XC2000/XE166 Family

AP16173

Sensorless Control of BLDC Motor using XE164F Microcontroller

Application Note

V1.0 2010-03

Microcontrollers

Edition 2010-03

Published by
Infineon Technologies AG
81726 Munich, Germany

LEGAL DISCLAIMER

THE INFORMATION GIVEN IN THIS APPLICATION NOTE IS GIVEN AS A HINT FOR THE IMPLEMENTATION OF THE INFINEON TECHNOLOGIES COMPONENT ONLY AND SHALL NOT BE REGARDED AS ANY DESCRIPTION OR WARRANTY OF A CERTAIN FUNCTIONALITY, CONDITION OR QUALITY OF THE INFINEON TECHNOLOGIES COMPONENT. THE RECIPIENT OF THIS APPLICATION NOTE MUST VERIFY ANY FUNCTION DESCRIBED HEREIN IN THE REAL APPLICATION. INFINEON TECHNOLOGIES HEREBY DISCLAIMS ANY AND ALL WARRANTIES AND LIABILITIES OF ANY KIND (INCLUDING WITHOUT LIMITATION WARRANTIES OF NON-INFRINGEMENT OF INTELLECTUAL PROPERTY RIGHTS OF ANY THIRD PARTY) WITH RESPECT TO ANY AND ALL INFORMATION GIVEN IN THIS APPLICATION NOTE.

Information

For further information on technology, delivery terms and conditions and prices, please contact the nearest Infineon Technologies Office (**www.infineon.com**).

Warnings

Due to technical requirements, components may contain dangerous substances. For information on the types in question, please contact the nearest Infineon Technologies Office.

Infineon Technologies components may be used in life-support devices or systems only with the express written approval of Infineon Technologies, if a failure of such components can reasonably be expected to cause the failure of that life-support device or system or to affect the safety or effectiveness of that device or system. Life support devices or systems are intended to be implanted in the human body or to support and/or maintain and sustain and/or protect human life. If they fail, it is reasonable to assume that the health of the user or other persons may be endangered.

XC2000/XE166 Family
Revision History: V1.0, 2010-03
Previous Version: none

Page	Subjects (major changes since last revision)
-	This is the first release

We Listen to Your Comments
Is there any information in this document that you feel is wrong, unclear or missing?
Your feedback will help us to continuously improve the quality of this document.
Please send your proposal (including a reference to this document) to:
mcdocu.comments@infineon.com

Table of Contents

1 Introduction

Because of their compact size, controllability and high efficiency, Brushless Direct Current (BLDC) motors are used in a diverse range of industries including appliance manufacturing, automotive, aerospace, consumer, medical, industrial automation equipment and instrumentation.

BLDC motors do not use brushes for commutation, but are electronically commutated instead. The BLDC motor is usually operated with rotor position sensors (Hall Effect sensors or Encoders), since the electrical excitation must be synchronous with the rotor position. It is desirable to eliminate position sensors for the reasons of cost, reliability and mechanical packaging. That makes it more important to control the BLDC motor without the position sensors (Sensorless Operation).

This application notes describes the implementation of a Sensorless control algorithm for BLDC motors using the Infineon XE164F microcontroller. The BLDC motor's Back-emf is used for commutation.

In the following chapters, the principle of the Sensorless control algorithm and the software implementation is discussed in detail. The advantages of the microcontroller peripherals are also discussed: CAPCOM6E (Capture and Compare Unit for modulation and PWM generation) and the fast 10-bit ADC (Analog-to-Digital Converter). These peripherals are specifically designed for motor control applications.

The motor control software is written in both C and assembly. This software uses the XE164F peripherals, while the mathematical computations such as the PI control algorithm use the microcontroller DSP Data processing (MAC Unit) functionality.

1.1 Motor Control using the XE164F Microcontroller

The XE164F microcontroller has dedicated peripherals specifically designed for motor control applications. The key features of the microcontroller are:

- High-performance CPU with five-stage pipeline
- Interrupt system with 16 priority levels for up to 83 sources
- Eight-channel interrupt-driven single-cycle data transfer with Peripheral Event Controller (PEC)
- Two Synchronizable A/D Converters with up to 16 channels, 10-bit resolution
- 16-channel general purpose capture/compare unit (CAPCOM2)
- Up to three capture/compare units for flexible PWM signal generation (CCU6x)
- Multi-functional general purpose timer unit with 5 timers and quadrature decoders
- On-chip MultiCAN interface (Rev. 2.0B active) with up to 128 message objects

With the intensive, autonomous use of dedicated peripherals designed for motor control, CPU load can be reduced. The CPU can then be used to perform other key application tasks.

1.2 Hardware and Software Components

The following hardware and software components are required:

- PC with Microsoft Windows 2000 or above
- Infineon XE164F Drive card [2]
- Infineon Low Voltage Inverter Board [3]
- Infineon Drive Monitor Stick [4]
- BLDC Motor – MAXON EC32 15W
- 24 V Power supply for Drive Board
- KEIL (µV3) Tool chain for Infineon XE164F

2 Principle of Sensorless Operation

This chapter describes the principles of Sensorless Operation.

2.1 Motor Theory

This application note focuses on control of the most popular and widely used 3-phase BLDC motors. The Brushless DC motor consists of a Permanent magnet that rotates, surrounded by three equally spaced windings that are fixed. Current flowing in each winding produces a magnetic field vector, which sums with the fields from the other windings.

By controlling currents in the three windings, a magnetic field of arbitrary direction and magnitude can be produced by the stator. Torque is produced by the attraction or repulsion between this net stator field and the magnetic field of the rotor.

2.2 Principle of Operation

The commutation of a BLDC motor is controlled electronically. The stator windings should be energized in a particular sequence to rotate the motor. It is important to know the rotor position to energize appropriate stator windings.

In sensor mode, the rotor position is sensed using Hall Effect sensors embedded in the stator or an encoder. In this application, rotor position is estimated using the Back-EMF.

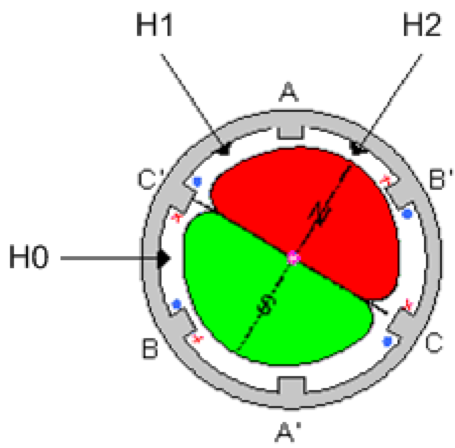


Figure 1 Single Pole Pair BLDC Motor with Hall Sensor

BLDC motors are commutated every 60° during a full cycle of 360° electrical, so in total there are 6 steps in one cycle. A transition from one step to another step is called commutation.

Two of the three coils are energized at any given time. In each commutation sequence one of the windings is energized to positive power (current enters into the winding), the second winding is negative (current exits the winding) and the third is in a non-energized condition. Torque is produced by the interaction between the magnetic field generated by the stator coils and the permanent magnets. Ideally, the peak torque occurs when these two fields are at 90 degrees to each other and falls off as the fields move together. In order to keep the motor running, the magnetic field produced by the windings should shift position as the rotor moves to catch up with the rotor field. What is known defines This sequence of energizing windings is known as 'Six-Step commutation' or 'Block Commutation'.

The motor takes six steps to complete one electrical cycle for a three phase machine. In general, the relationship between mechanical and electrical degrees is as stated in equation (1.1):

$$\textbf{Mechanical Revolution} = \frac{\textbf{Electrical Revolution}}{\textbf{Pole Pairs}} \quad \text{.....(1.1)}$$

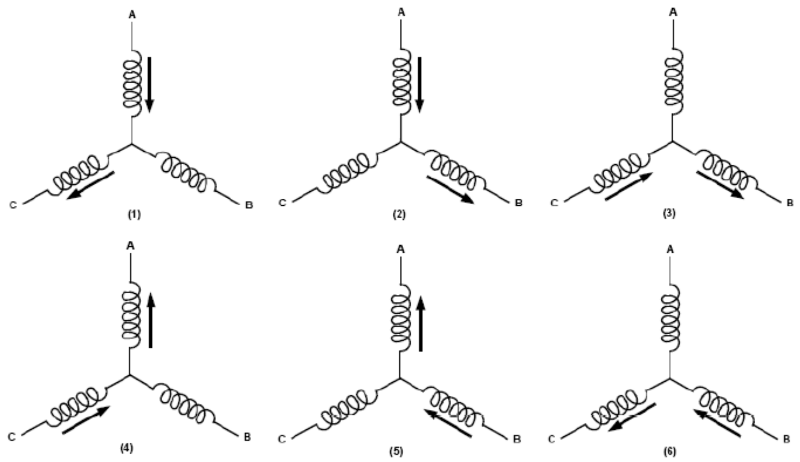


Figure 2 Six Step Commutation Sequence

2.3 Three Phase Inverter

An inverter is an electronic circuit for converting direct current to alternating current. The structure of a typical three phase voltage source power inverter is shown in Figure 3. The six MOSFETs are controlled by the input PWM signals (A+, A-, B+, B-, C+ and C-), that shape the input voltages supplied to the motor terminals.

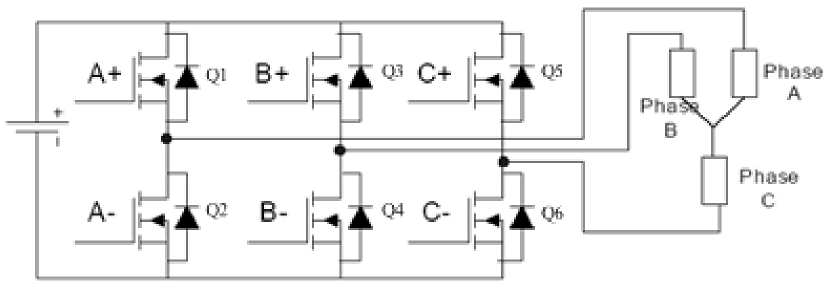


Figure 3 Three Phase Voltage Source Inverter

Note that whenever the MOSFET A+ is switched on, MOSFET A- must be switched off and vice-versa, to prevent damaging shoot-through current.

Table 1 Commutation Sequence

Phase A	+	0	-	-	0	+
Phase B	-	-	0	+	+	0
Phase C	0	+	+	0	-	-
A+	ON	OFF	OFF	OFF	OFF	ON
B+	OFF	OFF	OFF	ON	ON	OFF
C+	OFF	ON	ON	OFF	OFF	OFF
A-	OFF	OFF	ON	ON	OFF	OFF
B-	ON	ON	OFF	OFF	OFF	OFF
C-	OFF	OFF	OFF	OFF	ON	ON

2.4 Sensorless Mode of Operation

One of the most commonly used methods for acquiring position information is to monitor the induced EMF of the machine phases when they are not being energized.

Using the common six-step commutation, one phase is inactive for 33.33% of the time and two phases conduct at any given time. The commutation timing for Sensorless drive can be calculated by examining the induced EMF across the inactive phase. If the zero crossing of the phase back-emf is detected, then the commutation of the appropriate stator windings is possible.

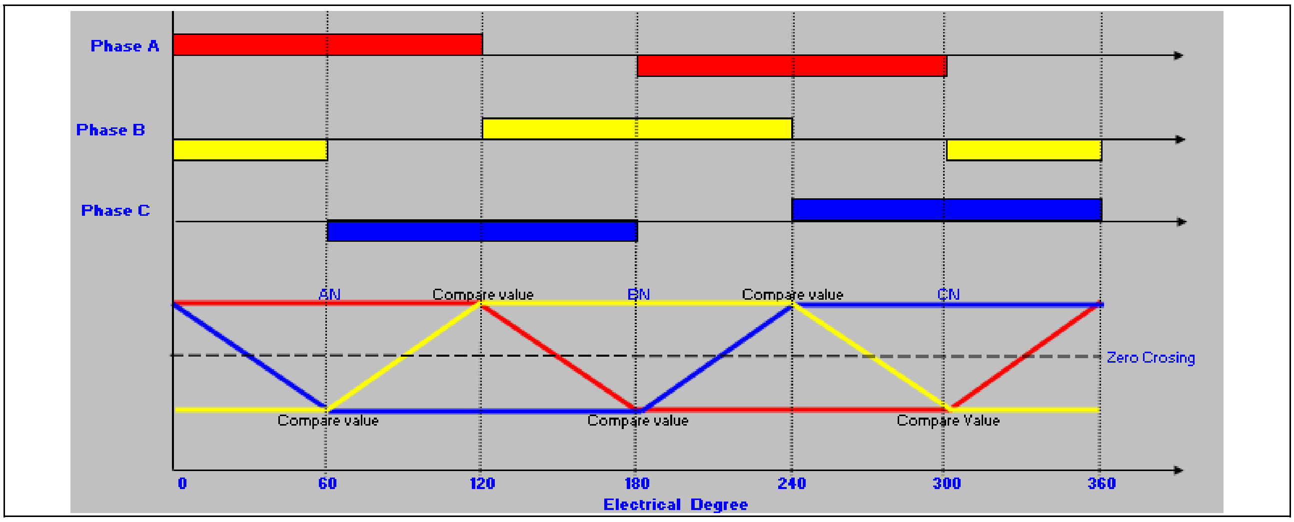


Figure 4 Phase Voltage and Induced EMF

As shown in Figure 4, the back-emf is trapezoidal in shape only two of the 3 phases are seen to conduct at any given time. The inverter switching pattern is easily derived from the back-emf. This switching pattern is organized into 6 commutation states.

Table 2 Motor Position and Commutation Sequence

Position	Energized Phase	Non Energized Phase
00	A+ ,B-	C
600	A+,C-	B
1200	B+,C-	A
1800	B+,A-	C
2400	C+,A-	B
3000	C+,B-	A

2.5 Back-EMF Measurement

In Block Commutation mode, while two phases are conducting the neutral voltage is approximately one half of the DC link voltage. The relation between phase voltage and back-EMF is:

$$Vp = R * I + L * \frac{dI}{dt} + Eemf \qquad \text{..... (1.2)}$$

Where :

V_p - Phase Voltage

R - Winding Resistance

L - Winding Inductance

I - Phase Current

di/dt - Rate of change of current over time

E_{emf} - Back-EMF

There is no current in the non-energized phase, so equation (1.2) becomes:

$$Vp = Eemf \qquad \text{..... (1.3)}$$

This means that by measuring the terminal voltage in the non-energized phase, the Back-EMF is easily determined. However the above conclusion is valid only when the two conducting phases are active. If one or both of the phases are being chopped, then the neutral voltage will vary and the relation between terminal voltage and back-emf will not be valid. For this reason, the terminal voltage measurement should be synchronized with the PWM signal used for chopping. This is shown in Figure 5.

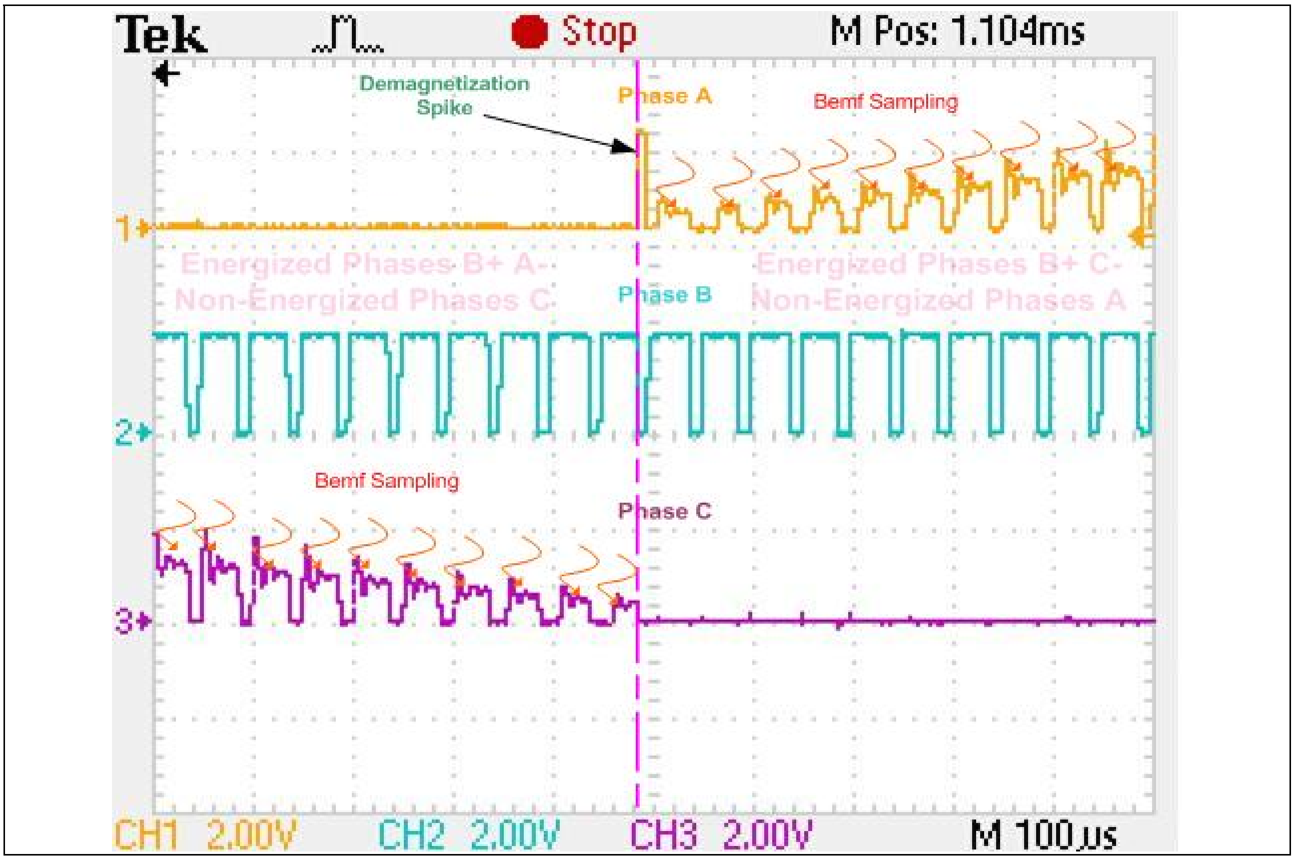


Figure 5 Phase Voltage and ADC Sampling Time

The disadvantage of using the ADC is that it is difficult to achieve a high speed range. This is because the ADC sampling is performed only once per PWM cycle. Therefore when the motor speed increases, the number of PWM cycles per commutation is decreased. However to obtain an accurate zero crossing measurement, a minimum of approximately 12 PWM periods per commutation are needed. This limits Sensorless operation at high speed, especially for motors with a large number of poles. This problem can be worse for applications that want to minimize switching losses by using a low PWM frequency.

2.6 Speed Control of BLDC Motor

The speed of the motor is directly proportional to the applied voltage. The average voltage applied to the motor can be varied using Pulse Width Modulation (PWM) by switching the MOSFET on or off.

At 100% PWM duty cycle, the motor will run at rated speed provided the rated dc voltage is supplied. To operate the motor at a desired speed below the rated speed, either the high side or low side transistor should be pulse width modulated.

Two control schemes are possible:

1. Open-loop speed Control (Voltage Control)
2. Closed-loop speed Control

In an Open loop speed control, the duty cycle is calculated based on the set reference speed. In a closed loop speed control, the actual speed is measured and compared with the reference speed to find the error difference. This error difference is supplied to a PI controller. The output from the PI controller is the new duty cycle.

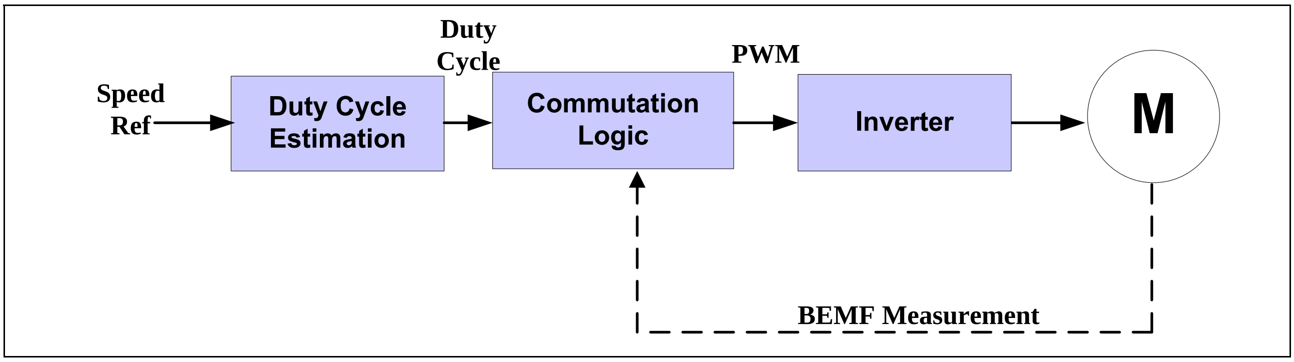


Figure 6 Open Loop Speed Control

Figure 6 shows the open loop speed control of a BLDC motor. The duty cycle for a set reference speed is estimated based on the nominal base speed of the motor.

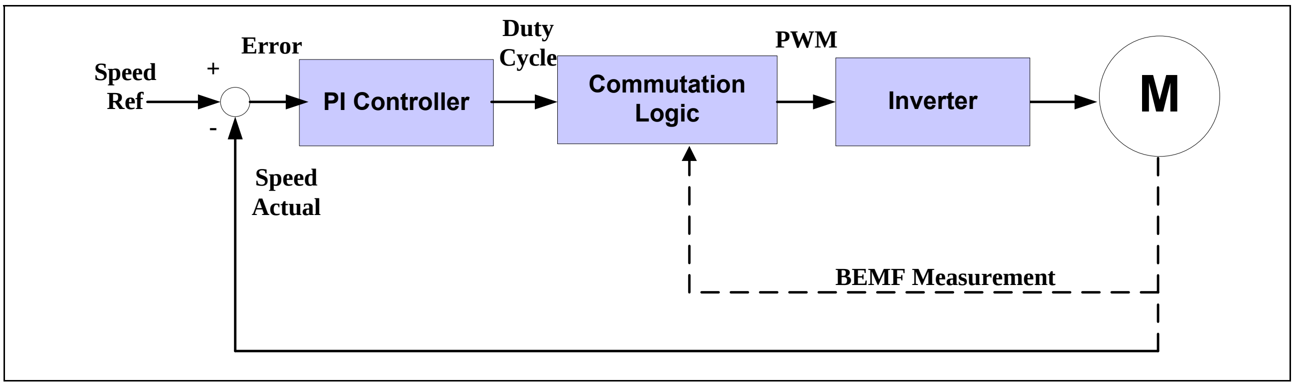


Figure 7 Close Loop Speed Control

3 Software Implementation

The implementation of Sensorless Speed control of a BLDC motor with a XE164F microcontroller and peripherals is discussed in this section.

3.1 Overview

Figure 8 shows an implementation of Sensorless Speed control of a BLDC inverter fed motor in closed loop.

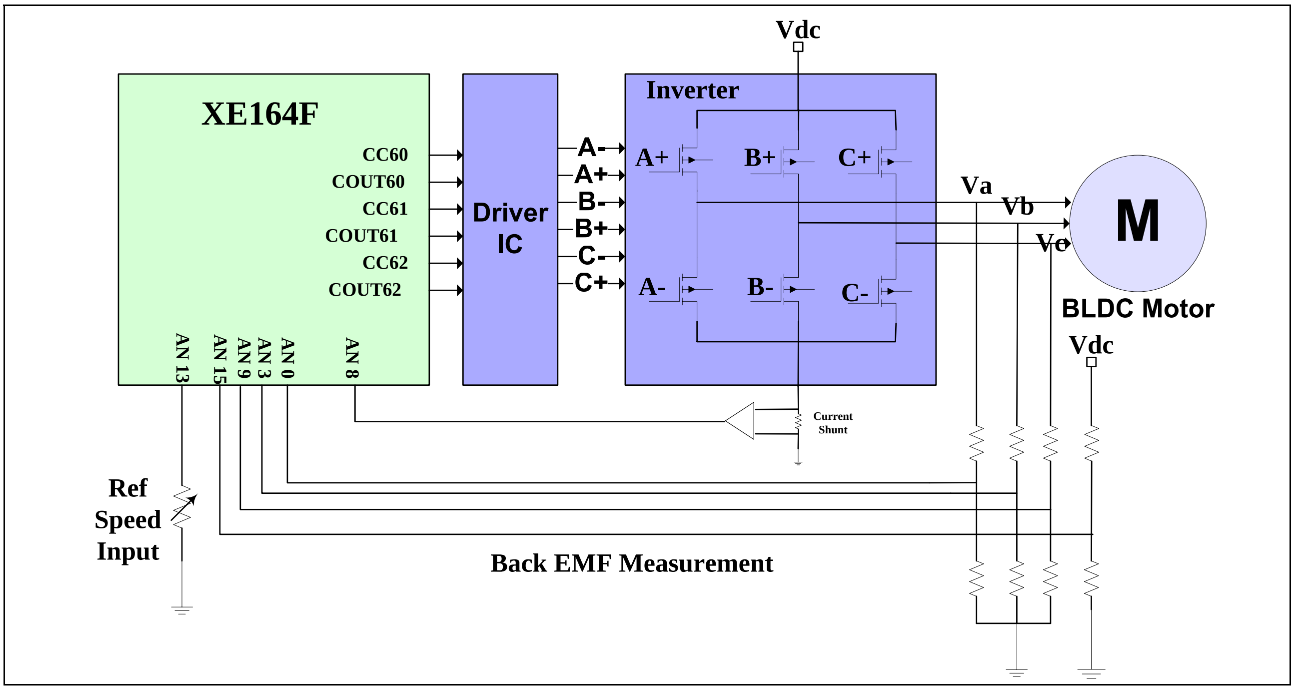


Figure 8 Block Diagram for Sensorless Control of BLDC Motor

Three on-chip peripheral modules are used to implement this application in the XE164 microcontroller at any given time:

- CCU6E (CAPCOM6E)
- ADC (Analog-to-Digital Converter)
- General Purpose Timer Unit (GPT1)

The software is divided into several routines:

Main Loop
Initialization (CPU, I/O ports, CAPCOM6, ADC and GPT1)

Interrupt Routines

- CAPCOM6
  - T13 Period Match
  - T13 Compare Match
  - T12 Compare match of Channel 2
  - T12 Period Match and CTRAP
- GPT1
  - Timer 2 Over Flow

All the parameters such as voltage, frequency, current and speed used in this algorithm are represented in the microcontroller in 1Q15 format; i.e. bit15 is the sign bit and bit14-bit0 refers to the value.

The equation for scaling is as follows:

$$Target\_Value = \frac{Actual\_Value * 2^{15}}{Ns} \qquad .....(3.1)$$

Where:

Target_Value - Value passed to the microcontroller

Actual_Value - Physical value

Ns - Normalization value (maximum physical value)

The Normalization value is the maximum physical value usable in the microcontroller without overflow.

3.2 Peripheral Configuration

All the necessary initialization routines have been performed, before the motor is started.

3.2.1 Port Initilization

P0.0 is used as output to Enable / Disable the Drive Board.

3.2.2 CAPCOM6 Initilization

The CCU60 module is used to generate the PWM control signals for the inverter. For this purpose the timers T12 and T13, the compare registers CC60SR, CC61SR and CC62SR, and the MCMOUT registers, are used.

Timer T12 and Timer T13 operation are configured for edge aligned Mode, and Timer T13 is also used for pulse width modulation to control the motor speed. Dead-time control is enabled for the six PWM signals to avoid shoot-through current.

- P10.0, P10.1, P10.2, P10.3, P10.4 and P10.5 are used as outputs for the CCU60 Channel Port pins (CC6x, COUT6x)
- Enable Multi-channel mode
- Set the passive output level as High (Based on Drive Board)
- Timer T13 Period value set to 50 µs (20 kHz)
- Timer T12 configured for Edge aligned mode
- MCMOUT register shadow transfer enabled during CCU61 compare match with optional synchronization on T13 zero matches.
- Trap function is enabled for emergency stop.

3.2.3 ADC Initilization

The ADC0 module is used to measure the induced EMF in the non energized phase, motor current, DC link voltage and the speed reference.

- The measured induced EMF value of Phase A (in channel 0), Phase B (in channel 3) and Phase C (in channel 9) are stored in result register 3.
- Channel 15 is configured to read the DC link voltage and the result is stored in result register 0.
- Channel 8 is configured to measure the motor current and the result is stored in result register 0.
- Channel 13 is configured to read the speed reference value via POT and the result is stored in result register 1.
- Select P5.0,P5.3, P5.9 (Channel 0,3,9) as AD channels, 10bit, conversion time 3.7 µs, Arbitration Parallel Source for measuring the phase voltages.
- Select P5.8, P5.13, P5.15 (Channel 8,13,15) as AD channels, 10 bit, sampling at T13 Period match, conversion time 3.7 µs, Arbitration Sequential Source 0 for measuring motor current, reference speed and DC link voltage respectively.

3.2.4 GPT1 – Timer 2 Initilization

In the GPT1 Timer 2 overflow Interrupt Service Routine (ISR), the reference speed value is calculated based on the POT input.

- Timer 2 is configured in timer mode, Count up control
- Start Timer2 after initialization
- Timer 2 Overflow value is 1 mS

After peripheral initialization, the motor initialization function is called which initializes motor control specific variables, and the motor start function is called to start the motor.

3.3 Interrupt Service Routines (ISRs)

In this implementation, six ISRs are used to execute motor control specific functions. The Interrupt priorities and function calls are listed in the following table.

Table 3 ISR Priority and Function Call

No	ISR	Period	Inerrupt Configuration		Function Call
			Group	Level	
1	T13 Period Match	50uS	1	6	Commutation Speed Ramp-up PI Controller Duty Cycle Update
2	T13 Compare Match	50uS	1	6	Channel Selection Current CV Measurement
3	T12 Compare Match on Channel 2	Commutation Time*	0	6	Speed Calculation
4	T12 Period Match	254mS	2	7	Motor Stop
5	CTrap	-	2	7	Motor Stop
6	GPT1 Timer2 Overflow	1ms	0	4	Read Speed

$$^{*}\text{Commutation Time} = \frac{60}{\text{MotorSpeed[RPM]} * \text{PolePair} * 6} \qquad \text{.....(3.2)}$$

3.4 CCU60 T13 Period Match ISR

This interrupt routine is executed for every 50 µs (PWM frequency is 20 K). The Commutation function is called and if the motor is running in closed loop (Sensorless Mode) Speed Ramp up, then PI controller and Duty Cycle Update functions are also called.

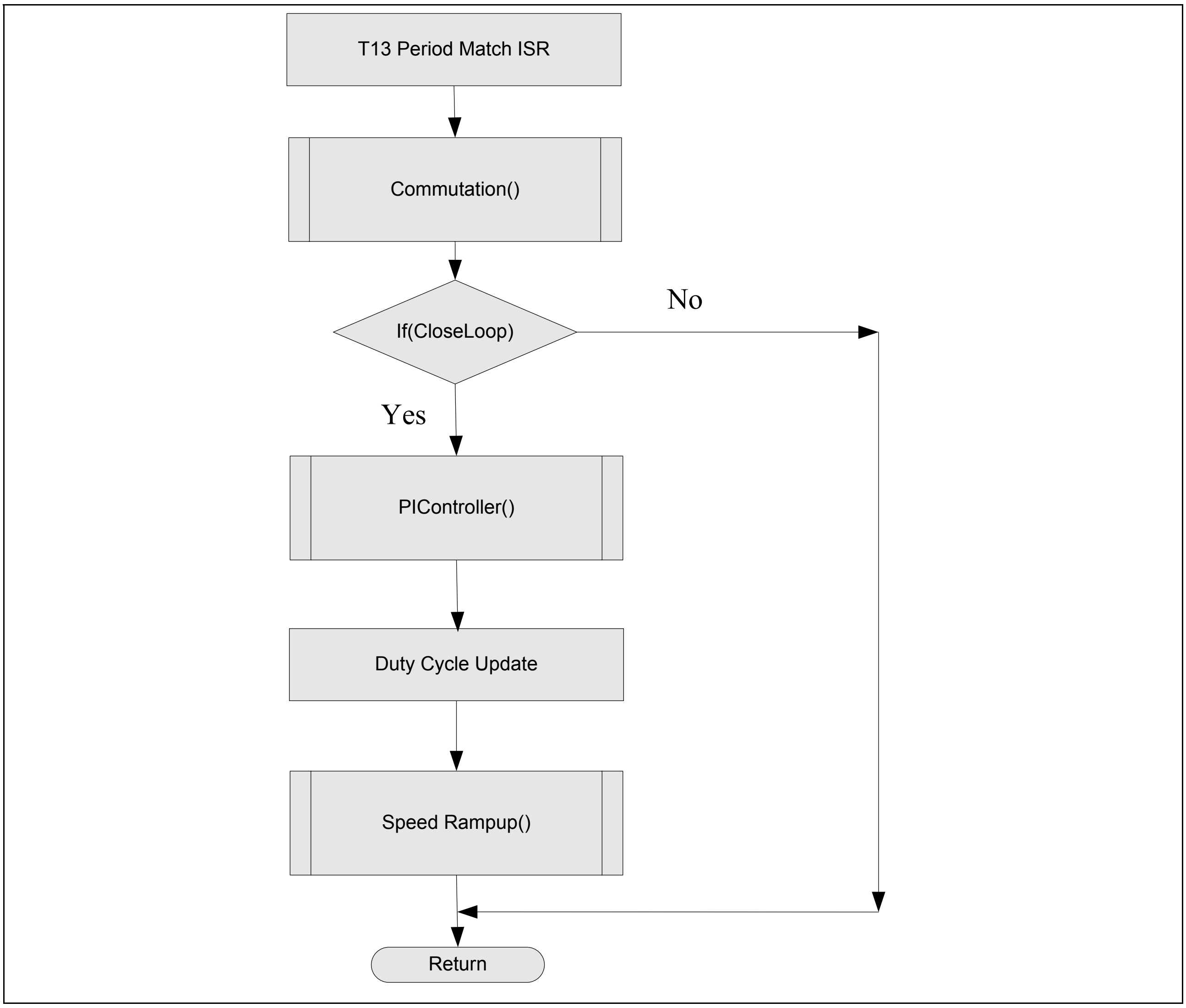


Figure 9 Flowchart CCU60 T13 Period Match ISR

3.4.1 Commutation Function

Back-EMF detection and commutation are implemented in the Commutation function.

Once the Motor Start function is called, the motor will start to run in Open Loop mode. During this phase, the commutation speed and the phase voltage are increased continuously until the Back-EMF voltage is interpretable. The application then switches to the closed loop mode and the motor is accelerated until it reaches the reference speed.

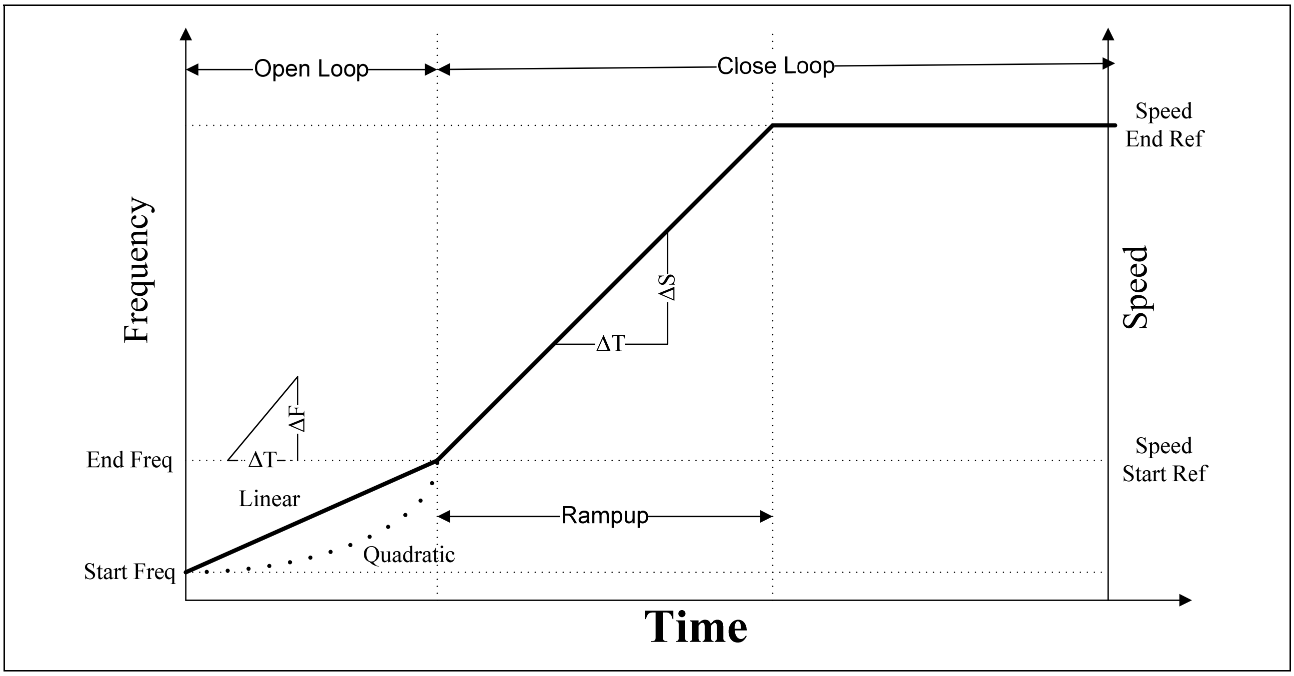


Figure 10 Motor Behaviour during Open and Close Loop

3.4.1.1 Open Loop Mode

When the motor is started, there will not be any Back-EMF. It is necessary to control the motor in an open loop configuration until sufficient Back-EMF has been generated. In this mode, the motor is started with a particular speed based on a start frequency value and the initial applied voltage is defined by a start duty cycle value. Then the applied voltage and commutation speed are increased based on a voltage increment value and a frequency increment value respectively.

The voltage increment and frequency increment values need adjustment based on the motor and load conditions. If the load is higher, then the voltage increment or frequency increment value should be adjusted to drive this load. This can be achieved by increasing the voltage increment value or decreasing the frequency increment value. Once the motor reaches the speed corresponding to the Back-EMF check frequency value, the actual Back-EMF value is checked. If the Back-EMF value is interpretable, then control will switch to close loop.

The actual frequency, voltage increment and frequency increment values used in the software are scaled. The following equation gives the relationship between actual and target values:

$$\mathbf{Frequency[T]} = \frac{\mathbf{Frequency[A] * f_{pwm}}}{\mathbf{128}} \qquad \text{.....(3.3)}$$

Where:

Frequency [T] - Target Value

Frequency [A] - Actual Value

F_{pwm} - PWM frequency [20kHz]

Normalization value (Maximum actual value) for frequency is around 200 Hz for 20 kHz PWM frequency.

$$\mathbf{Freq\_Increment[T]} = \frac{\mathbf{128}}{\mathbf{Freq\_Increment[A]}} \qquad \text{.....(3.4)}$$

Where:

Freq_Increment [T] - Target Value

Freq_Increment [A] - Actual Value

$$\mathbf{Volt\_Increment[T]} = \frac{\mathbf{V_{dc} * (F_{pwm})^2}}{\mathbf{F_{CPU} * Volt\_Increment[A]}} \qquad \text{.....(3.5)}$$

Where:

V_{dc} - DC link Voltage

F_{pwm} - PWM frequency [20kHz]

F_{cpu} - CPU frequency [66MHz]

Volt_Increment [T] - Target Value

Volt_Increment [A] - Actual Val

T13 Pre scaler - 1

$$\mathbf{Start\_Duty\_Cycle[T]} = \frac{\mathbf{Start\_Duty\_Cycle[T] * F_{cpu}}}{\mathbf{F_{pwm}}} \qquad \text{.....(3.6)}$$

Where:

Start_Duty_Cycle [T] - Target Value

Start_Duty_Cycle [A] - Actual Value

Table 4 Startup Parameter Value

Parameter	Physical value	Unit	Target Value
BEMF Start Frequency	5	Hz	781
BEMF Check Frequency	36	Hz	5625
BEMF End Frequency	40	Hz	6250
Frequency Increment	7	Hz/S	144
Voltage Increment	1	V/S	106
Start Duty Cycle	5	%	165

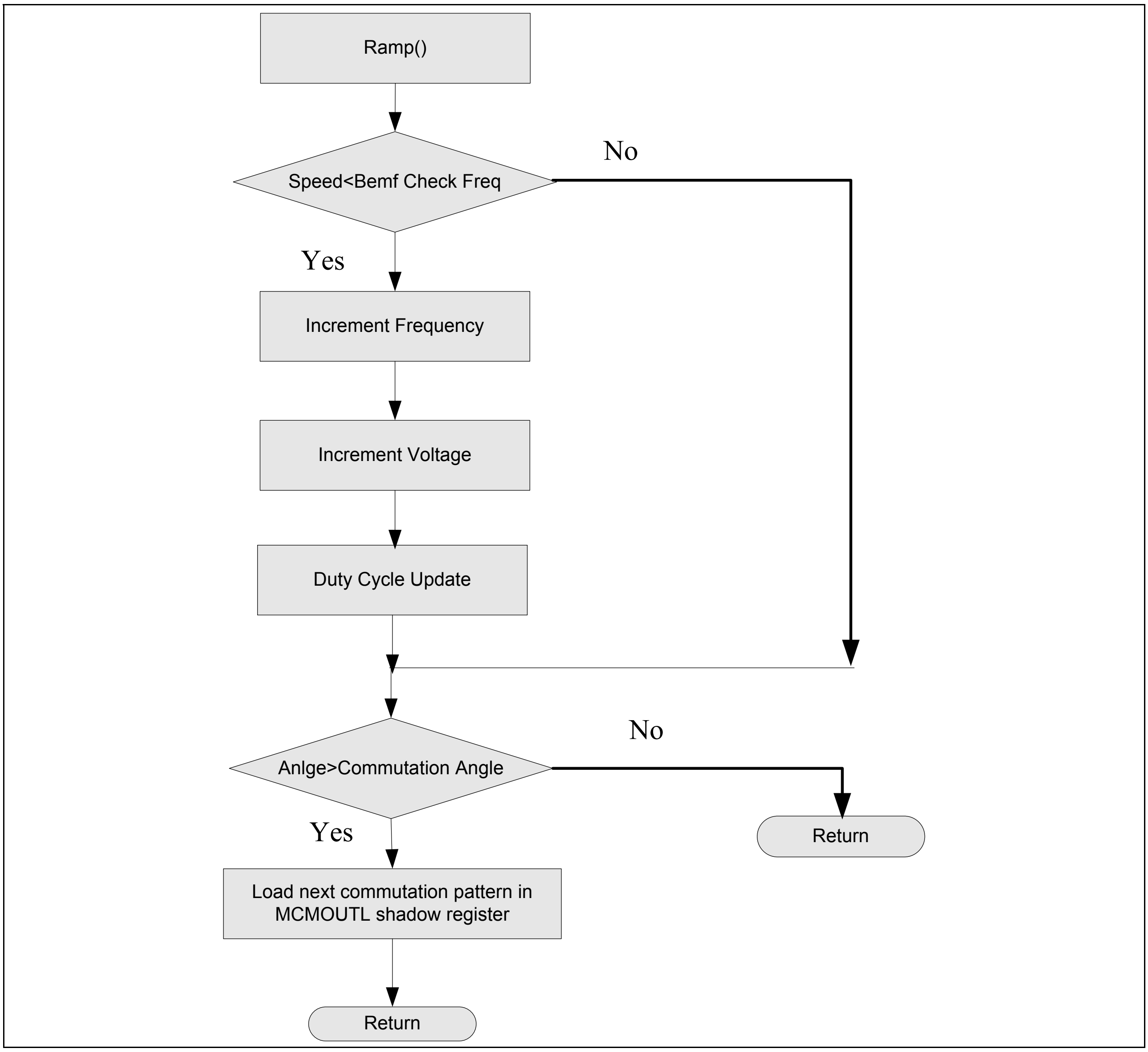


Figure 11 Flowchart for Rampup Function

3.4.1.2 Close Loop Mode (Sensorless Mode)

In this mode, the commutation time is calculated based on the induced EMF in the inactive phase. The motor speed will rampup from the start speed reference to the user speed.

As discussed in section 2.5, the Back-EMF measurement should be synchronized with the PWM signal used for chopping. In the implementation Timer T13 is used for chopping, so the unexcited phase voltage is measured during every CCU63 compare match event.

When a new commutation pattern has been loaded into the MCMOUT register, the unexcited phase voltage is measured for every CCU63 Compare ISR via ADC.

Demagnetization spikes will occur whenever a new commutation pattern is applied. This spike will affect the Back-EMF and may be interpreted as a zero crossing event. In order to ignore this spike, zero crossing detection is ignored for a predefined delay time after applying every new commutation pattern.

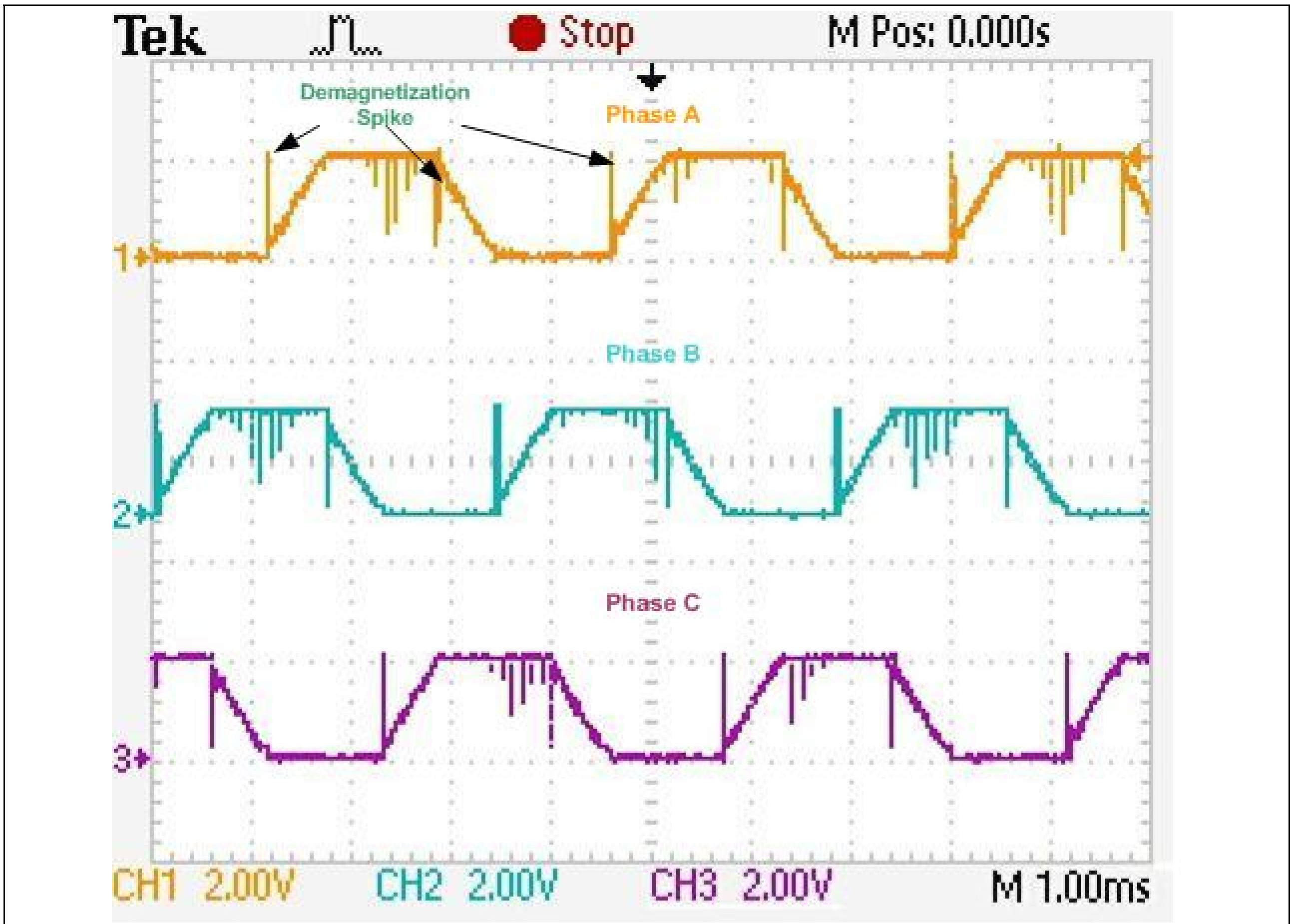


Figure 12 Phase Voltage at 100% Duty Cycle

If the voltage values on two consecutive measurements are greater than the zero crossing value for positive slopes (slope =1), or less than zero crossing value for negative slopes (slope =0), then timer T12 will be stopped and the timer value is captured. The commutation pattern is then updated in the MCMOUT register after half of the T12 timer value. The following steps are required to accomplish this:

- Half of the T12 timer value should be loaded into the CCU61 compare register
- Timer T12 should be reset and started again

The MCMOUT shadow transfer will happen during the CCU61 compare match event, and the next commutation pattern is loaded into the MCMOUT shadow register after the MCMOUT shadow transfer.

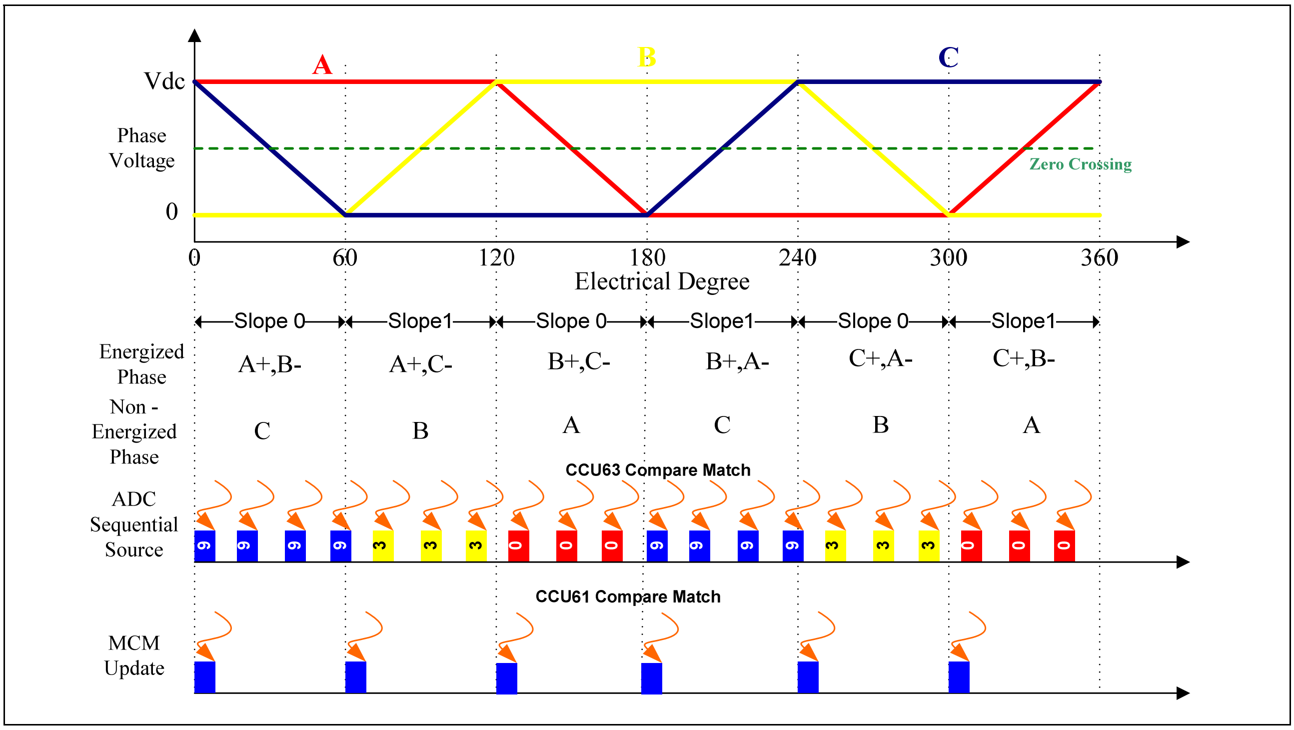
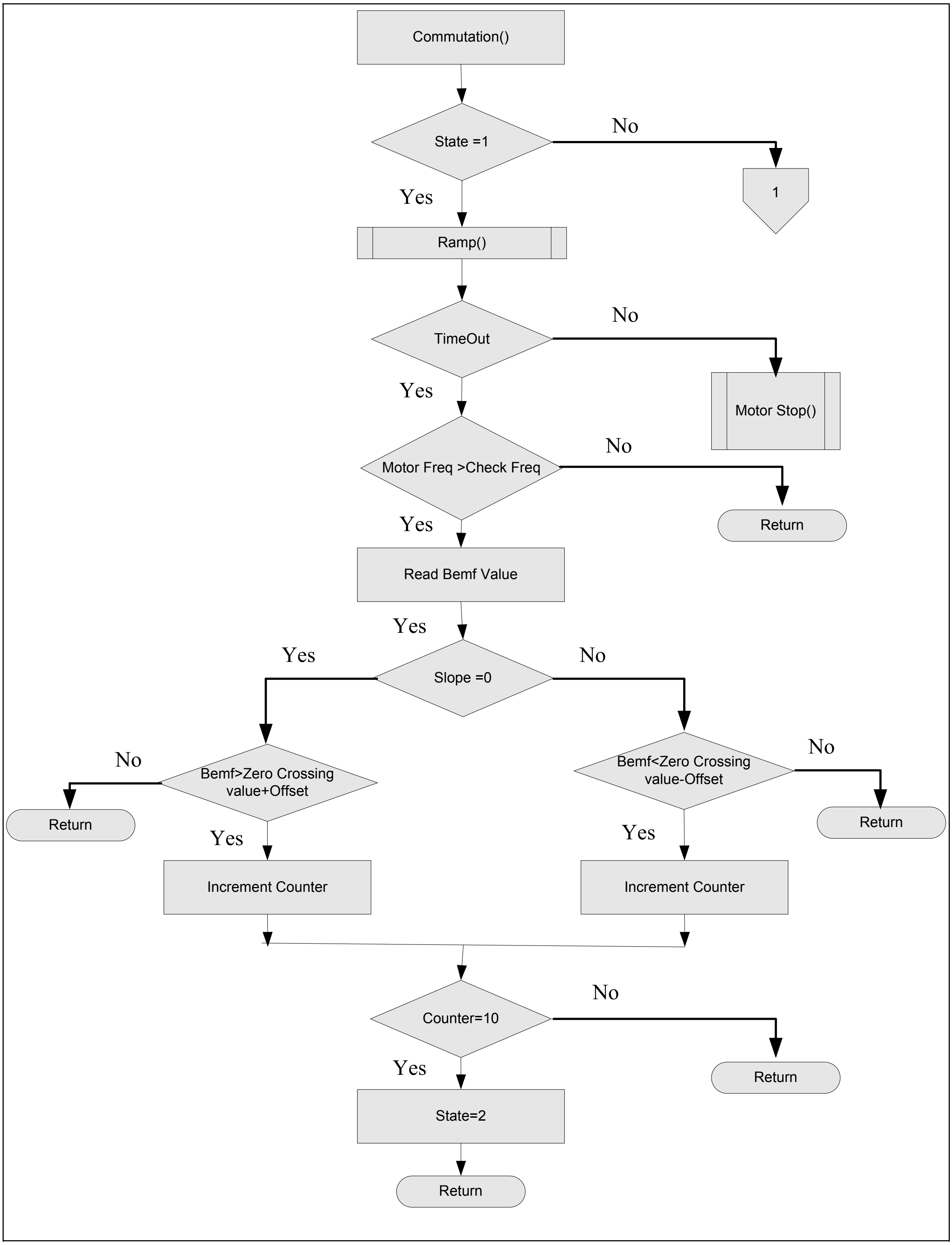


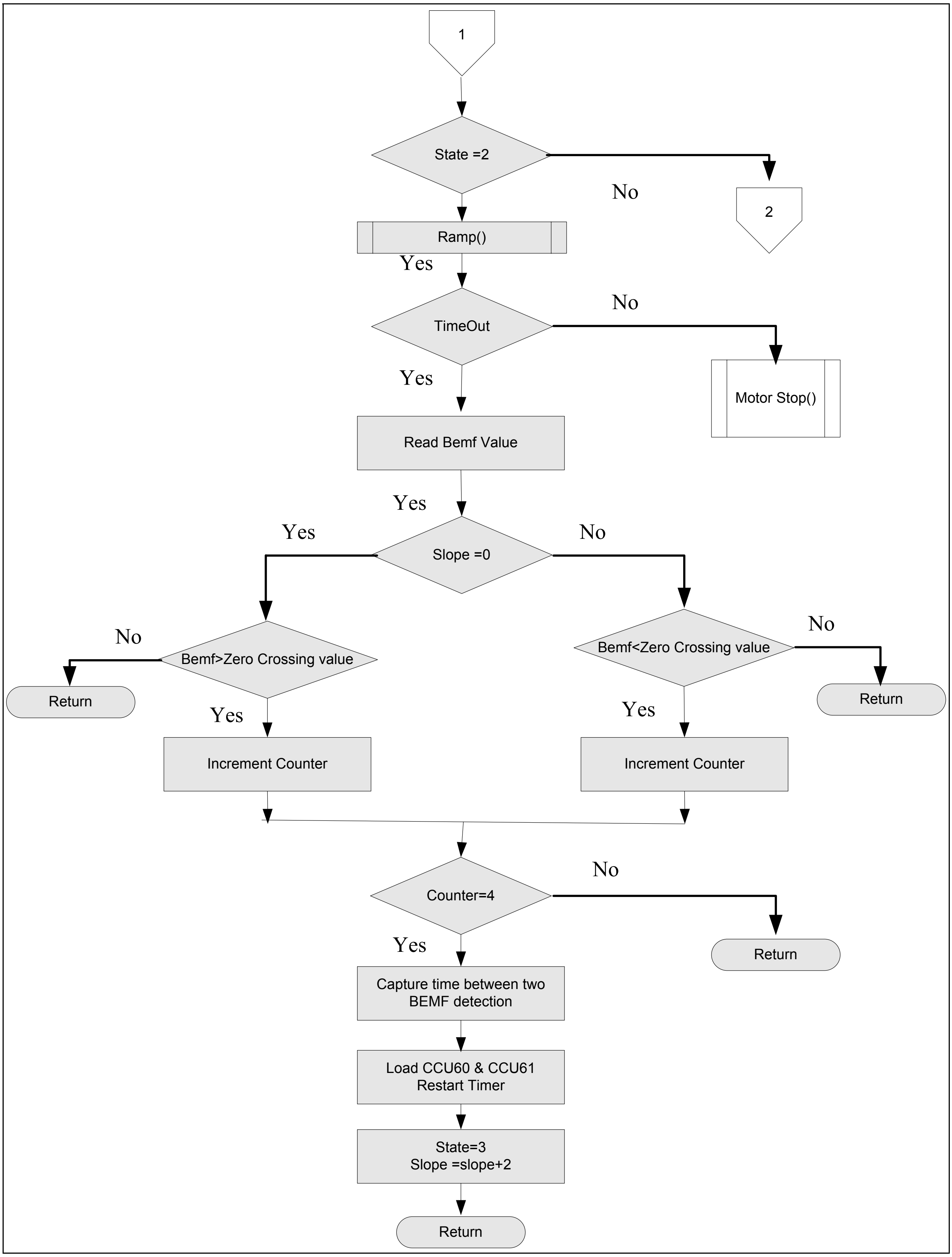
Figure 13 BEMF Detection Timing Diagram

This function can handle 4 different operation modes

Table 5 Motor Operation Modes

State	Action
1	Open Loop - Rampup Phase for BEMF Detection
2	Start of time Between two Zero Crossing
3	Normal Running Mode
4	Turn off Motor


Commutation()
State =1
No
Yes
1
Ramp()
TimeOut
No
Yes
Motor Stop()
Motor Freq >Check Freq
No
Yes
Return
Read Bemf Value
Yes
Yes
Slope =0
No
Bemf>Zero Crossing value+Offset
No
Return
Yes
Increment Counter
Bemf<Zero Crossing value-Offset
No
Return
Yes
Increment Counter
Counter=10
No
Yes
Return
State=2
Return


1
State =2
No
2
Ramp()
Yes
TimeOut
No
Motor Stop()
Yes
Read Bemf Value
Yes
Slope =0
Yes
No
Bemf>Zero Crossing value
No
Return
Yes
Increment Counter
Bemf<Zero Crossing value
No
Return
Yes
Increment Counter
Counter=4
No
Return
Yes
Capture time between two BEMF detection
Load CCU60 & CCU61 Restart Timer
State=3
Slope =slope+2
Return

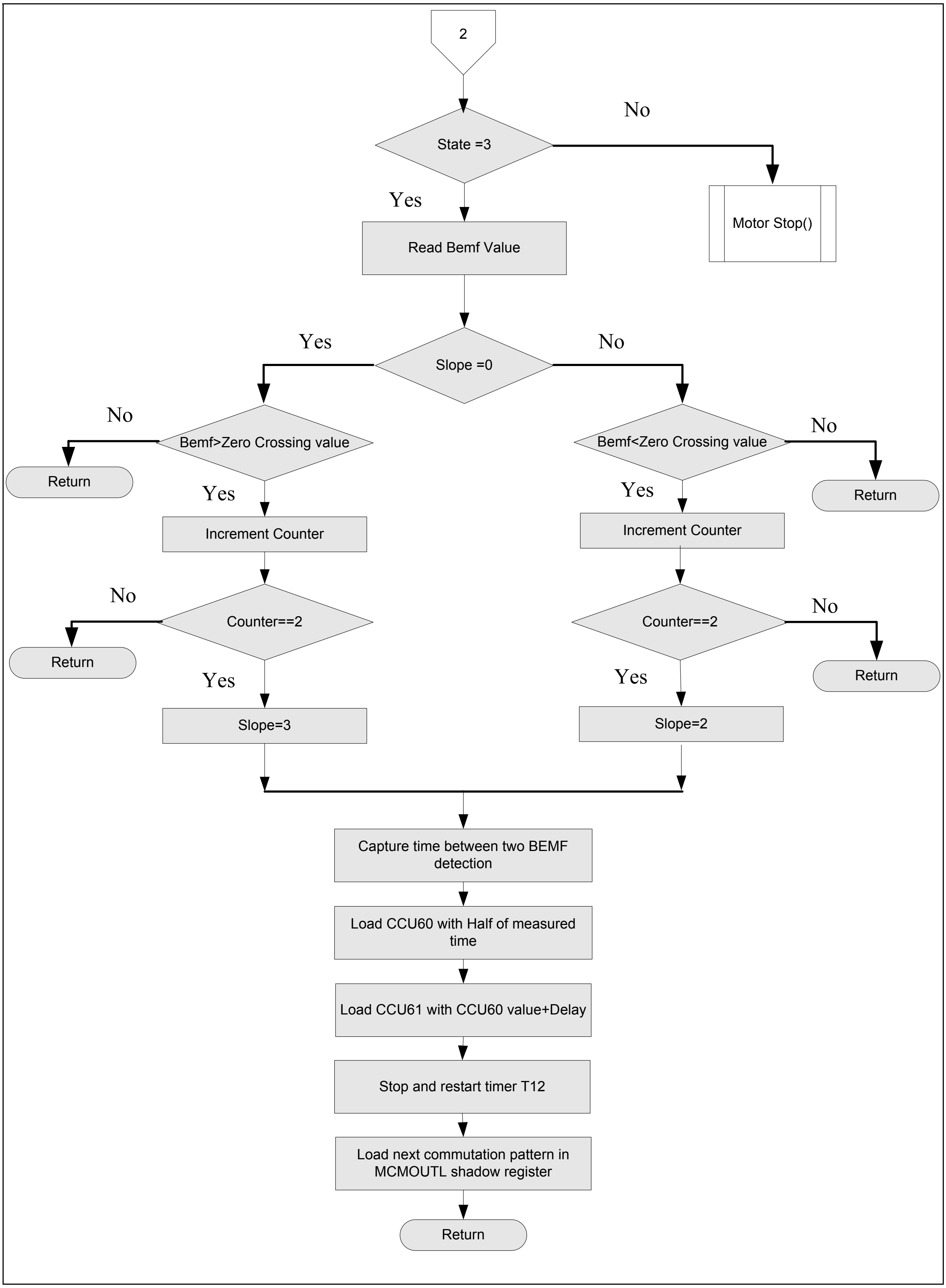


Figure 14 Flowchart for Commutation Function

3.4.2 Speed Rampup Function

In the Speed Rampup function, the user reference speed value is determined by user input via POT.

The motor reference speed is gradually increased/decreased up to the user reference speed, with the rate of speed increase/decrease based on the speed slew rate (RPM/Second).

The speed slew rate (RPM/Second) is based on the function call rate and the ramp scheduler value. The function call rate is fixed for a particular PWM frequency; for 20 kHz the function call rate is 50 µs. So the ramp scheduler value is calculated from the required speed slew rate and PWM frequency.

$$\textbf{RampScheduler} = \frac{\textbf{1}}{\textbf{Required Slew Rate} * \textbf{Function Call Rate}} \qquad \textbf{.....(3.7)}$$

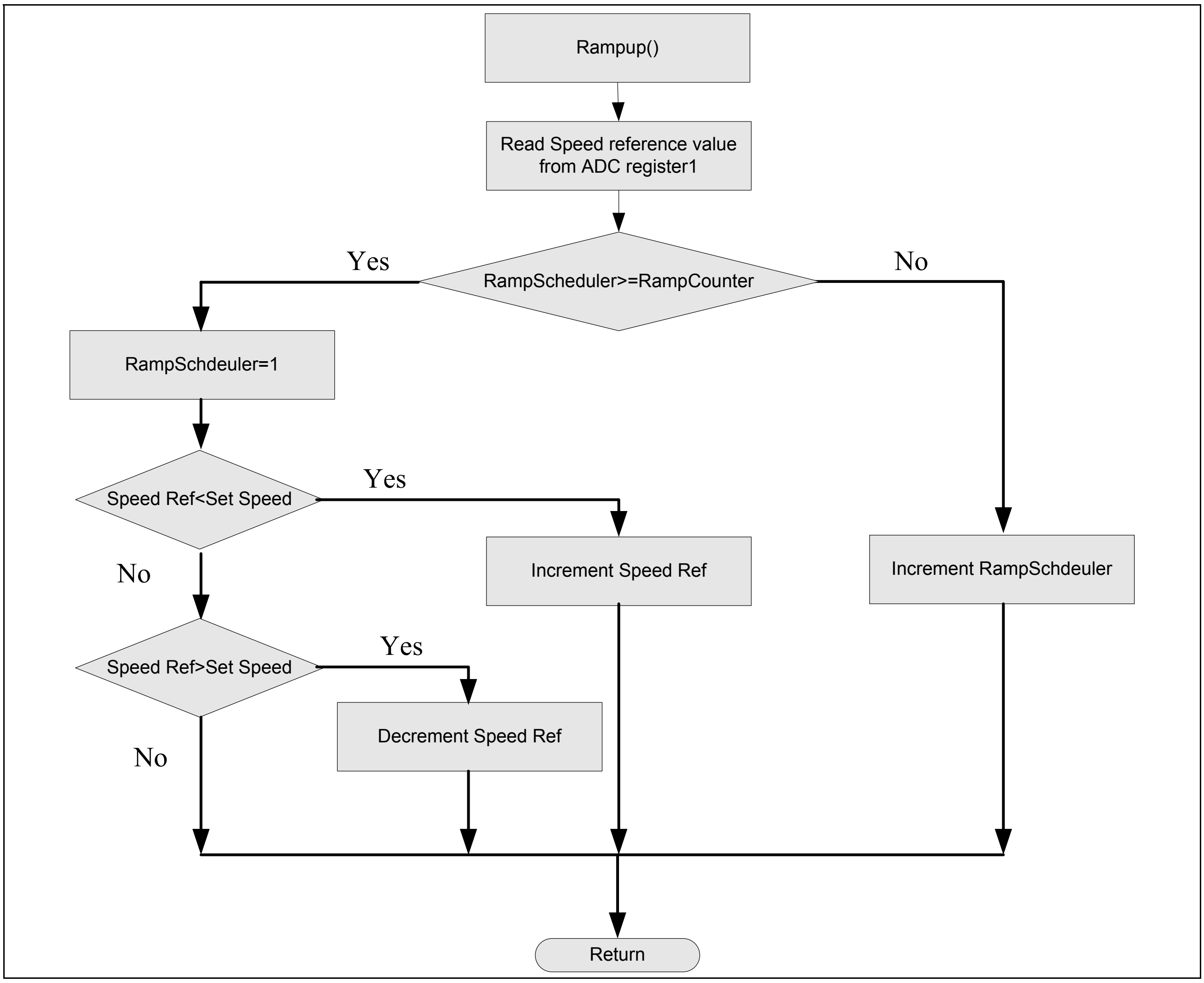


Figure 15 Flowchart for Speed Rampup Function

3.4.3 PI Controller Function

A PI controller is used for regulating speed. The error difference between the reference speed and the actual speed is fed to the controller. The PI controller functionality is shown in Figure 16.

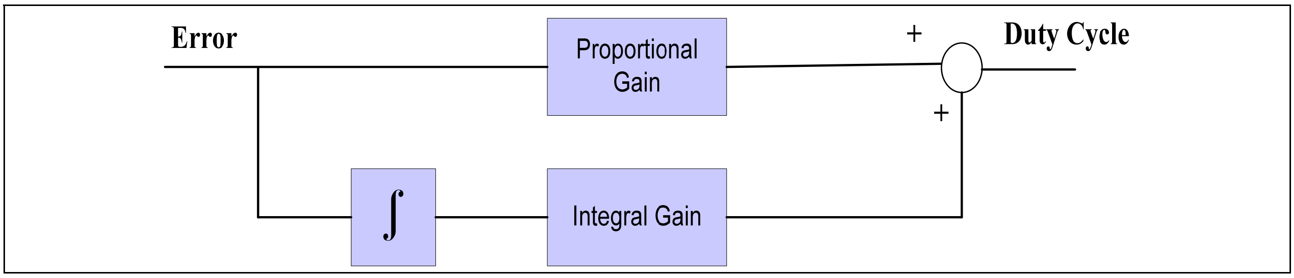


Figure 16 Block Diagram for PI controller

In continuous time domain, the duty cycle output is given by:

$$\textbf{Duty Cycle} = K\ \text{error} + K \int \text{error}\ dt \qquad \text{.....(3.8)}$$

In discrete time domain, the PI controller is implemented as described by the following equations.

$$Yn(k+1) = Yn(k) + Ki * e(k)$$

$$Y(k+1) = Yn(k+1) + Kp * e(k) \qquad \text{.....(3.9)}$$

Where:

Ki - Integral Gain

Kp - Proportional Gain

e(k) - Error value

y(k+1) - Next computed duty cycle

yn(k) - Integrated error value till last computation

yn(k+1) - Current Integrated error value

The actual Kp and Ki values are scaled and will be used in target as follows:

$$Kp[T] = kp * 2^{15} / 64$$

$$Ki[T] = ki * 2^{15} \qquad \text{.....(3.10)}$$

Where:

Kp[T] and ki[T] are the Scaled Proportional and Integral Gain values used in software.

The PI controller functionality implementation is written in assembly, and uses the microcontroller MAC unit functionality. The PI controller parameters are given to the function over structure:

Code Listing 1 PI Controller Parameters

```
Struct
{
    uword   ki;            //Ki Value
    uword   kp;            //Kp Value
    uword   Ymin;          //PI mimimum output value
    uword   Ymax;          //PI maximum output value
    slong   Ibuf;          //Integral Buffer
    uword   PI_Output;     //PI output
}PI_Array;
```

The reference value and the actual value are given directly to the PI controller function. The values are represented in 1Q15 format.

Code Listing 2 PI Controller Code using MAC

```
Uword PIControllerSpeed(uword *PI_Parameter, uword Actual, uword Ref)
{
#pragma asm
MOV     R12,MCW         ;Save MCW register
MOV     MCW,#1536       ;Set saturation and shift left
MOV     R11,ZEROS       ;Load Zero to R11
CoLOAD  R11,R10         ;Load accumulator(High) with Referenc
CoSUB   R11,R9          ;error =reference -actual
CoSTORE R9,MAS          ;Load error in R9
MOV     R1,[R8+]        ;Load Kp value to R1
MOV     R2,[R8+]        ;Load Ki value to R2
MOV     R3,[R8+]        ;Load Ymin value to R3
MOV     R4,[R8+]        ;Load Ymax value to R4
MOV     R5,[R8+]        ;Load Integral buffer(low) value to R5
CoLOAD  R5,[R8]         ;Load Integral buffer to accumulator
CoMAC   R2,R9           ;Yn =Ki*error+Yn
CoMIN   R11,R4          ;Limit MAX Yn
CoMAX   R11,R3          ;Limit MIN Yn
CoSTORE R6,MAH          ;Store Yn(high) in R6
CoSTORE R5,MAL          ;Store Yn (low) in R5
MOV     [R8],R6         ;Store R6 in Integral Buffer (high)
MOV     [-R8],R5        ;Store R5 in Integral Buffer (low)
CoMUL   R1,R9           ;Kp*error
CoSHL   #6              ;64*kp*error
CoADD   R5,R6           ;Y =Yn + 64*kp*erro
CoMIN   R11,R4          ;Limit MAX Yn
CoMAX   R11,R3          ;Limit MIN Yn
CoSTORE R4,MAS          ;Store y-high in R4 (return register)
MOV     MCW,R12          ;Restore MCW register value
#pragma endasm
}
```

PI Controller will command a duty cycle value to achieve the required speed. The duty cycle value is updated into CCU63 compare register.

3.5 T13 Compare Match ISR

The T13 Compare Match ISR is executed for every 50µs (if PWM frequency is 20K). During this ISR the channel selection and compare and current measurement functions are also called

3.5.1 Channel Selection Function

This function is used to find non-energized Phase winding and to select the appropriate ADC channel to measure the induced voltage.

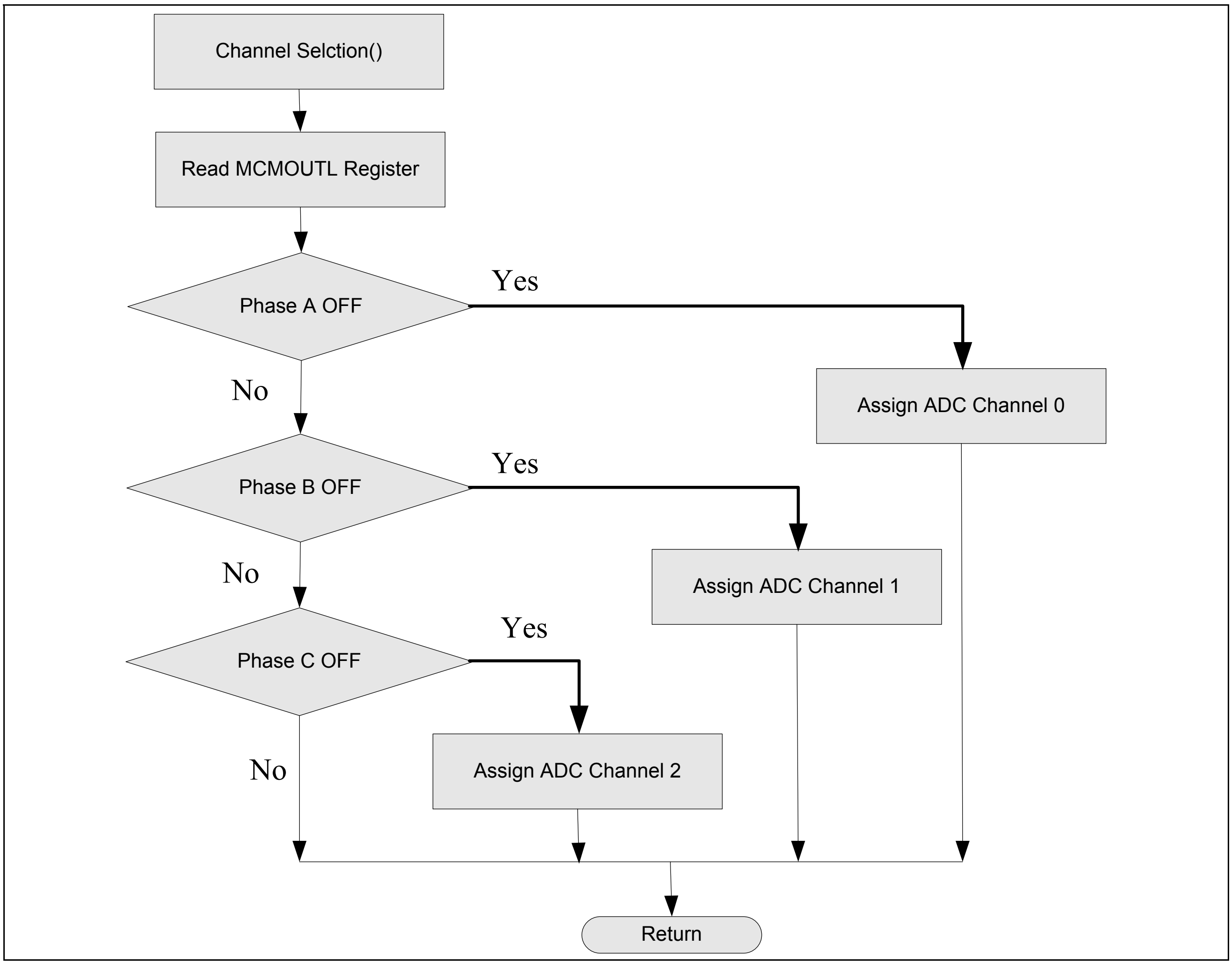


Figure 17 Flow Chart for Channel Selection Function

3.5.2 Compare & Current Measurement Function

During this function call, the motor current value or the DC link voltage value is read from the ADC result register. Over current protection is also implemented in the software. If the motor current value exceeds the set limit value, the motor will be stopped.

The maximum current range is defined as:

$$I_{max} = \frac{V_{adcref}}{R_{shunt} * G_{op}} \qquad .....(3.11)$$

Where:

V_{adcref} - ADC reference Voltage

R_{Shunt} - Current shunt resistor value

G_{OP} - Amplifier gain

In this implementation the 10 bit ADC value is multiplied by 8. The current scaling is:

$$N_I = \frac{I_{max} * 2^{15}}{8 * 2^{10}} \qquad .....(3.12)$$

3.6 CCU62 Compare Match ISR

During this ISR the Speed calculation function is executed. The speed calculation needs the time between zero crossing values. The time will be determined by Timer12 (CAPCOM6E). On every zero crossing, the Timer T12 will be stopped and the time between zero crossing values is stored in a circular memory.

To reduce the measurement errors, the time between two zero crossing events is averaged over (6 * Pole pairs) measured values. The time taken by the motor to complete one rotation can be calculated from the sum of (6 * Pole pairs) measurement and the timer T12 resolution.

$$T_{Speed} = T_{(n+0)} + T_{(n+1)} + ... + T_{(6*PolePair-1)} \qquad .....(3.13)$$

Where:

T_n - Time between two zero crossing events

The speed is calculated using the formula:

$$MotorSpeed = \frac{60}{T_{speed} * T12\,Resolution} \text{ [RPM]} \qquad .....(3.14)$$

The step size of Timer T12 will determine the range of speed values that can be measured.

3.7 T12 Period Match & CTRAP ISR

If the CCU6 trap input becomes active or if the T12 Period match (Timeout) occurred, the motor will be stopped for protection purposes. The ISR Motor Stop function is called during this function.

3.8 GPT1 Timer 2 Overflow ISR

This interrupt routine will be called every 1mS. During the ISR, the speed reference value is calculated based on the POT input.

4 Conclusion

This application note describes the implementation of the Back-EMF Sensorless algorithm for BLDC motors using the Infineon XE164F microcontroller. This software solution consumes only very limited CPU resources because of the high performance of the microcontroller and its dedicated peripherals for BLDC motor control.

5 Reference

1. XE1666 User’s Manual
2. AP16160, XE164 DriveCard Hardware Description Board
3. AP9001. Low Voltage 3phase Inverter with MOSFETs and 6ED driver - Hardware Description
4. AP08071, Hardware and Software Description DriveMonitor
5. AP16155, ADC Result Handling on XC2000/XE166 family of Micrcontrollers
6. AP16117, Speed Control of Brushless DC Motor with Hall Sensor Using DAVE Drive for Infineon XC164 CS/CM microcontrollers

w w w . i n f i n e o n . c o m

Published by Infineon Technologies AG