Application Note, V 1.0, Oct. 2005

AP16090

XC164CM

Driving of a Brushless DC Motor without Sensors

Microcontrollers

Infineon technologies

Never stop thinking.

Revision History:	2005-10	**V 1.0**
Previous Version:	-	

Page	Subjects (major changes since last revision)

Controller Area Network (CAN): License of Robert Bosch GmbH

We Listen to Your Comments

Any information within this document that you feel is wrong, unclear or missing at all? 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

Edition 2005-10

Published by
Infineon Technologies AG
81726 München, 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 your 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 your nearest Infineon Technologies Office.

Infineon Technologies Components may only be used in life-support devices or systems 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.

Table of Contents **Page**

Table of Contents

1 Introduction

1.1 Why use brushless DC motors (advantages/disadvantages)?

Brushless DC motors are synchronous motors suitable for use as a simple means of controlling permanent drives (e.g. ABS pumps, EHPS pumps, fuel pumps or cooling fans). This type of 3-, 4- or 5-phase brushless DC motor will increasingly replace brushed DC motors. Brushed DC motors require maintenance, e.g. to service carbon brushes and commutator. Another major problem with a brushed DC machine is the possibility of brush burnout in the event of an overload or stall condition.

One of the main advantages of brushless DC motors is their robustness against obsolescence (due to the absence of brushes), another major benefit is the high efficiency and high torque of a brushless DC motor.

1.2 Functional principle of a brushless DC motor

Figure 1 shows a three-phase brushless DC motor with two pole pairs. The rotation of the electrical field (vector) has to be applied twice as fast as the desired mechanical speed of the brushless DC motor. The three coils of the stator are split into two groups of coils (A, B, C and A', B', C'). When a brushless DC motor is energized in block commutation mode, only two of the three phases are active at a given time. Illustrated in Figure 1, coils A and C are energized and coil B is not energized. A 0° to 180° rotation will be shown in detail in section 2.1 to explain the setting of the appropriate switches of the B6 bridge pattern, the appropriate voltages relating to the coils, and the energized coils of the motor with the suitable rotor position between 0° and 180° mechanical.

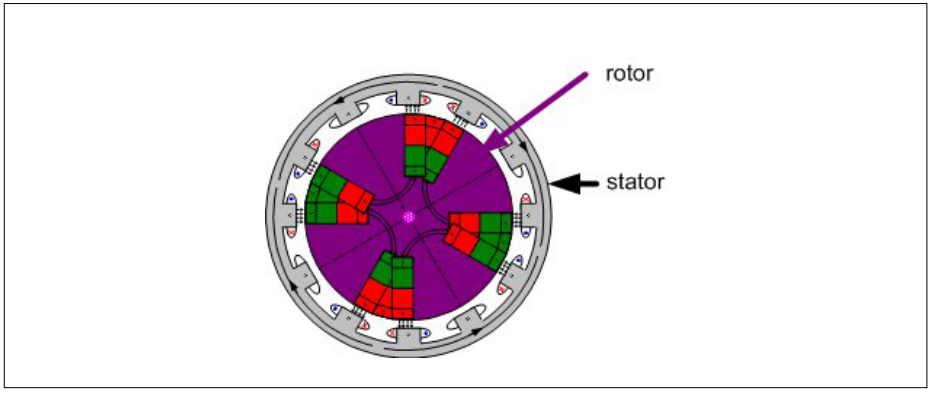


Figure 1 Functional principle of a brushless DC motor

1.3 Brushless DC motor with Hall sensors

A synchronous motor mainly depends on synchronous processes, e.g. ADC measurement has to be absolutely synchronous with the energizing pattern of the coils realized by the switches of the B6 bridge. In this context, the change of state of the three Hall sensors gives an indication of the specific time at which the next pattern of the B6 full-bridge has to be applied. Each of these Hall sensors costs about $0.15 - $0.30, so they have a big impact on the total cost of the whole application. Typically, external hardware is needed, such as three comparators for sensor signal conditioning.

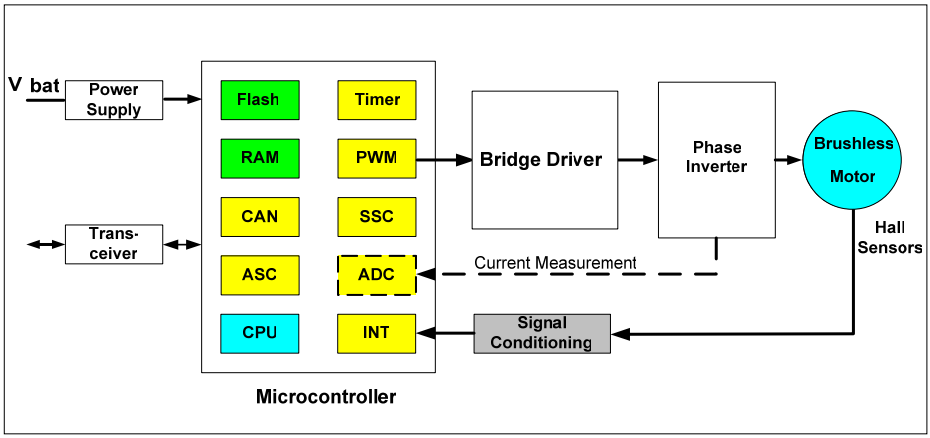


Figure 2 Block diagram for brushless DC motor drive with Hall Sensors

1.4 Brushless DC motor without Hall sensors

The aim of this application note is to eliminate position sensors altogether. The XC164CM 16-bit µC in its small P-TQFP 64 package (12*12 mm incl. pins) fits this type of application very well because of its flexible capture compare unit (CAPCOM6), with two 16-bit timers and 6 + 1 output channels. Many other features are listed at the end of the application. Three channels of the 10-bit ADC of the XC164CM and only a two-resistor voltage divider with a capacitor at each of the three phases are the hardware necessary to realize a viable sensorless brushless DC motor. The most important point – and at the same time a big advantage of this hardware configuration – is the aspect of minimized costs, because even the comparators that would normally be necessary are superfluous. A 16-bit microcontroller delivering high CPU performance, with a fast and accurate 10-bit ADC, has a big advantage at higher speeds because of the very acceptable CPU load even in these conditions. A very important point in relation to the cost-effective back-EMF sensing is the very low

leakage current of the XC164CM ADC (+/- 300 nA), because the back-EMF is low at low speeds and the use of a high-impedance voltage divider is necessary.

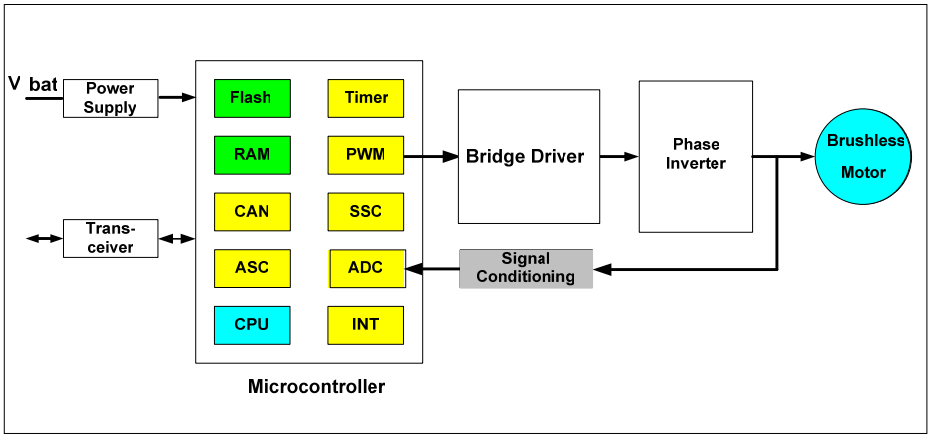


Figure 3 General block diagramm for sensorless BLDC motor drive

The brushless DC motor rotor position has to be evaluated by the zero crossing of the back-EMF which can be evaluated by measuring the non-energized phase. The voltage value of the back–EMF has to be half of the maximum value to detect the precise point for changing the coil-energizing pattern for the following 30° mechanical rotation. It is very important to find the precise point for the next change in the energizing pattern to ensure smooth operation of this type of motor. A big advantage in this context is the fact that the back–EMF is proportional to the mechanical rotation speed. This means that the XC164CM can process directly the 10-bit ADC values without any type of linearization algorithm. This allows the use of very fast regulation software. The amount of back-EMF is a function of the number of coil windings, mechanical speed, and flux ($E = N \times \phi \times n$). Taking account of a low rotation speed, it is very important to have a high resolution with regard to the ADC microcontroller (10-bit +/- 2 LSB minimum) to evaluate the Back-EMF with the application necessary demand. The 16-bit wide code and data core and the MAC unit as part of the XC164CM instruction set e.g. allows fast processing of the 16-bit integer or even PI speed regulator.

2 Operation of a brushless DC motor

2.1 Visualization of the rotation including appropriate switches of a 3-phase brushless DC motor (0° mechanical to 180° mechanical)

Only two of the three phases are energized in each state of the B6 full-bridge. The non-energized phase can therefore be used to detect the zero crossing of the back-EMF as explained later (cf. the black arrow in the voltage diagram shown in Figures 5, 7, 9, 11, 13, 15, 17). The following explanation will show you the turning of a two pole-pair brushless DC motor as a function of the state of the appropriate switches shown in Figures 4, 6, 8, 10, 12, 14, 16. In fact, driving a two pole-pair motor means that the rotation speed of the electrical field has to be double the speed of the desired mechanical rotation speed ($\Omega = \omega / p$, where Ω is the mechanical rotation speed, ω the rotation frequency of the electrical field and p is the number of pole pairs).

2.1.1 Position ‘ 0° ’

Figure 5 shows the voltage-energizing coils C and A (stator) and the permanent magnet realized with two pole pairs (rotor), as well as indicating the zero-crossing detection coil B. Figure 4 shows the active switches of the B6 bridge, and Figure 5 again the appropriate rotor position (0°):

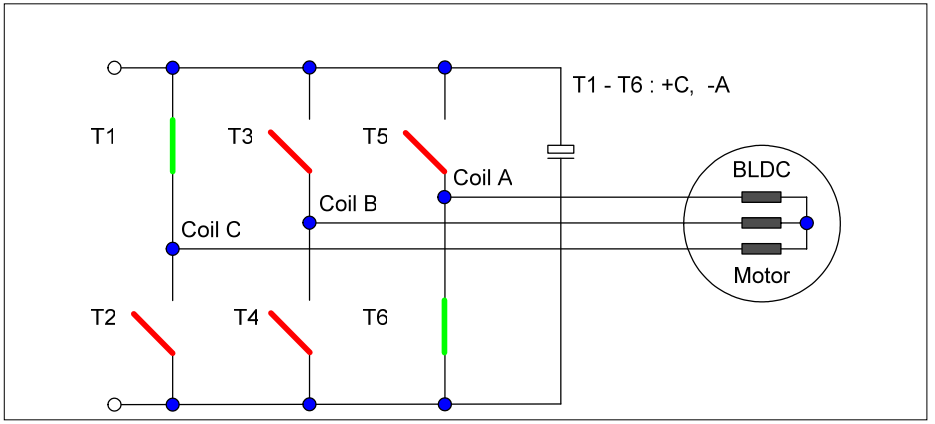


Figure 4 Appropriate switches T1, T6 of the B6 full-bridge (coil c:pos.,coil A:neg.)

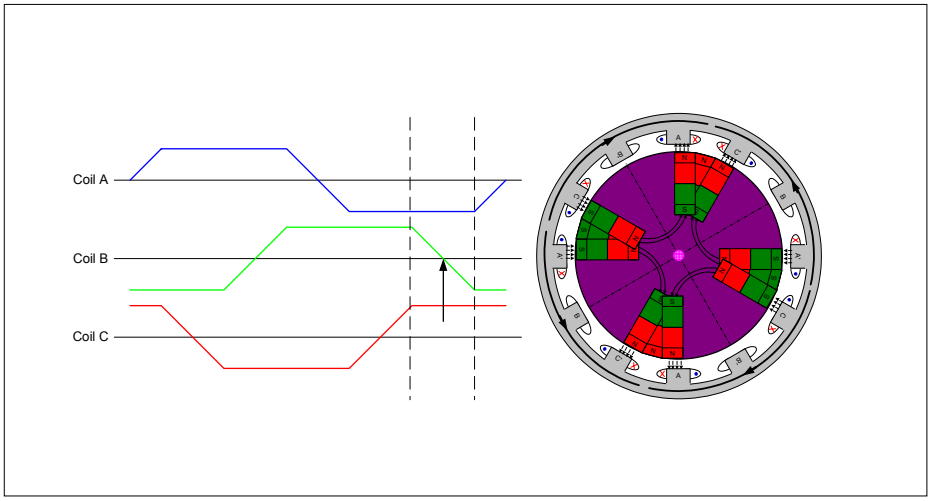


Figure 5 Energized coils C and A of a 2 pole-pair BLDC motor (mechanical position : 0 °)

2.1.2 Position ‘ 30° ’

Figure 7 shows the voltage-energizing coils C and B (stator) and the permanent magnet realized with two pole pairs (rotor), as well as indicating the zero-crossing detection coil A. Figure 6 shows the active switches of the B6 bridge, and Figure 7 again the appropriate rotor position (30°):

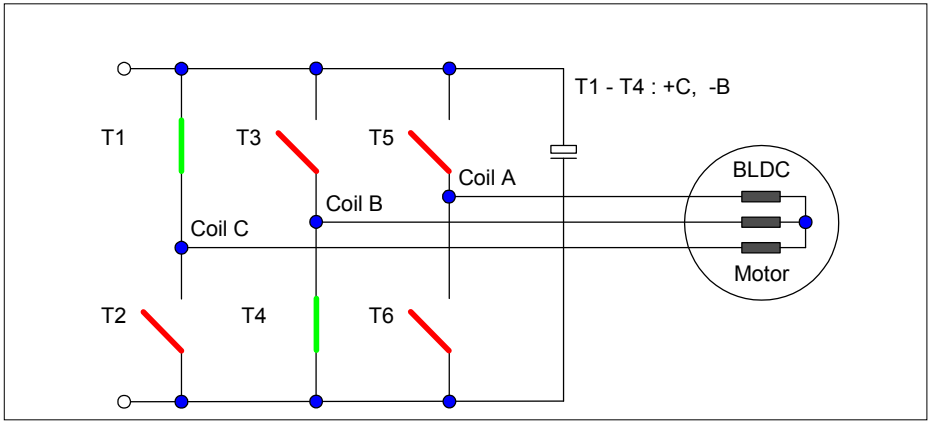


Figure 6 Appropriate switches T1, T4 of the B6 full-bridge (coil C:pos.; coil B:neg.)

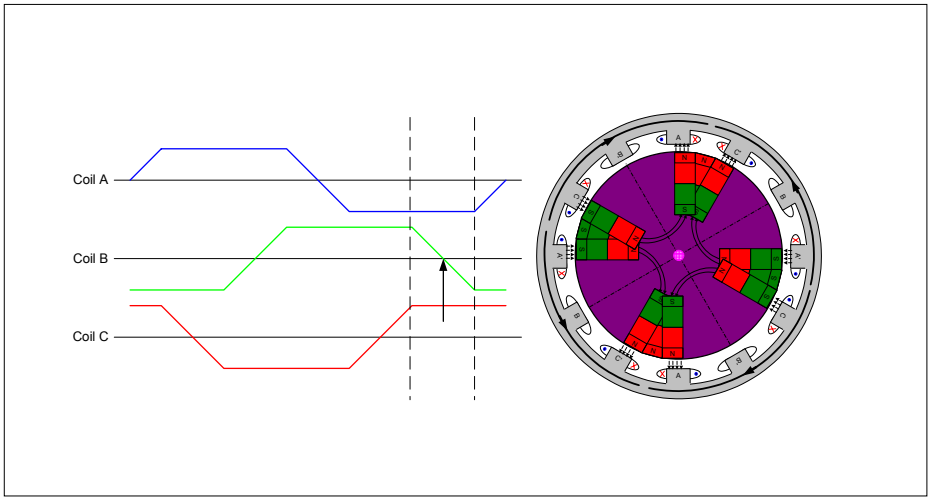


Figure 7 Energized coils C and B of a 2 pole-pair BLDC motor (mechanical position 30°)

2.1.3 Position ‘ 60° ’

Figure 9 shows the voltage-energizing coils C and B (stator) and the permanent magnet realized with two pole pairs (rotor), as well as indicating the zero-crossing detection coil C. Figure 8 shows the active switches of the B6 bridge, and Figure 9 again the appropriate rotor position (60°):

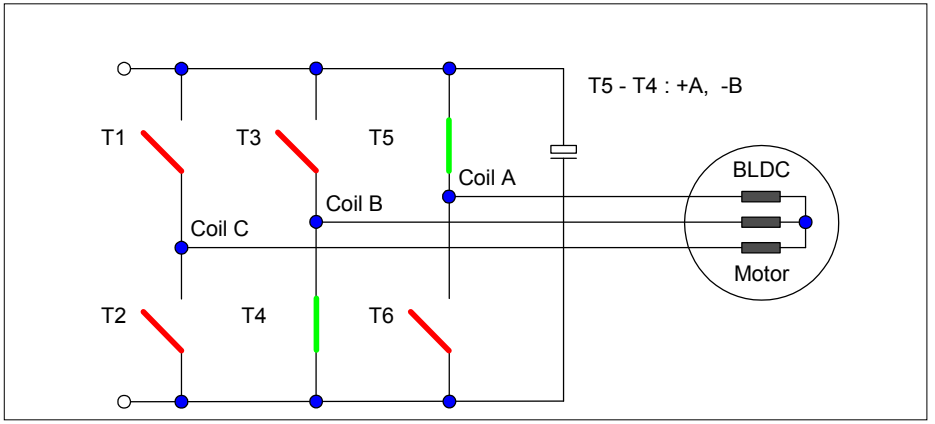


Figure 8 Appropriate switches T1, T4 of the B6 full-bridge (coil C:pos., coil B:neg.)

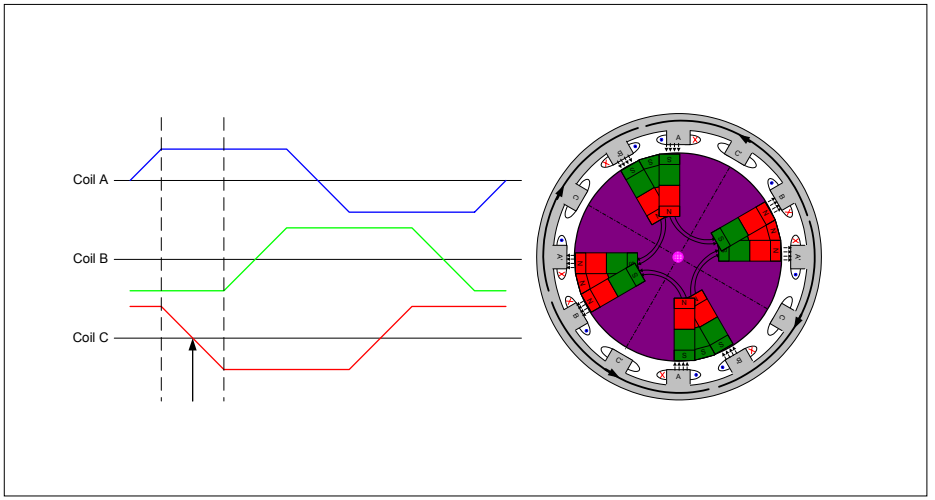


Figure 9 Energized coils A and B of a 2 pole-pair BLDC motor (mechanical position 60°)

2.1.4 Position ‘ 90° ’

Figure 11 shows the voltage-energizing coils A and C (stator) and the permanent magnet realized with two pole pairs (rotor), as well as indicating the zero-crossing detection coil B. Figure 10 shows the active switches of the B6 bridge, and Figure 11 again the appropriate rotor position (90°):

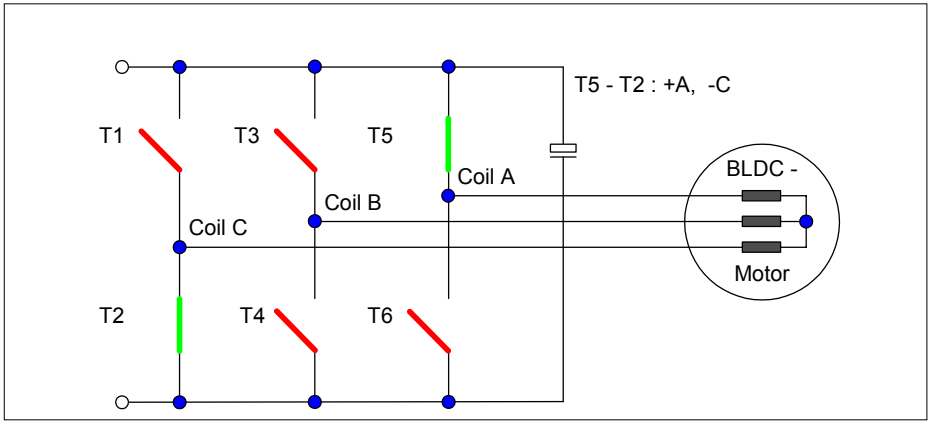


Figure 10 Appropriate switches T2,T5 of the B6 full-bridge (coil A: pos.,coil C:neg.)

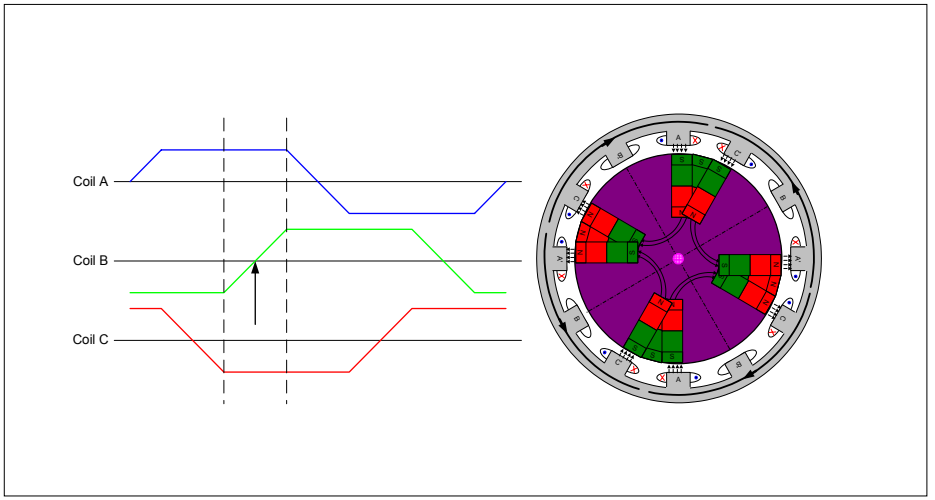


Figure 11 Energized coils A and C of a 2 pole-pair BLDC motor (mechanical position: 90°)

2.1.5 Position ‘ 120° ’

Figure 13 shows the voltage-energizing coils B and C (stator) and the permanent magnet realized with two pole pairs (rotor), as well as indicating the zero-crossing detection coil A. Figure 12 shows the active switches of the B6 bridge, and Figure 13 again the appropriate rotor position (120°):

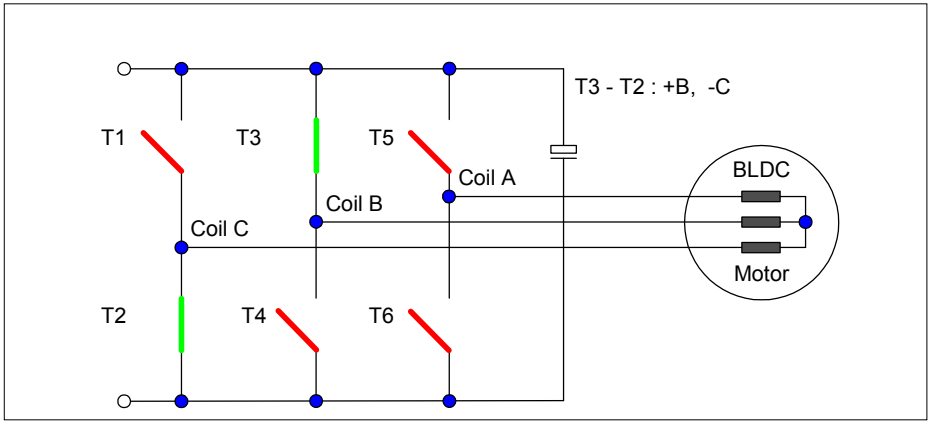


Figure 12 Appropriate switches T2, T3 of the B6 bridge (coil B: pos.; coil C:neg.)

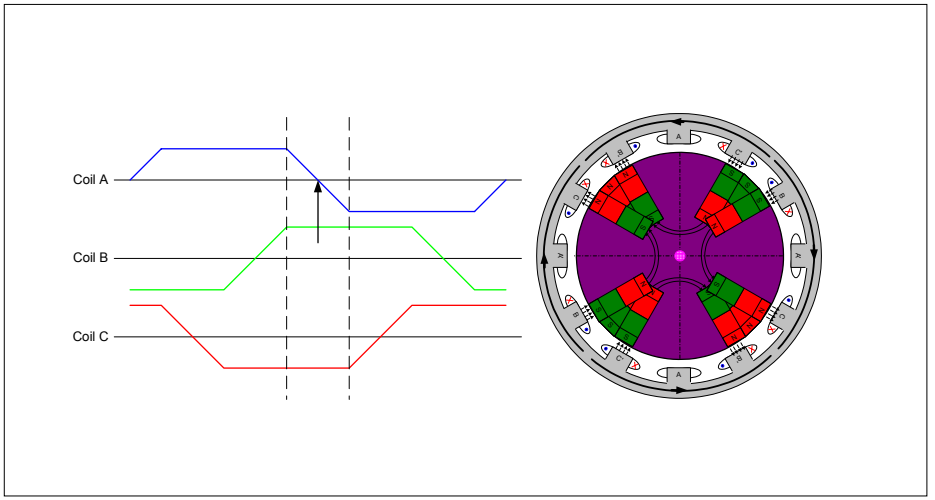


Figure 13 Energized coils B and C of a 2 pole-pair BLDC motor (mechanical position : 120°)

2.1.6 Position ‘ 150° ’

Figure 15 shows the voltage-energizing coils B and A (stator) and the permanent magnet realized with two pole pairs (rotor), as well as indicating the zero-crossing detection coil C. Figure 14 shows the active switches of the B6 bridge, and Figure 15 again the appropriate rotor position (150°):

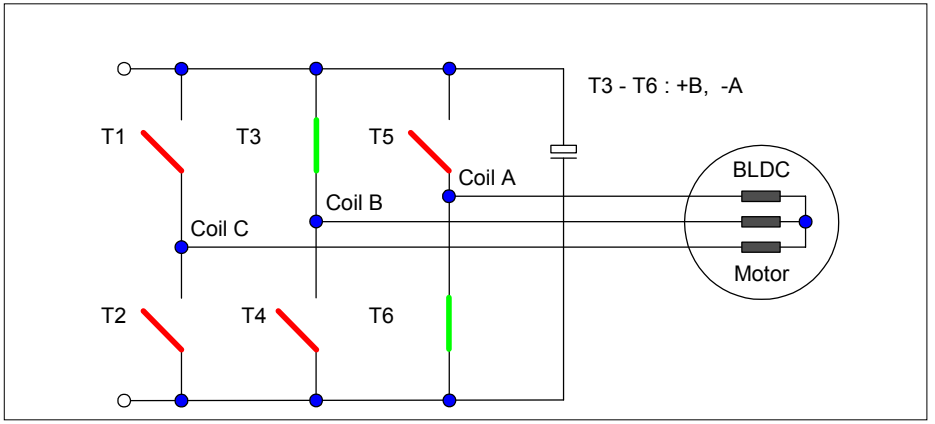


Figure 14 Appropriate switches T3, T6 of the B6 full-bridge (coil B: pos., coil A: neg.)

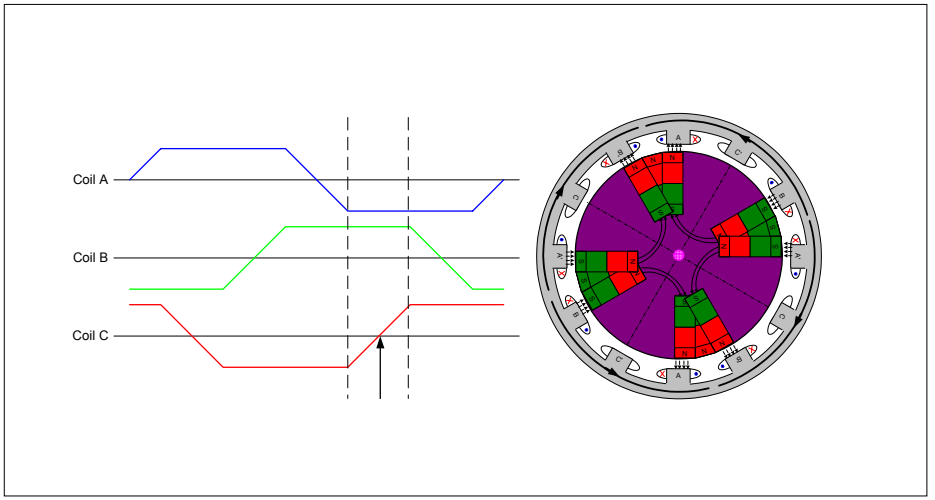


Figure 15 Energized coils A an C of a 2 pole-pair BLDC motor (mechanical position :150°)

Figure 17 shows the voltage-energizing coils C and A (stator) and the permanent magnet realized with two pole pairs (rotor), as well as indicating the zero-crossing detection coil B. Figure 16 shows the active switches of the B6 bridge, and Figure 17 again the appropriate rotor position (180°):

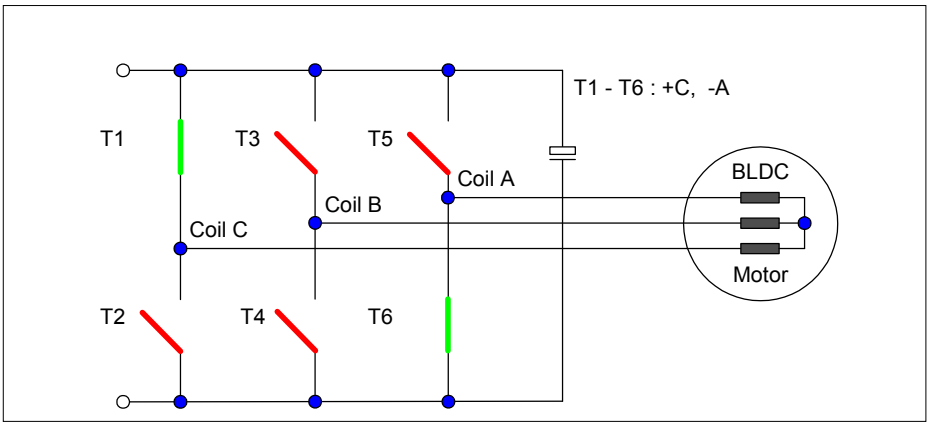


Figure 16 Appropriate switches T1, T6 of the B6 full.bridge (coil C: pos.; coild A: neg.)

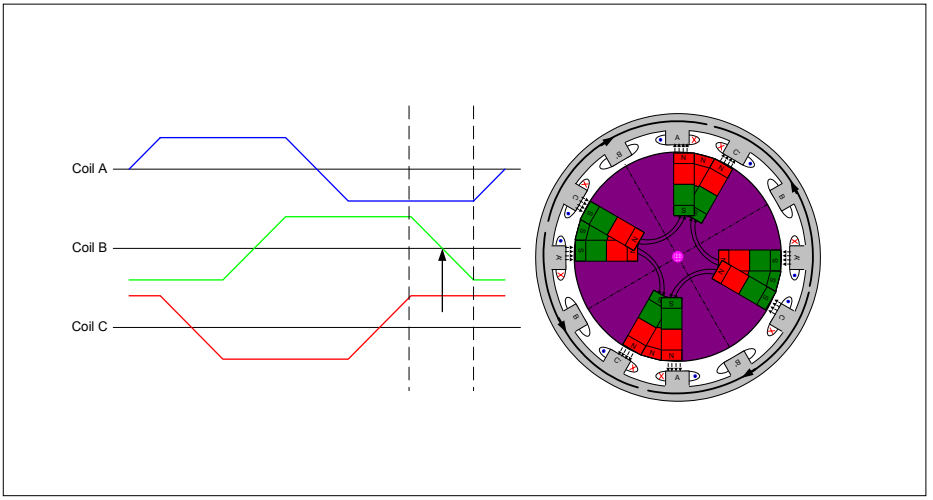


Figure 17 Energized coils A and C of a 2 pole-pair BLDC motor (mechanical position : 180°)

2.2 System description

2.2.1 Block commutation using Hall sensors

Figure 18 shows a typical structure of a three-phase BLDC motor drive suitable for use in automotive applications, e.g. fan cooling or water pump. The XC164CM microcontroller monitors the hall pattern and generates the rotation patterns for the stator field and can be used for speed or torque control.

The communication with other sub systems of the application may be realized with high speed CAN transceiver or with LIN transceiver.

The voltage regulator specially designed for microcontroller with dual supply voltage (2.6 V / 5 V) features an integrated reset circuitry, low quiescent current consumption and overtemperature/ overload protection. Further it handles the power sequencing for dual voltage microcontroller.

The BTS 7960 is a fully integrated high current half bridge for motor drive applications. It consists of one p-channel highside MOSFET, one n-channel lowside MOSFET and an integrated driver IC in one package. With three of them a 3 phase drive configuration can be combined. Dead time generation, adjustable slewrate, current limitation and current sense are only some of the mentioned features.

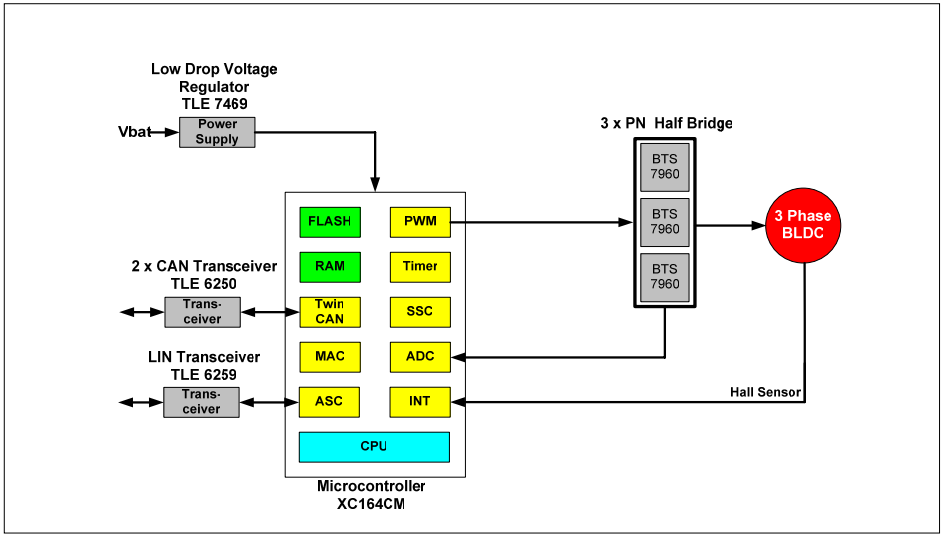


Figure 18 Block diagram of block commutation using Hall sensors

2.2.2 Block commutation without sensors

Figure 19 shows a typical structure of a three-phase BLDC motor drive with B-EMF detection. It looks similar to the BLDC drive application with Hall sensors. The only difference is shown in the detection of the rotor position. The back–EMF voltage is recorded directly at each phase. A signal conditioning unit is required to trim the voltage (filtering, etc.). The zero cross voltage is detected for calculation of the next commutation pattern via the AD channels.

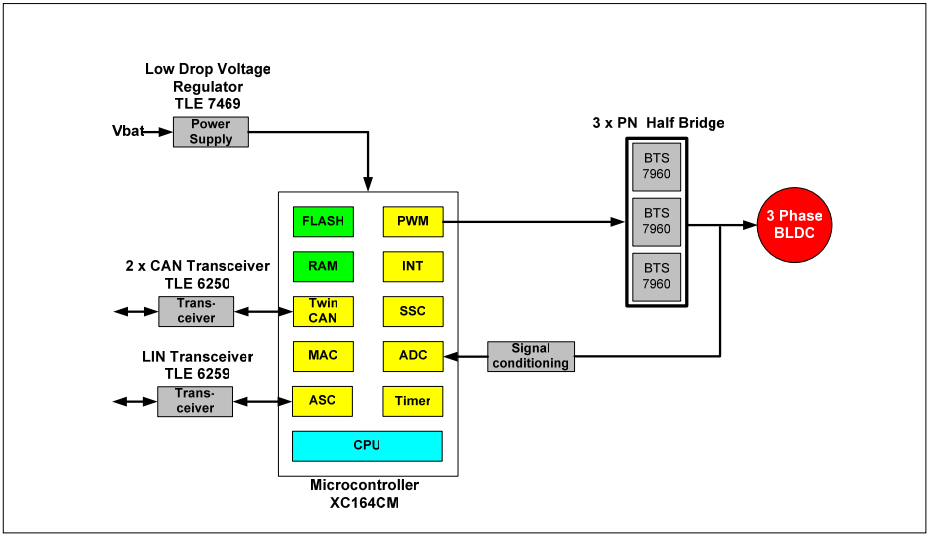


Figure 19 Block diagram of block commutation with B-EMF detection

3 Block commutation mode with BACK-EMF detection

3.1 General

As described in a previous section, the commutation of a brushless DC motor is performed electronically. The rectangular-shaped three-phase voltage system (see Figure 20) is used to generate a rotational field. Applied voltage with this easy-to-create shape helps ensure control and drive simplicity. The rotor position must be known at certain angles in order to align the applied voltage with the back-EMF, which is induced in the stator windings due to the movement of the permanent magnets on the rotor. Achieving alignment between the back-EMF and the commutation events is one of the main tasks. Under these conditions the motor behaves like a DC motor and runs at the best operating points. Simplicity of control and good performance makes this kind of motor a good choice in low-cost and high–efficiency applications.

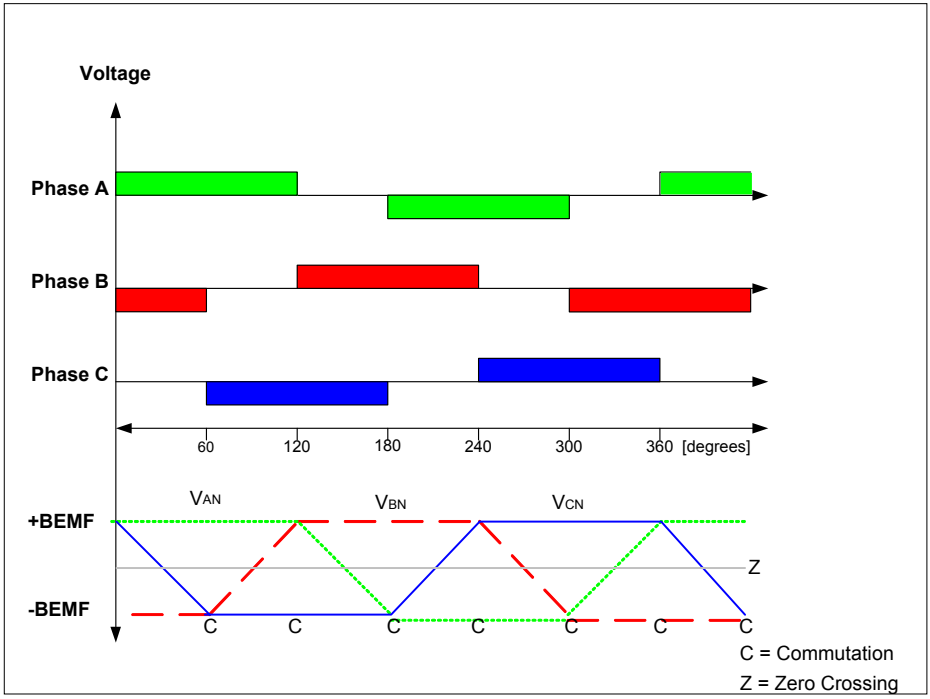


Figure 20 Simplified 3 phase voltage system/ back-EMF

3.1.1 Back–EMF sensing

The back-EMF sensing technique is based on the fact that only two phases of a DC-brushless motor are connected at the same time, so the third phase can be used to sense the back-EMF voltage. Figure 21 uses a simple model to illustrate the situation when phases A and B are energized and phase C is used to measure the back-EMF.

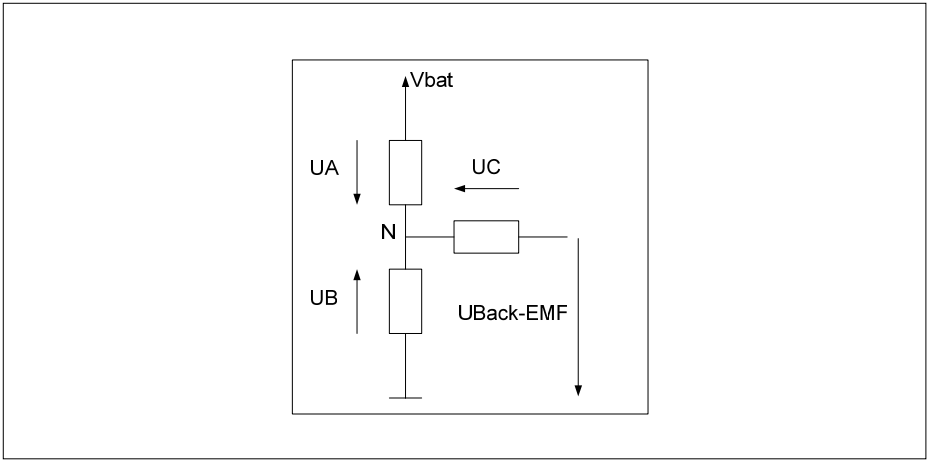


Figure 21 Simplified model of 3-phase BLDC (coils A and B are energized, coil C is analyzed)

General methods to detect the zero crossing of the back-EMF often described in the literature utilize additional hardware such as amplifiers or other circuitry. Also, a speed-dependent phase delay time has been implemented to ensure proper commutation, and the external circuitry has been configured for a fixed motor speed. Overall, the costs for these additional components have an impact on total system costs.

To reduce system costs further, provide more flexibility and even chop one or both of the phases, another system approach might be considered. Each phase is connected via voltage divider circuitry and a low-pass filter to an AD channel of the XC164CM. Each revolution of the rotor of a motor is divided into 6 commutation windows, each 60 degrees wide. For each window, an appropriate AD channel has been selected to detect the zero cross point in order to switch to the right commutation pattern after a dedicated phase delay time.

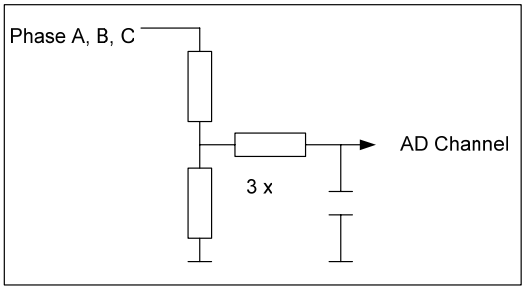


Figure 22 Passive network for zero cross detection

Basically, the external circuitry consists of a passive network such as that shown in Figure 22. The network includes a voltage divider circuit and a low-pass filter. The cut-off frequency is dependent of the application. One network is required for each phase.

Because the back-EMF voltage is a function of the rotor speed, an AD converter with a higher resolution, at least 10-bit, is required.

Figure 23 shows the monitored Back-EMF (Channels 1 to 3) of a 3-phase BLDC motor with one pole-pair. The rotation speed of the motor is around 6000 rpm.

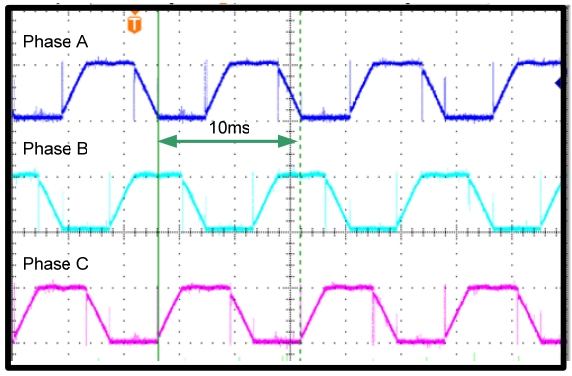


Figure 23 Back-EMF voltage after low-pass filter

3.2 Required peripherals

3.2.1 CAPCOM6

The CAPCOM6E unit of the XC164CM has been designed for applications with a requirement for digital signal generation and event capturing (e.g. pulse width modulation, pulse width measurement). It supports generation and control of timing sequences on up to three 16-bit capture / compare channels plus one additional 10-bit compare timer. This peripheral fulfills all requirements for controlling 3-phase or multiphase motor drives.

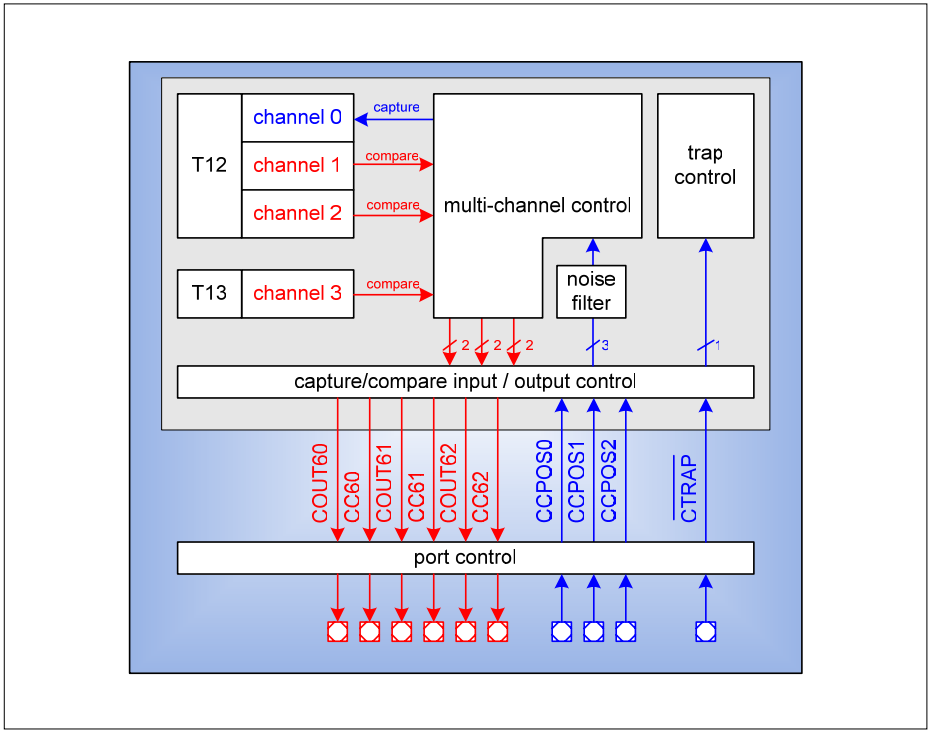


Figure 24 Block diagram of CAPCOM6E unit for Block Commutation

For BLDC Motors, usually the Multi-Channel Mode is used, see figure 24. The generated patterns need to be output in relation to the angular position of the motor. Typically Hall sensors or Back-EMF sensing are used to determine the rotor position.

Block commutation mode with BACK-EMF detection

The CAPCOM6 provides a mode for the Timer T12 block especially targeted for convenient control of a Brushless DC-Motors, see figure 27. In order to suppress spikes on the Hall inputs a noise filter (6 Bit timer) can be used. Channel 0 can be used to measure the speed of the motor. With channel 1 a phase delay may be implemented and channel 2 monitors a time out criteria. Timer 13 is used to modulate the outputs to influence the speed/torque of the motor. The CTRAP functionality provides a feature to switch off immediately the output pins if the trap input becomes active. Figure 25 shows are possible configuration for a Hall Senor Mode. For more detailed information about the CAPCOM6E unit please refer to the XC164CM manual.

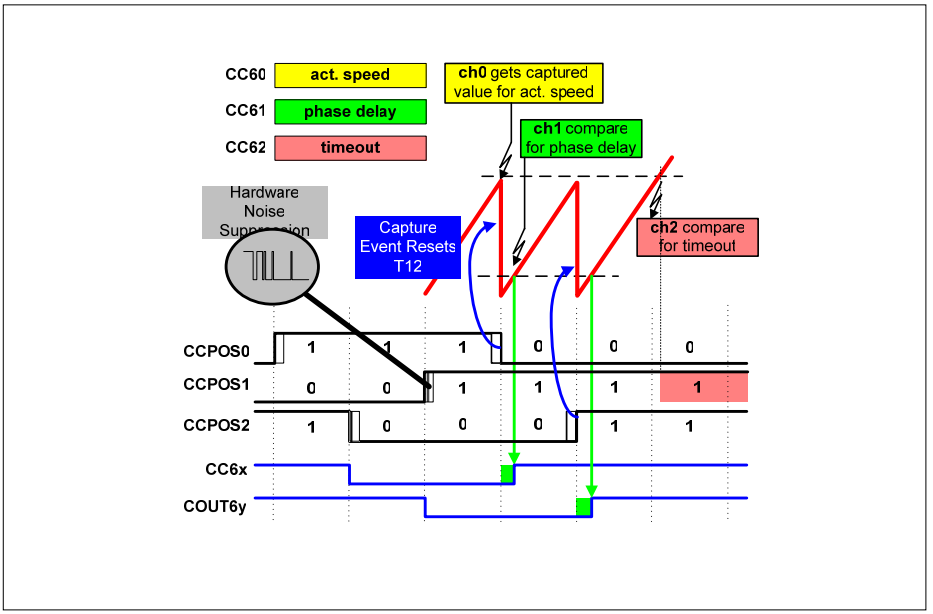


Figure 25 BLDC Motor Control with Timer T12 Block

3.2.2 ADC

The Analog Digital Converter of the XC164CM has been designed to measure analog sources with 8-bit or 10-bit resolution. It consists of a sample and hold circuit. A multiplexer allows up to 14 analog input channels to be selected. To fulfill the different requirements of embedded applications, the peripheral supports several conversion modes, such as fixed channel conversion, autoscan conversions or channel injection modes. By hardware the channel injection can be triggered with a period match of timer T13 and with a capture/compare event of CC31.

The minimum conversion time is dependent on the CPU frequency and can be configured individually. The minimum conversion time for a 10bit resolution is 2.55 µs (fcpu @ 40 MHz). For more detailed information please refer to the datasheet or the user manual.

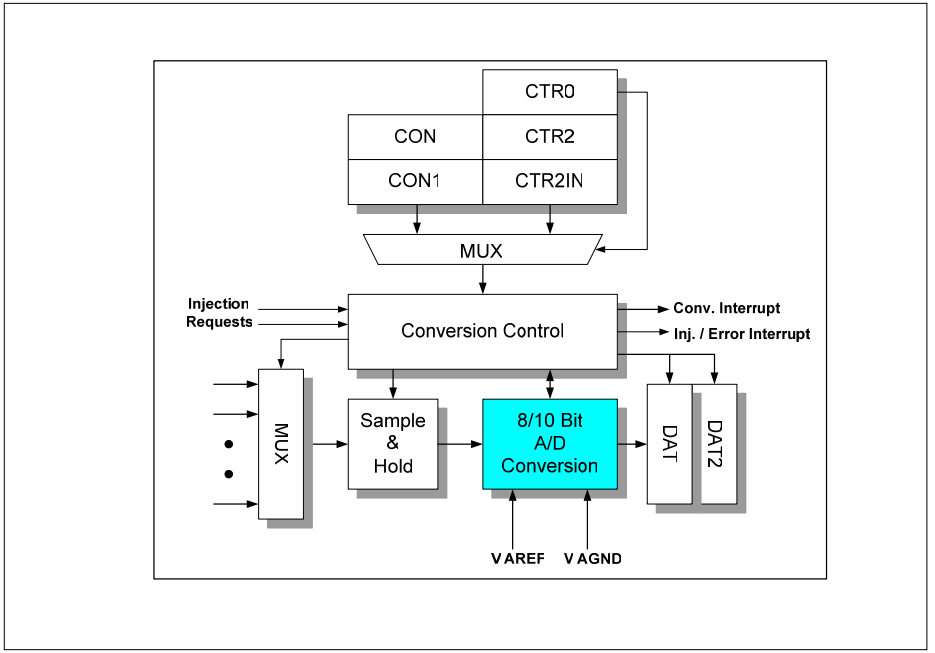


Figure 26 Block diagram of ADC unit

4 Implementation

4.1 Functional description

The application node includes an example code and a monitor program to configure and control the BLDC motor drive application.

4.1.1 General description

The application is divided into several sub-functions. First, all relevant peripherals are configured in the function **MAIN_vinit()**. In the function **START()** , the ramp-up phase of the motor is handled. During this phase, the commutation speed and the phase voltage are increased continuously until the back-EMF voltage is interpretable. Now the application switches to the closed loop mode and the motor is accelerated until it reaches the nominal voltage level. Further an optional PI controller for speed or torque control may be activated. The function **STOP()** is used to stop the motor in a regular way.

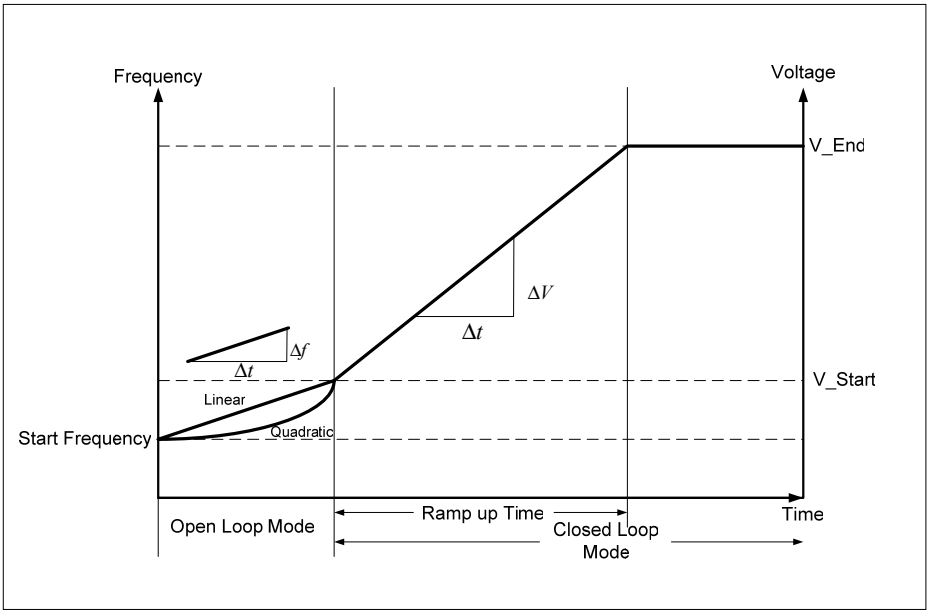


Figure 27 Acceleration phase

4.1.2 Management of commutation, demagnetization and zero cross detection

The commutation patterns are generated and modulated with timer T13 and parts of the capture-compare unit 6. Timer T12 and the corresponding register CC61SR are used to generate the phase delay and to control the defined time-out criteria. Register CC62 is used to trigger on each slope of the B-EM. Further the acceptance window for the zero cross detection is defined. CC60 can be used to measure the rotor speed. The measurement of the phase voltage is done by three ADC channels via PEC. These channels can be triggered and periodically analyzed either by the timer t13 period match or by CC31 of the CAPCOM2 unit. Additionally a fourth channel is used for current measurement. All tasks are interrupt driven and consume less CPU-time.

4.2 Implementation of the software

The software is divided into several routines:

Main loop:

- Initialization (CPU, I/O ports, CAPCOM6, CAPCOM2, ADC, ASC (monitor))
- Start, Stop, communication (monitor program)

Interrupt routines:

- CAPCOM 6
  - Timer T13
  - Error handling
  - Speed measurement
- CAPCOM2
- ADC
- ASC (monitor program)

4.2.1 Initialization

A number of small routines perform all the necessary initializations before the motor is started:

Port initialization:	P1L.x is used as alternate output for the CAPCOM6 output (CC6x,COUT6x).
Interrupt initialization:	CAPCOM6 emergency trap, timeout (CC62), timer 13 period match, speed measurement (CC60), ADC interrupt with PEC, ADC error, ASC receive interrupt.
CAPCOM6 initialization:	Enable multi-channel mode, enable hall sensor mode, passive output level is low, Timer T13 edge aligned mode. Enable trap function for emergency cases.
CAPCOM2 initialization:	Add additional time base timer T7, CC19, CC31 (optional if phase current /voltage is measured at the same time).
ADC initialization:	Select P5.1-P5.4 as AD channels, 10bit, fixed-channel single conversion, sample time 400ns, conversion time 2,55µs.

4.2.2 Function Main()

Figure 30 shows the call graph of the function **Main()**. All sub-programs can be called over the main function.

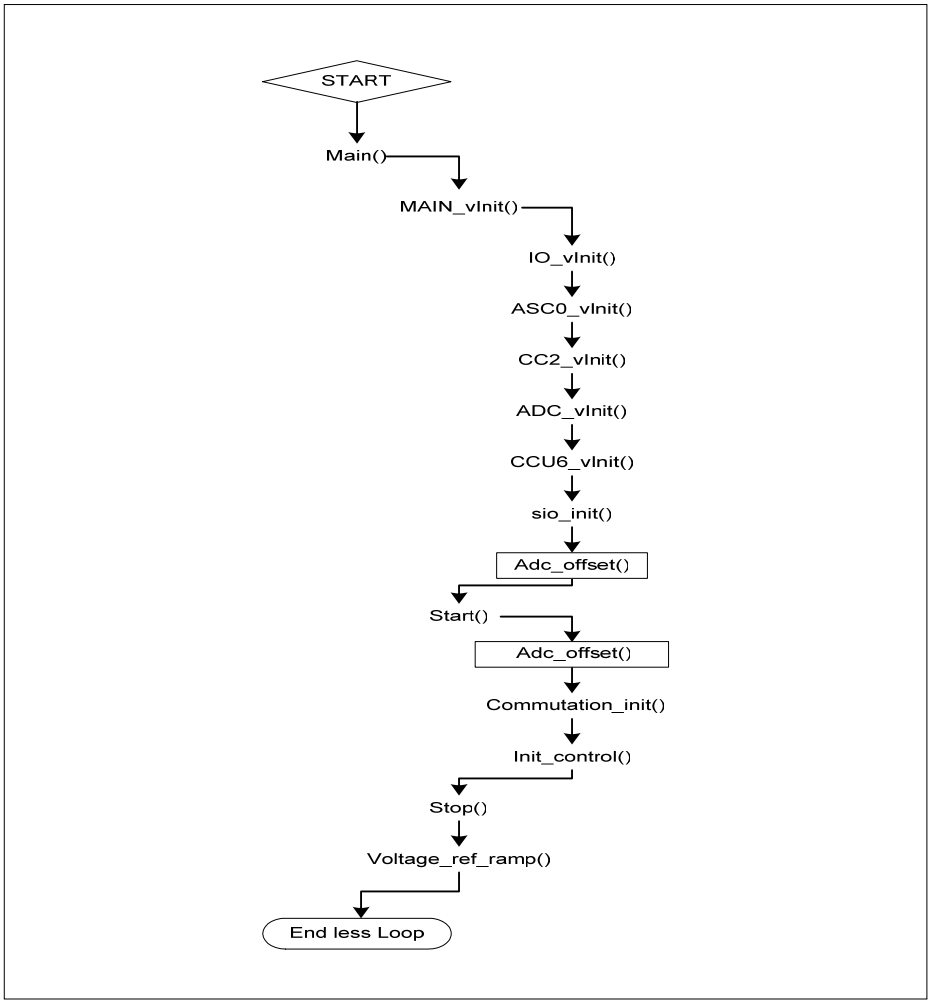


Figure 28 Call graph of function Main()

4.2.2.1 Function Start()

The variable Status_word includes the state on/off and all possible error flags. The variable v_start_ref which present the start voltage, is initialized every time when the function is activated. The function **Commutation_init()** activates the modulation of the CAPCOM6 output pins. The function **Init_control()** initializes all variables for the motor control.

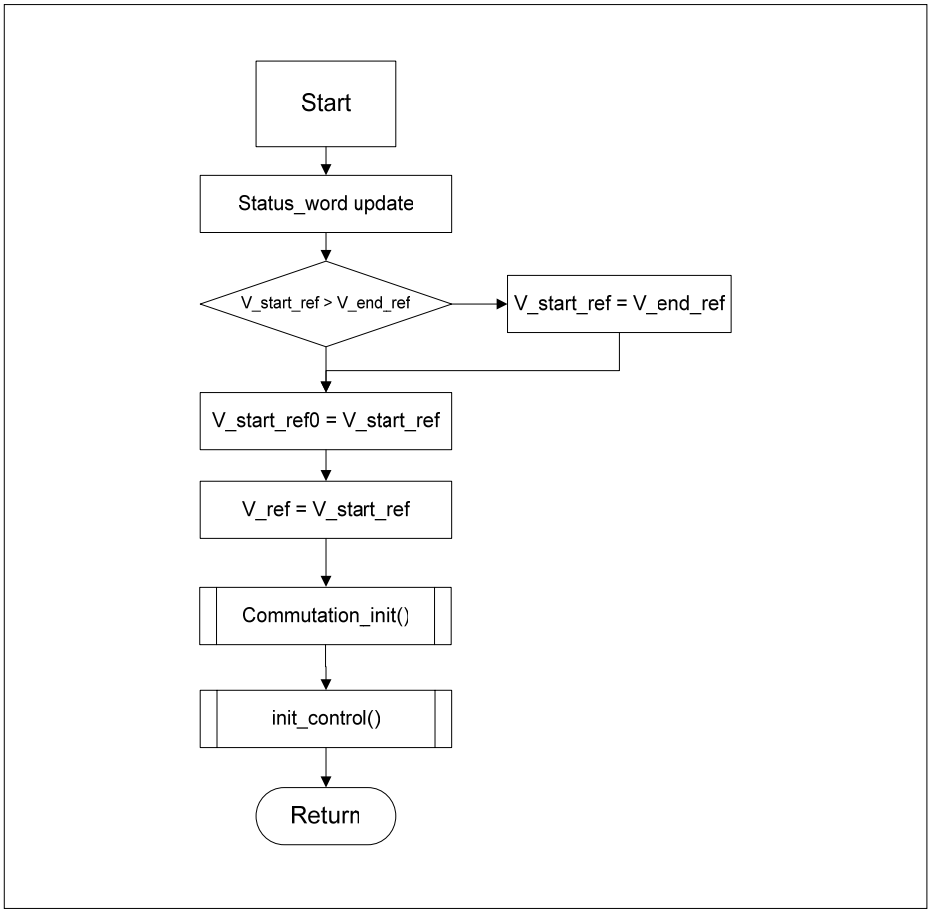


Figure 29 Function Start()

4.2.2.2 Function Stop()

The function **Stop()** is called either by the user or when an error occurres. Basically the function **Stop()** disable all relevant drivers, like CAPCOM6 or status variables.

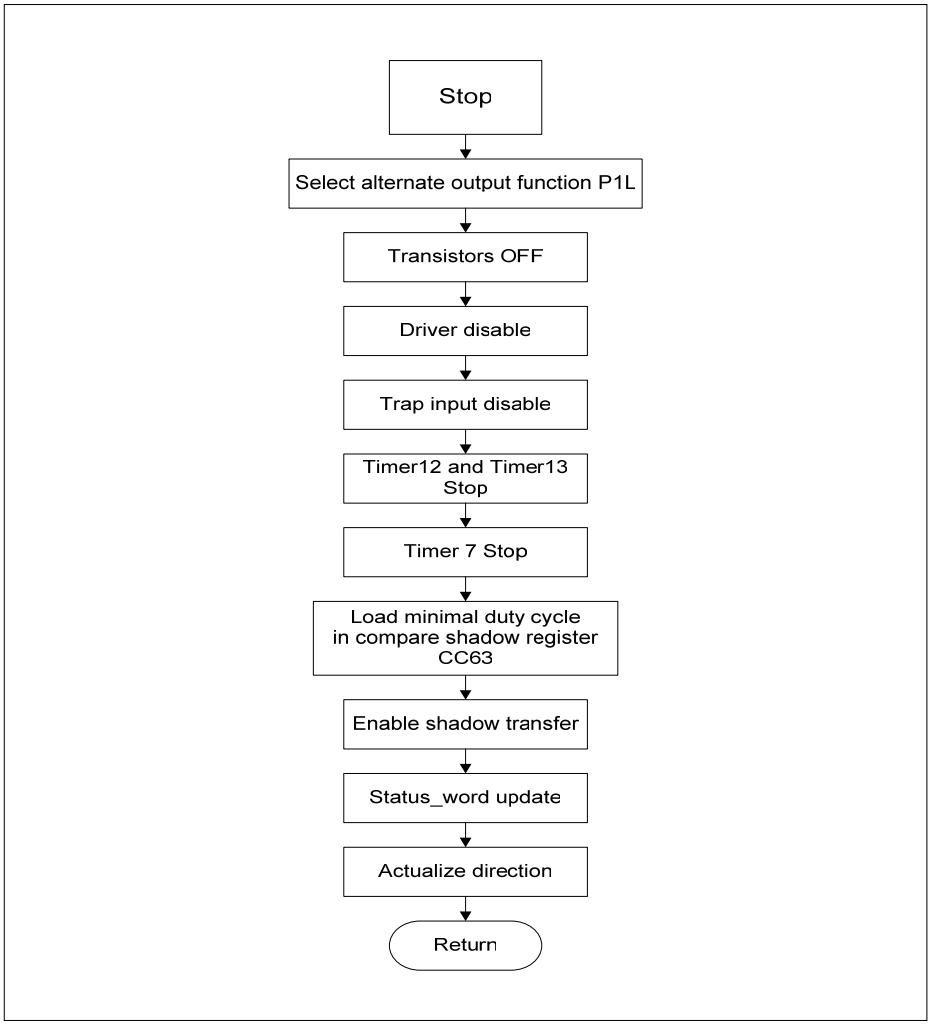


Figure 30 Function Stop()

4.2.3 Interrupt routines

The interrupt routine **CCU6_viINodeI2** is executed when a CC62 compare match occurs. If the motor is working in normal operation mode (state3) then a speed calculation can be done. The state of the Back EMF measurement is given in the variable slope.

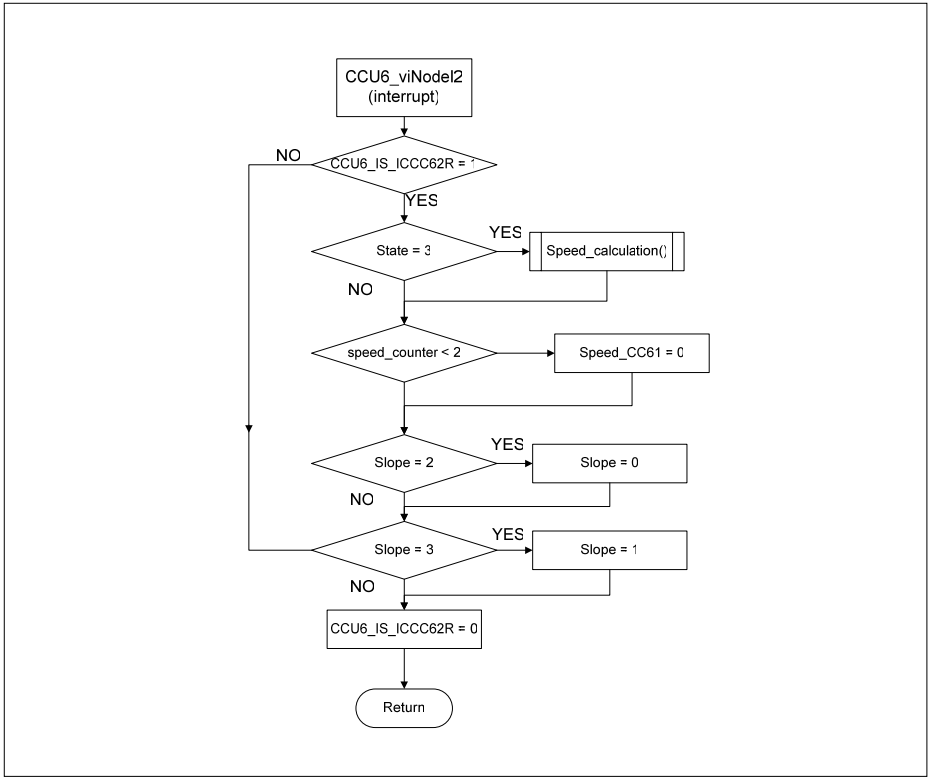


Figure 31 Interrupt speed measurement CCU6_viNodeI2

- Slope = 0 → B-EMF Measurement pos. edge with zero crossing detection
- Slope = 1 → B-EMF Measurement neg. edge with zero crossing detection
- Slope = 2 → B-EMF Measurement neg. edge without zero crossing detection
- Slope = 3 → B-EMF Measurement pos. edge without zero crossing detection

Function 32 shows the function **Speed_calculation()**.

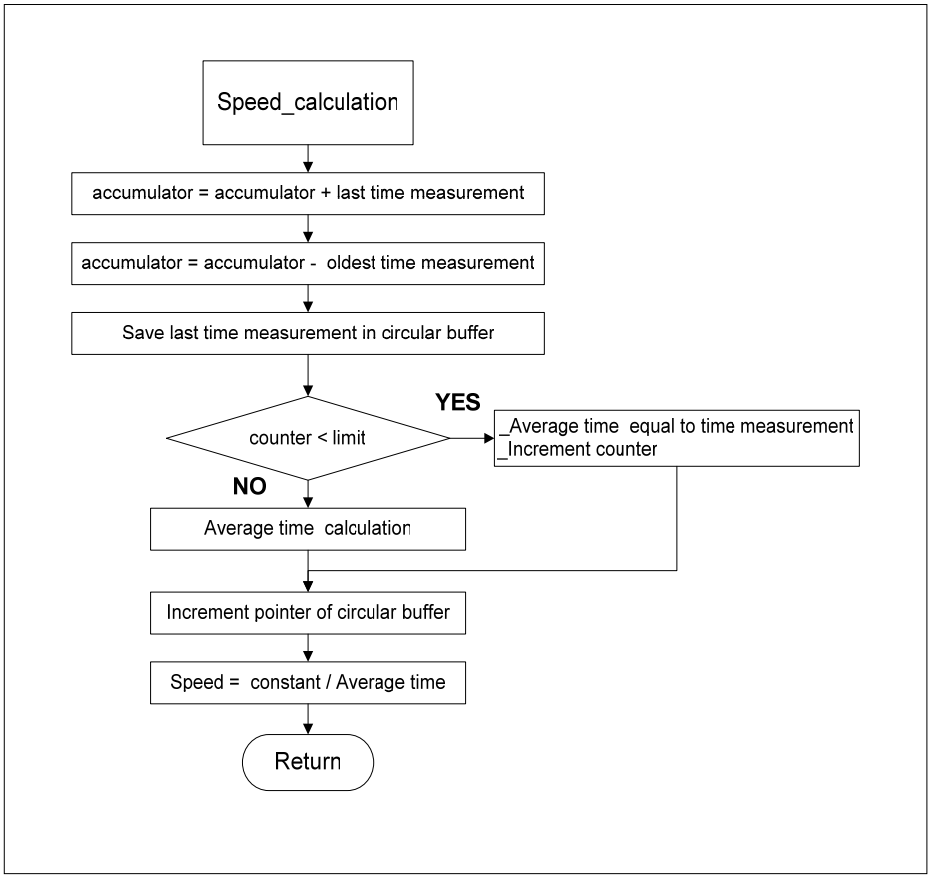


Figure 32 Function Speed_calculation

The values of the time measurement are saved in a circular buffer. Six values are used to minimize errors from B-EMF measurement. The variable counter counts the commutation events and builds the average value for the speed calculation.

The interrupt **CCU6_viNode3** is executed every 50 µs when the Timer T13 period match happens. At first the function **commutation()** is called. It is a state-machine and handles the different operation modes. Then, depending of the actual pulse pattern the accordant AD-channel is chosen for Back-EMF measurement.

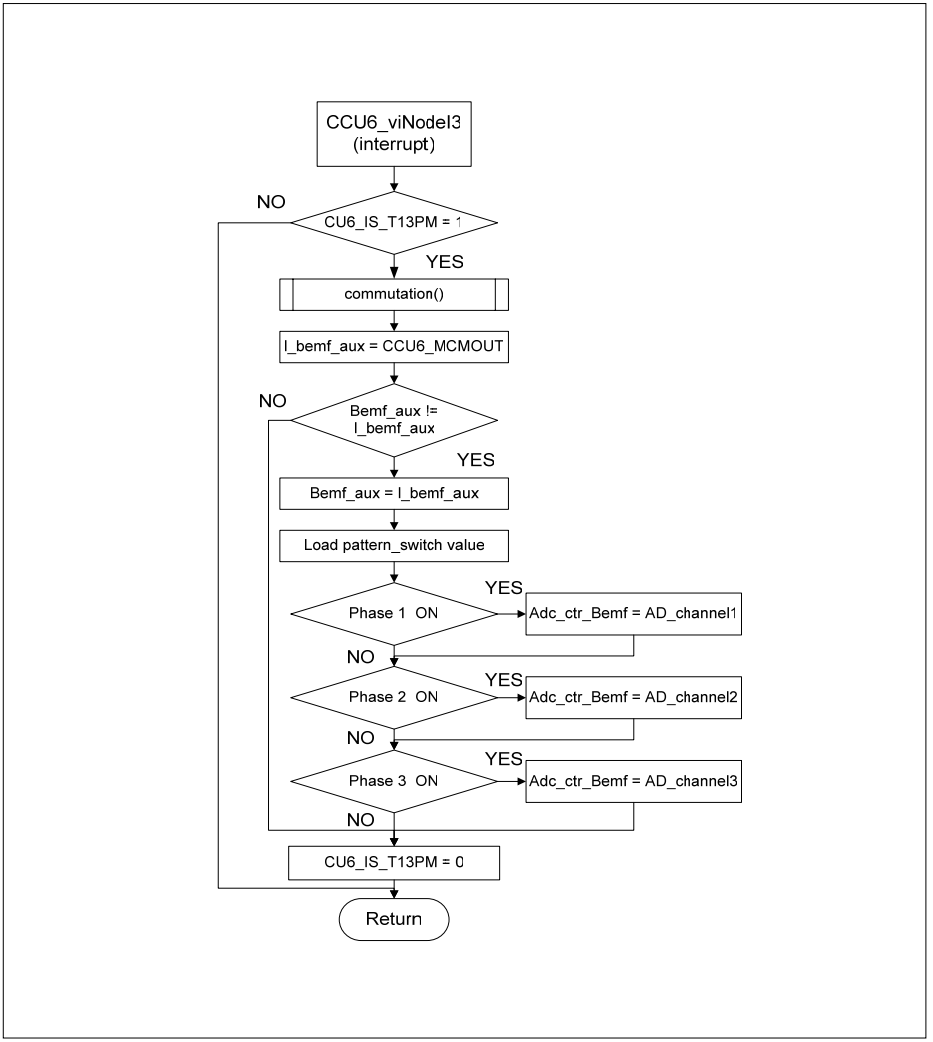


Figure 33 Interrupt Timer T13PM (CCU6_viNodeI3)

4.2.3.1 Function commutation()

This function can handle 5 different operation modes (state 0 to 4) of a motor application, which are illustrated in figure 34.

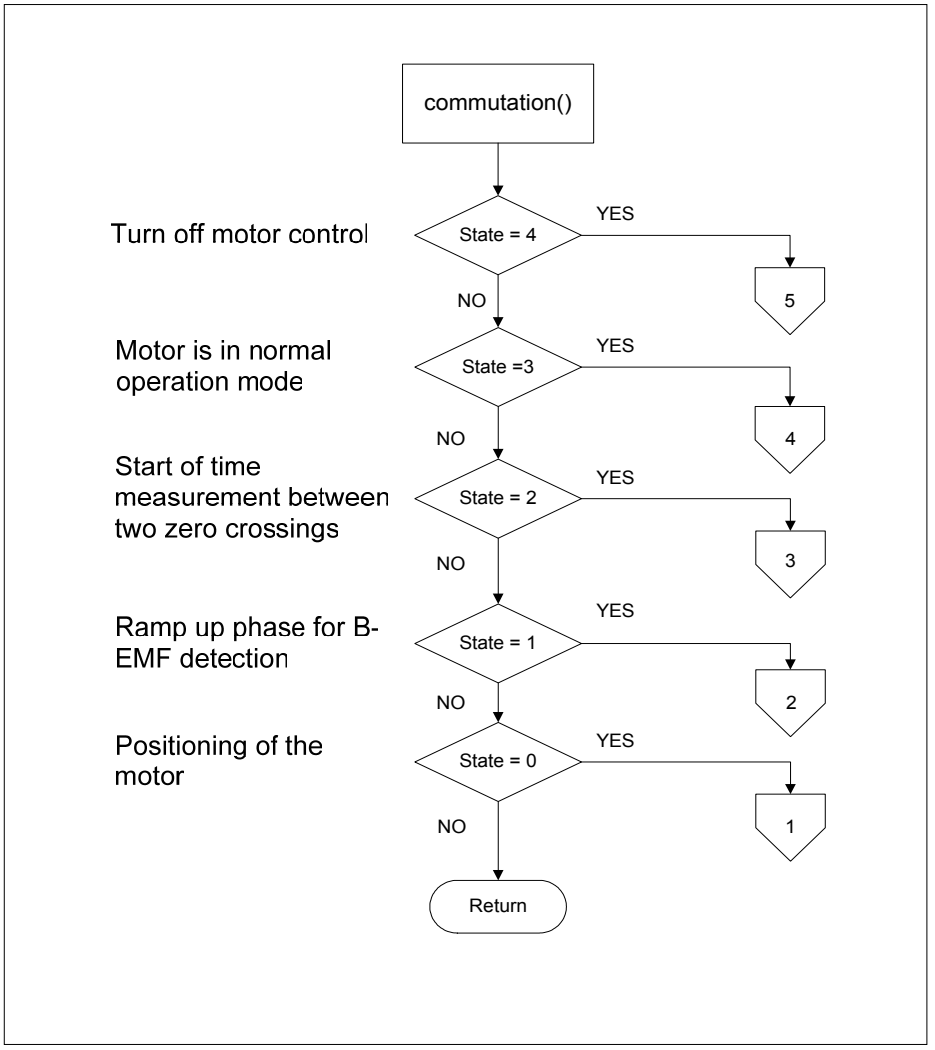


Figure 34 Function commutation()

4.3 CPU load

The following hardware requirements are needed:

CPU:	XC164Cx , fcpu @ 40 MHz
ROM:	5 kByte
RAM:	360 bytes plus system stack
Peripherals:	CAPCOM6 unit:
	ADC: 4-channel, 10-bit resolution, 2,55µs conversion time
	CCU2: Timer: 16 timer, 25ns resolution
CPU Load:	12 % (independent of the motor speed)

The CPU load is measured in a simple way. A flag of a dedicated port pin is set each time an interrupt routine is entered. The flag is reset after the interrupt routine is exited.

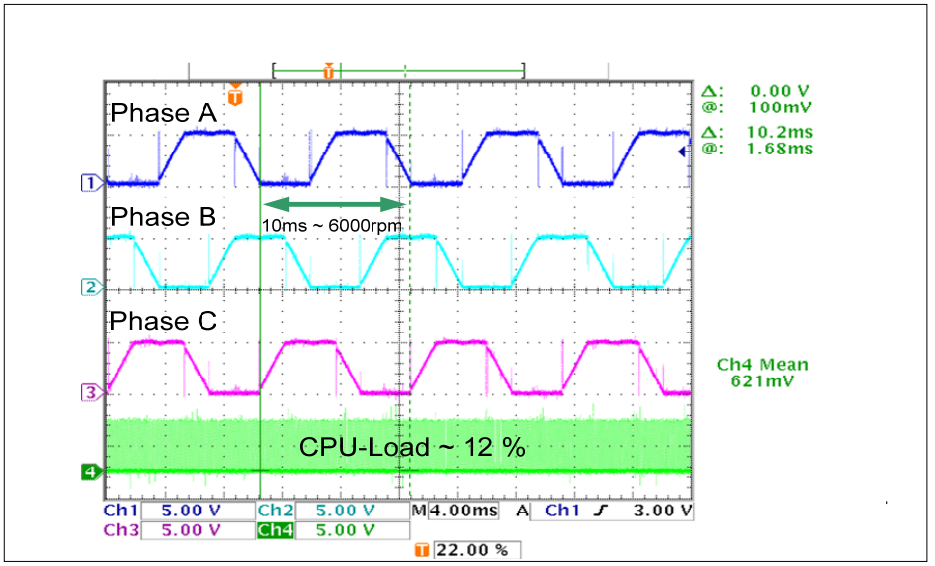


Figure 35 CPU load

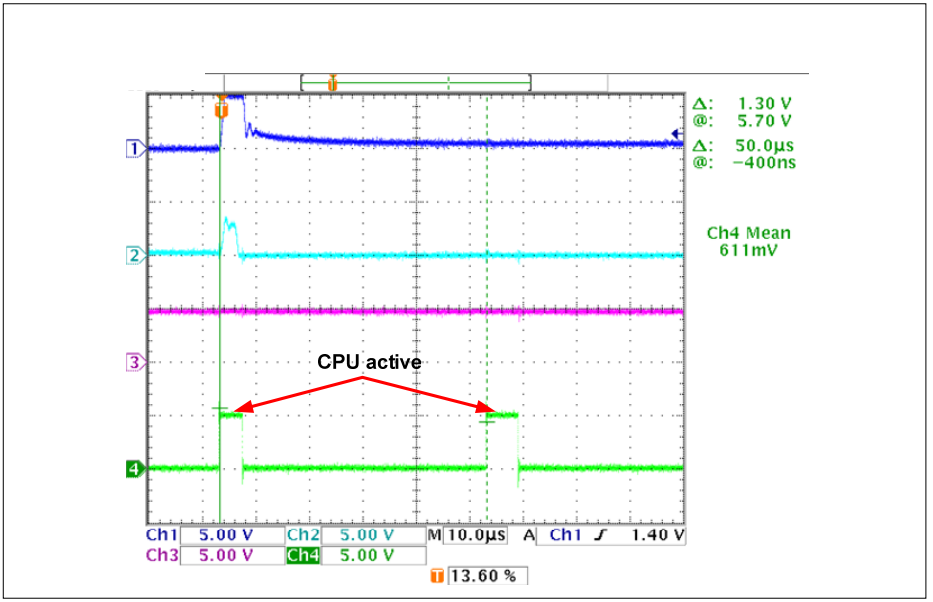


Figure 36 CPU load in detail

In this implementation the CPU load is independent of the motor speed, because each 50µs a B-EMF phase analysis is done. The CPU load can be reduced by the following actions:

- Reduce the number of phase samples (currently each 50µs)
- Focus the sampling on the outer field of the zero crossing

5 Get it started

5.1 General

Additional parts of a complete evaluation platform are available:

- Source code for the XC164x family for the Toolchain Keil, Tasking
- Monitor program based on the RS232 interface

The target board and the BLDC motor should be customer specific and is not part of this evaluation platform.

The source code can be programmed in the internal Flash with MEMTOOL via RS232 interface. Of course using one of the mentioned tool chains the code can be downloaded and debugged.

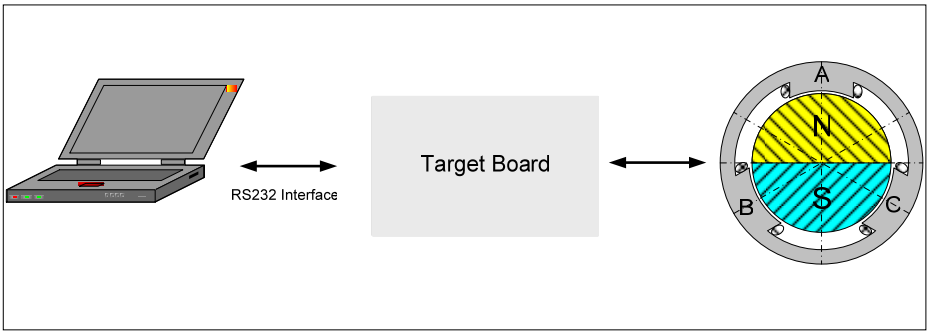


Figure 37 Overview

5.2 Monitor

The monitor brdc_monitor controls the application. Three windows can be selected.

Display (Display of variables)

Input (Control and initialization of the application / variables)

Graphic (Up to three curves can be stored)

Within the major window the com port is selected and the communication with the application board is established.

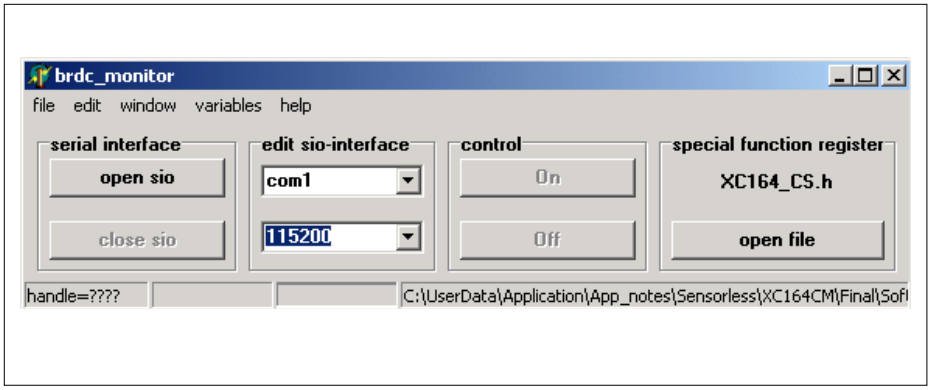


Figure 38 Major window

Before the application is started the initialization of the application is mandatory. Therefore the Input window is selected, all parameters are checked and the data are sent to the microcontroller. After that the application can be turned on.

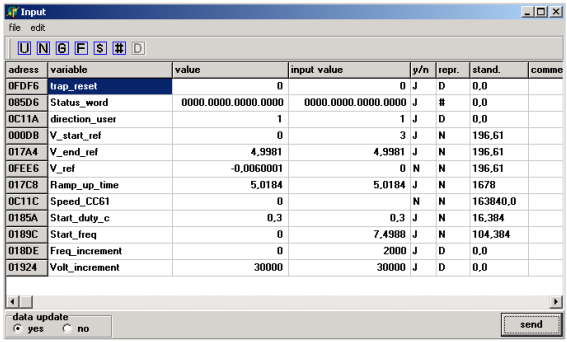


Figure 39 Input Window

If additional special function registers or variables should be monitored an easy to handle drag and drop functionality is available.

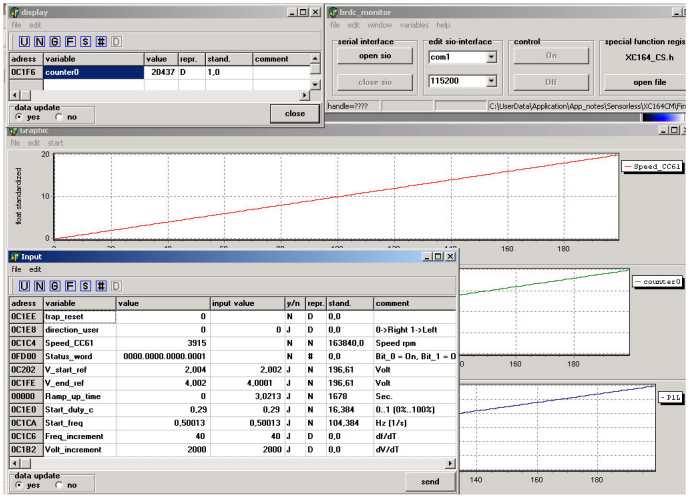


Figure 40 All windows

6 Conclusion

There is an increasing demand in the automotive industry for applications using BLDC motors. In the past, many applications were addressed with DC motors. Automotive suppliers would like to use BLDC motors in order to achieve greater robustness and efficiency. Currently, the only limitation is the price gap between BLDC motors and DC motors. On the other hand, automotive suppliers are intensely price-driven and need smart solutions to meet the requirements of the market.

This application note aims to provide an understanding of the functionality of a BLDC motor explains how such a motor is driven, describes all the necessary components and the necessity of knowing the positioning of a synchronous motor. Furthermore, the back-EMF technique is discussed and a method to implement the back-EMF algorithm in the XC164CM is presented. To achieve a good price/performance ratio, all the components used for back-EMF detection have been replaced by software. Thanks to the high performance of the microcontroller and its powerful peripherals, this software solution consumes only limited CPU resources.

The system approach discussed might be considered as a starting point for designing a complete system-specific closed-loop BLDC application

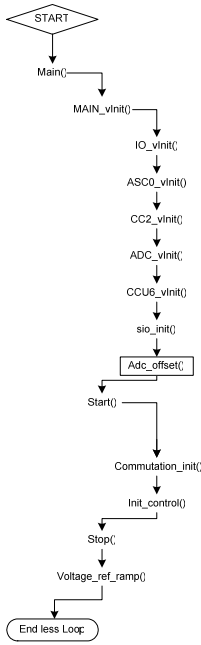
7 Glossary

ADC:	Analog Digital Converter
B-EMF :	Back electromagnetic force
B6 :	Bridge with 6 switches (e.g. MOSFET) for 3 phases
BLDC:	Brushless DC
CAN:	Controller Area Network
CAPCOM:	Capture Compare Unit, peripheral that measure events or provide PWM signal
CPU:	Central Processor Unit
EHPS:	Electro Hydraulic Power Steering
LIN:	Local Interconnect Network
PI:	Proportional Integral
PWM:	Pulse width modulation
ROM:	Read only memory
RAM:	Read Access Memory

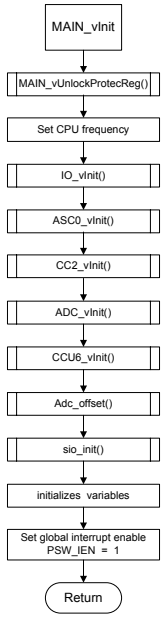
8 Addendum

8.1 Flow Charts

8.1.1 Call graph of Main Function



8.1.2 Flow chart of Main_vinit Function



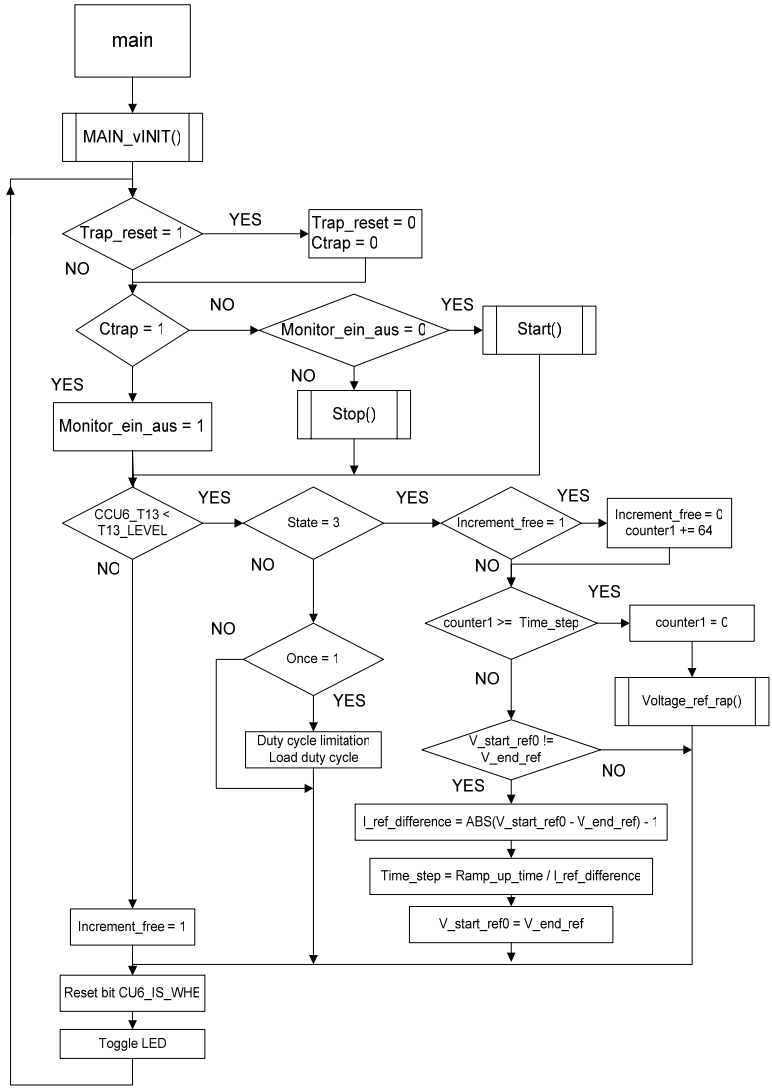
Name	Type	Init value	Change
direction	**Int**	**0x0000**	**Intern**
Once	**Int**	**0x0000**	**Intern**
Status_word	**Int**	**0x0002**	**Intern**

direction gives the rolling direction of the motor.

Once is a flag, which will be set in the start program.

Status_word is a variable, which shows the status- and error bits of the control.

8.1.3 Flow chart of Main Function



Name	Type	Init value	Change
monitor_ein_aus	bit	0	User
Ctrap	Bit	0	Intern
increment_free	Int	0x0000	Intern
Time_step	long	0x0000	Intern
Trap_reset	Int	0x0000	User
counter1	long	0x0000	Inter
Ramp_up_time	long	0x0000	User
V_start_ref0	Int		Intern
V_end_ref	Int		User
Duty_cycle	Unsigned Int		Intern

monitor_ein_aus is the On/Off signal from the monitor program resp. the user.

CTrap is an error flag from the driver board.

increment_free is a flag for the reference ramp.

Time_step gives the time basis for the increment of reference ramp.

Trap_reset is used from user to delete the Trap.

Counter1 is a timer for the determination of time basis of the reference ramp.

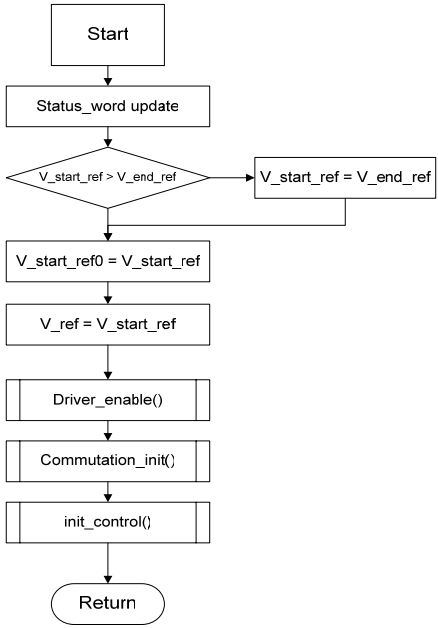
Ramp_up_time determine the time for the reference ramp.

V_start_ref0 is the memory for the start value of the voltage reference ramp in operation mode.

V_end_ref is the final value of the voltage reference ramp.

Duty_cycle is the value of the duty cycle.

8.1.4 Flow chart of Start Function



.

Name	Type	Init value	Change
Once	**Int**	**0x0000**	**Intern**
Status_word	**Int**	**0x0002**	**Intern**
V_start_ref	**Int**		**User**
V_start_ref0	**Int**		**Intern**
V_ref	**Int**		**Intern**
V_end_ref	**Int**		**User**

Once is a flag, which will be set in the start program.

Status_word is a variable, which shows the status- and error bits of the control.

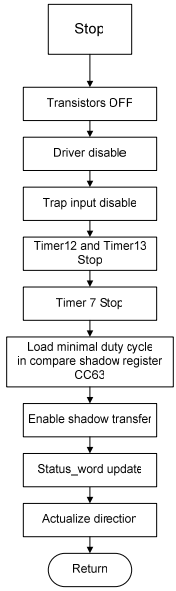
V_ref is the value of voltage reference for the control.

V_start_ref is the start value of the voltage reference ramp by startup.

V_start_ref0 is the memory for the start value of the voltage reference ramp in operation mode.

V_end_ref is the final value of the voltage reference ramp.

8.1.5 Flow chart of Stop Function



Name	Type	Init value	Change
On	bit	0x0000	Intern / User
Off	bit	0x0000	Intern / User
Direction	Int	0x0000	Intern
direction_user	Int	0x0000	User

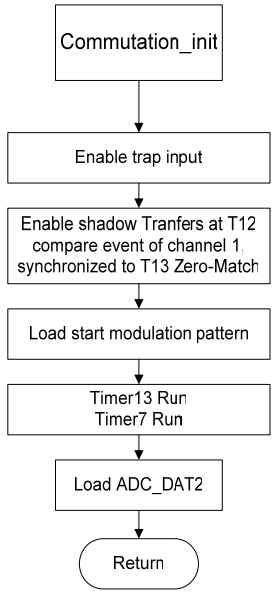
On is a bit, which shows that the motor is running (Status_word).

Off is a bit, which shows that the motor is stopped (Status_word).

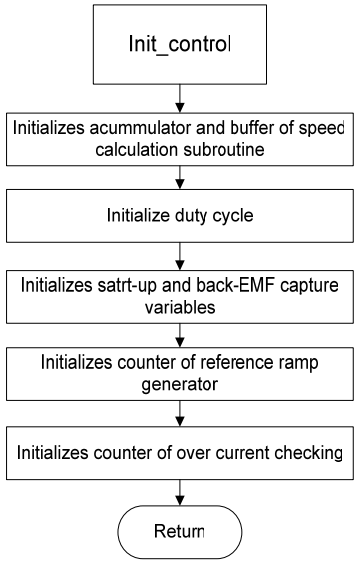
direction gives the rolling direction of the motor.

direction_user is a variable to choose the rolling direction of the motor by the user.

8.1.6 Flow chart of Commutation_init Function

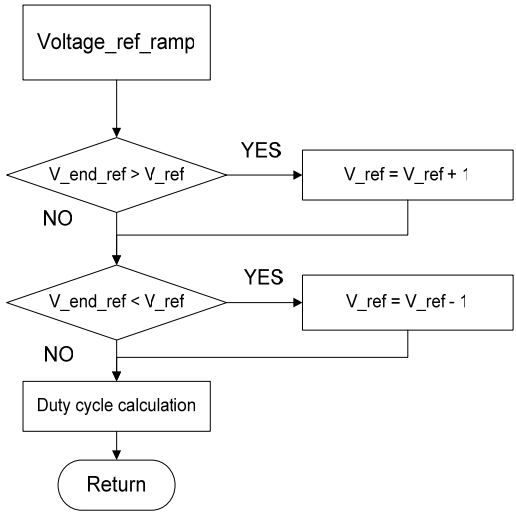


8.1.7 Flow chart of Init_control Function



.

8.1.8 Flow chart of voltage_ref_ramp Function



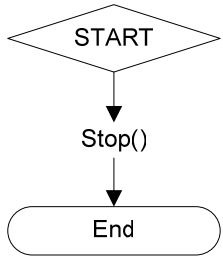
Name	Type	Init value	Change
V_ref	**Unsigned Int**	**0x0000**	**Intern**
V_end_ref	**Unsigned Int**	**0x0000**	**User**
Duty_cycle	**Unsigned Int**		**Intern**

V_ref is the value of voltage reference for the control.

V_end_ref is the final value of the voltage reference ramp.

Duty_cycle is the value of the duty cycle.

8.1.9 Call graph of Error Interrupt-Program (CCU6_viNodel1)



8.1.10 Flow chart of Interrupt CCU6_viNodel1

Name	Type	Init value	Change
Time_out	**Bit**	**0**	**Intern**
Ctrap	**Bit**	**0**	**Intern**

Time_out is the error flag from Timer 12 overflow.

CTrap is the error flag from driver board.

8.1.11 Call graph CC62 Interrupt (CCU6_viNodeI2)

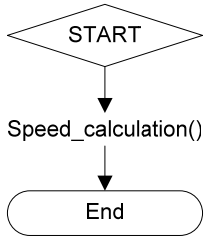
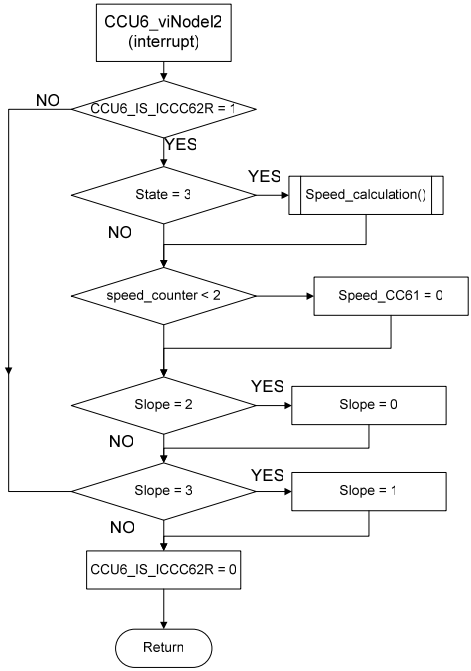
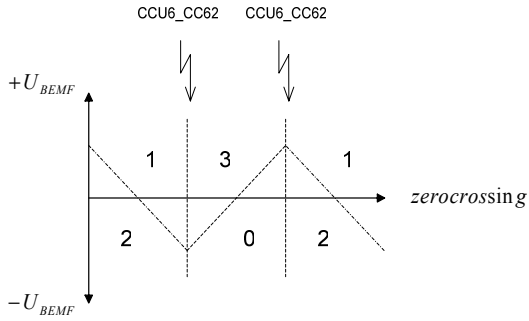


Figure 4.2.2.13 CC62_INT (CC62 compare-match)

.

8.1.12 Flow chart of Interrupt CCU6_viNodeI2





- Slope = 0 → B-EMF Measurement pos. edge with zero crossing detection
- Slope = 1 → B-EMF Measurement neg. edge with zero crossing detection
- Slope = 2 → B-EMF Measurement neg. edge without zero crossing detection
- Slope = 3 → B-EMF Measurement pos. edge without zero crossing detection

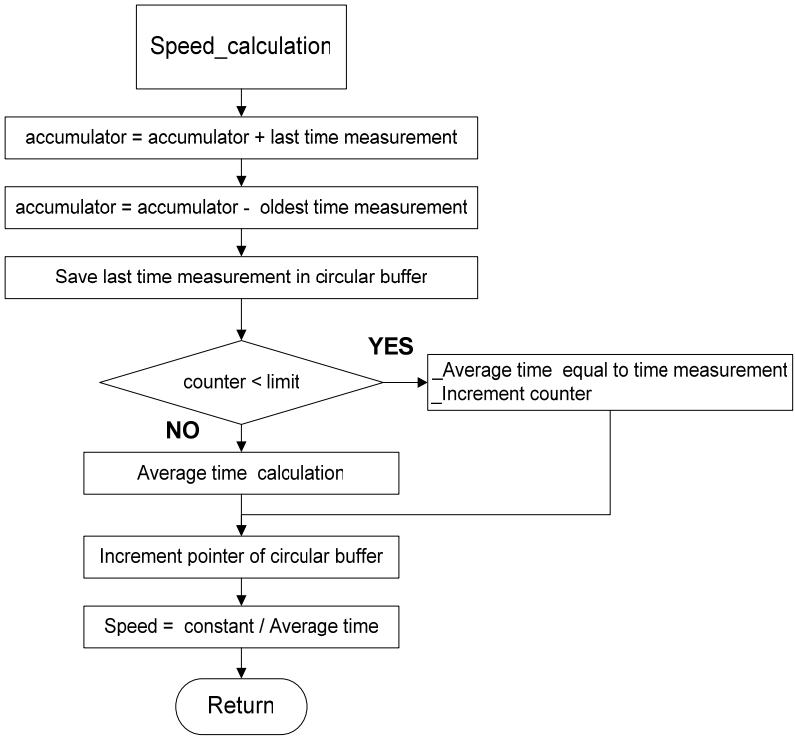
Name	Type	Init value	Change
Time_measur	**Unsigned Int**	**0x0000**	**Intern**
speed_counter	**Int**	**0x0000**	**Intern**
Slope	**Int**	**0x0000**	**Intern**
Speed_CC61	**Unsigned Int**	**0x0000**	**Intern**

Time_measur is the time between two zero crossings and serves as parameter for speed calculation.

speed_counter counts the number of commutations and serves as parameter for speed calculation.

Speed_CC61 is the actual speed.

8.1.13 Flow chart of Speed_calculation Function



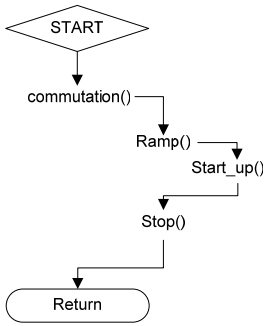
Name	Type	Init value	Change
Speed_buffer[6*P_p]	**Int**	**0x0000**	**Intern**
circ_index	**Int**	**0x0000**	**Intern**
Speed_accumm	**Long**	**0x0000 0000**	**Intern**

Speed_buffer[6*P_p] is the circular memory for average calculation of time measurement.

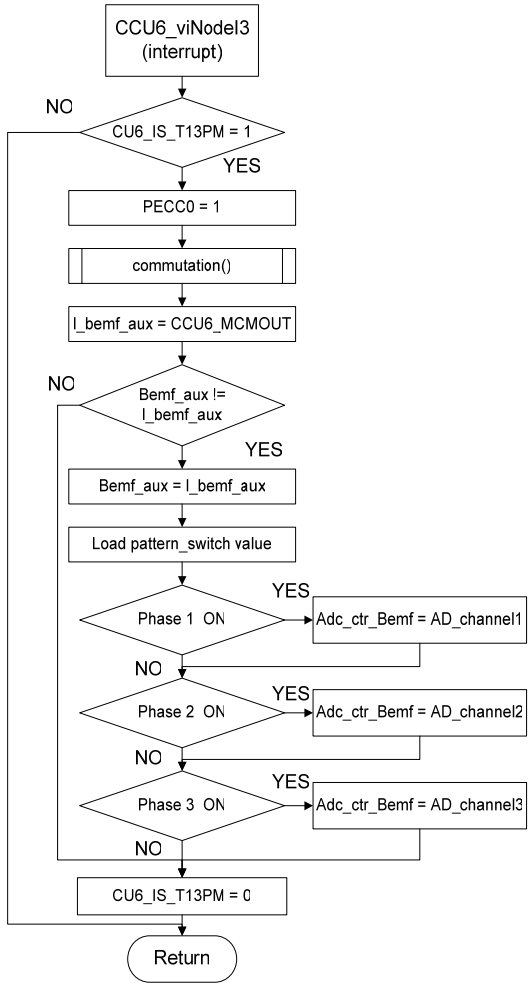
circ_index is a counter, which contains the offset for the circular memory.

Speed_acumm is the integral of the measured values for time determination.

8.1.14 Call graph T13 period match Interrupt Program (CCU6_viNodeI3)



8.1.15 Flow chart CCU6_viNodeI3 Program

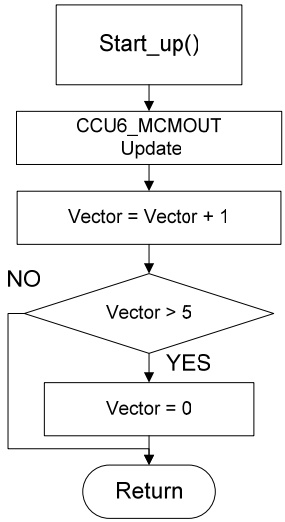


Name	Type	Init value	Change
Bemf_aux	**Int**	**0x0000**	**Intern**
Adc_ctr_Bemf	**Int**	**0x9691**	**Intern**

Bemf_aux is the memory for the old pulse pattern, which is compared with the new one.

Adc_ctr_Bemf contains the configuration of the ADC_CTR register to start a fixed channel single conversion. This happens with PECC2-Transfer.

8.1.16 Flow chart of Start_up Function

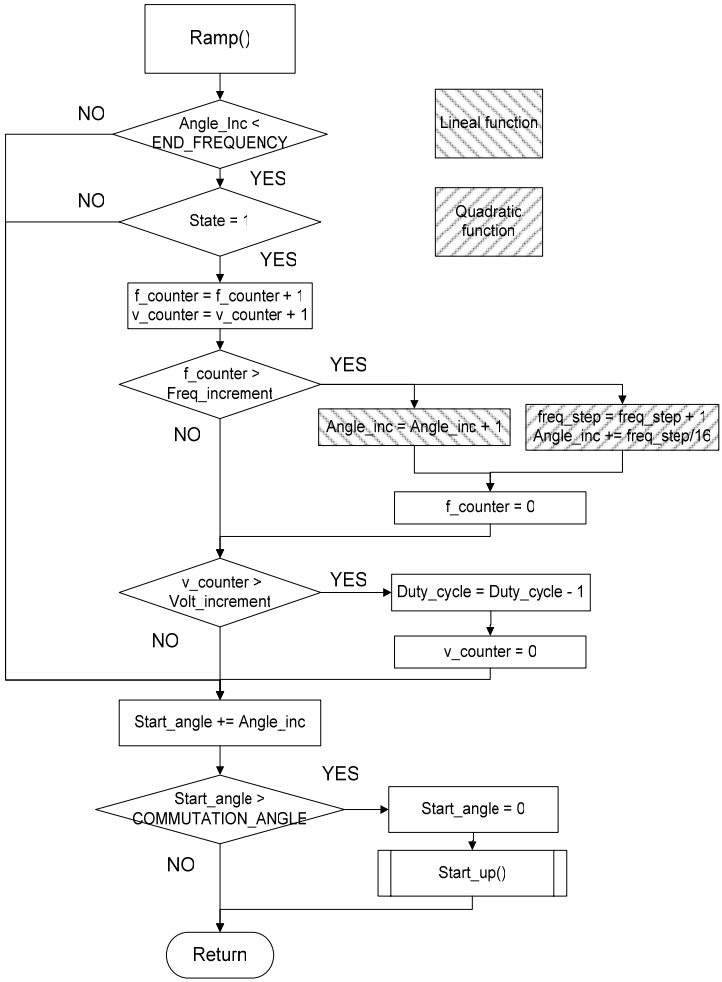


Name	Type	Init value	Change
direction	**Int**	**0x0000**	**Intern**
Vector	**Int**	**0x0000**	**Intern**

direction gives the rolling direction of the motor and it is the offset for the pulse pattern table.

Vector is a pointer, which rotates and indicates to the corresponding pulse pattern.

8.1.17 Flow chart of Ramp Function



Name	Type	Init value	Change
Freq_increment	**Int**		**User**
Volt_increment	**Int**		**User**
Start_freq	**Int**		**User**
Freq_step	**Int**	**0x0000**	**Intern**
v_counter	**Int**	**0x0000**	**Intern**
f_counter	**Int**	**0x0000**	**Intern**
State	**Int**	**0x0000**	**Intern**
Angle_inc	**Int**	**0x0000**	**Intern**
Start_angle	**Long**	**0x00490000**	**Intern**
Duty_cycle	**Unsigned Int**		**Intern**

Volt_increment is a variable to change the increase of voltage at the phase.

Freq_increment gives the value for the increase of frequency.

Angle_Inc represents a linear increase.

freq_step is an upstream integrator and represents a quadratic increase.

Start_angle specifies the frequency and the moment of commutation.

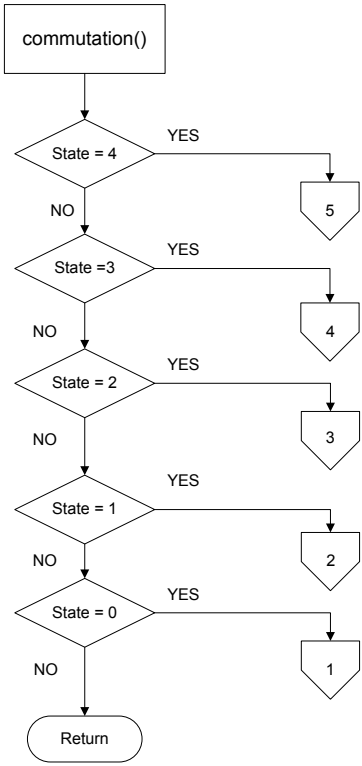
State is the state of the state machine.

Duty_cycle is the value of the duty cycle.

v_counter is a counter for time basis of voltage ramp.

f_counter is a counter for time basis of frequency ramp.

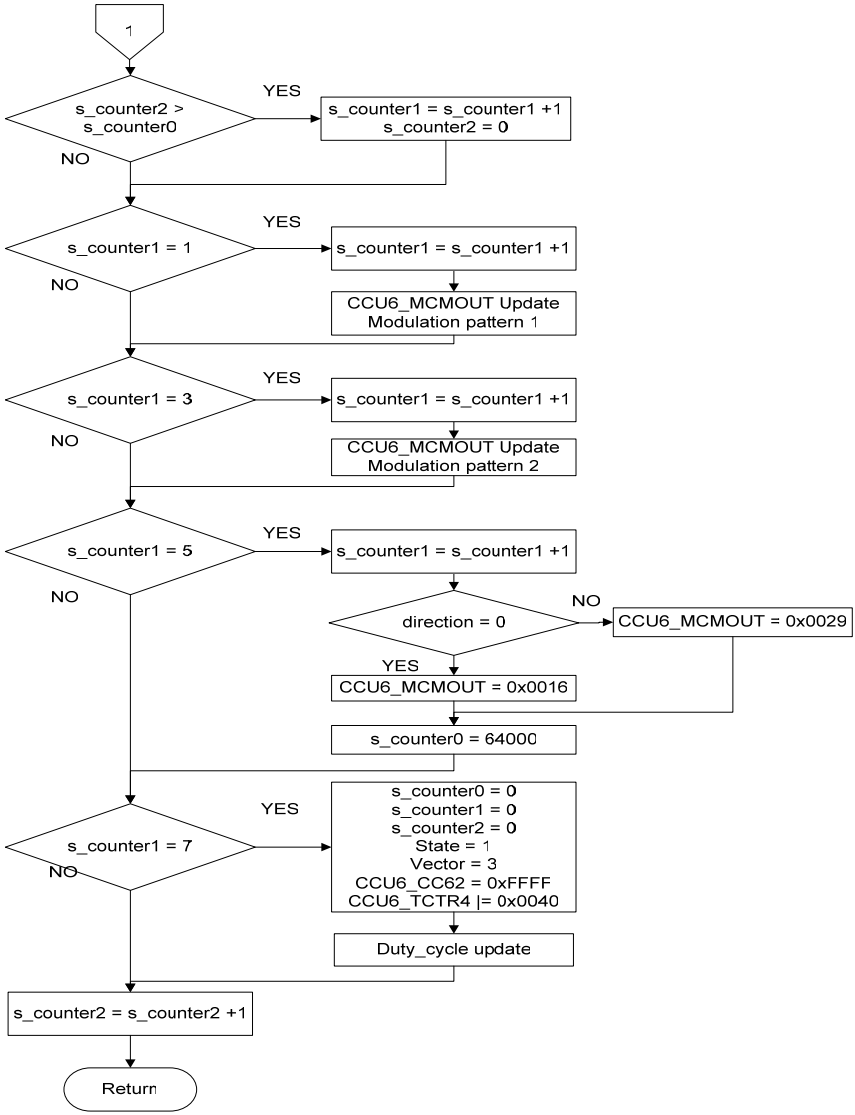
8.1.18 Flow chart of Commutation Function



Name	Type	Init value	Change
State	**Int**	**0x0000**	**Intern**

State is the state of the state machine.

8.1.19 Flow chart State 0 of Commutation Function (Positioning)



Name	Type	Init value	Change
s_counter0	**Unsigned Int**	**4000**	**User**
s_counter1	**Unsigned Int**	**0x0000**	**User**
s_counter2	**Unsigned Int**	**0x0000**	**Intern**
State	**Int**	**0x0000**	**Intern**
Duty_cycle	**Unsigned Int**		**Intern**
direction	**Int**	**0x0000**	**Intern**
Vector	**Int**	**0x0000**	**Intern**

State is the state of a state machine.

Duty_cycle is the duty cycle.

direction gives the rolling direction of the motor and it is the offset for the pulse pattern table.

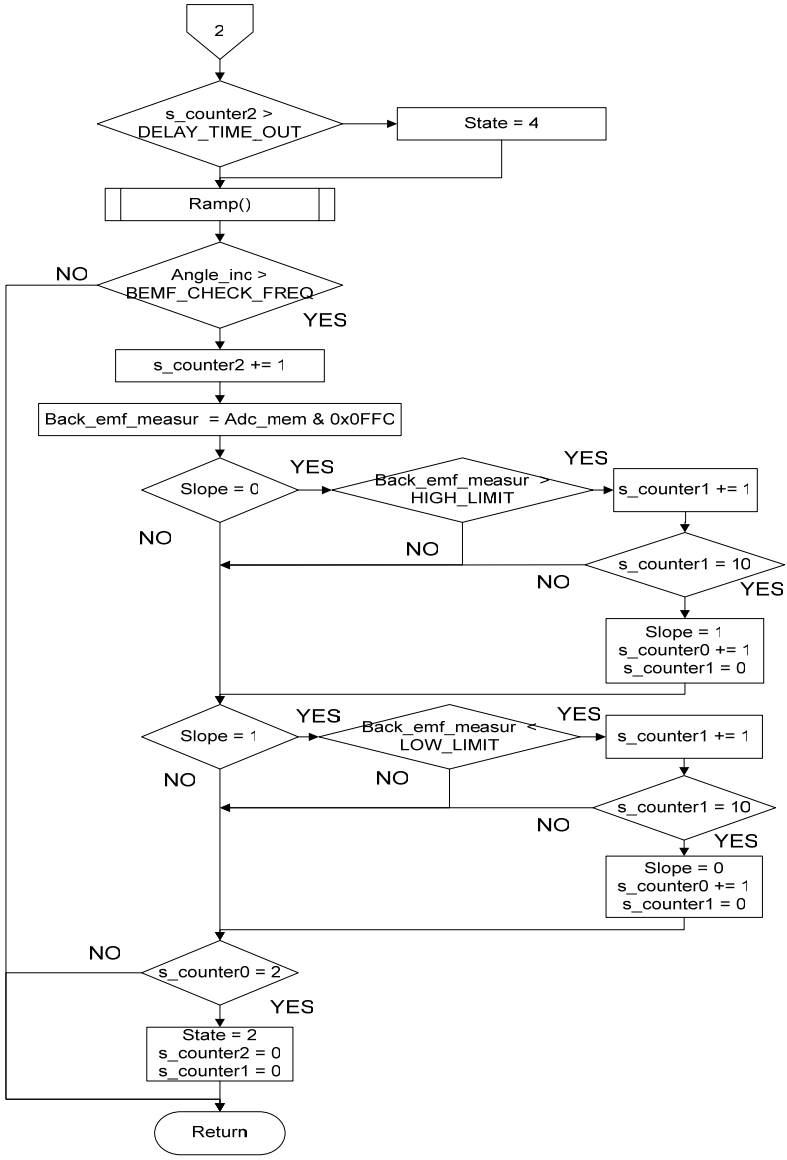
Vector is a pointer, which rotates and indicates to the corresponding pulse pattern.

s_counter0 gives the duration of each voltage vector.

s_counter1 is a counter for the switching of the voltage vector.

s_counter2 gives the time basis for the switching of the voltage vector.

8.1.20 Flow chart State 1 of Commutation Function (frequency ramp)



Name	Type	Init value	Change
s_counter0	**Unsigned Int**	**0x0000**	**User**
s_counter1	**Unsigned Int**	**0x0000**	**User**
State	**Int**	**0x0000**	**Intern**
Angle_inc	**Unsigned Int**	**0x0000**	**Intern**
Back_emf_measur	**Int**	**0x0000**	**Intern**
Slope	**Int**	**0x0000**	**Intern**

Slope is a state variable, which indicates the state of BACK-EMF measurement.

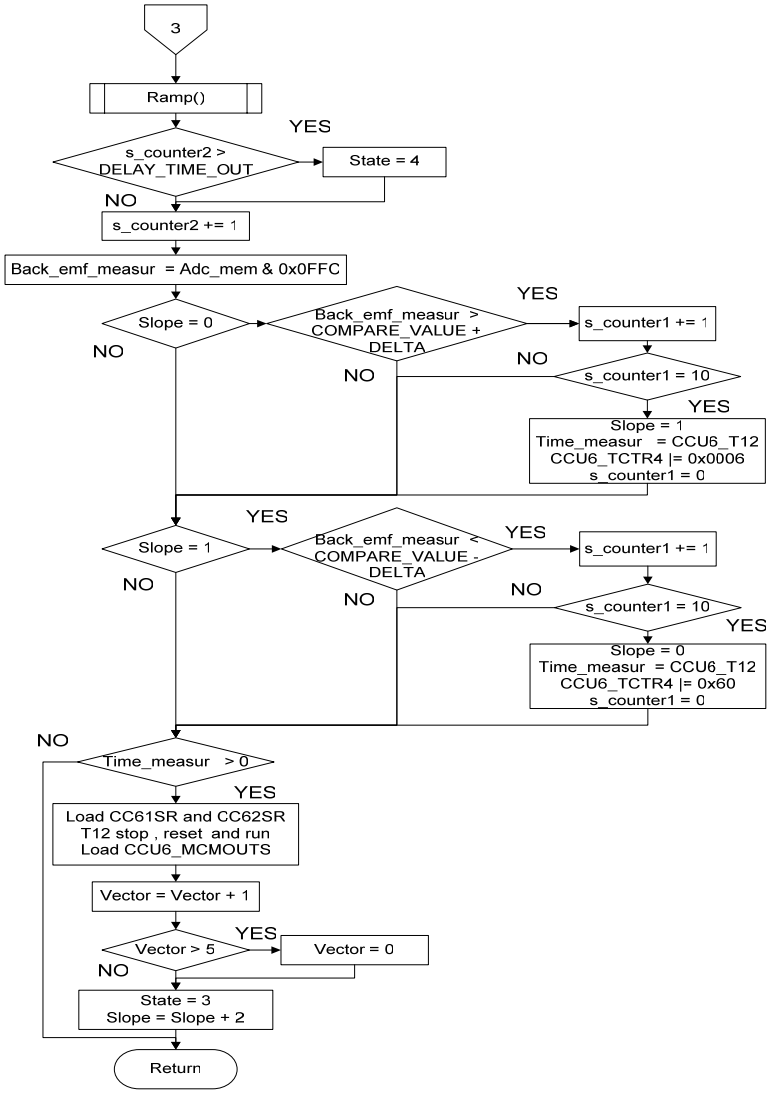
Angle_Inc is a value, which is proportional to the frequency.

Back_emf_measur is the measurement of BACK-EMF.

s_counter0 counts the events, when the BACK-EMF is higher or lower of the limit. ***s_counter1*** serves as filter for the comparison of the BACK-EMF measurement.

8.1.21 Flow chart State 2 of Commutation Function

The time measurement between two zero crossings of the B-EMF.



Name	Type	Init value	Change
s_counter2	Unsigned Int	0x0000	Intern
s_counter1	Unsigned Int	0x0000	Intern
State	Int	0x0000	Intern
Back_emf_measur	Int	0x0000	Intern
Slope	Int	0x0000	Intern
Vector	Int	0x0000	Intern
Time_measur	Unsigned Int	0x0000	Intern

Time_measur is the time between two zero crossings.

State is a state of the state machine.

Slope is a state variable, which indicates the state of BACK-EMF measurement.

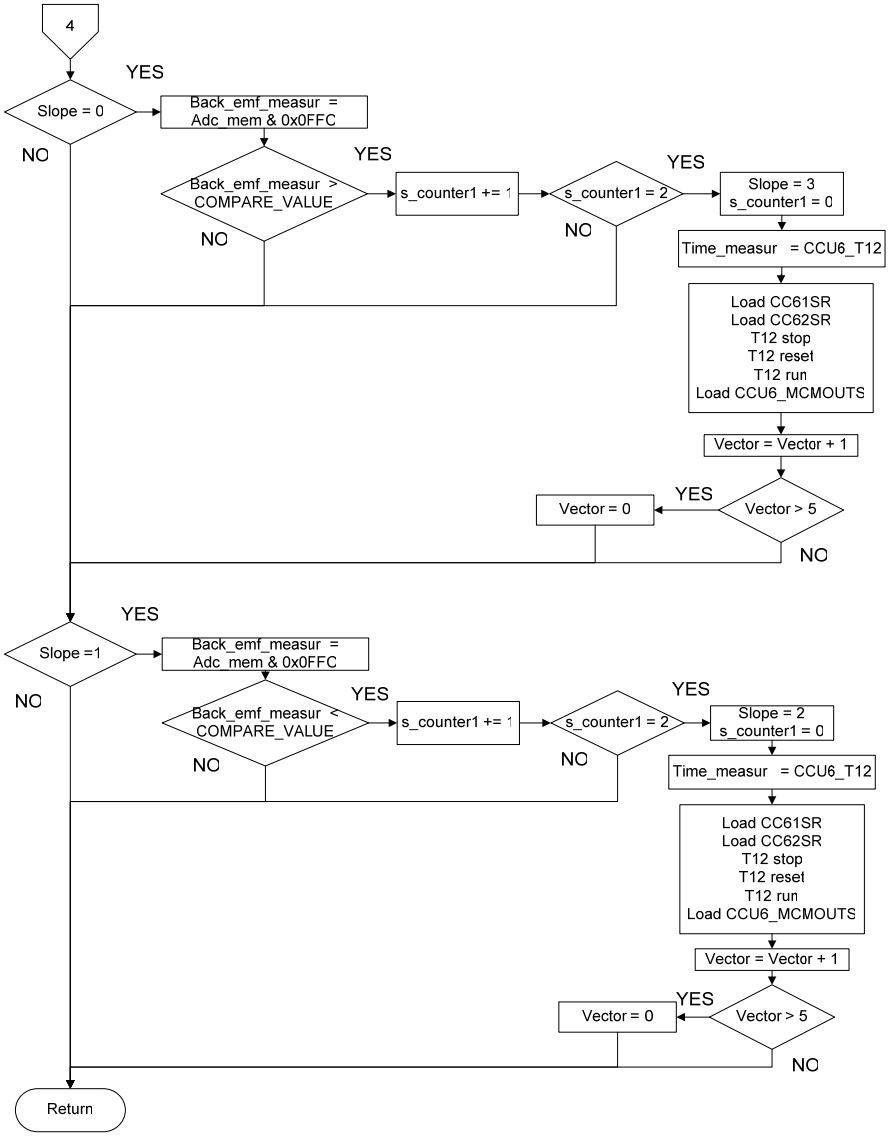
Vector is a pointer, which rotates and indicates to the corresponding pulse pattern.

Back_emf_measur is the measurement of the BACK-EMF.

s_counter2 is the counter for time out function.

s_counter1 serves as filter for the comparison of the BACK-EMF measurement.

8.1.22 Flow chart state 3 of Commutation Function (normal operation mode)



Name	Type	Init value	Change
s_counter1	Unsigned Int	0x0000	Intern
Back_emf_measur	Int		Intern
Slope	Int	0x0000	Intern
Vector	Int	0x0000	Intern
Time_measur	Unsigned Int	0x0000	Intern

Time_measur is the time between two zero crossings.

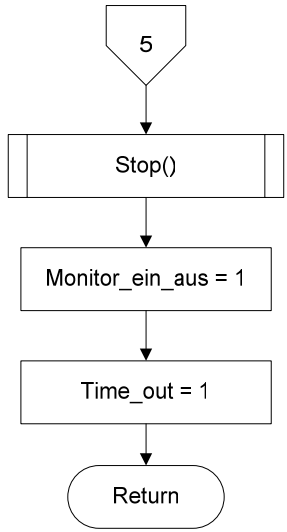
Slope is a state variable, which indicates the state of BACK-EMF measurement.

Vector is a pointer, which rotates and indicates to the corresponding pulse pattern.

Back_emf_measur is the measurement of the BACK-EMF.

s_counter1 serves as filter for the comparison of the BACK-EMF measurement.

8.1.23 Flow chart State 4 of Commutation Function

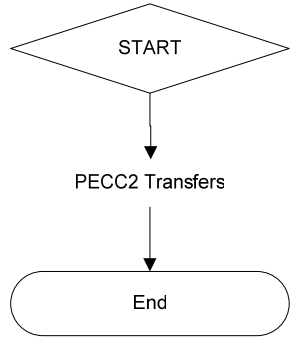


Name	Type	Init value	Change
Monitor_ein_aus	**Bit**	**0**	**User**
Time_out	**Bit**	**0**	**Intern**

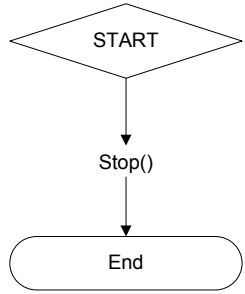
Time_out is the error flag of Timer 12 overflow.

monitor_ein_aus is the On and Off signal for the monitor program resp. the user.

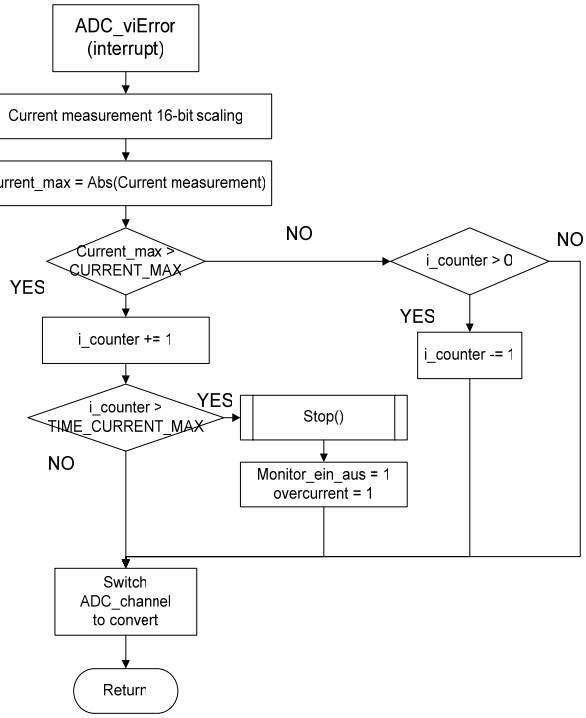
8.1.24 Call graph CC2-Interrupt



8.1.25 Call graph AD-Converter Interrupt (ADC_viError)



8.1.26 Flow chart ADC_viError Interrupt



Name	Type	Init value	Change
Current	**Unsigned Int**		**Intern**
i_counter	**Unsigned Int**	**0x0000**	**Intern**
Overcurrent	**Bit**	**0**	**Intern**
i_mpattern	**Int**	**0x0000**	**Intern**

Current is the actual measurement of current

i_counter is a time counter for current monitoring.

i_mpattern is the memory for the actual pulse pattern.

Overcurrent is the flag for over current (Status_word).

http://www.infineon.com

Published by Infineon Technologies AG