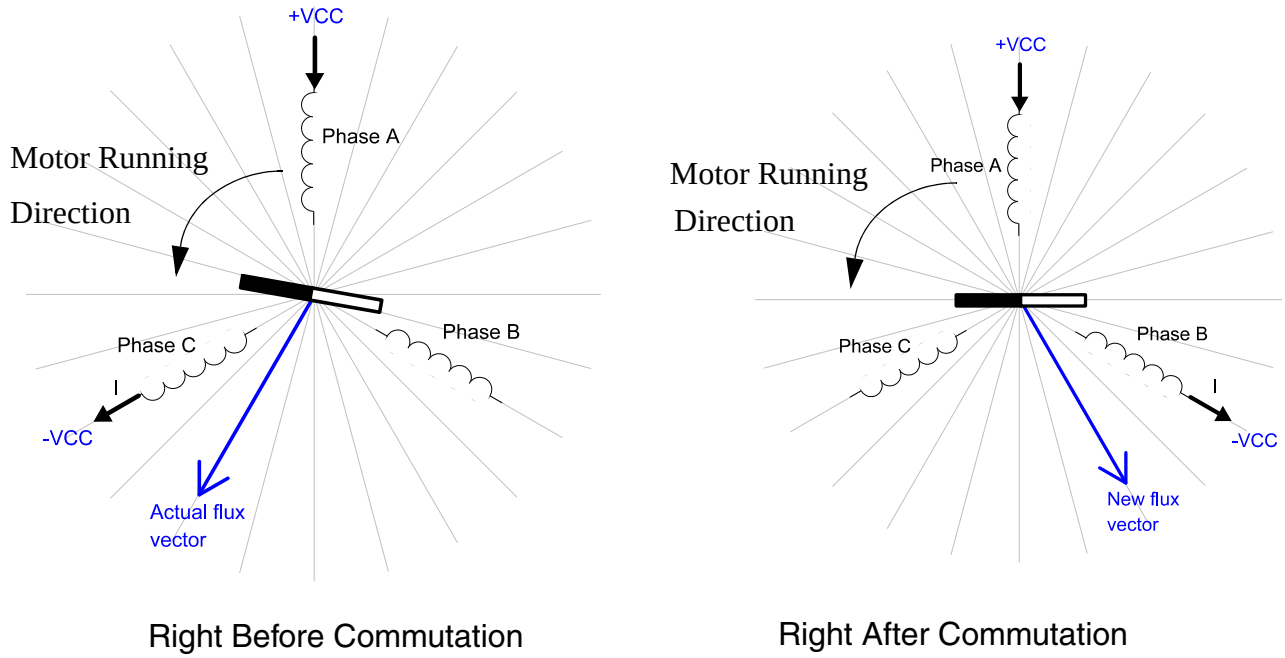

Figure 2-4. 6-Step Commutation Stator Field Relative to Rotor

2.2.1 3-Phase Power Stage

The 3-phase 6-step voltage system (see Figure 2-2) is created by a 3-phase power stage with six IGBTs (MOSFET) power switches controlled by the MCU on-chip PWM module (see Figure 2-5). When using MC9S08MP16, the PWM module is realized by six channels of its Flextimer module.

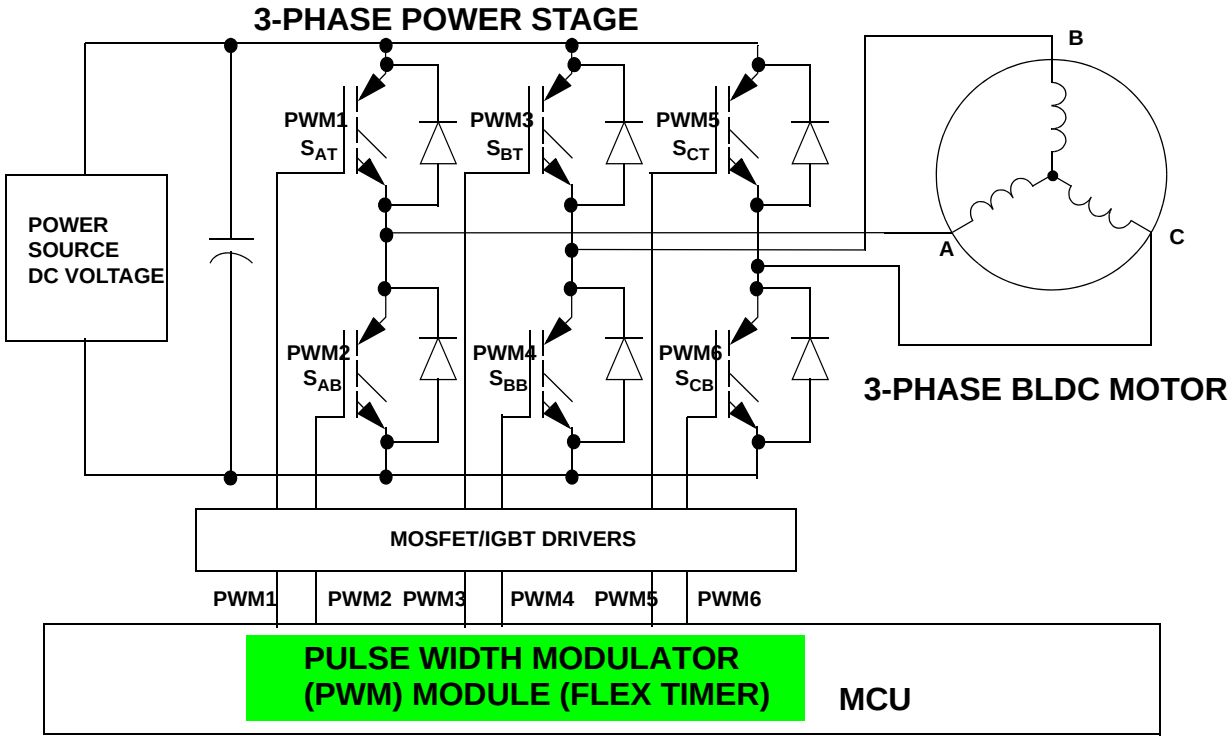

Figure 2-5. Power Stage — Motor Topology

2.2.2 Voltage Amplitude Controlled by PWM

As described in the previous sections, the amplitude of a 3-phase voltage system needs to be controlled. The most common BLDC control topology uses the power stage from Figure 2-5 with a constant POWER SOURCE DC VOLTAGE. Therefore, the 3-phase average voltage amplitude is controlled by a PWM technique of the top and bottom transistors. The 6-step controller uses one of two PWM techniques:

1. Bipolar PWM switching
2. Unipolar PWM switching

There are a few derivatives of the two PWM switching techniques. According to the operating quadrants of the power stage voltage and current:

1. 4-quadrant power stage control
2. 2-quadrant power stage control

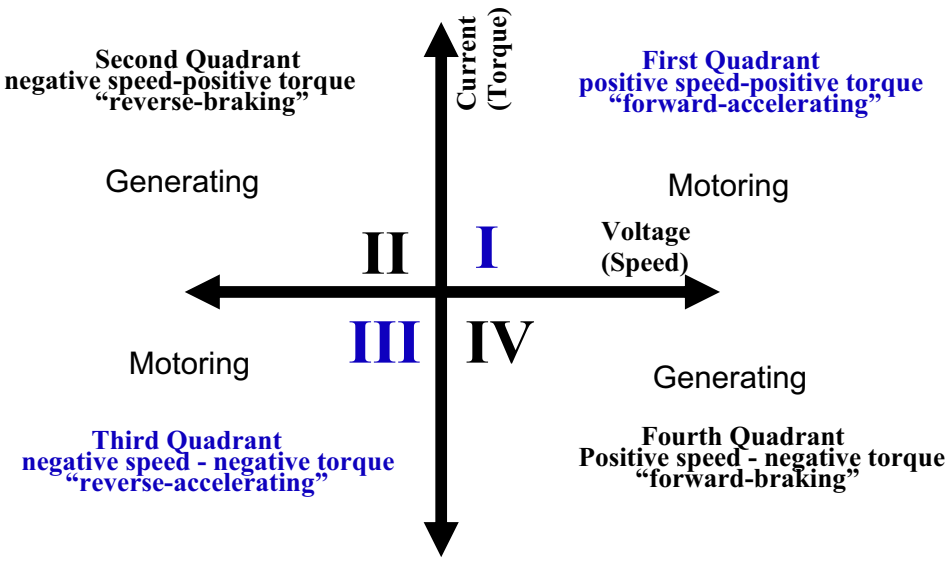

Figure 2-6. Quadrant of Operations

The bipolar/unipolar switching and the operating quadrant control is determined by all the six S_{At} S_{Cb} switchings.

The 2-quadrant operation provides a defined voltage and current of the same polarity (positive voltage with positive current or negative voltage and negative current) which are the operating quadrants I and III in Figure 2-6.

The 4-quadrant PWM switching covers the operation of the generated voltage in all four quadrants. This is provided using complementary switching of the top and bottom transistors. The benefits of the 4-quadrant PWM operation are:

- Motoring and generating mode control (possibility of braking the motor).
- Linear operation in all four quadrants.

The MC9S08MP16 is able to control the 3-phase power stage with unipolar and bipolar PWW with the more advanced 4-quadrant operation. Therefore, in the following sections, we will cover the transistor control pattern for the 4-quadrant PWM technique.

2.2.2.1 3-Phase BLDC 6-Step Control with Bipolar (Hard) PWM Switching

The PWM commutation pattern of the bipolar complementary (4-quadrant) PWM switching technique is shown in Figure 2-7.

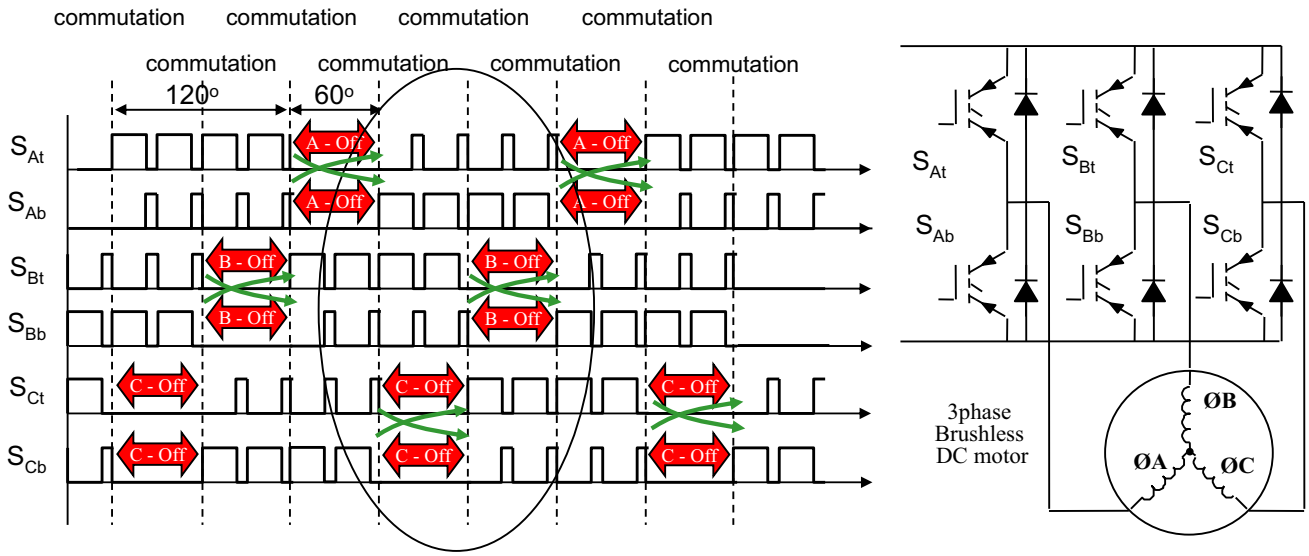

Figure 2-7. Bipolar PWM Switching

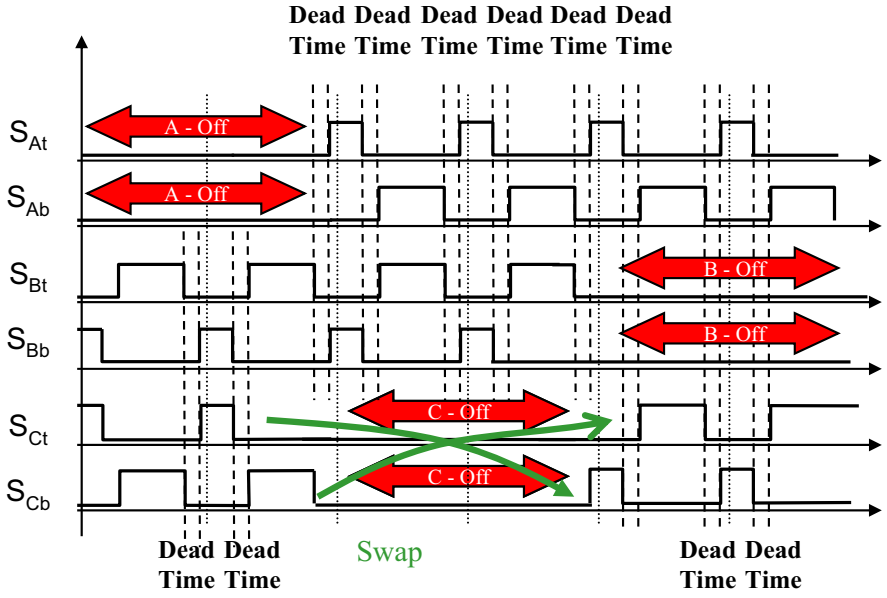

Figure 2-8. Bipolar PWM Switching — Detail

Details of the technique are shown in Figure 2-8. From Figure 2-8 we can see the characteristic of the bipolar 4-quadrant (complementary) switching. The bipolar switching requires that the top and bottom switch PWM signals need to be swapped. Another important detail is the introduction of dead time insertion in the complementary top and bottom signals. This dead time insertion is typical for all 4-quadrant power stage operations. The 4-quadrant operation is enabled by the complementary operation of the top and bottom switches (the bottom switch of one phase is almost the negative of the top switch).

This requires the insertion of a dead time, since the switching transient will cause a DC-Bus short circuit with fatal power stage damage.

The bipolar PWM switching is not as popular as the unipolar switching due to a worse electromagnetic emission of the motor. This is because the PWM ripple is twice that of the DC-Bus voltage. On the other hand, this switching is better for sensorless rotor position sensing. This switching can be implemented using MC9S08MP16.

2.2.2.2 3-Phase BLDC 6-Step Control with Unipolar (Soft) PWM Switching

There are more modifications to the unipolar PWM switching. One of the most common 6-step commutation with unipolar complementary (4-quadrant) PWM switching techniques is shown in Figure 2-9.

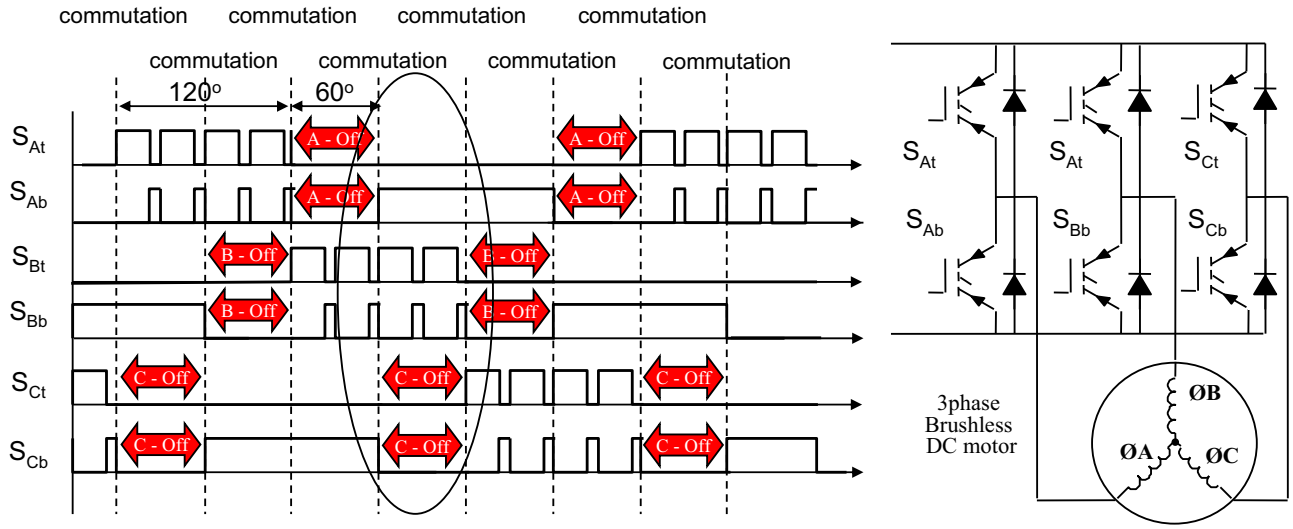

Figure 2-9. Unipolar PWM Switching

The unipolar PWM switching can control the BLDC motor with an almost identical voltage and current performance. The only difference is that the PWM duty cycle = 0 creates an average voltage of 0. The main advance of the unipolar PWM switching is better EMC compatibility, due to half of the voltage ripple compared to the bipolar switching. This switching is implemented as default for this application, using the MC9S08MP16.

2.3 Why Sensorless Control?

As explained in the previous section, rotor position must be known in order to drive a brushless DC motor. If any sensors are used to detect the rotor position, the sensed information must be transferred to a control unit (see Figure 2-10). Therefore, additional connections to the motor are necessary. This may not be acceptable for some applications. There are at least two reasons why you might want to eliminate the position sensors:

- Inability to make additional connections between position sensors and the control unit.
- Cost of the position sensors and wiring.

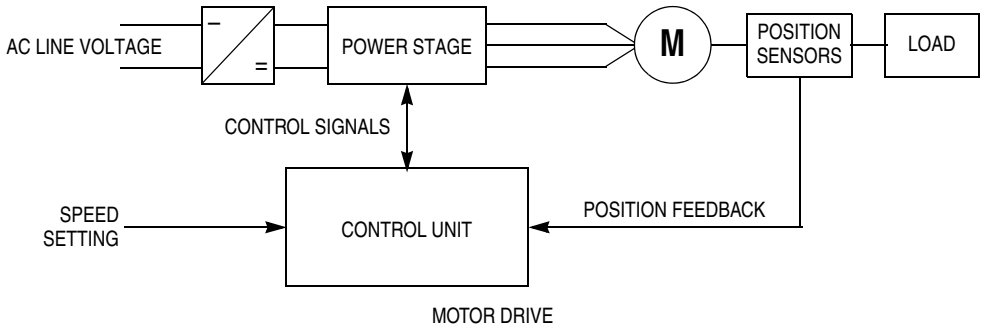

Figure 2-10. Classical System

2.4 Sensorless Technique Based on Back-EMF Zero-Crossing

The rotor position estimation is based on the back-EMF voltage induced in the stator phases due to rotor flux (permanent magnet) rotation. The back-EMF voltage phase corresponds with the rotor position relative to the stator position.

2.4.1 Back-EMF Zero-Crossing Sensing Background

A BLDC motor is controlled with a 6-step commutation. The 6-step commutation specific feature is that one of the three phases is off at a time. The off phase is determined by the 6-step commutation sector.

After the commutation transient (current recirculation — the fly-back diodes are conducting the decaying phase current), the current of phase x:

- $I_{Sx} = 0$ and so U_{Sx} = BEMFC

See Figure 2-11.

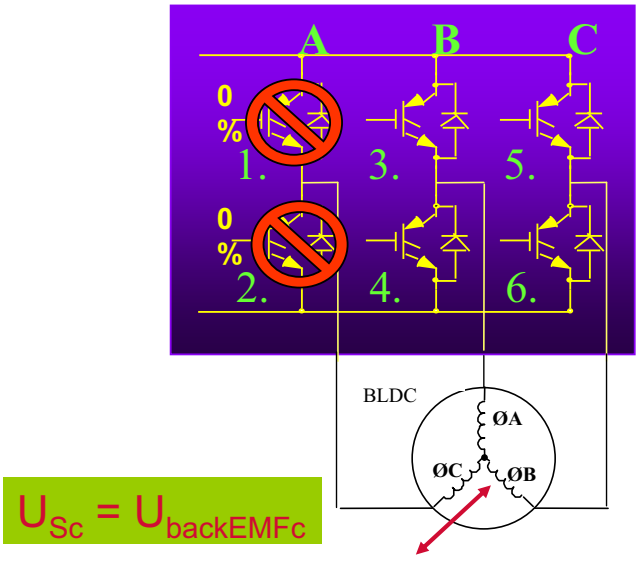

Figure 2-11. 6-Step Commutation Specifics

So, the phase back-EMF voltage can be simply measured when using the 6-step commutation.

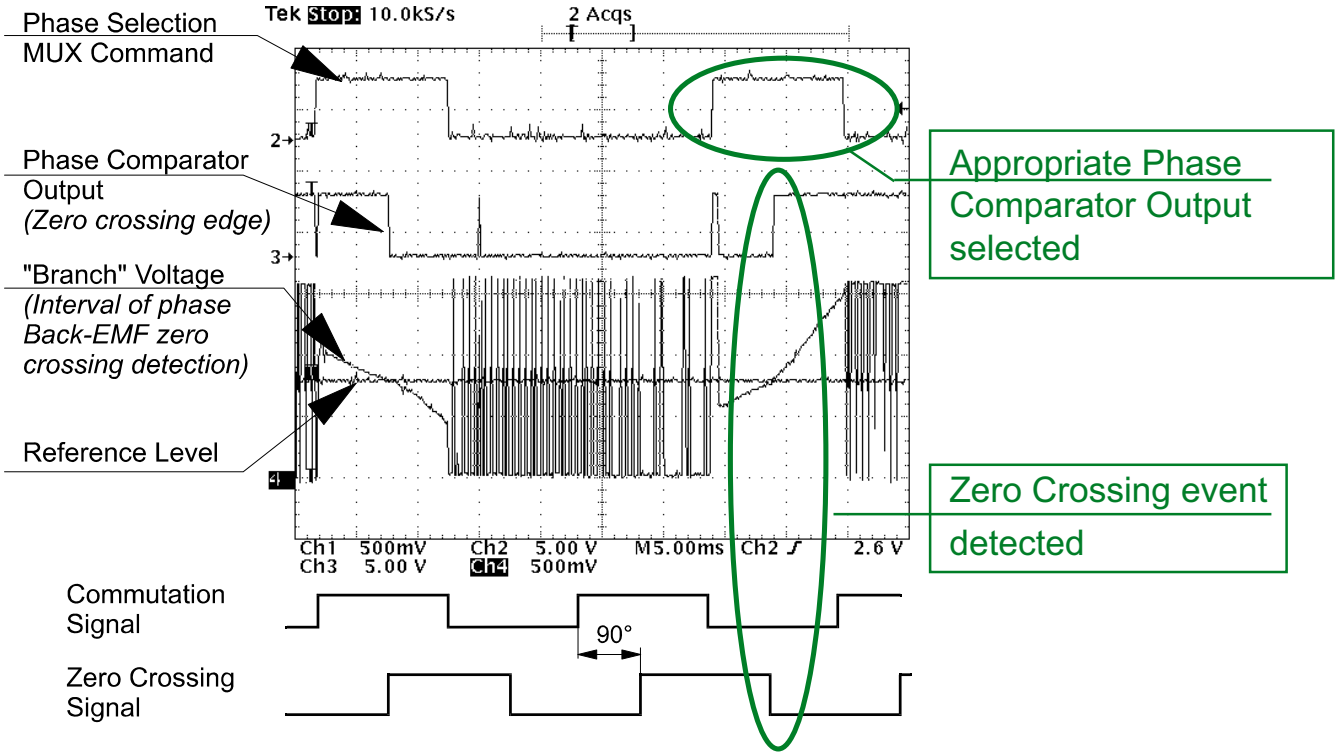

Figure 2-12. The Back-EMF Zero-Crossing Detection

Figure 2-12 shows the relationship between back-EMF zero-crossing and an ideally synchronized commutation. For the ideal commutation time instant shown in Figure 2-4, the phase back-EMF zero-crossing is in the middle of the commutations.

So the back-EMF zero-crossing is used to synchronize the 6-step commutation with the rotor position in order to get the required constant torque performance of the BLDC motor.

2.4.2 Power Stage — Motor System Model

One of the issues of the phase back-EMF zero-crossing sensing is, that the motor stator phases central point is usually not accessible. This section describes how to get the right zero-crossing comparator reference signal.

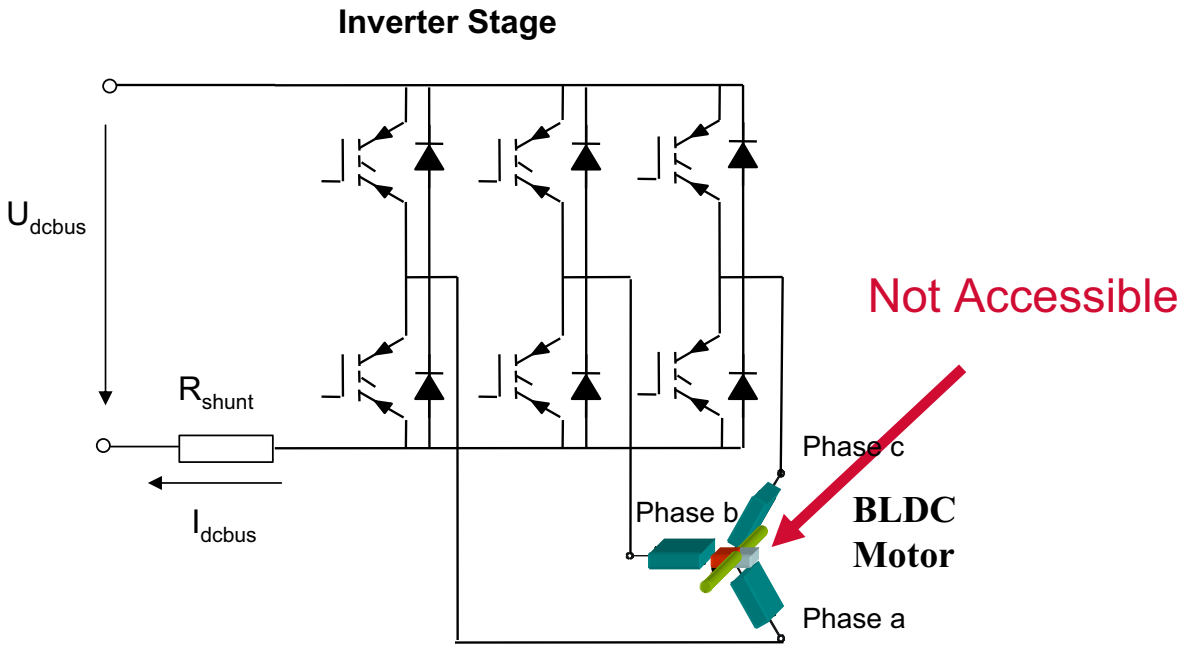

Figure 2-13. Stator Phase Central Point Not Accessible

In order to explain and simulate the idea of back-EMF sensing techniques, a simplified mathematical model based on the basic circuit topology has been created. See Figure 2-14 or Figure 2-17.

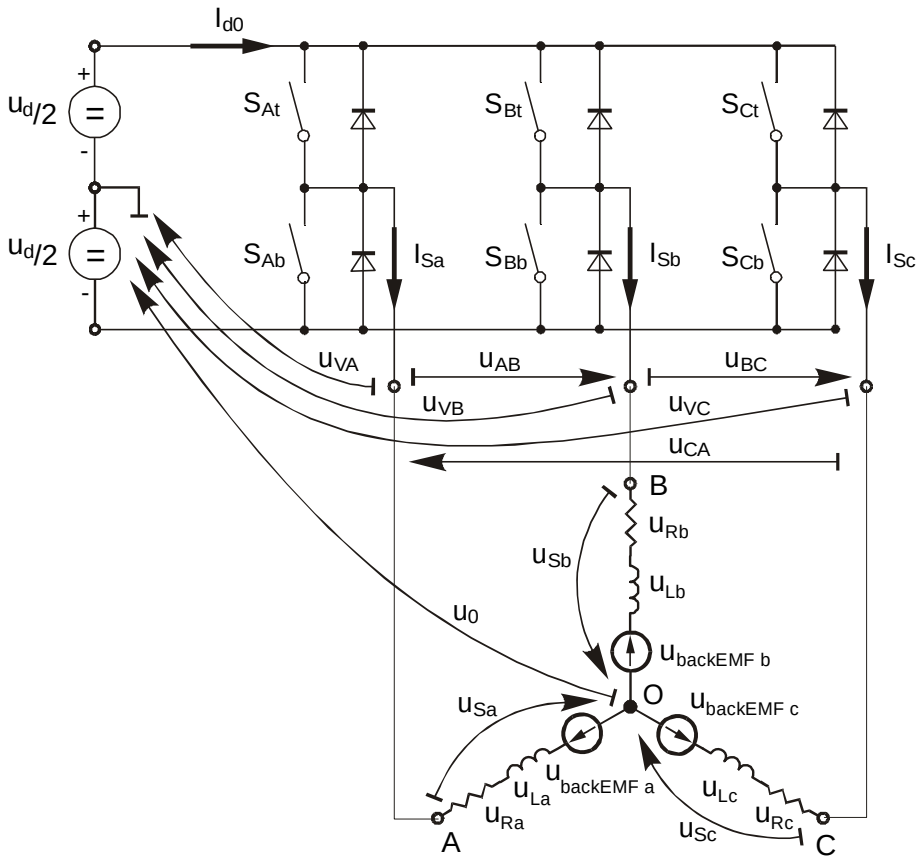

Figure 2-14. Power Stage — Motor Topology with a DC-Bus Voltage/2 Reference

The motor-drive model consists of a normal 3-phase power stage plus a brushless DC motor. Power for the system is provided by a voltage source (U_d). Six semiconductor switches ($S_{A/B/C\ t/b}$), controlled elsewhere, allow the rectangular voltage waveforms (see Figure 2-2) to be applied. The natural voltage level of the whole model is put at one half of the DC-Bus voltage. This simplifies the mathematical expressions.

2.4.2.1 Stator Winding Equations

The BLDC motor is usually very symmetrical. All phase resistances, phase and mutual inductances, flux-linkages can be thought of as equal to, or as a function of the position q with a 120° displacement. The task of this section is to explain the background of the back-EMF sensing and to demonstrate how the zero-crossing events can be detected. Parasitic effects that negatively influence the back-EMF detection are discussed and their nature analyzed.

The electrical BLDC motor model then consists of a set of the following stator voltage equations;

$$\begin{bmatrix} u_{Sa} \\ u_{Sb} \\ u_{Sc} \end{bmatrix} = R_S \begin{bmatrix} i_{Sa} \\ i_{Sb} \\ i_{Sc} \end{bmatrix} + \frac{d}{dt} \begin{bmatrix} \psi_{Sa} \\ \psi_{Sb} \\ \psi_{Sc} \end{bmatrix} \qquad \textit{Eqn. 1}$$

it is also evident that the stator currents sum is 0:

Eqn. 2

$$i_A + i_B + i_C = 0$$

The Equation 1 can be rewritten as:

Eqn. 3

$$\begin{bmatrix} u_{Sa} \\ u_{Sb} \\ u_{Sc} \end{bmatrix} = R_S \begin{bmatrix} i_{Sa} \\ i_{Sb} \\ i_{Sc} \end{bmatrix} + \begin{bmatrix} u_{backEMFa} \\ u_{backEMFb} \\ u_{backEMFc} \end{bmatrix} + \begin{bmatrix} L_{Saa} & L_{Sab} & L_{Sac} \\ L_{Sba} & L_{Sbb} & L_{Sbc} \\ L_{Sca} & L_{Scb} & L_{Scc} \end{bmatrix} \frac{d}{dt} \begin{bmatrix} i_{Sa} \\ i_{Sb} \\ i_{Sc} \end{bmatrix}$$

In the Equation 3, we assume that the phase resistance of all phases is constant and equal.

For further calculation we need to introduce voltage u_0, which is the voltage between the motor winding central point and the virtual ground.

Eqn. 4

$$u_0 = u_{VA} - u_{Sa} \qquad u_0 = u_{VB} - u_{Sb} \qquad u_0 = u_{VC} - u_{Sc}$$

And so:

Eqn. 5

$$u_0 = u_{VC} - u_{backEMFc} - R \times i_{Sc} - \sum_{x=A}^{C} L_{Scx} \frac{d}{dt} i_{Sx}$$

The Equation 5 can be written for any motor phase, and so we can get Equation 6:

Eqn. 6

$$3 \times u_0 = \sum_{x=A}^{C} u_{Vx} - \sum_{x=A}^{C} u_{backEMFx} - \sum_{y=A}^{C} \sum_{x=A}^{C} L_{Syx} \frac{d}{dt} i_{Sx}$$

Since the central stator winding point is not accessible, we need to know the relationship between the motor phase u_{Sx} voltage and the branch voltage u_{VX}. The Equation 7

$$u_{Sc} = u_{VC} - u_0$$

Eqn. 7

$$u_{Sc} = u_{VC} - \frac{1}{3}\left(\sum_{x=A}^{C} u_{Vx} - \sum_{x=A}^{C} u_{backEMFx} - \sum_{y=A}^{C} \sum_{x=A}^{C} L_{Syx} \frac{d}{dt} i_{Sx}\right)$$

leads into the Equation 8:

Eqn. 8

$$u_{VC} = \frac{3}{2} u_{Sc} + \frac{1}{2}(u_{VA} + u_{VB}) - \frac{1}{2} \sum_{x=A}^{C} u_{backEMFx} - \frac{1}{2} \sum_{y=A}^{C} \sum_{x=A}^{C} L_{Syx} \frac{d}{dt} i_{Sx}$$

The Equation 8 can be written for any of the three motor branch and phase voltages.

2.4.2.2 Switching Phases of Bipolar and Unipolar PWM

Let us assume a usual situation, where the BLDC motor is driven in 6-step commutation mode using a PWM technique.

The standard bipolar PWM switching technique (see Figure 2-15 and section 2.2.2.1) has two phases:

- 1(4) Top and Bottom switches in diagonal on
- 2(3) Top and Bottom switches in (inverse) diagonal on

The inverse phase 2(3) is same as 1(4) but the top and bottom switches are swapped on both phases.

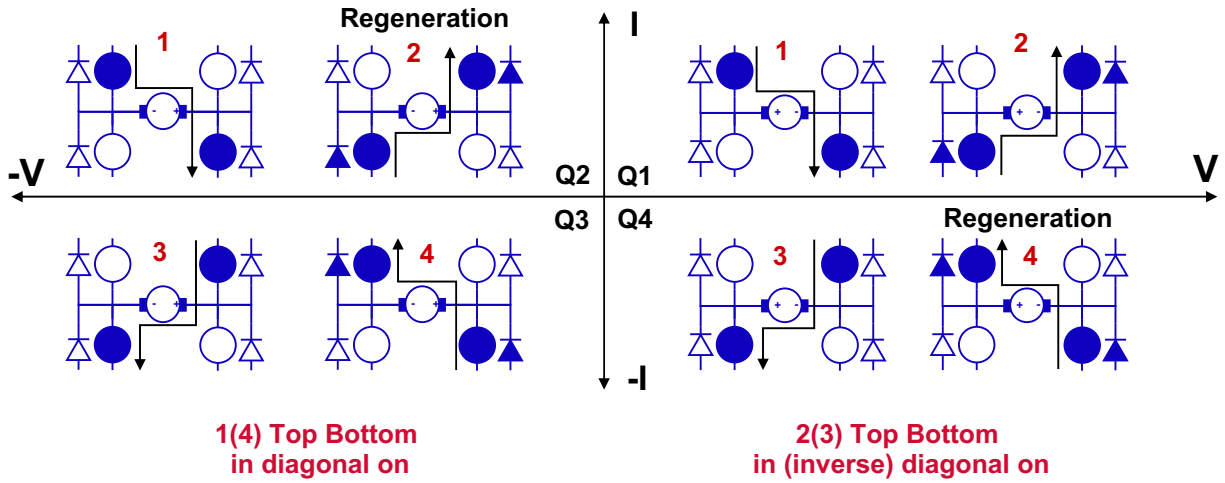

Figure 2-15. Bipolar (Hard) PWM Switching Phases

The unipolar PWM switching technique (see Figure 2-16 and section 2.2.2.2) has other two switching phases;

- 1(3) Top and Bottom switches in diagonal on
- 2(4) Two Bottom switches on — low voltage drop

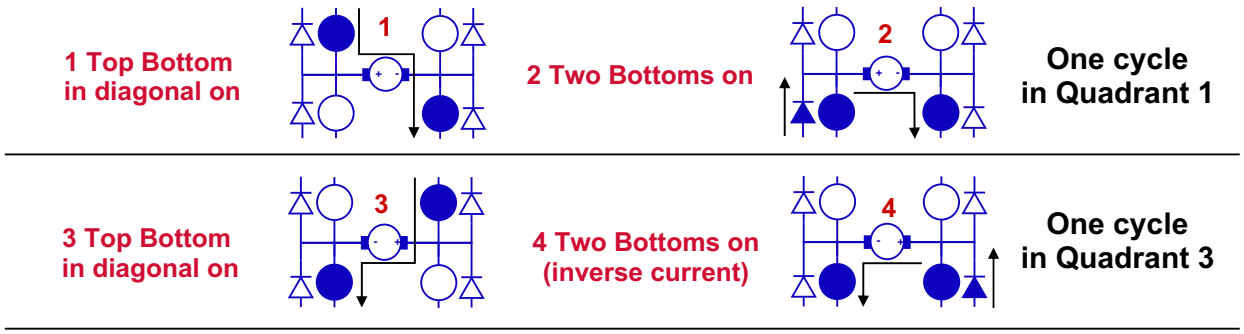

Figure 2-16. Unipolar (Soft) PWM Switching Phases

2.4.2.3 Indirect Back-EMF Sensing — Top and Bottom Switches in Diagonal On

Both common PWM switching techniques (bipolar and unipolar switching) have a PWM switching phase where the top and bottom switches in a diagonal are switched on. In this section, the situation where the Top and Bottom switches are in diagonal on will be analyzed.

For demonstration of the back-EMF sensing, we show the commutation state where the motor phases A and B are powered, and phase C is free, having no current. So, the phase C can be used to sense the back-EMF voltage. This is described by the following conditions:

$$S_{Ab}, S_{Bt} \leftarrow On \qquad U_{QaBot} = U_{QbTop}$$

$$u_{VA} = -\frac{1}{2}Ud + Uqa \qquad u_{VB} = +\frac{1}{2}Ud - Uqb$$

$$i_{Sb} = -i_{Sa} = i \qquad di_{Sb} = -di_{Sa} = di$$

$$i_{Sc} = 0 \qquad di_{Sc} = 0$$

$$u_{Sc} = u_{backEMFc} + (L_{Scb} - L_{Sca})\frac{d}{dt}i$$

Eqn. 9

The U_{QaBot} and U_{QbTop} are the voltage drops over the switching devices. Currents flowing through the top switch S_{Bt} and the bottom switch S_{Ab} are the same, and both go through transistors (or both go through the reverse diodes) we can assume the difference in the voltages across the top, and bottom switches can be ignored. The back-EMF voltages $u_{backEMFa}$ and $u_{backEMFb}$ of an ideally trapezoidal BLDC motor should be as according to Equation 10;

$$u_{backEMFb} = -u_{backEMFa}$$

Eqn. 10

The branch voltage u_{VC} can be calculated using the above conditions in Equation 8:

$$u_{VC} = u_{backEMFc} + (L_{Scb} - L_{Sca})\frac{di}{dt}$$

Eqn. 11

In reality, some of the BLDC motors have the back-EMF voltage close to sinusoidal. The back-EMF voltage sum of an ideal 3-phase symetrical sinusoidal motor is:

$$\sum_{x=A}^{C} u_{backEMFx} = 0$$

Eqn. 12

and the branch voltage u_{VC} will then be:

$$u_{VC} = \frac{3}{2} u_{backEMFc} + (L_{cb} - L_{ca}) \frac{di}{dt} \qquad \textit{Eqn. 13}$$

In a real BLDC motor, the relationship between the branch u_{VC} and BEMF voltages is somewhat between the equation Equation 11 and Equation 11. For the back-EMF zero-crossing detection the scale does not matter.

2.4.2.3.1 Indirect Back-EMF Sensing — Top and Bottom Switches in Diagonal On Conclusion

- With top and bottom switches in diagonal on, the back-EMF zero-crossing of the off phase can be sensed from the branch voltage u_{VX}, which is the off phase terminal with the half DC-Bus voltage ($u_d/2$) reference, shown in Figure 2-14.
- Unbalanced mutual inductance brings error to this indirect back-EMF sensing method.

Fortunately, close to the back-EMF zero-crossing of any phase, the L_{ca} and L_{cb} are balanced $L_{ca} = L_{cb}$, so we get the Equation 14:

$$u_{VC} \cong u_{backEMFc} \qquad \textit{Eqn. 14}$$

2.4.2.4 Indirect Back-EMF Sensing — Two Bottom Switches On (Low Drop)

The unipolar PWM switching technique has the second PWM switching phase (besides the previous one) where the two bottom switches are switched on (see Figure 2-16 and section 2.2.2.2). In this section, the situation where the two bottom switches are on will be analyzed.

For a demonstration of the back-EMF sensing, we show the commutation state where the motor phases A and B are powered, and phase C is free, having no current. So, the phase C can be used to sense the back-EMF voltage. This is described by the following conditions:

Now we have a situation where only the two bottom switches are turned on. Such a case is described by Figure 2-17.

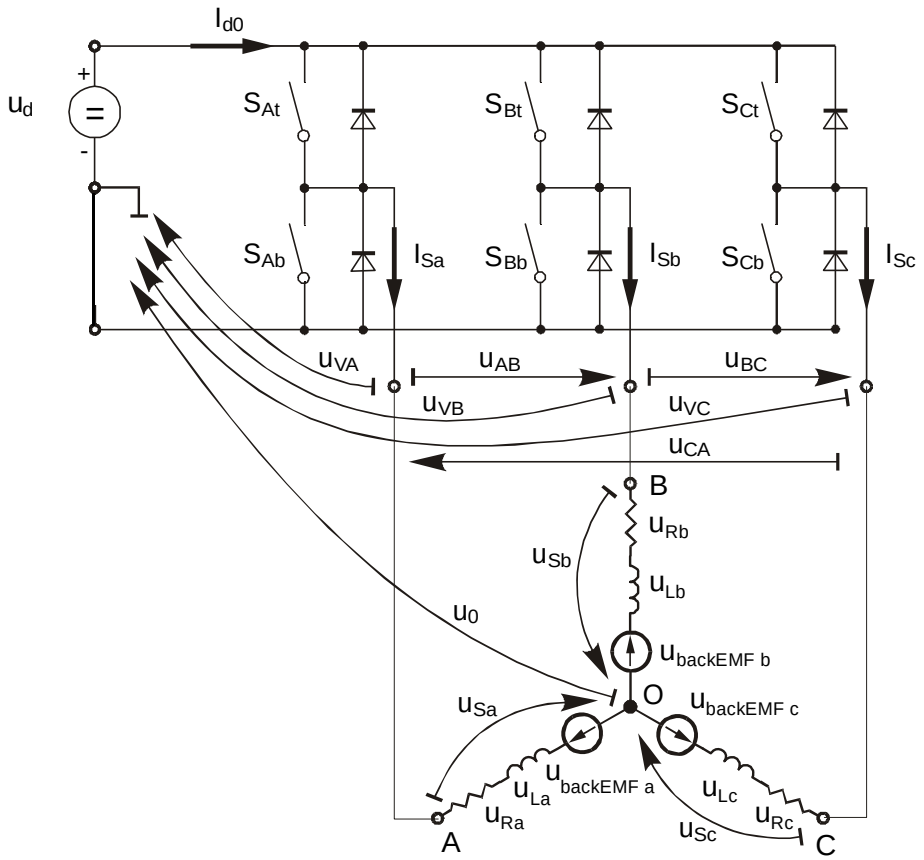

Figure 2-17. Power Stage — Motor Topology with GND Reference

The motor phases A and B are powered, and phase C is free, having no current. So, phase C can be used to sense the back-EMF voltage. This is described by the following conditions:

$$S_{At}, S_{Bt} \leftarrow On \qquad U_{QaBot} \neq U_{QbBot}$$

$$u_{VA} = 0 + U_{QaBot} \qquad u_{VB} = 0 - U_{QbBot}$$

$$i_{Sb} = -i_{Sa} = i \qquad di_{Sb} = -di_{Sa} = di$$

$$i_{Sc} = 0 \qquad di_{Sc} = 0$$

$$u_{Sc} = u_{backEMFc} \qquad u_{backEMFb} = -u_{backEMFa}$$

Eqn. 15

In such a case, one of the switches is conducting via a transistor (switch A) while the second (switch B) is conducting via the reverse diode. Therefore, the voltage drops over the switching devices U_{QaBot} and U_{QbBot} are not equal.

The branch voltage u_{VC} can be calculated using the above conditions in Equation 8:

$$u_{VC} = u_{backEMFc} + \frac{1}{2}(u_{QaBot} - u_{QbBot}) + (L_{cb} - L_{ca})\frac{di}{dt} \qquad \textit{Eqn. 16}$$

2.4.2.4.1 Indirect Back-EMF Sensing — Two Bottom Switches On Conclusion

However, the branch voltage seems to be similar to Equation 13, there are significant differences.

- The reference point for the back-EMF zero-crossing is the DC-Bus rail! At the two bottom switches on the PWM phase, the back-EMF zero-crossing of the off phase can be sensed from the branch voltage u_{VX}, which is the off phase terminal with the DC-Bus rail reference Figure 2-14.
- The difference in the bottom switches voltage drop, usually diode versus transistor, brings the reference voltage error.

The reference voltage error might be significant, mainly at low-voltage applications. This error can be rectified with reversing MOSFET control gate control (this is used for 4-quadrant power stage operations) and with some offsetting of the sensed voltage.

Unbalanced mutual inductance error is usually not a issue, as explained in Section 2.4.2.3.1, “Indirect Back-EMF Sensing — Top and Bottom Switches in Diagonal On Conclusion.” So we get the Equation 17:

$$u_{VC} = u_{backEMFc} + \frac{1}{2}(u_{QaBot} - u_{QbBot}) \qquad \textit{Eqn. 17}$$

2.4.3 Back-EMF Zero-Crossing Synchronization with PWM

The back-EMF zero-crossing needs to be synchronized with PWM in order to compare to a correct reference voltage, as explained in Section 2.4.2, “Power Stage — Motor System Model. Another reason for the synchronization is, that the zero-crossing detection window needs to be even slightly shorter than the period of the Top and Bottom Switches in Diagonal On PWM stage. This is in order to stop the zero-crossing sampling during dead time and during PWM transient spikes. Those spikes appear on the measured phase due to power stage switches transitions, and mainly due to mutual capacitance between the motor phases (see Section 1.3.1, “Bibliography” 5 for details). Figure 2-18 shows the Zero-Crossing Sampling window for position sensing at the Top and Bottom switches in diagonal on.

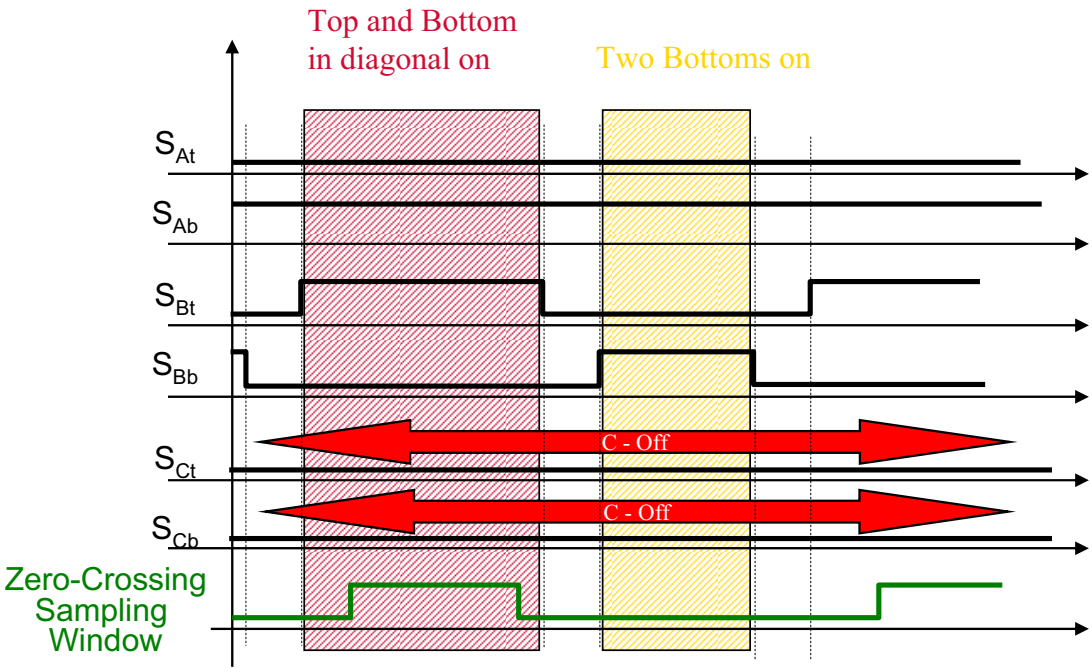

Figure 2-18. Zero-Crossing Detection Measurement Window Unipolar Switching

2.4.4 Back-EMF Zero-Crossing Sensing Circuit

In this Sensorless BLDC Control application, the zero-crossing is detected using a high-speed comparator. According to the previous sections, the comparator topology:

- Phase analog multiplexer
  - This multiplexes the sampled motor phase according to the PWM sector.
- Reference signal
  - Figure 2-19 shows the reference signal for top bottom in diagonal on.
- Comparator
- Sampling Window Generator
  - Generates the measurement window synchronized with the power stage PWM according to Figure 2-18.

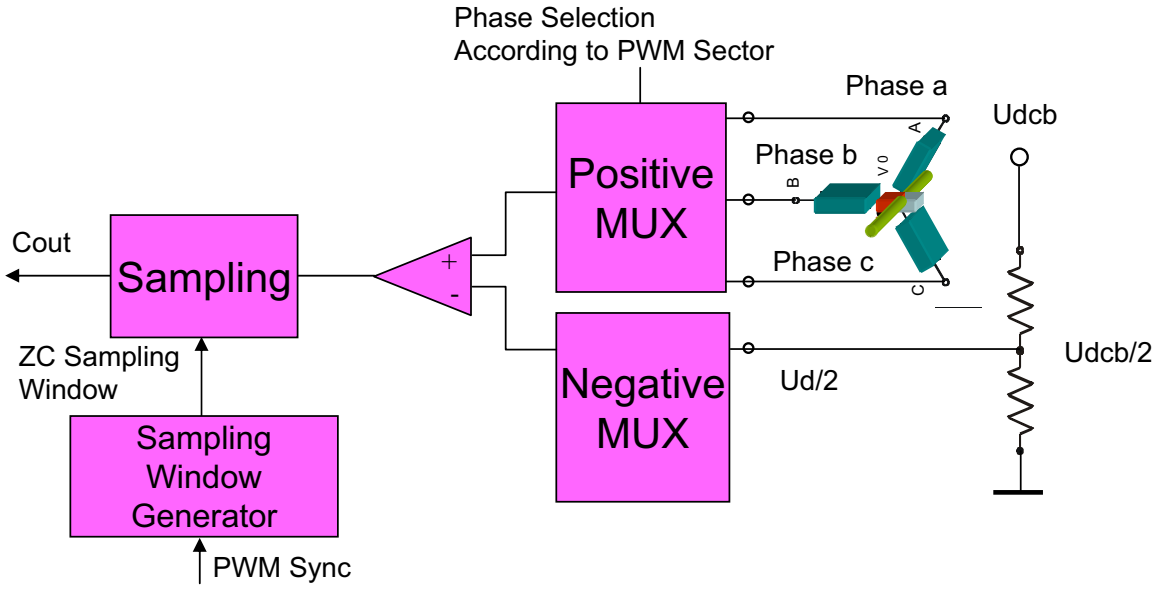

Figure 2-19. Back-EMF Zero-Crossing Sensing Circuit

2.4.5 The Zero-Crossing Sensing States

During the 6-step commutation there is a current recirculation transient (the fly-back diodes are conducting the decaying phase current) before the phase current is 0. During this transient, the back-EMF zero-crossing cannot be measured, because the phase is not off. See Figure 2-20.

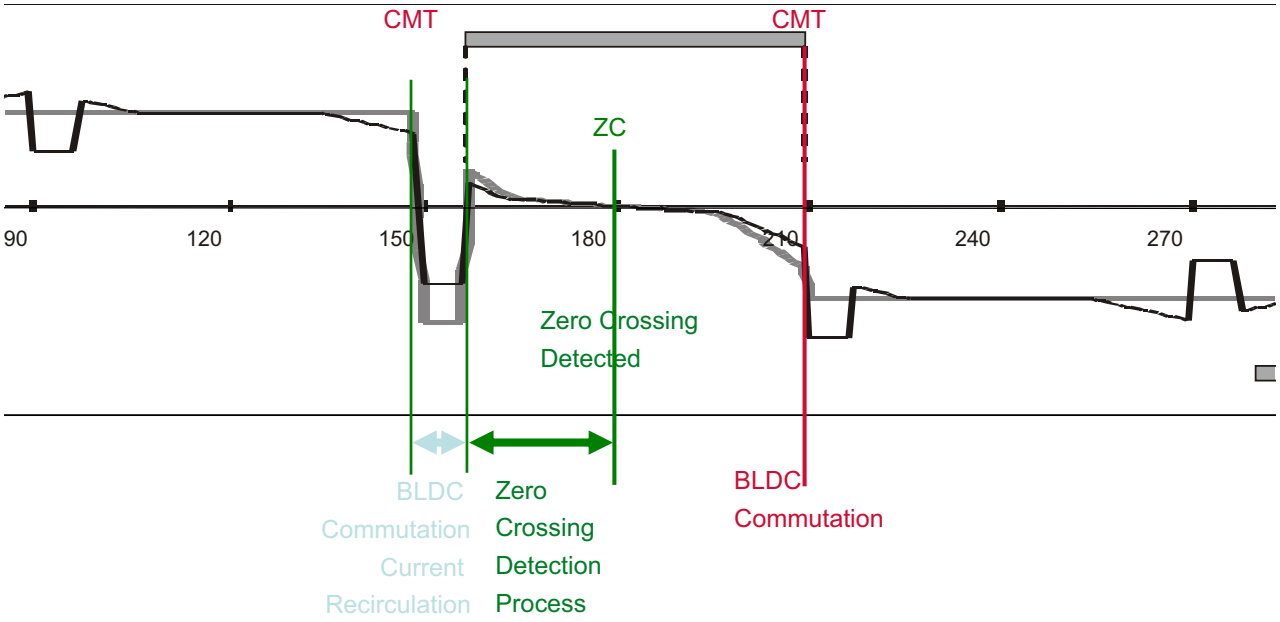

Figure 2-20. Zero-Crossing Sensing States

The zero-crossing detector also needs to cover the situation which might appear when the zero-crossing sensing is a state where the 6-step commutation is executed before the back-EMF zero-crossing is detected. This is also described in Section 2.5.3, "Sensorless Run — Direct Commutation Calculation."

The zero-crossing detection states are shown in Figure 2-21.

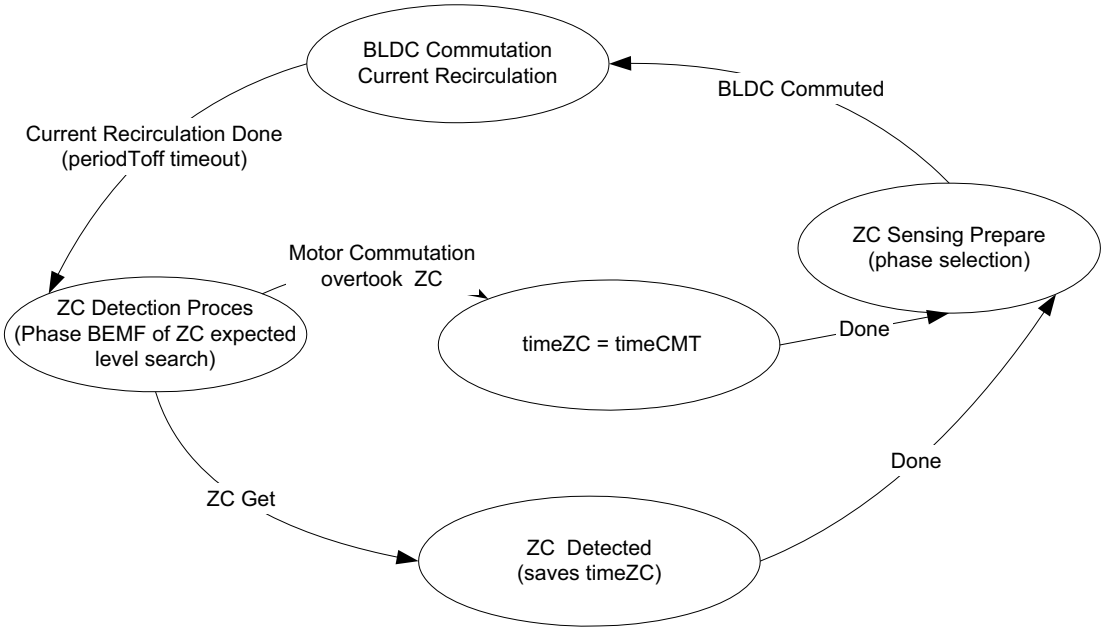

Figure 2-21. Zero-Crossing Sensing States

2.5 Sensorless Commutation Control

This section presents the sensorless BLDC motor commutation with the Back-EMF Zero-Crossing technique.

In order to start and run the BLDC motor, the control algorithm has to go through the following states:

- Alignment
- Open-Loop Start
- Sensorless Run — Direct Commutation Calculation
- Sensorless Run — Synchronized Phase-Locked-Loop (PLL)
- Sensorless Run — Forced Phase-Locked-Loop (PLL) — Forced Cmt

Figure 2-22 shows the transitions between the states. Firstly the rotor is aligned to a known position; then the rotation is started without the positional feedback. When the rotor moves, back-EMF is acquired so the position is known, and can be used to calculate the speed and processing of the commutation in the running state.

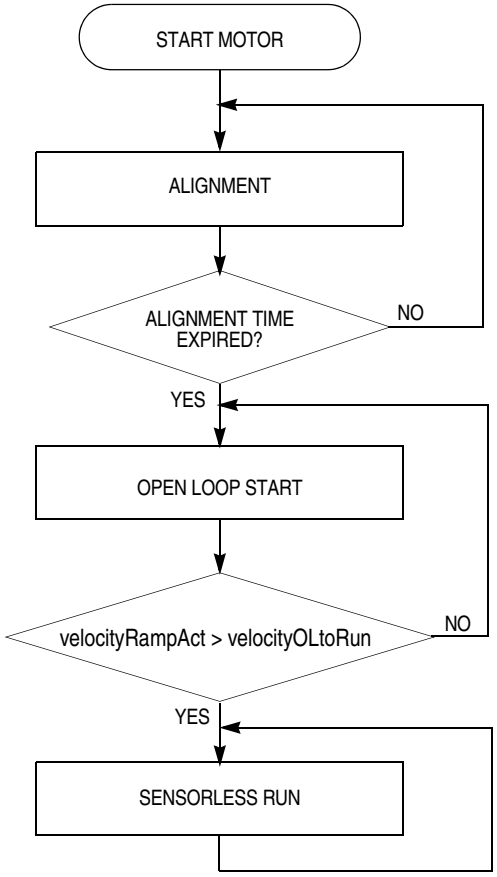

Figure 2-22. Commutation Control Stages

2.5.1 Alignment

Before the motor starts, there is a short time (depending on the motor's electrical time constant) when the rotor position is stabilized by applying PWM signals to only two motor phases (no commutation). The current controller keeps the current within predefined limits. This state is necessary in order to create a high startup torque. When the preset timeout expires, then this state is finished.

The BLDC motor rotor position with flux vectors during alignment is shown in Figure 2-23.

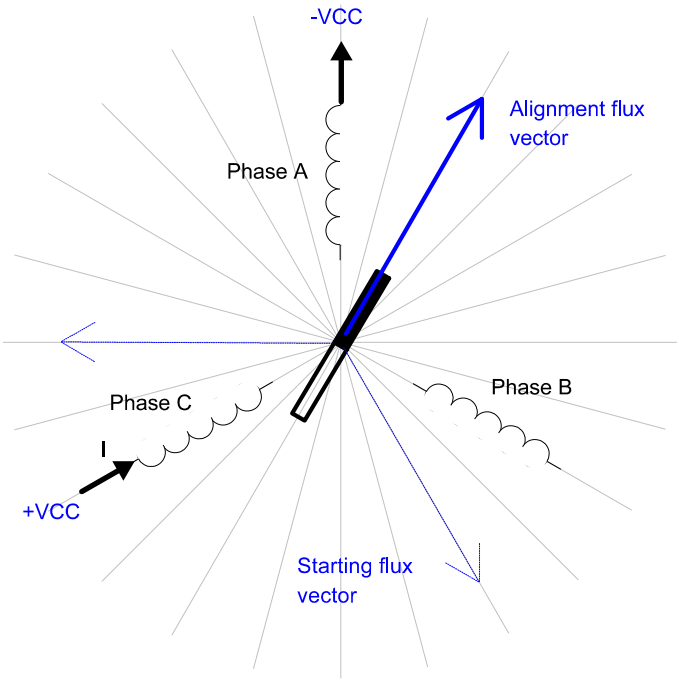

Figure 2-23. Alignment

2.5.2 Open-Loop Start

In the open-loop start, the motor commutation starts the open-loop without rotor positional feedback. The commutation period is controlled by a linear velocity ramp. The rotor and stator flux need to be in an approximately 90 degree relation in order not to lose synchronization. The torque is lower and not constant. Therefore, the position resonance can cause the rotor synchronization loss. The open-loop start needs to be a short state at a very low speed where the back-EMF is too small, so the zero-crossing cannot be reliably detected.

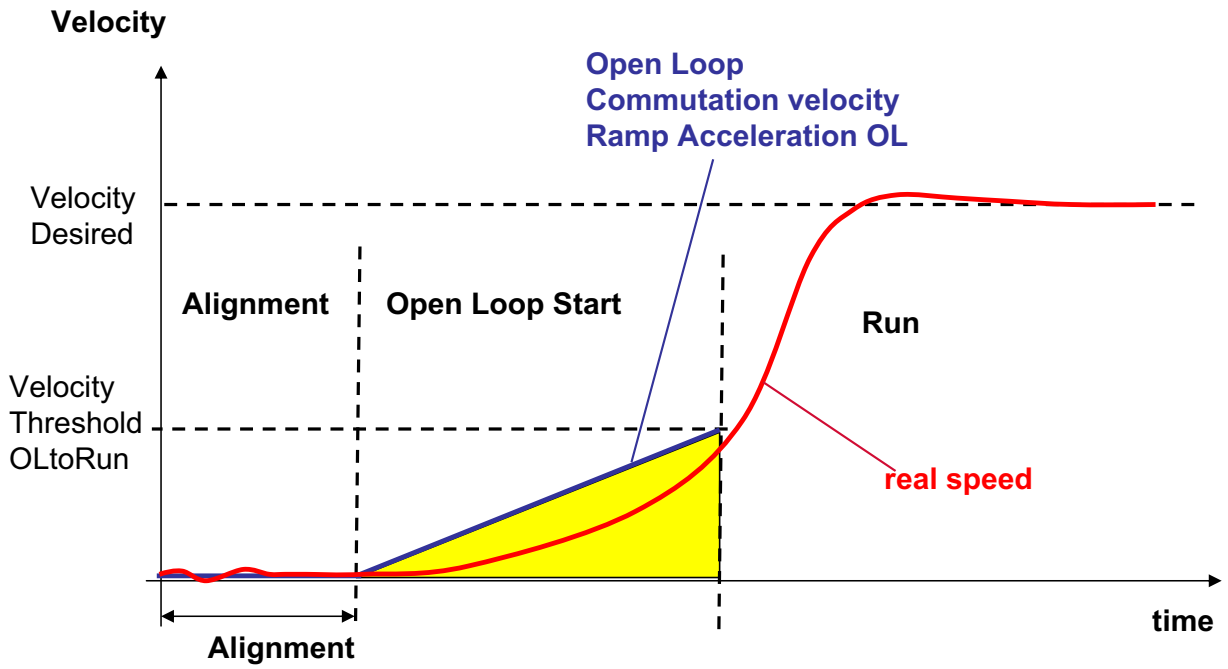

Figure 2-24. Open-Loop Start

2.5.3 Sensorless Run — Direct Commutation Calculation

The commutation process is a series of states which ensure:

- The back-EMF zero-crossing is successfully captured.
- The new commutation time is calculated.
- The commutation is performed.

The following processes need to be provided:

- BLDC motor commutation service.
- Back-EMF zero-crossing moment capture service.
- Calculation of commutation time.
- Interactions between these commutation processes.

From the diagrams, an overview of how the commutation works can be understood. After commuting the motor phases, a time interval periodZcToff[k] is set that allows the shape of the back-EMF to be stabilized (after the commutations the fly-back diodes are conducting the decaying phase current; therefore, sensing of the back-EMF is not possible). Then the new commutation time (timeBLDCCmt[k]) is preset according to periodCmtPreset. The new commutation will be performed at this time if the back-EMF zero-crossing is not captured. If the back-EMF zero-crossing is captured before the preset commutation time expires, then the exact calculation of the commutation time timeCmt[k] is made from this, based on the captured zero-crossing time (timeBLDCZc[k]) and perZcToCmt[k]. The new commutation is performed at this new time.

If for any reason the back-EMF feedback is lost within one commutation period, corrective actions are taken in order to return to the regular states.

The flowchart explaining the principle of BLDC commutation control with back-EMF zero-crossing sensing is shown in Figure 2-25.

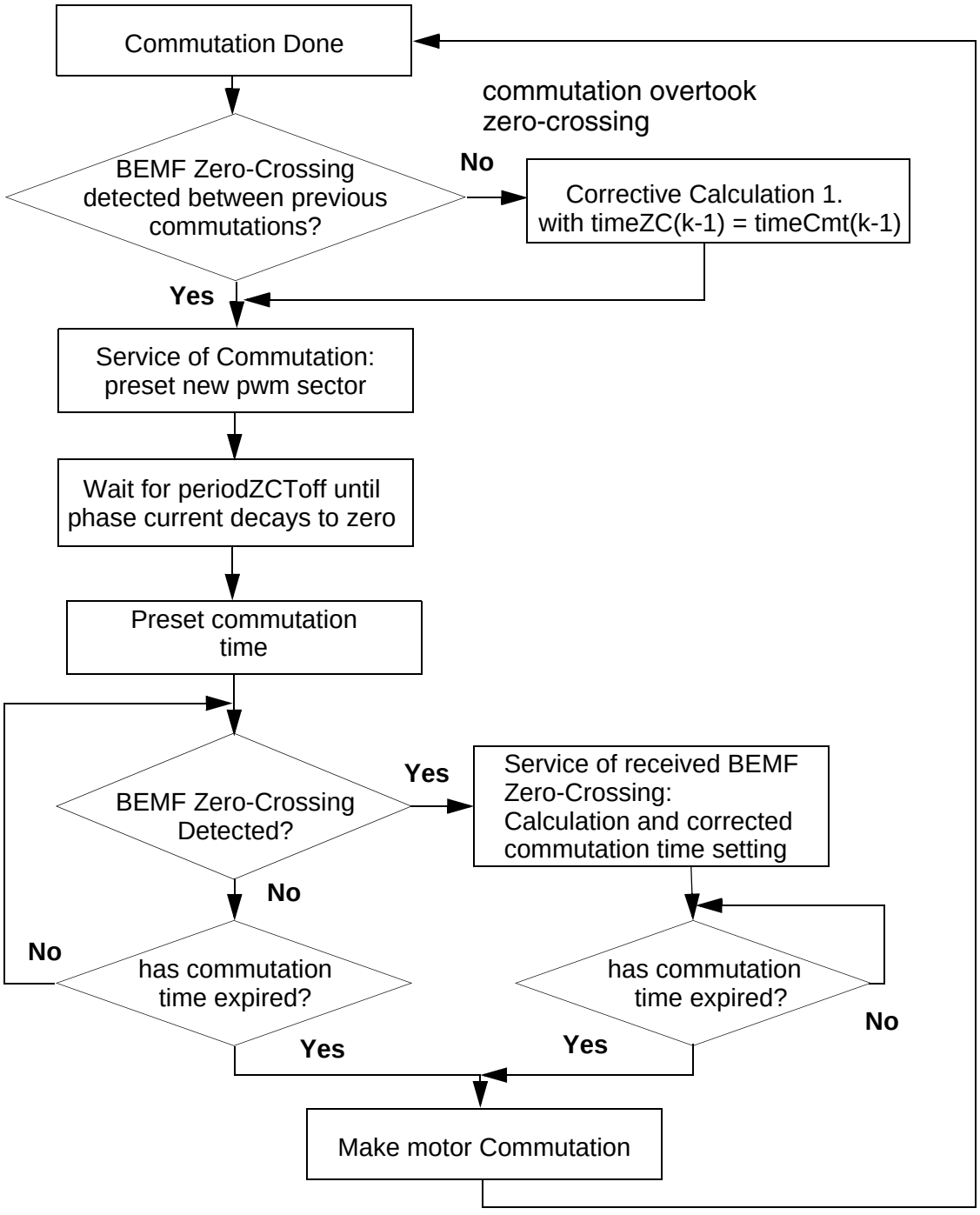

Figure 2-25. Flow Chart — BLDC Commutation with Direct Commutation Calculation

Commutation time calculation is shown in Figure 2-27.

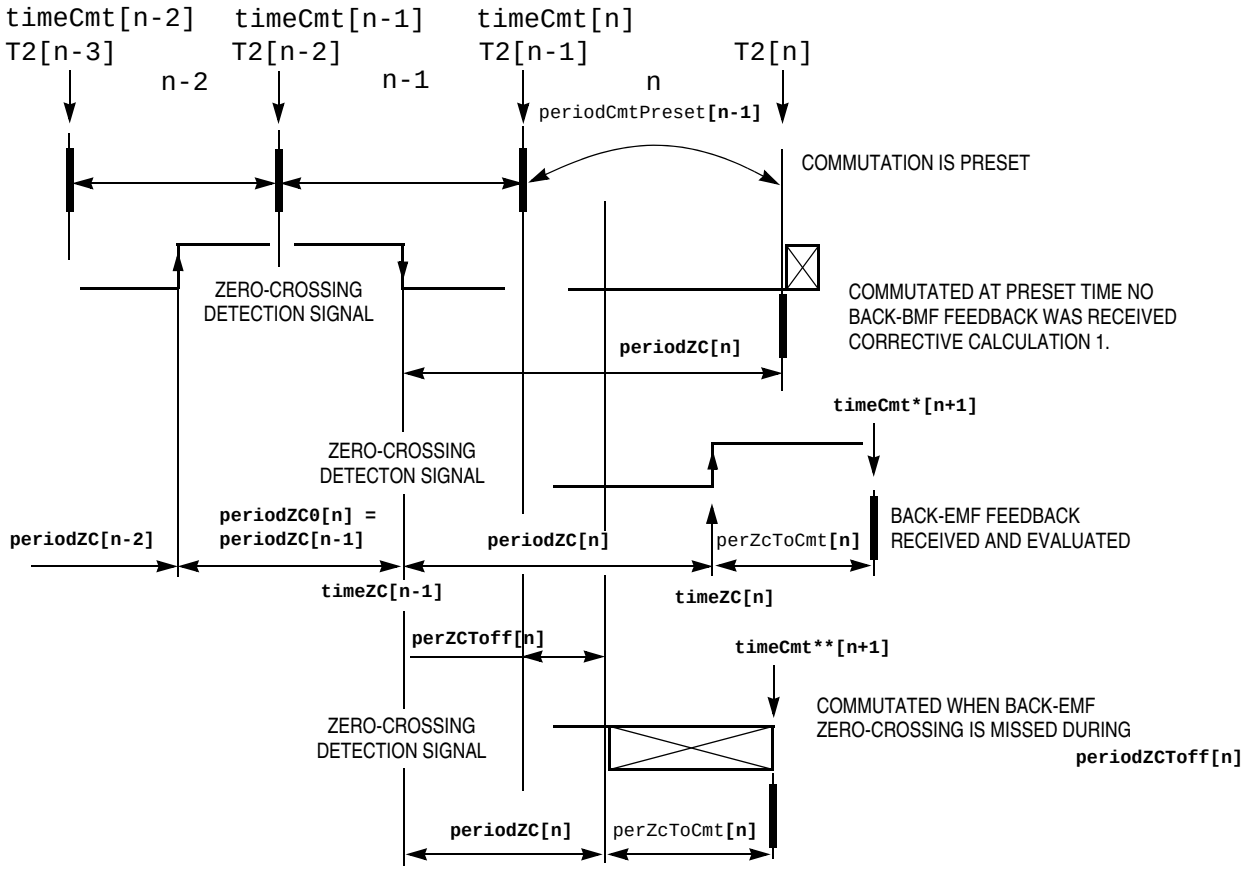

Figure 2-26. . BLDC Commutation Time with Direct Commutation Calculation

The periods from Figure 2-26 are calculated using the equations below:

$$\text{periodZcFlt} = \frac{\text{periodZc} + \text{periodZc0}}{2} \qquad \textit{Eqn. 18}$$

$$\text{perZcToCmt} = \text{periodZcFlt} \times \text{coefZcToCmt} \qquad \textit{Eqn. 19}$$

$$\text{periodCmtPreset} = \text{periodZcFlt} \times \text{coefCmtPreset} \qquad \textit{Eqn. 20}$$

Where, theoretically, the coefZcToCmt = 0.5, since the back-EMF zero-crossing is between the commutations — see Figure 2-12. In a real application the coefficient is slightly different. The coefCmtPreset is >1 and usually less or equal to 2.

Using the Sensorless Run — Direct Commutation Calculation technique, the motor commutation is synchronized with the rotor position. The motor velocity is then controlled using the zero-crossing period feedback provided to the speed regulator, as described in Section 2.6, “Speed Controller with Current (Torque) Limitation.” The speed regulator controls the 3-phase power stage PWM.

2.5.4 Sensorless Run — Synchronized Phase-Locked-Loop (PLL)

This mode differs from the Section 2.5.3, “Sensorless Run — Direct Commutation Calculation” using a closed feedback loop.

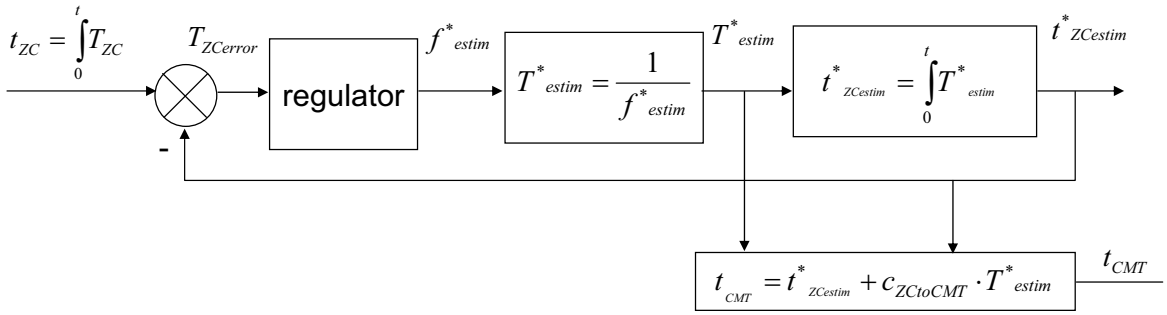

Figure 2-27. Synchronized Phase-Locked-Loop

The motor commutation period $T_{CMT} = T^*_{estim}$ is estimated using a closed loop. The closed-loop feedback is the period $T_{ZCerror}$ between the estimated and the measured back-EMF zero-crossings. A benefit of this feedback loop is that the back-EMF zero-crossing detection disturbance is filtered. Finally, the BLDC motor commutation is smoother and less noisy, especially at a constant speed. For some application ranges with high dynamics, accelerations and decelerations, the Section 2.5.3, “Sensorless Run — Direct Commutation Calculation” state might be more suitable.

Using the Sensorless Run — Synchronized Phase-Locked-Loop (PLL) technique, the motor commutation is synchronized with the rotor position. The motor velocity is then controlled using the estimated period feedback T^*_{estim} provided to the speed regulator, as described in Section 2.6, “Speed Controller with Current (Torque) Limitation.”

2.5.5 Sensorless Run — Forced Phase-Locked-Loop (PLL) — Forced Cmt

The Forced PLL is similar to Section 2.5.4, "Sensorless Run — Synchronized Phase-Locked-Loop (PLL)." But the commutation period T_{CMT} is constant, and the period $T_{ZCerror}$ between estimated and measured amplitude.

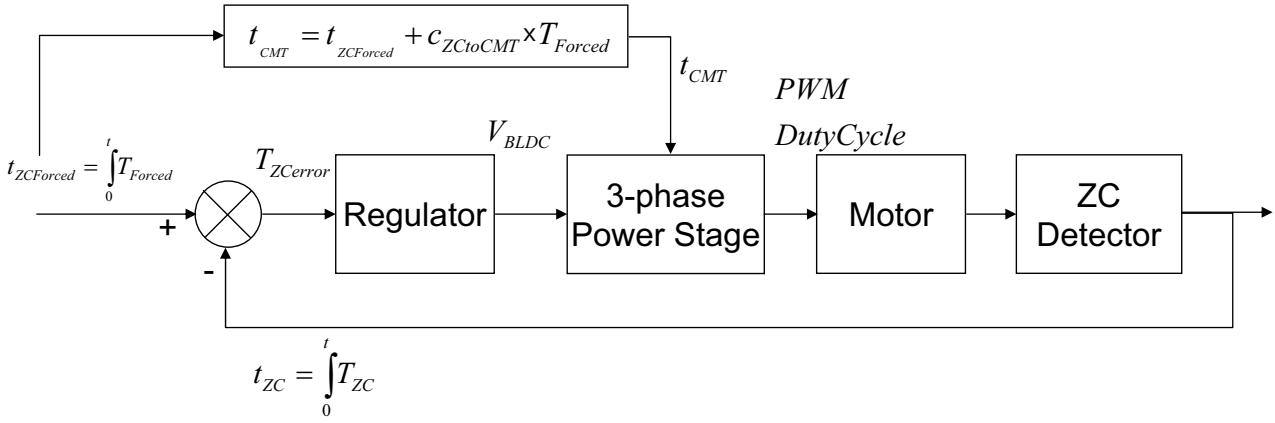

Figure 2-28. Forced Phase-Locked-Loop

Back-EMF zero-crossings is maintained by the voltage amplitude of the 3-phase system applied on the motor.

The benefit of this feedback loop is that the commutation period T_{CMT} is constant. This is required for some applications. The motor noise might also be reduced using this technique. This technique is only used when the motor phase current is below maximal limits. Therefore, for high load disturbance ripples, the state is leaved.

Using the Sensorless Run — Forced Phase-Locked-Loop (PLL) — Forced Cmt technique, the motor commutation determines the motor velocity.

2.6 Speed Controller with Current (Torque) Limitation

The motor velocity is then controlled using the zero-crossing period feedback provided to the speed regulator. The outer current regulator limits the motor current. This provides the torque limitation in order to limit the maximal motor current. The speed regulator controls the 3-phase power stage PWM. This regulator controls the velocity or torque in the Sensorless Run — Synchronized Phase-Locked-Loop (PLL) or Sensorless Run — Direct Commutation Calculation states.

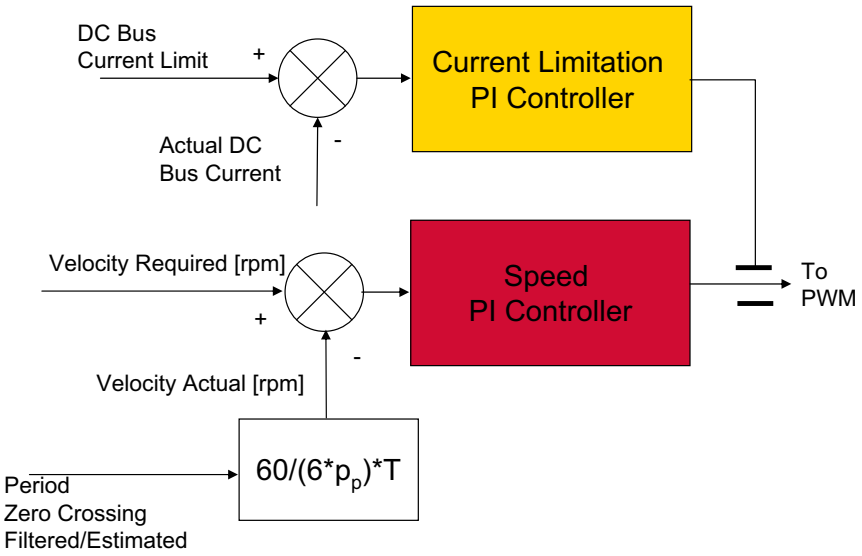

Figure 2-29. Speed Control with Torque Limitation

Chapter 3
System Concept

3.1 Application Description

As was already mentioned in previous sections, the 3-Phase Sensorless BLDC Motor Control Using MC9S08MP16 application is based on the MC9S08MP16 MCU. A standard system concept was chosen for the drive (see Figure 3-1). The system incorporates this hardware:

- 3-Phase BLDC/PMSM Low-Voltage Motor Control Drive
- MC9S08MP16 Controller Daughter Board for BLDC/PMSM Motor Control Drive
- 3-phase BLDC motor (default configuration for LINIx 45ZWN24-40 motor)

The MC9S08M16 populated on the controller daughter board executes the control algorithm. In response to the user interface and feedback signals, it generates PWM signals for the board called the 3-Phase BLDC/PMSM Low-Voltage Motor Control Drive. Voltage waveforms generated by the 3-phase inverter are applied to the motor. Input voltage is 24 V DC.

3.2 Control Process

The speed control is calculated according to the state of the control signals (Start/Stop, Desired Speed from FreeMASTER). Then the speed command is processed by means of the speed ramp algorithm.

Depending on the enabled control algorithm, utilizing the commutation period is set according to the velocity ramp. This is for the open loop starting state or the Forced PLL control technique. The comparison between the actual ramp velocity obtained from the ramp algorithm output and the measured speed obtained from 1/estimated period generates a speed error. The speed error is processed by the speed PI controller that generates a new corrected motor voltage amplitude.

The PWM 3pps generation process provides a 6-step BLDC commutation of a 3-phase voltage system of required amplitude (PWM), includes dead time, and finally the 3-phase PWM motor control signals are generated. The DC-Bus voltage and the DC-Bus current (optional informative value) are measured during the control process. They are used for over-voltage, under-voltage protection (DC-Bus voltage), and torque limitation (DC-Bus current) of the drive. The voltage protection is performed by software.
The over-current fault signal utilizes a fault input of the microcontroller. If any of the above mentioned faults occur, the motor control PWM outputs are disabled in order to protect the drive and the fault state of the system is displayed.

The rotor positional estimation with back-EMF zero-crossig detection is based on the HSCMP comparator.

The application can be controlled via the FreeMASTER control page from a host PC. The FreeMASTER communicates via a serial interface which is converted into USB on the MC9S08MP16 daughter board.

The application state machine of the drive manages the operating states of the drive (FAULT, INIT, INIT, and RUN with sub-states). The actual operating state is indicated by the FreeMASTER control page. In the case of over-voltage, under-voltage, or over-current, the signals for the 3-phase inverter are disabled and the fault state is displayed.

3.3 3-Phase Sensorless BLDC Drive Using MC9S08MP16

Figure 3-1shows the system functional blocks and their sub-blocks. Most of the sub-blocks are implemented with the MC9S08MP16 modules.

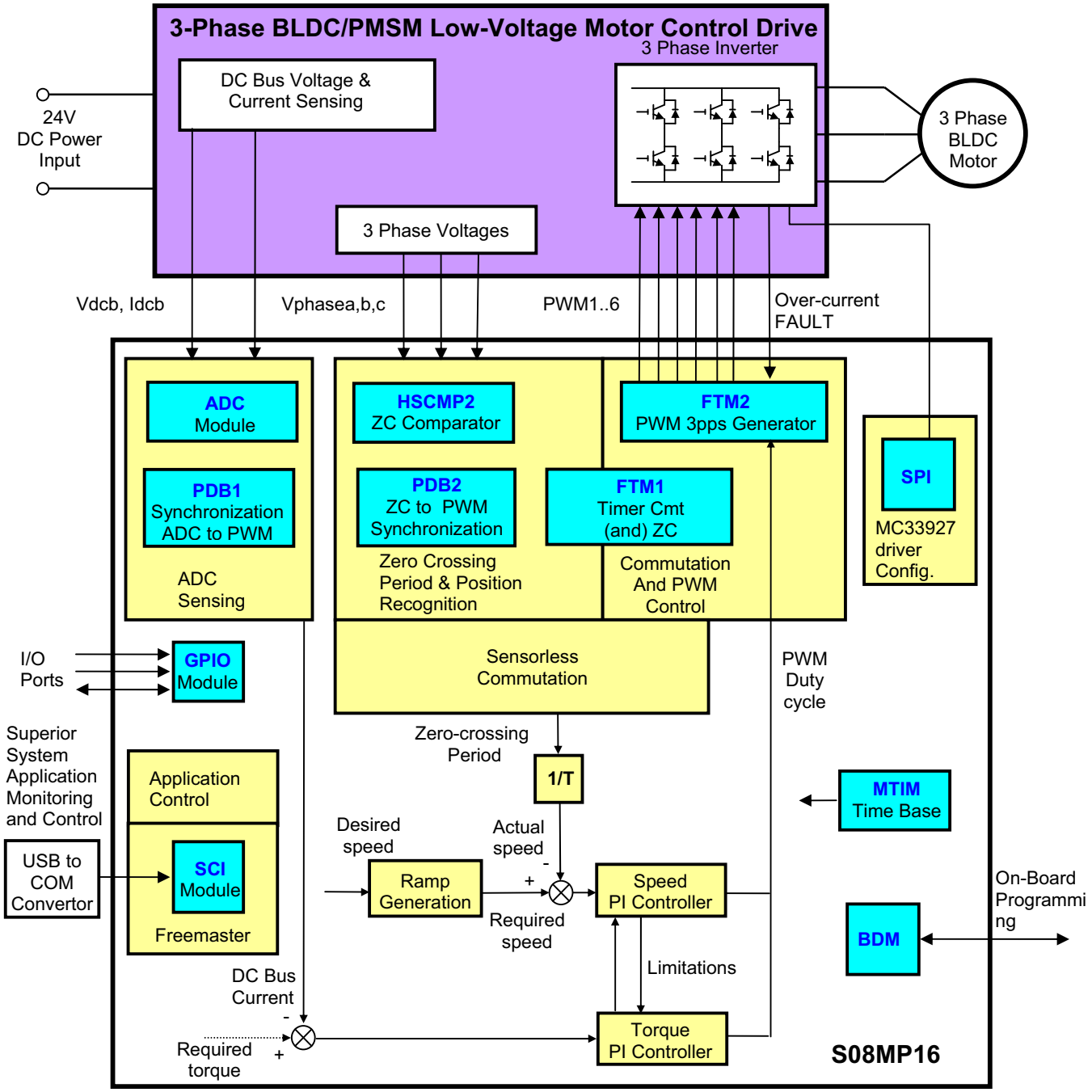

Figure 3-1. 3-Phase Sensorless BLDC Drive Using MC9S08MP16 — System Concept

3.3.1 S08MP16 Modules implementation and H/W Synchronization

One of the most valuable features of the S08MP16 device is the hardware synchronization between internal modules. Figure 3-2 shows the functional blocks and synchronization signals used for the sensorless BLDC motor control. This figure also shows how the functional blocks are implemented using the S08MP16 internal modules and synchronization signals.

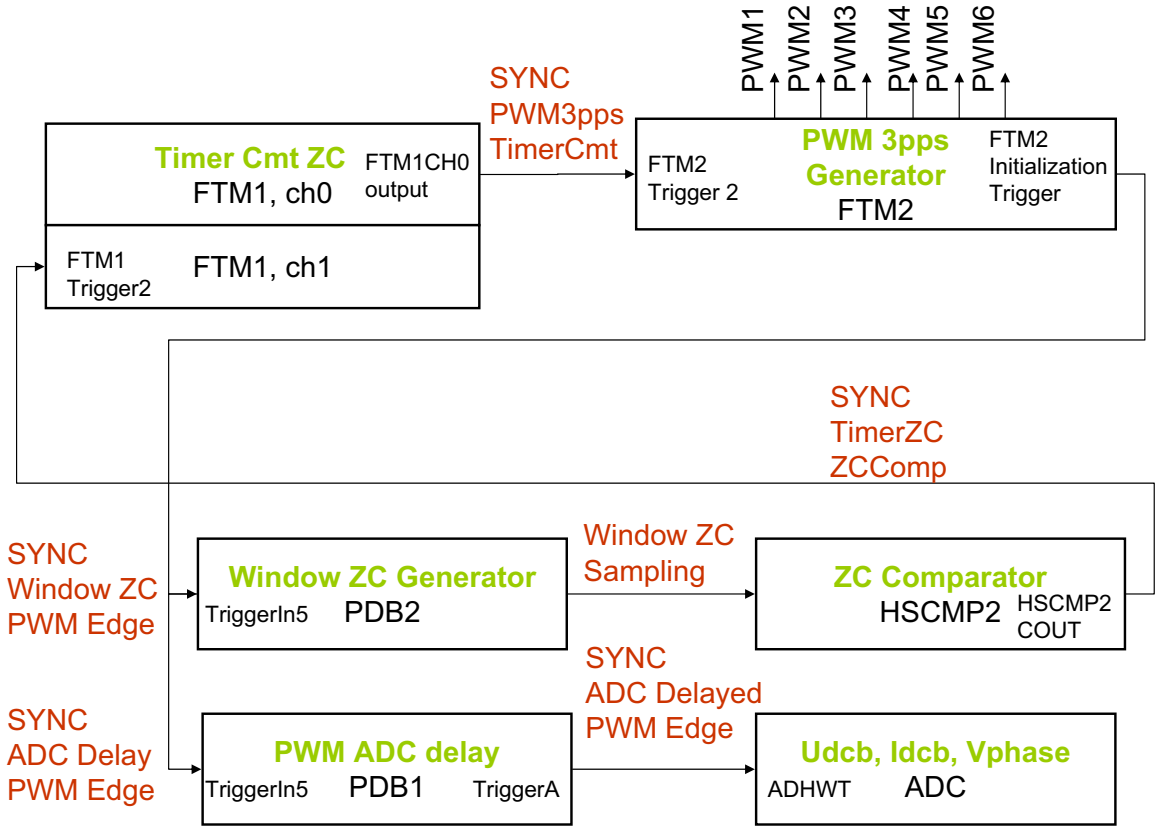

Figure 3-2. S08MP16 Modules and Synchronization

A description of the system functional blocks, sub-blocks, and their realization is in the following sections.

3.3.2 Sensorless Commutation

This functional block provides sensorless BLDC commutation with the functionality described in Section 2.4, "Sensorless Technique Based on Back-EMF Zero-Crossing." This functional block is provided by software and the blocks: Commutation and PWM Control, Zero-Crossing Period and Position Recognition.

3.3.3 Commutation and PWM Control

This functional block provides the BLDC commutation and PWM duty cycle control. It consists of:

- PWM 3pps Generator

- Timer Cmt Zc

3.3.3.1 PWM 3pps Generator

This is the PWM generator for the 3-phase BLDC control. It provides the 6-step commutation with unipolar or bipolar (as defined in precompiler) PWM switching, as described in Section 2.2.2.1, "3-Phase BLDC 6-Step Control with Bipolar (Hard) PWM Switching" and Section 2.2.2.2, "3-Phase BLDC 6-Step Control with Unipolar (Soft) PWM Switching."

This PWM is realized by all six channels of the Flextimer FTM2 module. The FTM2 is used in FTM complementary combined mode. The 6-step commutation is provided automatically using the hardware synchronization signal SYNC PWM 3pps Timer Cmt.

The module has hardware fault protection. The fault is generated by the over-current signal from the power stage driver.

3.3.3.1.1 SYNC PWM 3pps Timer Cmt

Synchronizes the PWM 3-phase BLDC module with the commutation period timer. The next commutation vector is executed just when the synchronization signal is set.

The synchronization is realized by the FTM1CH0 output (must be previously cleared with bit FTM1OUTINIT.CH0OI = 0 and FTM1MODE.INIT = 1) connected as FTM2 Trigger2 (bit SOPT2.FTM2T2S = 1).

3.3.3.2 Timer Cmt Zc

This is a timer which serves the two functional blocks, Commutation and PWM Control and Zero-Crossing Period and Position Recognition. This timer is realized by the Flextimer FTM1 module in TP mode.

Channel0 is used for the Commutation and PWM Control functional block as an output compare for the commutation timing (see timeCmt in Figure 2-26). The FTM1CH0 output is used for SYNC PWM 3pps Timer Cmt for hardware commutation.

The FTM1 channel1 is used for Zero-Crossing Period and Position Recognition as input capture for the zero-crossing time instant (see timeZC in Figure 2-26). The hardware trigger FTM1 Trigger2 triggers the input capture with the SYNC Timer Zc ZC Comp.

3.3.4 Zero-Crossing Period and Position Recognition

This functional block provides the back-EMF zero-crossing detection with period and position recognition.

- ZC Comparator
- Timer Cmt Zc

3.3.4.1 ZC Comparator

Back-EMF zero-crossing comparator. Provides zero-crossing detection as described in Section 2.4.3, "Back-EMF Zero-Crossing Synchronization with PWM" and Section 2.4.4, "Back-EMF Zero-Crossing Sensing Circuit."

The ZC Comparator is realized by the HSCMP2 module. The motor phase terminal input is selected by the register HSCMP2CR0 according to the PWM Sector. The Zero-Crossing sampling window is realized by the Window ZC Sampling generated by the PDB2 module.

3.3.4.1.1 SYNC Timer Zc ZC Comp

The zero-crossing comparator output COUT triggers the Timer Zc input capture. The synchronization is realized by the HSCMP2 COUT output which is connected to the FTM1 Trigger 2.

3.3.4.1.2 Window ZC Sampling

The signal Window ZC Sampling synchronizes the ZC comparator and is generated by the PDB2 module.

3.3.4.1.3 SYNC Window ZC PWM Edge

This signal is used to trigger the PDB2 module to generate Window ZC Sampling. The signal is generated as an FTM2 Initialization trigger.

3.3.5 ADC Sensing

3.3.5.1 Udcb, Idcb, Vphase ADC

An analog-to-digital convertor for the analog variables conversion. This is realized by an ADC internal module. The conversion is synchronized with the PWM edge SYNC ADC Delayed PWM Edge.

3.3.5.1.1 SYNC ADC Delay PWM Edge

This signal is used to trigger the PDB1 module to an SYNC ADC Delayed PWM Edge. The signal is generated as an FTM2 Initialization trigger.

3.3.5.1.2 SYNC ADC Delayed PWM Edge

The signal SYNC ADC Delayed PWM Edge synchronizes the ADC start and is generated by the PDB2 module.

3.3.6 BLDC State Control, Alignment, and Speed/Torque Close Loop

The BLDC state control process provides application state control as described in Section 2.5, "Sensorless Commutation Control" and Chapter 5, "Software Design. Section 2.6, "Speed Controller with Current (Torque) Limitation."

3.3.6.1 MTIM Time Base (TimB)

A time base of a long period is used for speed control, alignment timing, and other functionality in the BLDC control process state machine with long period (default setting 3 ms). It is realized by the MTIM module.

3.3.7 MC33927 Driver Config

The 3-Phase BLDC/PMSM Low-Voltage Motor Control Drive board utilizes a MOSFET MC33927 predriver. This device is configured via the SPI protocol. The S08MP16 SPI module provides the communication.

3.3.8 FreeMASTER

This is a tool for communication with a superior system for application monitoring and control. The superior system is a personal computer running the FreeMASTER protocol.

The physical layer of the FreeMASTER can be a BDM module or a serial port. In this application, we use the serial communication using the SCI module. Today, personal computers usually do not have the COM port. Therefore, the MC9S08MP16 Controller Daughter Board for BLDC/PMSM Motor Control Drive is equipped with a USB-to-COM converter.

3.3.9 I/O Ports

Are realized by the GPIO module.

3.3.10 On-Board Programming

This is realized by the BDM module. The BDM module can be accessed using a connector on the daughter board. This can be connected to a personal computer using one of two possible devices:

- P&E USB Multilink device, or
- HCS08 Open Source BDM device.

Both devices use a USB connector for connection to a PC.

Chapter 4
Hardware

4.1 Hardware Implementation

The BLDC sensorless application runs on Freescale's 3-phase BLDC/PMSM Low Voltage Motor Control Drive board with an MC9S08MP16 controller daughter board and a LINIX 45ZWN24-40 BLDC motor. The application hardware system configuration is shown in Figure 4-1.

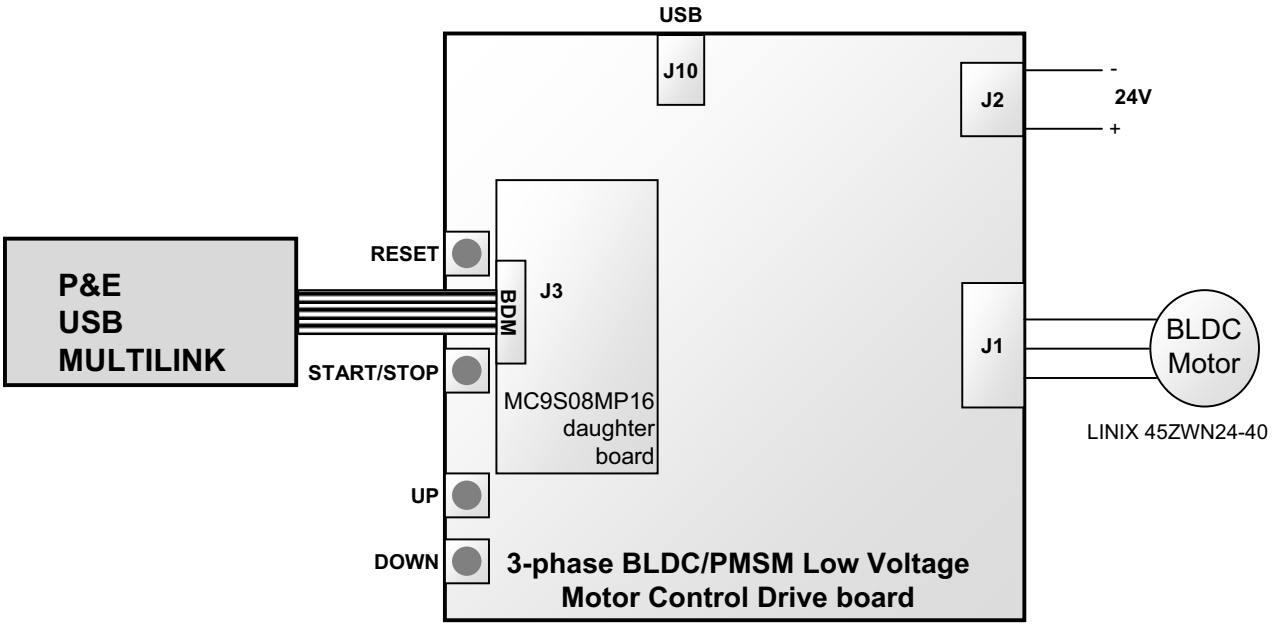

Figure 4-1. Hardware System Configuration

All system parts are supplied and documented:

- MC9S08MP16 Controller Daughter Board for BLDC/PMSM Motor Control Drive:
  - — Uses Freescale's MC9S08MP16 as the controller.
  - — Described in Section 1.3.1, "Bibliography."
- 3-Phase BLDC/PMSM Low-Voltage Motor Control Drive:
  - — Low-voltage, 3-phase power board with DC input 12–50 V AC and 4,000 VA variable voltage 3-phase MOSFET bridge output.
  - — Described in Section 1.3.1, "Bibliography."

A detailed description of each individual board can be found in the appropriate user manual, or on the Freescale web site (www.freescale.com). The user manuals include a schematic of the board, a description of individual function blocks, and a bill of materials (parts list).

4.2 Component Descriptions

The MC9S08MP16 sensorless BLDC demo is based on a 3-phase BLDC/PMSM Low-Voltage Motor Control Drive board module which provides a broad solution for testing and developing low-power drives and demos with a list of microcontrollers and DSC daughter boards (see Figure 4-2). It demonstrates the abilities of the MC9S08MP16 and provides a hardware tool to help in the development of applications using the MC9S08MP16 targeted at motor control applications. A detailed description, including the hardware specification of the 3-phase BLDC/PMSM Low-Voltage Motor Control Drive board, is located in the user's manual under the number LVMCDBLDCPMSMUG. The user's guide contains the schematic of the board, description of individual function blocks and a bill of materials.

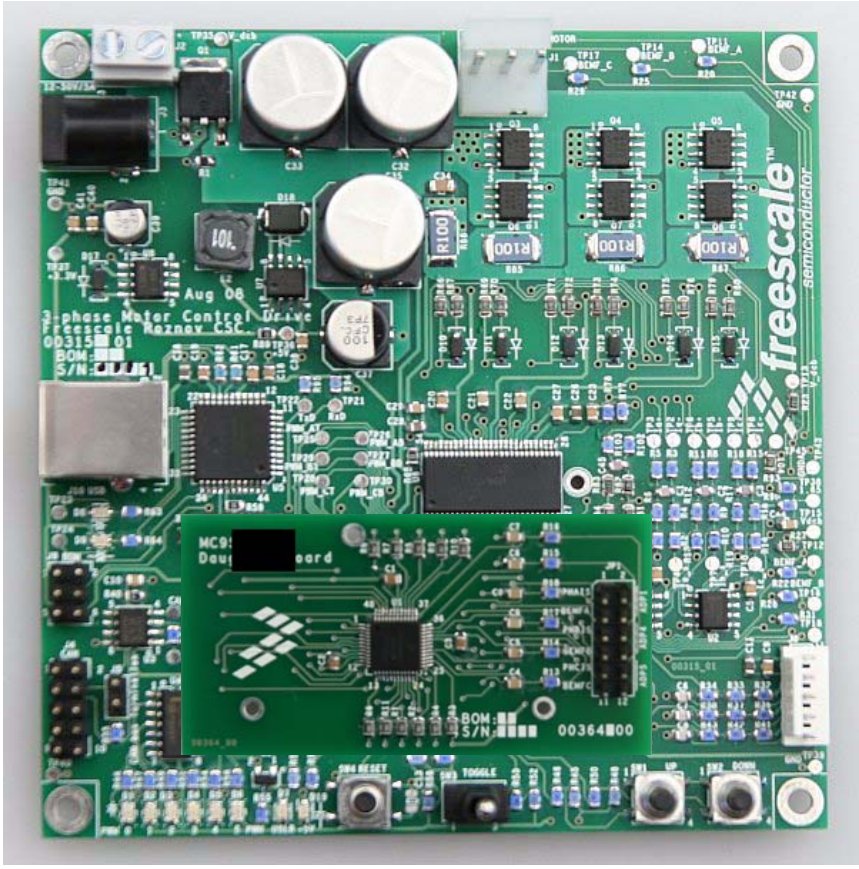

Figure 4-2. 3-Phase BLDC/PMSM Low-Voltage Motor Control Drive Board with Daughter Board

The motor used in this application is a standard production BLDC with a Hall sensor mounted on the shaft, though not used in the control. The motor has the following specifications:

Table 4-1. Specifications of the Motor

Motor Specification:	**Motor Type:**	**LINIX 45ZWN24-40 BLDC Motor**
	Nominal voltage (line-to-line)	24 V RMS
	Nominal speed	4000 RPM
	Nominal current (phase)	2.34 A
	Nominal torque	0.0924 Nm

Chapter 5
Software Design

5.1 Introduction

This section describes the design of the software blocks of the drive. The software will be described in terms of:

- Application Software Processes and Data Flow Diagram
- Application Software States
- Application Software Flowchart
- FreeMASTER Software
- Software Setting and Overview
- Application Parameters (see 5.7/5-16)

5.2 Application Software Processes and Data Flow Diagram

The application data flow diagram with the main processes and variables is shown in Figure 5-1.

The individual processes of the control routines are described in the following sections. The variables are described at the end of this section.

5.2.1 BLDC Application Main Process

Provides the main state machine control. The state is defined by the bldcStateIndex variable. The process is mainly realized in the main background loop and the MTIM Tim Base Overflow Interrupt (see Section 5.4.3, “MTIM TimB Overflow ISR”).

5.2.2 FreeMASTER Process

The FreeMASTER process provides the communication between the BLDC control application and the superior system — a personal computer. The physical interface between the personal computer and MC9S08MP16 is a serial communication based on the SCI module (however due to RS232 port unavailability the hardware board transfers the RS232 to USB — see Section 3.3.8, “FreeMASTER”). The FreeMASTER is used to control the application via variables such as velocityDesired and appControlFlags, and also to monitor the application variables like the actual BLDC motor velocity, DC-Bus current, and voltage. It is also used for tuning application parameters rampAccelOL, regSpeedPIparams, and so on. This process is mainly realized in the main background loop.

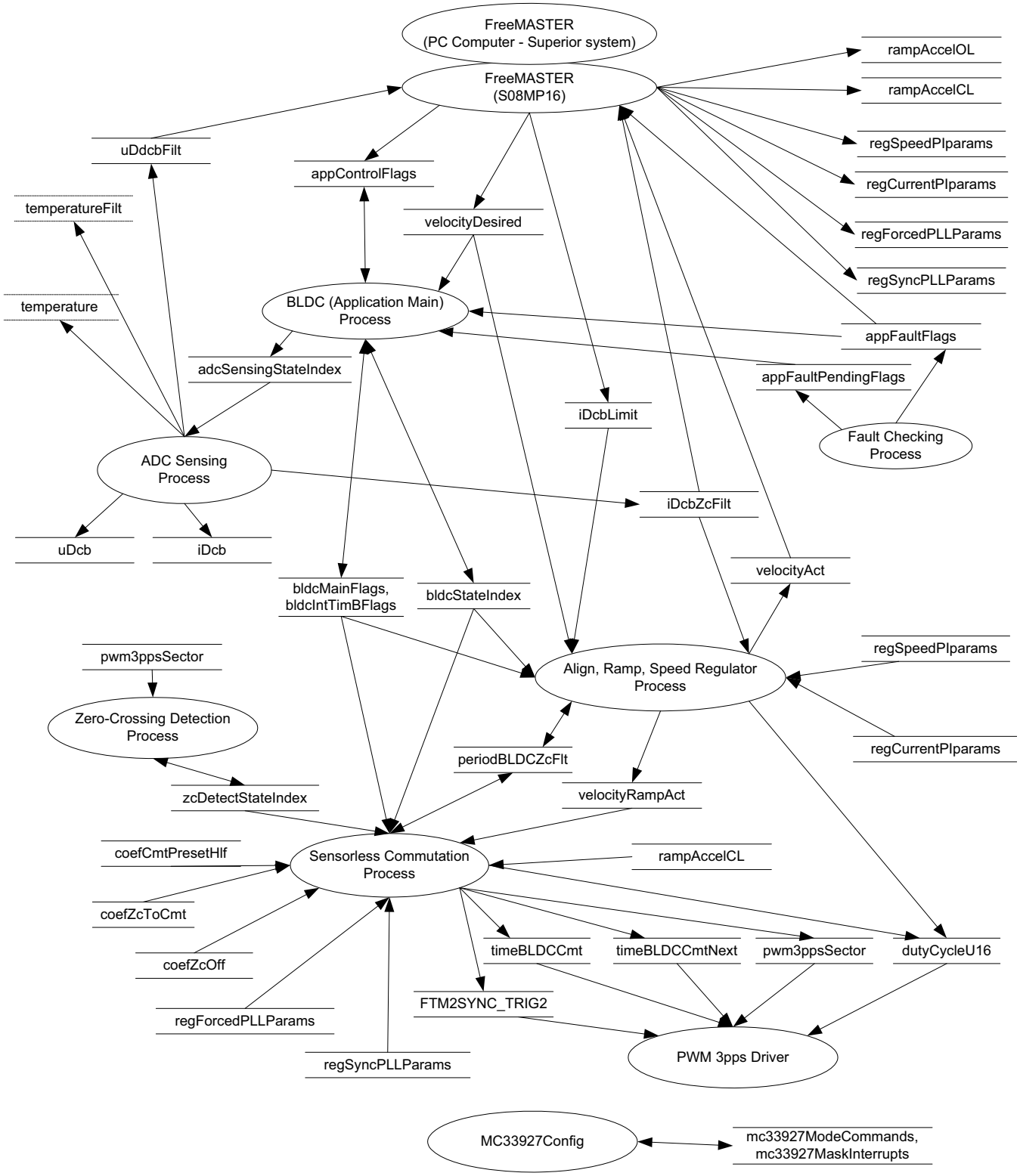

Figure 5-1. BLDC Control Data Flow Diagram

5.2.3 ADC Sensing Process

Is analog variables sensing process. It provides DC-Bus current iDcb sensing synchronized with the PWM 3pps generator, and the voltage uDcb and temperature sensing (the software is prepared for temperature or other variable sensing, but this is not implemented in the default hardware). This process also provides filtered variables with an average from a defined number of steps.

For example, uDcbFilt = uDcbSum32/adcSensing_AverageUDCBCounter

The process is mainly realized in the ADC Sensing Complete Interrupt (see Section 5.4.2, “ADC ADC Sensing Complete ISR”).

5.2.4 Fault Checking Process

The fault checking process consists of the fault interrupt which is triggered by hardware over-current detection (see Section 5.4.7, “FTM2 pwm3pps Fault ISR”), uDcb voltage, or other variables checking (see Section 5.4.1, “Application Background Loop”).

5.2.5 Zero-Crossing Detection Process

Provides the back-EMF zero-crossing detection with the state described in Section 2.4.5, “The Zero-Crossing Sensing States.” The process state is defined by the zcDetectStateIndex variable. This process is mainly realized in the interrupt subroutines Timer Cmt Zc OC ISR (see Section 5.4.4, “FTM1 ch0 Timer Cmt Zc OC ISR”), TimerZc ZC IC ISR (see Section 5.4.5, “FTM1 ch1 TimerZc ZC IC ISR ISR”) and ZC Comparator Cur. Recirc. Done ISR (see Section 5.4.6, “HSCMP ZC Comparator Cur. Recirc. Done ISR”).

5.2.6 Sensorless Commutation Process

The sensorless commutation process provides all the functionality for the sensorless BLDC commutation control according to the state (bldcStateIndex). This process is mainly realized in the interrupt subroutines Timer Cmt Zc OC ISR (see Section 5.4.4, “FTM1 ch0 Timer Cmt Zc OC ISR), TimerZc ZC IC ISR (see Section 5.4.5, “FTM1 ch1 TimerZc ZC IC ISR ISR”) and ZC Comparator Cur. Recirc. Done ISR (see Section 5.4.6, “HSCMP ZC Comparator Cur. Recirc. Done ISR”).

5.2.7 PWM 3pps Driver

This is the 3pps PWM driver for 6-step BLDC control. It serves the Section 5.2.6, “Sensorless Commutation Process” and Section 5.2.8, “Align, Ramp, Speed Regulator Process” processes.

5.2.8 Align, Ramp, Speed Regulator Process

Provides alignment timing, open-loop ramp, closed-loop ramp, and speed controller with torque limitation. The process is mainly realized inside the Time Base Interrupt (see Section 5.4.3, “MTIM TimB Overflow ISR”).

5.2.9 MC33927Config

This process provides configuration of the MC33927 3-phase power stage driver via the SPI module. The driver is utilized by the 3-Phase BLDC/PMSM Low-Voltage Motor Control Drive hardware board. The configuration is provided via SPI.

5.2.10 Application Data Flow Variables

- adcSensingStateIndex — adc sensing state index variable
- appControlFlags — application control flags
- appFaultFlags — application fault flags
- appFaultPendingFlags — application fault pending flags
- bldcFlags — bldc control process flags
- bldcIntTimBFlags — bldc control process flags
- bldcMainFlags — bldc control process flags
- bldcStateIndex — bldc control process state index variable
- coefCmtPresetHlf — coefficient 1/2 of the multiplicant for commutation period preset
- coefZcOff — coefficient for calculation of the period where zero-crossing detection sampling is off
- coefZcToCmt — coefficient for calculation of the period from zero-crossing to commutation
- FTM2SYNC_TRIG2 — hardware trigger used for synchronizing the PWM 3pps Sync Timer Cmt
- iDcb — DC-Bus current sample
- iDcbLimit — DC-Bus current limit (desired value for the current-torque regulator)
- iDcbZcFilt — DC-Bus current at zero-crossing point filtered
- mc33927ModeCommands — MC33927 mode setting commands
- mc33927MaskInterrupts — MC33927 mask interrupts — determines which event generates the MC33297 interrupt (the interrupt pin is connected to MC9S08MP16 fault input)
- periodBLDCZcFlt — period between back-EMF zero-crossings filtered (average or PLL loop)
- pwm3ppsSector — sector of the PWM 3pps — determines the 6-step vector
- rampAccelCL — acceleration ramp closed-loop
- rampAccelOL — acceleration ramp open-loop
- regCurrentPIparams — current regulator parameters
- regForcedPLLParams — regulator parameters for Forced PLL control
- regSpeedPIparams — speed regulator parameters
- regSyncPLLParams — regulator parameters for Synchronized PLL control
- temperature — measured temperature — NOT APPLICABLE FOR DEFAULT HARDWARE
- temperatureFilt — temperature filtered — NOT APPLICABLE FOR DEFAULT HARDWARE
- uDdcb — DC-Bus voltage sampled
- uDdcbFilt — DC-Bus voltage filtered
- velocityDesired — velocity desired

- velocityRampAct — velocity ramp actual value
- velocityRequired — velocity required
- zcDetectStateIndex — zero-crossing detection state index variable

5.3 Application Software States

The main application state machine with the transition between the states is shown in Figure 5-2.

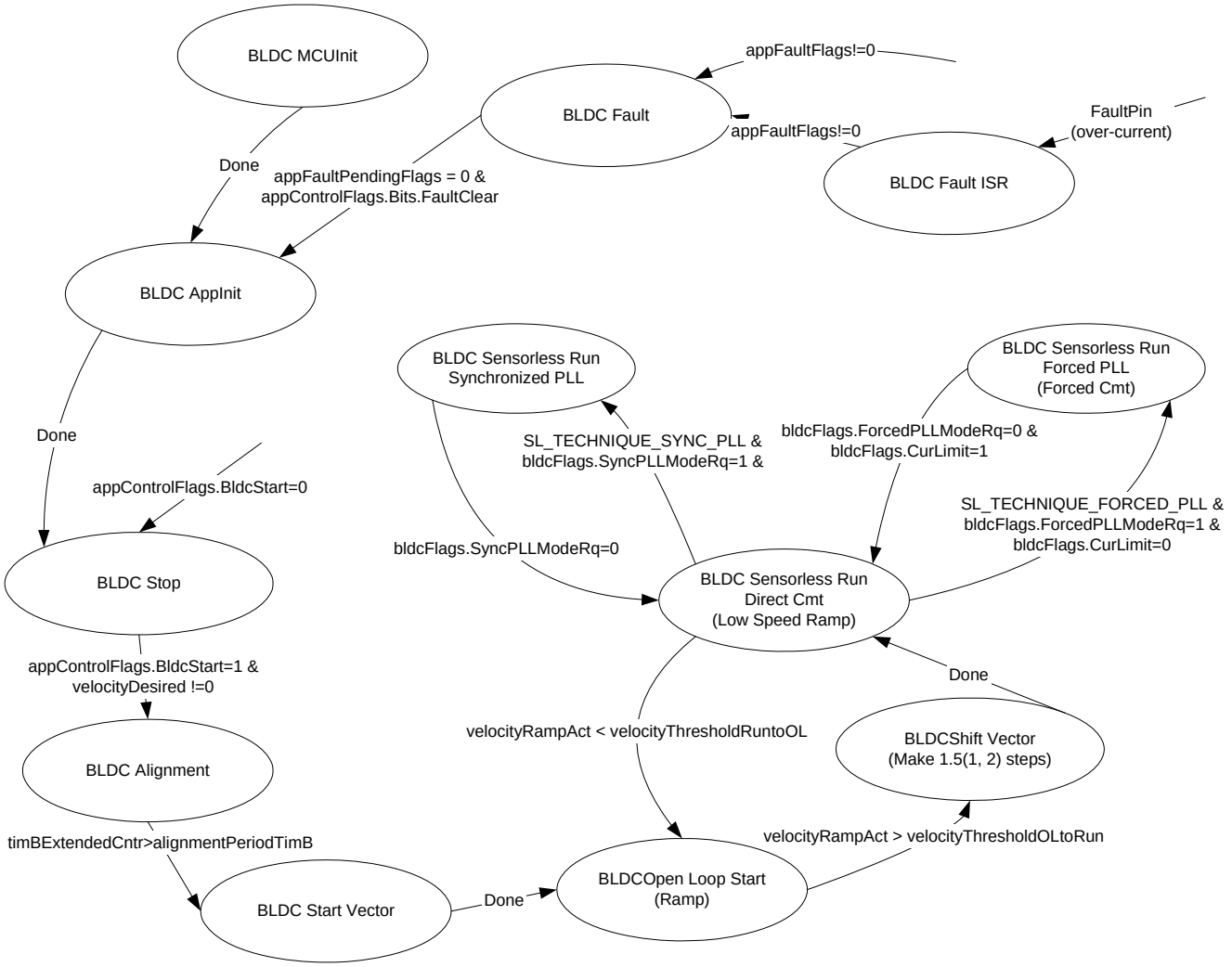

Figure 5-2. BLDC Control State Diagram

The application states are described in the following sections.

5.3.1 State Application MCU Init

The application goes to this state immediately after reset. In this state the software initializes all the general application modules:

- general purpose registers initialization
- MCU clock initialization
- input/output registers initialization
- SPI Initialization
- MC33927 driver configuration

5.3.2 State BLDC MCU Init

In this state, the BLDC application dedicated modules are initialized.

- the FTM2 module is initialized as pwm3pps for complementary BLDC 6-step control
- the FTM1 timer ch0 is initialized as the zero-crossing output compare Timer Cmt
- the FTM1 timer ch1 is initialized as the zero-crossing input capture Timer Zc
- the HSCMP comparator is initialized as a zero-crossing comparator
- the PDB2 delay block is initialized to a zero-crossing sampling window
- the ADC comparator is initialized for current sensing synchronized with the 3-phase PWM and voltage (possibly temperature where supported by hardware) sampling
- the PDB1 delay block is initialized for 3-phase PWM synchronization with the ADC
- the MTIM timer is initialized for Time Base interrupts (default 3 ms)
- the App Init state is entered

5.3.3 State BLDC App Init

In this state, the current offset calibration is provided. DC-Bus current samples are added to the iDcbSum register with the adcSensing_CalibrationCounter increment. After a defined number of samples (ADC_SENSING_CALIBRATION_COUNT), the average iDcboffset is calculated.

In this state the following processes are active:

- The ADC Sensing process — measures the DC-Bus current iDcb, voltage uDcb (possibly temperature)
- FreeMASTER communication with a superior PC
- Application and Fault Control Process

5.3.4 State BLDC Stop

The 3-phase power stage is off. The software checks the state of the appControlFlags.Bits.BldcStart flag, which is set by a superior system. The superior system is a personal computer with a FreeMASTER communication interface.

In this state, the following processes are active:

- The ADC Sensing process — measures the DC-Bus current iDcb and voltage uDcb (possibly temperature)
- FreeMASTER communication with a superior PC
- Application and Fault Control Process

5.3.5 State BLDC Alignment

Provides a constant vector with a defined ALIGNMENT_DUTY_CYCLE. The alignment state is timed using the MTIM Tim Base. It is exited when timBExtendedCntr>alignmentPeriodTimB.

In this state, the following processes are active:

- The ADC Sensing process — measures the DC-Bus current iDcb, voltage uDcb (possibly temperature)
- FreeMASTER communication with a superior PC
- Speed / Alignment/ Ramp Process — Alignment timing
- Application and Fault Control Process

5.3.6 State BLDC Start Vector

The start vector is set before the open-loop commutation.

5.3.7 State BLDC Open-Loop Start

The BLDC motor is commuted with a linear open-loop starting ramp of rampAccelOL acceleration, as described in Section 2.4.2, “Power Stage — Motor System Model.” When velocityRampAct > velocityThresholdOLtoRun, the sensorless feedback states will be entered with the intermediate state Section 5.3.8, “State BLDC Shift Vector.”

In this state, the following processes are active:

- The ADC Sensing process — measures the DC-Bus current iDcb, voltage uDcb (possibly temperature)
- FreeMASTER communication with a superior PC
- Speed / Alignment/ Ramp Process
  - Speed/Torque Closed-Loop Controller
  - Velocity Ramp with acceleration rampAccelOL
- Application and Fault Control Process

5.3.8 State BLDC Shift Vector

The stator flux vector, relative to the rotor, needs to change from an open-loop start to the regular run position — see Figure 2-4.

5.3.9 State BLDC Sensorless Run — Direct Cmt

The BLDC motor is controlled by the direct commutation calculation technique according to Section 2.5.3, "Sensorless Run — Direct Commutation Calculation."

In this state the following processes are active:

- The ADC Sensing process — measures the DC-Bus current iDcb, voltage uDcb (possibly temperature)
- FreeMASTER communication with a superior PC
- Speed / Alignment/ Ramp Process
  - — Speed/Torque Closed-Loop Controller
  - — Velocity Ramp with acceleration rampAccelCL
- Application and Fault Control Process
- ZC Detection Process
- Sensorless Commutation Process — Direct Cmt Mode

5.3.10 State BLDC Sensorless Run — Synchronized PLL

This state provides a better speed precision (the commutation period is constant). It is entered after the state State BLDC Sensorless Run — Direct Cmt.

If the constant SL_TECHNIQUE_SYNCHRONIZED_PLL is defined for the precompiler (*main.h*) and the commutation is stabilized, bldcFlags.SyncPLLModeRq=1, and current limitation is not active, bldcFlags.CurLimit=0, then the state is entered.

In this state, the following processes are active:

- The ADC Sensing process — measures the DC-Bus current iDcb, voltage uDcb (possibly temperature)
- FreeMASTER communication with superior PC
- Speed / Alignment/ Ramp Process
  - — Speed Closed-Loop Controller
  - — Velocity Ramp with acceleration rampAccelCL
- Application and Fault Control Process
- ZC Detection Process
- Sensorless Commutation Process — Synchronized PLL Mode

5.3.11 State BLDC Sensorless Run — Forced PLL (Forced Cmt)

This state provides a better speed precision (the commutation period is constant). It is entered after the state State BLDC Sensorless Run — Direct Cmt.

If the constant SL_TECHNIQUE_FORCED_PLL is defined for the precompiler (*main.h*) and the commutation is stabilized, bldcFlags.ForcedPLLModeRq=1, and current limitation is not active, bldcFlags.CurLimit=0, then the state is entered.

In this state, the following processes are active:

- The ADC Sensing process — measures the DC-Bus current iDcb, voltage uDcb (possibly temperature)
- FreeMASTER communication with superior PC
- Speed / Alignment/ Ramp Process
  — Velocity Ramp with acceleration rampAccelCL
- Application and Fault Control Process
- ZC Detection Process
- Sensorless Commutation Process — Forced PLL Mode

5.3.12 State BLDC Fault ISR

This fault interrupt subroutine state is initiated by a hardware pin FTM2FAULT (over-current) which generates an FTM2 pwm3pps Fault ISR. The MC9S08MP16 microcontroller provides programmable fault protection where a fault protection can disable any combination of PWM pins. These faults are generated by logic one on any of the fault pins. The fault pins are assigned to both Flextimer modules. When the fault protection hardware disables the PWM pins, the PWM generator continues to run, and only the output pins are deactivated. If a fault is latched in, it must be cleared and the fault state is invoked prior to enabling the PWM, to prevent an unexpected interrupt.

5.3.13 State BLDC Fault

This fault state is entered when appFaultFlags!=0. The fault flags are set by the ADC Sensing Process, or in the State BLDC Fault ISR initiated by a hardware pin FTM2FAULT (over-current) which generates an FTM2 pwm3pps Fault ISR.

The fault state is exited when no fault is pending (appFaultPendingFlags = 0) and where appControlFlags.Bits.FaultClear = 1 is set by a superior system. The superior system is a personal computer with a FreeMASTER communication interface.

In this state, the following processes are active:

- The ADC Sensing process — measures the DC-Bus current iDcb and voltage uDcb (temperature)
- FreeMASTER communication with a superior PC
- Application and Fault Control Process

5.4 Application Software Flowchart

The application software states are described in Section 5.3, “Application Software States.” The flow chart in Figure 5-3 shows all the interrupts and main software loop, and gives us information on the processes execution. Due to the software complexity the software flowchart in Figure 5-3 describes the software functionality in the state BLDC Sensorless Run (Direct Cmt/Forced PLL sub-states). This state is most descriptive, since most of the processes are active (see Section 5.3.9, “State BLDC Sensorless Run — Direct Cmt”).

5.4.1 Application Background Loop

The endless application background provides

- Application and Fault Control process with:
  - microcontroller initialization
  - FreeMASTER polling function *FMSTR_Poll()*
  - BLDC Process state machine
  - Fault Control
  - Watchdog periodic feeding.

The main application control tasks are executed in interrupt service routines, which interrupt the background loop.

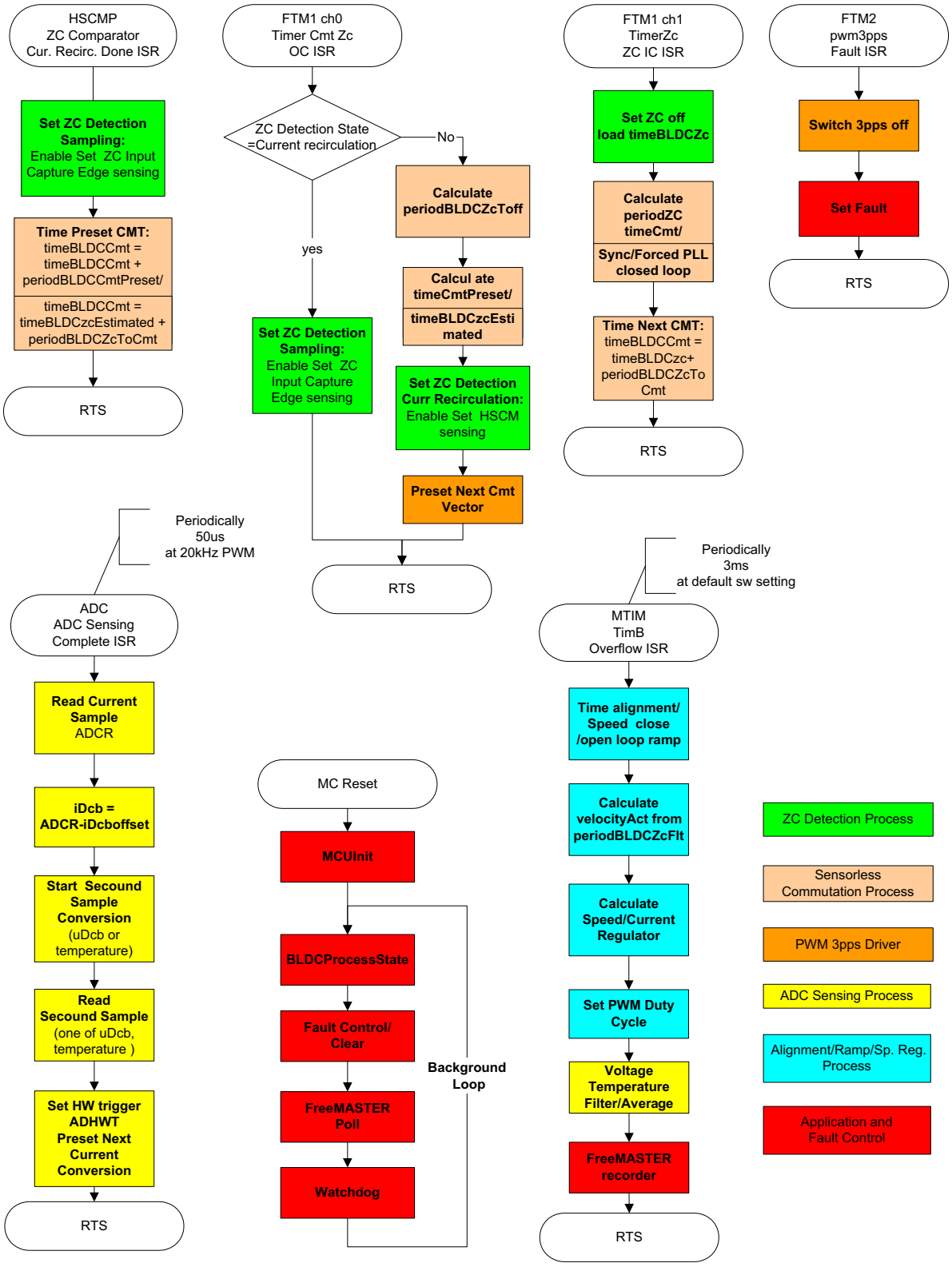

Figure 5-3. Application Flow Chart — State BLDC Sensorless Run (Direct Cmt/Forced PLL)

5.4.2 ADC ADC Sensing Complete ISR

This interrupt subroutine is mainly utilized for:

- Analog Sensing Process.

It provides the reading of two analog samples per one interrupt. It starts with:

  — the read of the synchronized DC-Bus current iDcb.

The second conversion is either the DC-Bus voltage or the temperature (temperature sensing is not implemented on the default hardware).

  — This second conversion is started by software and read out after the conversion is complete. At the end, the first conversion hardware trigger ADHWT is enabled.

5.4.3 MTIM TimB Overflow ISR

This is periodically called (default 3 ms). It is used by the following processes:

- Speed/Alignment/Ramp Process
  - — Time alignment/Speed close /open-loop ramp
  - — Calculate velocityAct from periodBLDCZcFlt
  - — Calculate Speed/Current Limitation PI controller
  - — Set PWM duty cycle
- ADC Sensing Process
  - — Voltage/temperature sample average calculation
- FreeMASTER Recorder

5.4.4 FTM1 ch0 Timer Cmt Zc OC ISR

This commutation and zero-crossing timer output capture interrupt is used by the following processes:

When ZC Detection State = Current recirculation

- Zero-Crossing Detection Process
  - — Set ZC Detection Sampling

When ZC Detection State != Current recirculation

- Sensorless Commutation Process
  - — Calculate periodBLDCZcToff
  - — Calculate timeCmtPreset (when in the Direct Commutation state) or
  - — timeBLDCzcEstimated (when in one of the Sensorless Run PLL states)
- PWM 3pps Driver
  - — Preset the Next Cmt Vector
- Zero-Crossing Detection Process
  - — Sets ZC Detection Curr Recirculation

5.4.5 FTM1 ch1 TimerZc ZC IC ISR ISR

This zero-crossing timer input capture interrupt is used by the following processes:

- Zero-Crossing Detection Process
  - — Set zero-crossing off
  - — load the zero-crossing instant load timeBLDCZc
- Sensorless Commutation Process
  - — Calculate periodZC timeCmt (when Direct Commutation state) or
  - — Calculate the Sync/Forced PLL closed-loop regulator (when in one of the Sensorless Run PLL states)
  - — Time Next CMT:
  - — time the next commutation timeBLDCCmt = timeBLDCzc+ periodBLDCZcToCmt

5.4.6 HSCMP ZC Comparator Cur. Recirc. Done ISR

The comparator interrupt Current recirculation Done is utilized by:

- Zero-Crossing Detection Process
  - — Set the Zero-Crossing Detection Sampling
- Sensorless Commutation Process
  - — time the next commutation preset (the preset commutation will be performed when no zero-crossing appears):
    - – Time Preset timeBLDCCmt = timeBLDCCmt + periodBLDCCmtPrese (when Direct Commutation state) or
    - – timeBLDCCmt = timeBLDCzcEstimated + periodBLDCZcToCmt (when one of the Sensorless Run PLL states)

5.4.7 FTM2 pwm3pps Fault ISR

This hardware fault interrupt subroutine is utilized by:

- PWM 3pps Driver
  - — Switch the 3pps off
- Application and Fault Control
  - — Set the Fault

5.5 FreeMASTER Software

FreeMASTER software was designed to provide a debugging, diagnostic, and demonstration tool for the development of algorithms and applications. Moreover, it's very useful for tuning the application for different power stages and motors, because almost all the application parameters can be changed via the FreeMASTER interface. This consists of a component running on a PC and another part running on the target MCU, connected via an RS-232 serial port. A small program is resident in the MCU that communicates with the FreeMASTER software to parse commands, return status information to the PC,

and process control information from the PC. FreeMASTER software executing on the PC uses Microsoft Internet Explorer as the user interface.

5.5.1 FreeMASTER Serial Communication Driver

The presented application includes the FreeMASTER Serial Communication Driver. The FreeMASTER Serial Communication Driver fully replaces the former PC Master driver. The new FreeMASTER driver remains fully compatible with the communication interface provided by the old PC Master drivers. It brings, however, many useful enhancements and optimizations.

The main advantage of the new driver is a unification across all supported Freescale processor products, as well as several new features that were added. One of the key features implemented in the new driver is target-side addressing (TSA), which enables an embedded application to describe the memory objects it grants the host access to. By enabling the so-called TSA-Safety option, the application memory can be protected from illegal or invalid memory accesses.

To include the new FreeMASTER Serial Communication Driver in the application, the user has to manually include the driver files in the CodeWarrior project. For the presented application, the driver has already been included.

The FreeMASTER driver files are located in the following folders:

- {Project}sources — contains *freemaster_cfg.h*, *freemaster.h*.
- {Project}sources\freemaster — contains platform-dependent driver C-source and header files

All C files included in the freemaster folders are added to the project for compilation and linking (see support group in the project). The master header file *freemaster.h* declares the common data types, macros, and prototypes of the FreeMASTER driver API functions. This should be included in your application (using *#include* directive), wherever you need to call any of the FreeMASTER driver API functions.

Note that the FreeMASTER driver does not perform any initialization or configuration of the SCI module it uses to communicate. Therefore, it is the user's responsibility to configure the communication module before the FreeMASTER driver is initialized by the *FMSTR_Init()* call. The default baud rate of the SCI communication is set to 9600 baud.

NOTE

Higher communication speeds than 9600 baud can cause communication instability mainly when the Recorder function is used.

FreeMASTER uses a poll-driven communication mode. It does not require the setting of interrupts for SCI. Both communication and protocol decoding are handled in the application background loop. The polling mode requires a periodic call of the *FMSTR_Poll()* function in the application main().

The driver is configured using the *appconfig.h* header file. Changes to the file are preferably made through the provided Quick Start graphical configuration tool (in CodeWarrior toolbar Project/Configuration Tool). The user has to modify this file to configure the FreeMASTER driver. The FreeMASTER driver C-source files include the configuration file, and use the macros defined there for conditional and parameter compilation.

A detailed description of the FreeMASTER Serial Communication Driver is provided in AN2471, “PC Master Software Communication Protocol Specification.”

5.5.2 FreeMASTER Recorder

Part of the FreeMASTER software is also a recorder, which is able to sample the application variables at a specified sample rate. The samples are stored in a buffer and read by the PC via an RS-232 serial port. The sampled data can be displayed in a graph or the data can be stored. The recorder behaves like a simple on-chip oscilloscope with trigger / pre-trigger capabilities. The size of the recorder buffer and the FreeMASTER recorder time base can be defined in the *appconfig.h* configuration.

The recorder routine must be called periodically from the loop in which you want to take the samples. The following line must be added to the loop code:

```
FMSTR_Recorder(); /* FreeMASTER recorder routine call */
```

In this application, the FreeMASTER recorder is called from the MTIM Time Base Overflow interrupt, which creates a 3 ms time base for the recorder function. A detailed description of the FreeMASTER software is provided in AN2395, “PC Master Software Usage.”

5.5.3 FreeMASTER Control Page

The FreeMASTER control page creates a graphical user interface (GUI) for the 3-phase sensorless BLDC control. Start the FreeMASTER software window’s project by clicking on the *BLDC_Sensorless_S08MP16.pmp* file. A user is able to monitor all the important quantities of the motor. By clicking the speed gauge, the motor is started and the desired speed is set. The actual motor speed, motor currents, and voltages are displayed on the control page gauges. Application status is displayed.

The following FreeMASTER software control page actions are supported:

- Setting the required speed of the motor
- Switch running motor on/off

The FreeMASTER software control page displays:

- Actual and required speed
- DC-bus current and voltage
- Application (fault) status

5.6 Software Setting and Overview

The application supports three sensorless synchronous commutation algorithms — Direct commutation calculation, Synchronized PLL and Forced PLL. Switching between algorithms can be done in the source code (*main.h*) using *#define* for permanent change of the control routine (SL_TECHNIQUE_SYNCHRONIZED_PLL, SL_TECHNIQUE_FORCED_PLL). The zero-crossing detection process has two modes. The current decay sampling mode detects where the current decay finishes. If the mode is not set, the current decay is defined by periodBLDCToff time-out. The mode permanent change is defined by *#define* ZC_MODE_CURRENT_DECAY_SAMPLING. The software default switching technique is unipolar PWM. The bipolar pwm is defined by

PWM_BIPOLAR_SWITCHING. The benefits of a structured modular software design are well perceived. This is especially true for complex motor control systems with many interacting software sub-blocks. Due to these reasons, a Freescale library set for the HCS08 has been developed.

5.6.1 Library Functions

The application source code uses the new Freescale Embedded Software Library for the S08 family of microcontrollers. The library consists of two packs. The basic math primitives (S08math) and the embedded control functions (ECLIB). For detailed information, follow [10].

5.7 Setting the Software Parameters for a Specific Motor

The default software parameter settings have been tuned for a default hardware setup with the motor LINIX 45ZWN24-40 and Freescale's 3-phase BLDC/PMSM Low-Voltage Motor Control Drive board. When another motor and hardware are used, the software settings need to be changed according to their specific parameters.

All application parameters dedicated to the motor or application ratings (max. velocity...) are defined in the *main.h* file and commented to help users modify the parameters according to their requirements.

Part of parameter file:

```
#define ALIGNMENT_PERIOD_MS     1000.0      /* Alignment state period [ms] */
#define ALIGNMENT_PERIOD_TIMB   ((ALIGNMENT_PERIOD_MS)*1000.0/(TIMB_PERIOD_US))
                                            /* Alignment state period [Time Base scale ms] */
```

All application parameters dedicated to the hardware boards and processor (pin assignment, clock rating, and so on.) are defined in the *hw_config.h* file and commented to help users modify the parameters according to their requirements.

Part of the parameter file:

```
#define CPU_FREQUENCY_MHZ           40.0       /* CPU FREQUENCY 40 000 000 [Hz] */
#define SYSTEMCLOCK_FREQUENCY_MHZ   (CPU_FREQUENCY_MHZ)
#define BUSCLOCK_FREQUENCY_MHZ      (CPU_FREQUENCY_MHZ/2.0)
```

5.8 Microcontroller Memory Usage

Table 5-1 shows how much memory is needed to run the 3-phase BLDC Motor Control application. A significant part of the microcontroller memory is still available for other tasks.

Memory Type	Available on MC9S08MP16	Used
Program Flash	16 KByte	7186 Bytes
Unified Data/Program RAM	1 KByte	412 Bytes

Table 5-1. Application Memory Usage

The Table 5-1 shows the code size without the FreeMASTER functionality.

5.9 Conclusion

The design of a speed closed-loop drive with a 3-phase BLDC motor was described in this Reference Design Manual. It is based on Freescale's MC9S08MP16 microcontroller. It illustrates the drive from a system point of view, power stage, hardware around the microcontroller and finally, the software.

The described design shows simplicity and efficiency in use of the MC9S08MP16 microcontroller for motor control, and introduces it as an appropriate candidate for different low-cost applications in industrial appliance and automotive fields.