Application Note AN 2013-09
V1.0 September 2013

300W Motor Control Application Kit

Mitja Rebec, IFAT IPC APS AE
Ralf Walter, IFAT PMM APS SE DC

Edition 2011-02-02
Published by
Infineon Technologies Austria AG
9500 Villach, Austria

AN-LV-09-2013-V1.1-EN-049
Revision History: 13-09-01, V1.1
Previous Version: 05/2013, V1.0
Subjects: 300W Motor Control Application Kit

Authors: Mitja Rebec, IFAT IPC APS AE
Ralf Walter, IFAT PMM APS SE DC

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: [Ralf.Walter@infineon.com]

Table of contents

1 Overview....5

1.1 Key Features....5

2 300W Motor Control Application Kit....6

2.1 PCB....6

2.2 Communication Interface....7

2.3 Dedicated Software for the Onboard 8-bit Microcontroller....7

2.4 Infineon Board Control Graphical User Interface (IBC-GUI)....8

2.5 BLDC Motor (BLDCM)....8

3 Running the 300W Motor Control Application Kit....9

3.1 Connecting the board to a BLDC Motor, Power Supply and PC....9

3.2 IBC-GUI installation....10

3.3 First Running of IBC-GUI....10

3.4 Using PC GUI....10

3.4.1 Establishing communication with USB adapter....10

3.4.2 Selecting operational modes....11

3.4.3 Setting reference and reading actual board and motor values....13

3.4.4 Commands and Statuses....14

3.4.5 Parameters....15

4 Hardware Description....16

4.1 Power Supply....16

4.2 Two Level Three Phase Inverter....17

4.2.1 Current Measurement....18

4.3 XC836M and MOSFETs Driver 6ED003L02....19

4.3.1 Driver 6ED003L02....20

4.3.2 Microcontroller XC836M....21

4.4 Digital Outputs....22

4.5 Hall Sensors Digital Inputs....23

4.6 Communication Port....23

5 PCB....24

5.1 Schematic....24

5.2 Placement 25
5.3 Bill of Material....26
6 Microprocessor Software Description 27
6.1 Overview 27
6.2 Motor Control Algorithm 29
6.2.1 Unipolar Block Commutation.... 29
6.3 Scheduler 35
6.4 Serial Communication Protocol....36
6.5 Parameters....38

1 Overview

Brushless motors are taking over the cordless power-tool industry, with just about every manufacturer either already selling brushless tools or preparing to do so. Manufacturers claim brushless motors give more power, require less maintenance and extend the life of cordless tools.

Brushless motors differ from brushed motors in three main ways: computer circuitry replaces the commutator, the electromagnets are stationary and conventional magnets can move freely.

Relatively new to the power-tool industry, these motors are generating interest among customers and manufacturers. Complete lines of cordless tools are being developed, and because brushless motors can generate more power than brushed motors, some tasks previously thought too tough for a cordless tool are no longer off limits.

In a traditional cordless power tool motor, the power supply (battery) uses carbon brushes to conduct electricity to the commutator, which acts as an electric switch. The commutator changes the polarity of the electromagnets, which are attached to a free-spinning shaft and surrounded by fixed magnets, creating the magnetic field for pushing and pulling against.

Brushless motors eliminate the wasted energy created by the physical connection of carbon brushes in a brushed motor. Computer circuitry replaces the commutator. Since the electromagnets are stationary, brushes aren’t needed to deliver power. Conventional magnets spin freely within a ring of electromagnets because they don’t require an electrical connection, thereby generating power to the tool.

Infineon´s Power Tool Kit is addressing cordless power-tool industry by using simple plug-and-play system. Customers can easily run a BLDC motor and test the efficiency of Infineon devices mounted on a PCB.

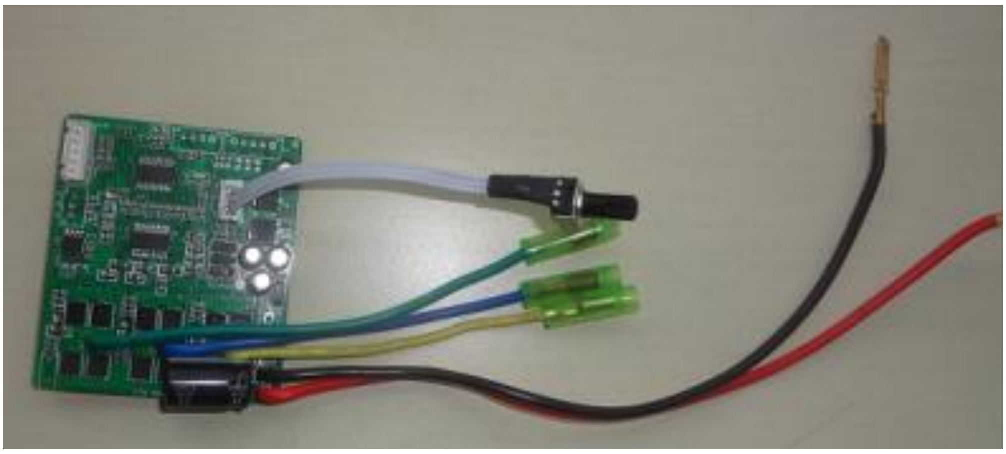


Figure 1.1: Power Tool Demo Board

1.1 Key Features

- High efficiency Infineon MOSFETs and MCU
- Easy plug and play software for fast testing of the board
- Suitable for BLDC motors with hall sensors
- Voltage, Current (Torque) and Speed control possible

2 300W Motor Control Application Kit

Within the 300W Motor Control Application Kit you can find:

- Power Tool PCB
- USB/Serial communication adapter
- USB memory stick including documentation PC and MCU software

The PCB functionality is controlled by using Infineon 8-bit microprocessor (XC863M2FRI), which is already programmed with **dedicated motor control software**.

2.1 PCB

The PC board has different functional parts in order to control BLDC motor:

- DC/DC converter transforming battery voltage into 5V (MCU supply) and 12V (driver supply).
- Two level three phase inverter using six Infineon OptiMOS™ Power-MOSFETs BSC016N06NS (100A, $R_{DS(on)}$=1.6mΩ). The inverter is equipped with shunt resistor on DC link return path for current measurement.
- Three phase inverter driver IC (6ED003L02) with over-current protection and fault signalization.
- 8-bit microprocessor (XC836M2FRI).
- Hall signal circuitry for running BLDC motors.
- Serial communication for programming and controlling the MCU software.
- Inputs: Motor Temperature Measurement, Potentiometer, 3 digital inputs.
- Outputs: 2 digital outputs.

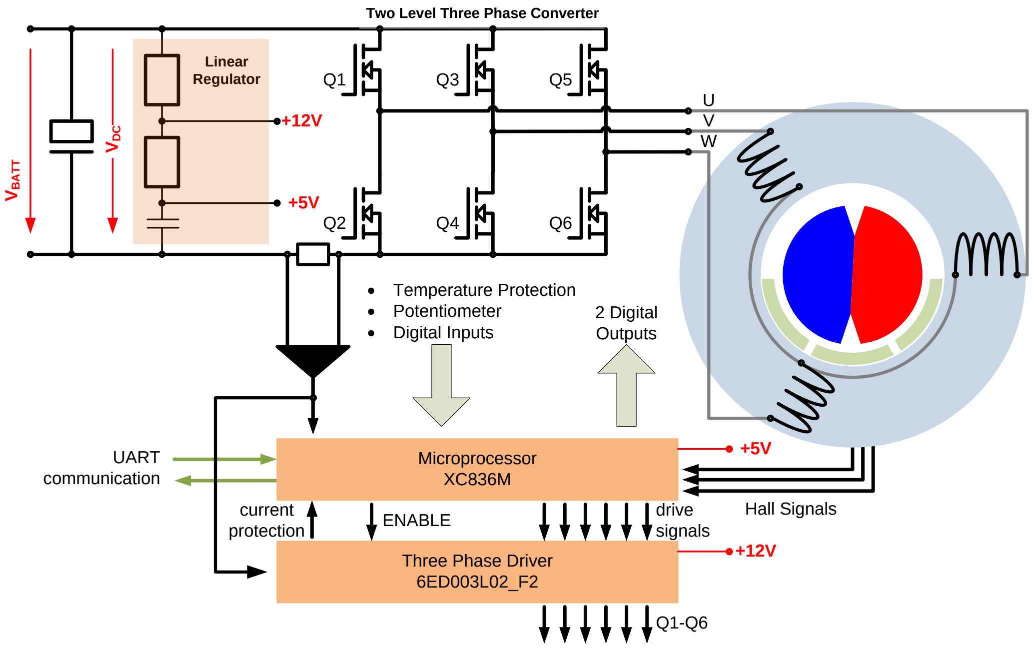


Figure 2.1: PCB Functional Blocks

2.2 Communication Interface

The Communication interface is a link between PC user interface and the board. It was realized with a commercially available cable from FTDI TTL-232R-5V.

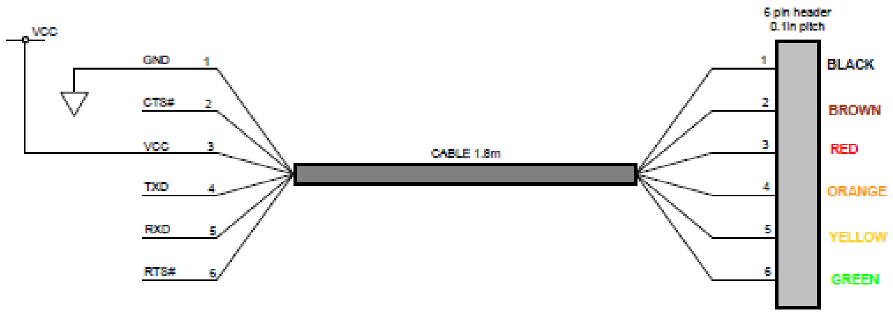
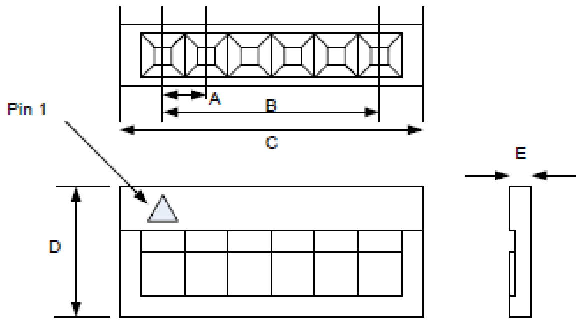


Figure 4.1 TTL-232R-5V and TTL-232R-3V3, 6 Way Header Pin Out

The mechanical details of the 6 way connector are shown in the following diagram



	Dimensions (mm)
A	2.54 +/- 0.05
B	12.70 +/- 0.2
C	15.24 +/- 0.2
D	14.0 +/- 0.2
E	2.54 +/- 0.05

Figure 4.2 TTL-232R-5V TTL-232R-3V3, 6 Way Header Mechanical Details

Figure 2.2: FTDI Communication Adapter

2.3 Dedicated Software for the Onboard 8-bit Microcontroller

The 8-bit MCU is already programmed with dedicated software. The software is implementing different functions:

- Block commutation for running BLDC motor with voltage, current or speed control.
- Hall auto-tuning in order indentify hall sequence using different motor.
- Unlocking the processor for programming using serial communication with user´s software.
- Changing and saving parameters to MCU flash permanently.
- Enabling/Disabling inputs for controlling motor.

2.4 Infineon Board Control Graphical User Interface (IBC-GUI)

The KIT comes with a GUI used to control MCU software execution using serial communication. IBC-GUI is a general program used to control Infineon demo or reference boards. In order to control the 300W Motor Control Application Kit, IBC-GUI must be opened with project file “PowerTool.iproj”. The project file defines parameter, commands, states, scaling factors and displayed values.

IBC-GUI can perform different tasks:

- Displaying and selecting motor operational modes.
- Reading and writing (set or clear) microprocessor different statuses.
- Real time values reading (current, voltage, frequency, etc.).
- Changing and saving parameters (switching frequency, dead time, etc.) to a project file or MCU permanent memory (FLASH).
- Selecting two different microprocessor operational modes: normal and program mode.

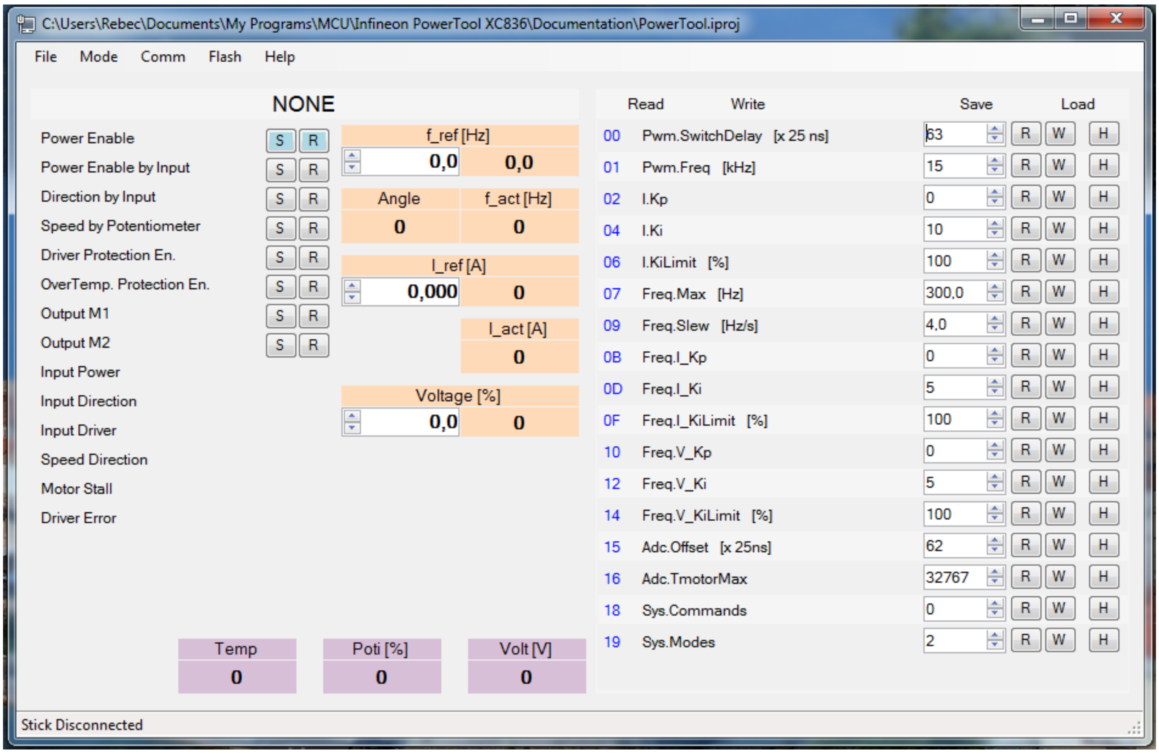


Figure 2.3: Infineon Board Control Graphical User Interface (IBC-GUI)

2.5 BLDC Motor (BLDCM)

The 300W Motor Control Application Kit can run every BLDC motor equipped with Hall Sensors.

3 Running the 300W Motor Control Application Kit

The 300W Motor Control Application Kit contains: Demo board, FTDI cable and USB memory stick. Running the KIT can be done in three steps:

1. The demo board must be connected properly to a BLDC motor, power supply and PC.
2. IBC-GUI (provided on USB memory stick) must be installed on the PC.
3. Tuning and running the motor.

3.1 Connecting the board to a BLDC Motor, Power Supply and PC

In order to run the board, connect the DC power supply (+18V, GND), 3 BLDC motor phases (U, V, W), BLDC motor hall sensors (+5V, GND, H1, H2, H3) and serial communication (GND, T_x, R_x). Additionally connect 2 digital inputs to control power ON/OFF and motor direction by using switches instead of GUI. When DC power supply is switched ON, the microprocessor starts to send messages using serial communication, therefore the UART/USB adapter hast to be connected to the PC and the dedicated GUÍ has to be installed for interpreting received messages.

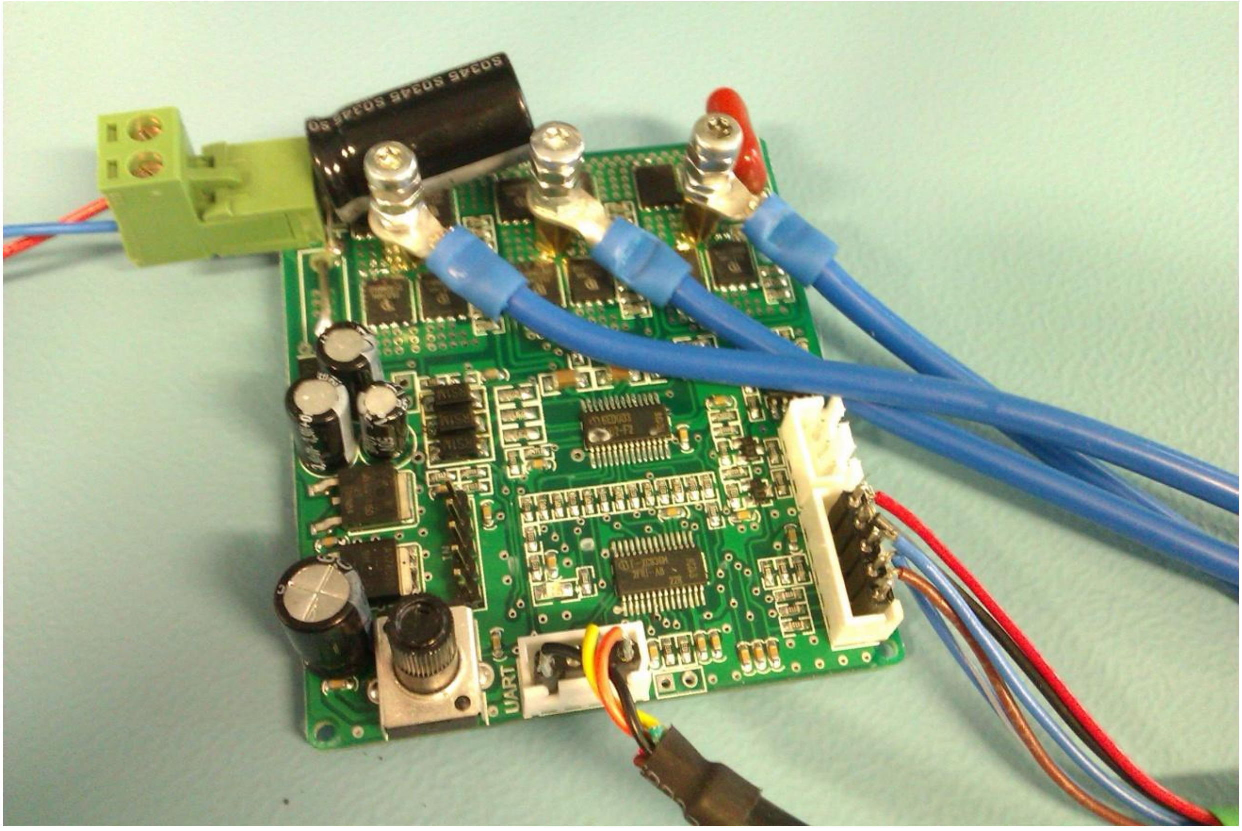


Figure 3.1: Board Wirings

3.2 IBC-GUI installation

To install the IBC-GUI, you must run the installation program "Setup.exe", which is provided on USB memory stick. The setup program will install USB/UART adapter driver and create a desktop shortcut "Infineon Demo Board Control".

3.3 First Running of IBC-GUI

If you run the PC software for the first time, you must select the correct project file. Project files have the extension "*.iproj". The project file for Power Tool KIT can be found in application folder

C:\Program Files\Infineon Technologies\Infineon Board Control\Projects\PowerTool.iproj. Double click it or run "Infineon Demo Board" program and select "File => Open". When selected, the project file path is written in the application header.

3.4 Using PC GUI

3.4.1 Establishing communication with USB adapter

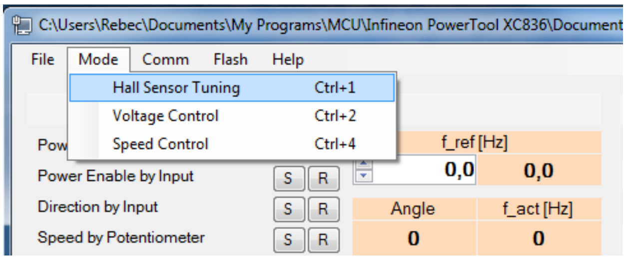
If the USB adapter is plugged into PC port, the application will automatically establish communication with it. If communication is not established automatically you can force the adapter to connect by selecting menu "Comm => Open Communication" or clicking F5. If the other part of the adapter is connected to functional

Power Tool PCB, you will see some activity in the PC GUI. Connection status is visible in GUI footer (left bottom corner of the GUI).

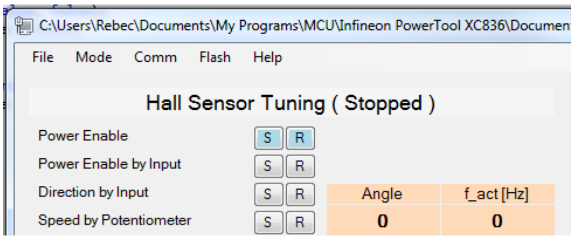
3.4.2 Selecting operational modes

There are three different operational modes that can be selected in order to control the motor. First of all the hall sequence has to be tuned according to the phase connection sequence. This operation will create the right commutation table. If the motor phases or hall signals connections change again, this operation must be repeated. The procedure is the following:

(1) Select Mode => Hall Sensor Tuning.

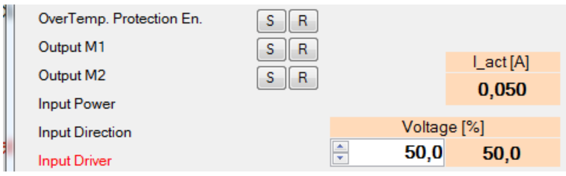


(2) Click button **S** (Set) for Power Enable command (this will run the selected operational mode). Power Enable will be colored in red (active).



(3) Increase the voltage [%] (duty cycle) till the motor rotor start to sweep and angle is changing (this mode is positioning the rotor in six different angles and hall sensor sequence is saved for each of those angles).

(4) When all angles are fired you can stop the power by clicking the button **R** (Reset) for Power Enable.



(5) After doing that commutation table is formed in RAM (temporary memory). In order to save the commutation table to FLASH (permanent memory) you must select menu Flash => Save to Flash.

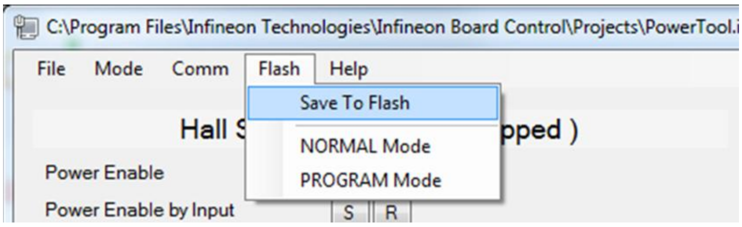


Figure 3.2: Hall Sensor Tuning

When the sensors are tuned, the motor can be run by selecting two remaining operational modes: **“Voltage Control”** and **“Speed Control”**. Operational mode will be started and stopped by clicking buttons **“S”** or **“R”** for **“Power Enable”** command like for hall tuning.

- **“Voltage Control”** is used to control the duty cycle or voltage [%] that is exciting motor stator. By increasing the voltage the motor start to rotate.
- **“Speed Control”** is used to control the frequency of the rotor using the speed feedback obtains from reading hall sensors. Duty cycle is regulated in order to keep this frequency stable to the selected one.

Modes and its statuses are shown in a banner bellow the GUI menus as followed: Mode (Status).

Mode: Speed Control, Status: Stopped

Mode: Speed Control, Status: Running

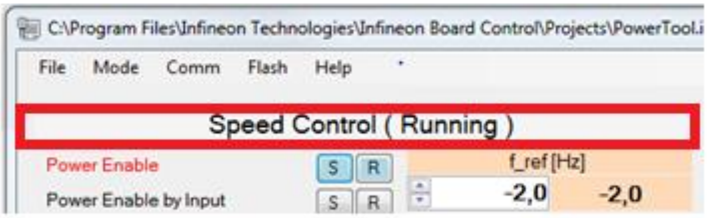


Figure 3.3: Operational Mode in Mode/Status Banner

Name	Remark
Operational Modes	
Hall Sensor Tuning	Creating commutation table
Voltage Control	Setting duty cycle
Speed Control	Setting rotor speed
Possible Statuses	
Stopped	Motor is stopped, driver is disabled, control algorithm is blocked
Boot	Boot sequence; Bootstrap capacitor is charging before the motor starts running
Start	All the main motor control variables are initialized
Running	Motor control algorithm is running, block commutation is active
Driver Error	MOSFETs driver current protection is active
Temperature Protection	Motor temperature is too high

Table 3.1: GUI Operational Mode and its Status banner

3.4.3 Setting reference and reading actual board and motor values

Some important values are sent from microprocessor periodically. Those values are displayed in orange GUI boxes. There are two values that can be set in microprocessor depending on operational modes: Voltage (duty cycle) and frequency.

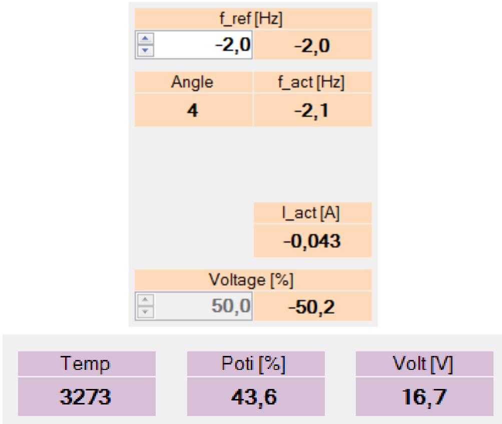


Figure 3.4: Values which can be read or written in real time

Name	Remark
	Monitored values
f_ref [Hz]	Ramped reference frequency used as an input to speed PI controller
Angle	Hall sensor combination
f_act [Hz]	Actual frequency calculated from hall signals
I_act [A]	Actual current read from shunt on negative DC link path
Voltage [%]	Duty cycle
Temp	Motor temperature
Poti [%]	Potentiometer
Volt [V]	DC link voltage
	Values that can be set
f_ref [Hz]	Reference frequency used in Speed Control operational mode
Voltage [%]	Duty Cycle used in Voltage Control operational mode

Table 3.2: MCU Command and Status Flags

3.4.4 Commands and Statuses

Behavior of MCU software can be controlled using a set of commands. Using status flags, we can easily monitor some main errors, warnings or messages that are happening when the MCU software is active.

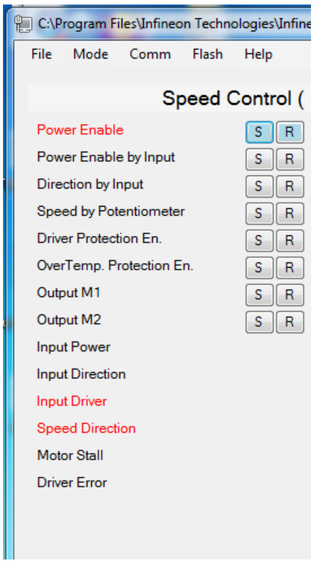


Figure 3.5: MCU Command and Status Flags

Name	***Remark***
Command	
[0] Power Enable	Enabling/Disabling operational mode (driver, algorithm)
[1] Power Enable by Input	Enabling/Disabling operational mode by external input
[2] Direction by Input	Selecting rotor direction by external input
[3] Speed by Potentiometer	Controlling speed or voltage using potentiometer
[4] Driver Protection En.	Enabling/Disabling driver protection (switching power off when it occurs)
[5] OverTemp. Protection En.	Motor over-temperature protection (switching power off when it occurs)
[6] Output M1	Output M1 ON/OFF
[7] Output M2	Output M2 ON/OFF
Status flags	
Input Power	External input for power is ON/OFF
Input Direction	External input for rotor direction is ON/OFF
Input Driver	Driver error output is ON/OFF (this output is normally ON)
Speed Direction	Direction of motor
Motor Stall	Motor is stand still or not
Driver Error	Status of driver error

Table 3.3: MCU Command and Status Flags

3.4.5 Parameters

MCU parameters are values that can be saved in FLASH and copied to working memory whenever the PCB start is operational. Those values can be changed and saved to FLASH in order to modify the PCB behavior. By clicking Read/Write, the GUI will read or write all parameters into MCU working memory. By clicking Save/Load, the GUI will save or load all parameter in project file. To save this parameters to FLASH (permanent memory) select menu “Flash => Save To Flash”.

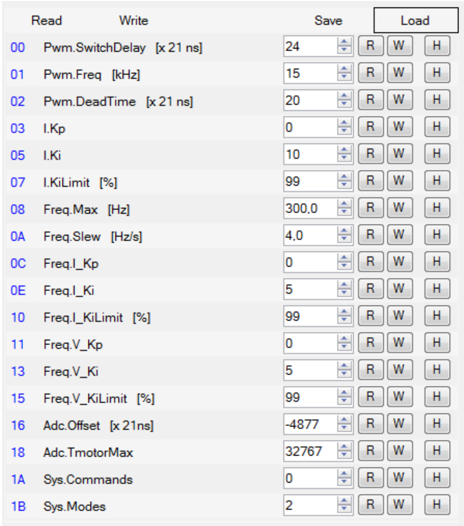


Figure 3.6: MCU Parameters

Name	Remark
Pwm.SwitchDelay [x21ns]	Setting the exact moment when current is sensed from shunt
Pwm.Freq [kHz]	PWM switching frequency
Pwm.DeadTime [x21ns]	Dead time between high side and low side
I.Kp	(NOT USED) Current controller proportional parameter
I.Ki	(NOT USED) Current controller integral parameter
I.KiLimit [%]	(NOT USED) Current controller integral limit parameter
Freq.Max [Hz]	Maximum motor frequency
Freq.Slew [Hz/s]	Frequency ramp
Freq.I_Kp	(NOT USED) Frequency current controller proportional parameter
Freq.I_Ki	(NOT USED) Frequency current controller integral parameter
Freq.I_KiLimit [%]	(NOT USED) Frequency current controller integral limit parameter
Freq.V_Kp	Frequency voltage controller proportional parameter
Freq.V_Ki	Frequency voltage controller integral parameter
Freq.V_KiLimit [%]	Frequency voltage controller integral limit parameter
Adc.Offset	Setting the zero for sensed current
Adc.TmotorMax	Maximum motor temperature (after this error is signalized)
Sys.Commands	Start up command when PCB start working
Sys.Modes	Start up operational mode

Table 3.4: MCU Parameters

4 Hardware Description

4.1 Power Supply

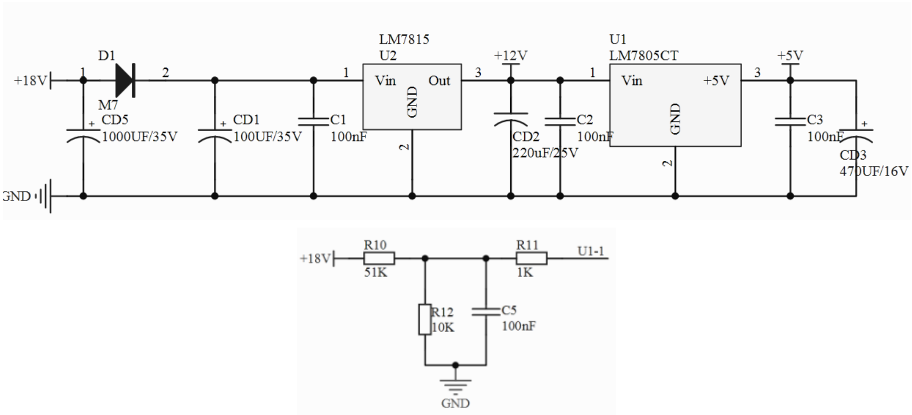


Figure 4.1: Auxiliary Power

The nominal input voltage is 18-24V. This voltage is used to supply the two level three phase inverter. Supplied voltage is converted to 12V and 5V by using voltage regulators LM7812. 12V is used to supply the MOSFET drivers and 5V is used to supply MCU and hall sensors.

Name	Remark
Elements	
Voltage regulator LM7812	Transforming 18V into 12V
Voltage regulator LM7805CT	Transforming 12V into 5V
Bulk capacitor	1000uF / 35V
Input Signals	
Mains power supply	18-24V supplied from battery
Ground	-
Output Signals	
DC link voltage	18V supplying 2 level three phase inverter
DC link voltage sense (U1-1)	5V on MCU is 30.5V on DC link
12V voltage line	Supplying MOSFETs driver
5V voltage line	Supplying MCU, hall sensors, potentiometer, …
Ground (GND)	0V

Table 4.1: Auxiliary Power

4.2 Two Level Three Phase Inverter

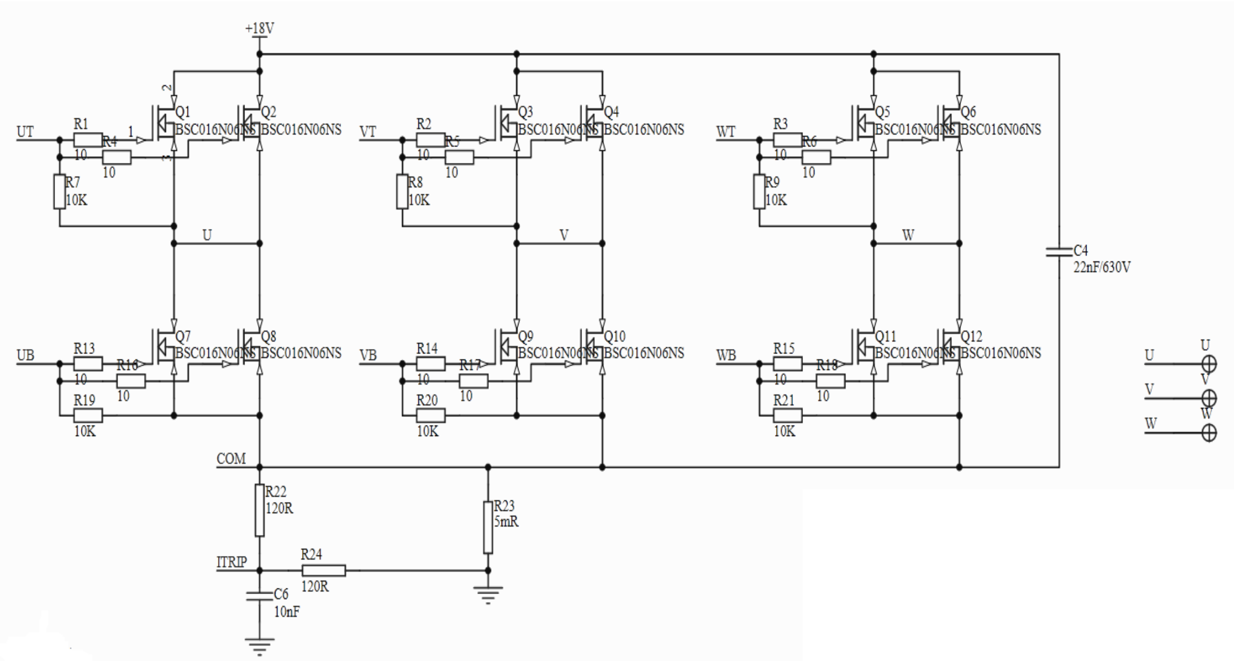


Figure 4.2: Schematic Two Level Three Phase Inverter

This inverter stage is designed to drive BLDC motors up to 300W continuously without additional heatsinks or fans using two MOSFETs in parallel. Higher motor power ratings are possible using an optimized thermal management (heatsink, fan etc.). There is a shunt on negative DC link path used for current measurement and driver current protection signal too.

Name	Remark
Parameters	
MOSFETs current rating (BSC016N06NS)	100A (package limited)
MOSFETs $R_{DS(ON)}$	1,6mΩ
MOSFETs V_{DS}	60V
Input Signals	
DC Link Voltage	18V
Ground	GND
Gate drive signals for high side MOSFET x 3 (UT, VT, WT)	Applied voltage 12V to U, V, W
Gate drive signals for low side MOSFET x 3 (UB,VB,WB)	Applied voltage 12V to COM
Output Signals	
Positive shunt voltage (COM)	499mV / 1A supplying operational amplifier
Current protection (ITRIP)	248mV / 1A supplying driver protection input
Phases U, V, W	

Table 4.2: Description of the Two Level Three Phase Inverter Circuitry

4.2.1 Current Measurement

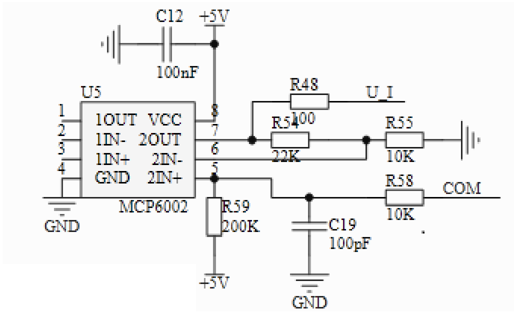


Figure 4.3: Current Measurements

Operational amplifier is adapting and shifting the current signal in a range of MCU analog to digital channel. The signal´s offset is increased in order to measure negative current as well. The offset is corrected in MCU by subtracting the transformed value for $\left({}^{0{,}762V * 2^{15}}/_{5V} = 4993\right)$. This is done using parameter "Adc.Offset".

Name	Remark
Parameters	
Operational amplifier (MPC6002)	
Input Signals	
Positive shunt voltage (COM)	499mV / 1A supplying operational amplifier
Ground	GND
5V power supply	
Output Signals	
Output voltage (U_I)	$U\_I = \left(1 + \frac{22k}{10k}\right)\left(COM \cdot \frac{200k}{200k + 10k} + 5V \cdot \frac{10k}{200k + 10k}\right)$ $U\_I = COM \cdot 3{,}05 + 0{,}762V$ $U\_I = I \cdot 1{,}522 + 0{,}762V$ This voltage is supplied to ADC MCU input

Table 4.3: Shunt Current Measurements

4.3 XC836M and MOSFETs Driver 6ED003L02

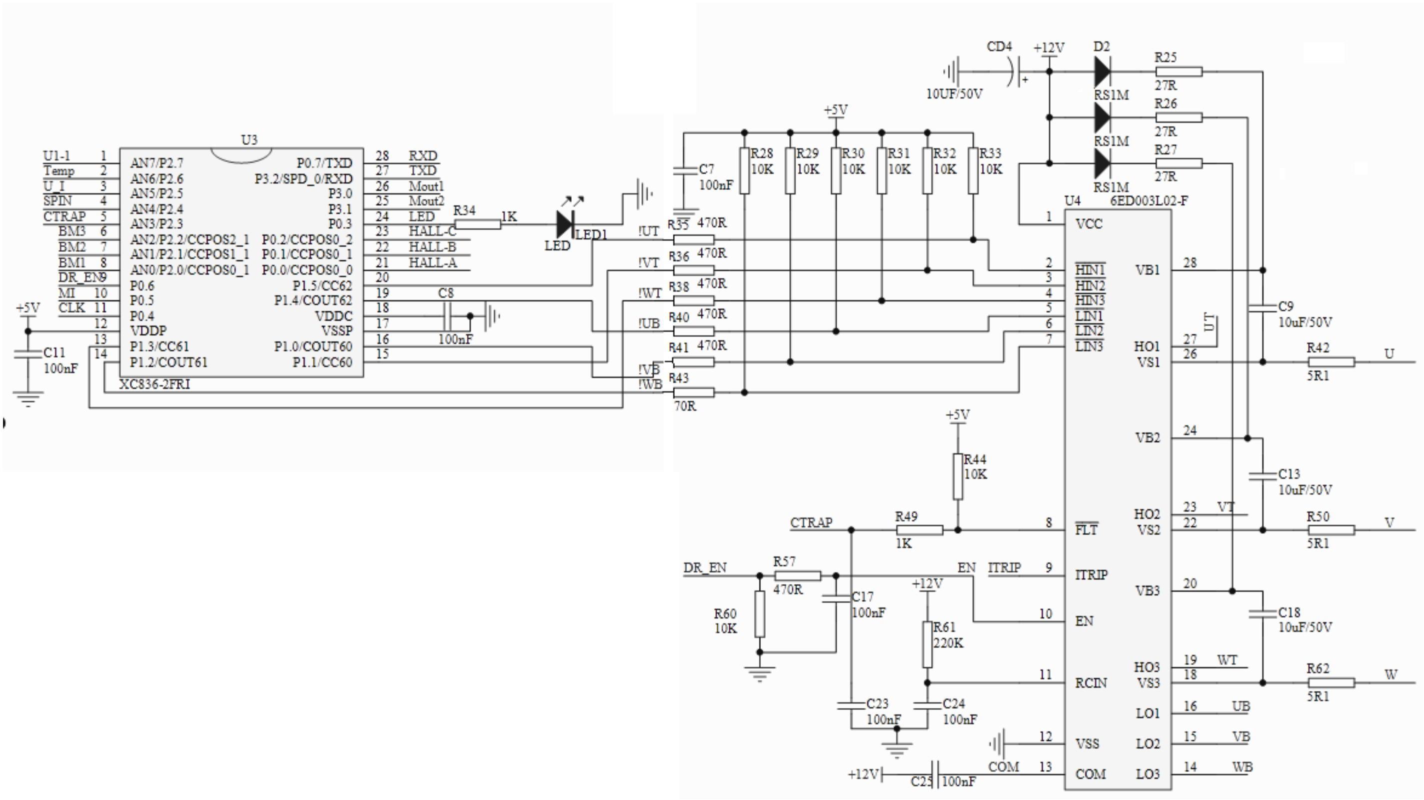


Figure 4.4: XC836 and MOSFET Driver 6ED003L02

4.3.1 Driver 6ED003L02

The gate driver (6ED003L02-F) is a two level three phase inverter driver to control power devices such as MOSFETs. The main features of this device are:

- Thin-film-SOI-technology
- Insensitivity of the bridge output to negative voltages up to -50V given by SOI-technology
- Maximum blocking voltage +180V
- Power supply of the high side drivers via boot strap
- CMOS and LSTTL compatible input (negative logic)
- Signal interlocking of every phase to prevent cross-conduction
- Detection of over-current and under-voltage supply
- 'Shut down' of all switches during error conditions
- Externally programmable delay for fault clear after over current detection

Name	Remark
Parameters	
ITRIP shut down threshold	0.46V
ITRIP input low pass corner frequency fc	3.386 MHz
Input Signals	
Power supply logic VCC	12V supplying driver and MOSFETs gate
Ground	GND
Driver enable (active high)	5V/0V supplied by MCU
6 control signals for low and high side switch	5V/0V supplied by MCU
Current protection (ITRIP)	248mV / 1A supplied by two level three phase inverter
Output Signals	
Gate drive signals for high side MOSFETs x 3 (UT, WT, VT)	12V to U, V, W
Gate drive signals for low side MOSFETs x 3 (UB, WB, VB)	12V to COM
Inverter fault output (/CTRAP) (active low)	5V / 0V

Table 4.4: Description of Driver Section

4.3.2 Microcontroller XC836M

The Demoboard is controlled by 8-bit Infineon microcontroller XC836M. The XC83x-Series makes the entry level for Infineon's 8-bit microcontroller family XC800 with real-time-control capabilities. The vector computer co-processor (MDU + CORDIC) boosts up standard 8-bit processing performance and supports field oriented motor control at 8-bit cost. This enables more efficient and intelligent designs for motor control (e.g. sensorless motors).

Name	Remark
Parameters	
XC836M system clock frequency (f_{SYS})	Internal oscillator runs with 48 MHz important for instruction execution time. Typical instruction is $2/f_{SYS}$ = 20,8ns.
Input Signals	
Voltage supply 5V	5V DC
Ground (GND)	0V
MOSFETs driver error signal (/CTRAP)	Digital input 0V or 5V supplied by MOSFETs driver circuitry
DC link voltage cense (U1-1)	Analog input from 0V to 5V supplied by power supply circuitry
DC link current cense (U_I)	Analog input from 0V to 5V supplied by 2 level 3 phase inverter circuitry
Temperature sense (Temp)	Analog input from 0V to 5V supplied by temperature protection circuitry
Potentiometer input (SPIN)	Analog input from 0V to 5V supplied by speed control circuitry
Digital input (BMI3)	Digital input 0V or 5V supplied by digital inputs circuitry
Digital input (BMI2)	Digital input 0V or 5V supplied by digital inputs circuitry
Hall sensor signals inputs (HALL A,B,C)	Digital inputs 0V or 5V supplied by hall circuitry
Output Signals	
Driver Enable Signal (active high)	Enabling/disabling MOSFET driver
Driver control signals (COUT6x and CC6x)	Inputs to MOSFET driver
Digital outputs (Mout1,2)	To digital output circuitry
LED digital output (LED)	
Communication	
Receive Input (RXD)	From Communication Port
Transmit Output (TXD)	To Communication Port

Table 4.5: Description of XC836 Circuitry

4.4 Digital Outputs

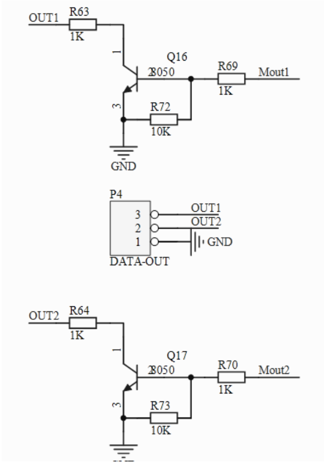


Figure 4.5: Digital Outputs

Name	Remark
Input Signals	
Digital outputs (Mout1,2)	0V/5V supplied from MCU
Ground (GND)	
Output Signals	
Digital outputs (OUT1,2)	Optional Output (like LED Control Light)
Ground (GND)	

Table 4.6: Digital Outputs

4.5 Hall Sensors Digital Inputs

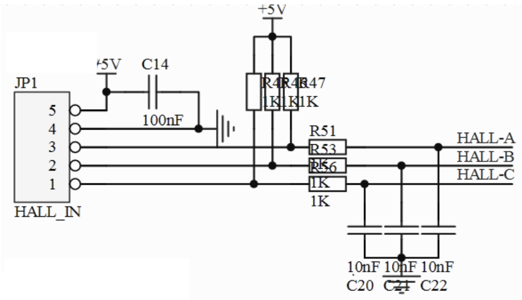


Figure 4.6: Hall Sensor Inputs

Name	Remark
Input Signals	
Hall sensor inputs (PIN 1,2,3)	0V/5V
5V power supply	
Ground (GND)	
Output Signals	
Hall sensor outputs (HALL A,B,C)	0V/5V supplied to MCU

Table 4.7: Hall Sensor Inputs

4.6 Communication Port

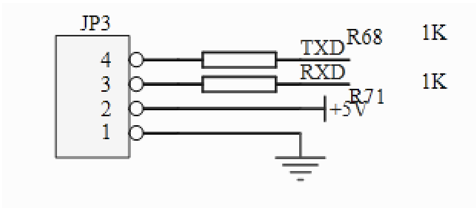


Figure 4.7: Communication Port

Pin Num.	Name
4	Transmit Output (RTS)
3	Receive Input (RXD)
1	Ground (GND)

Table 4.8: Description of Communication Port

5 PCB

5.1 Schematic

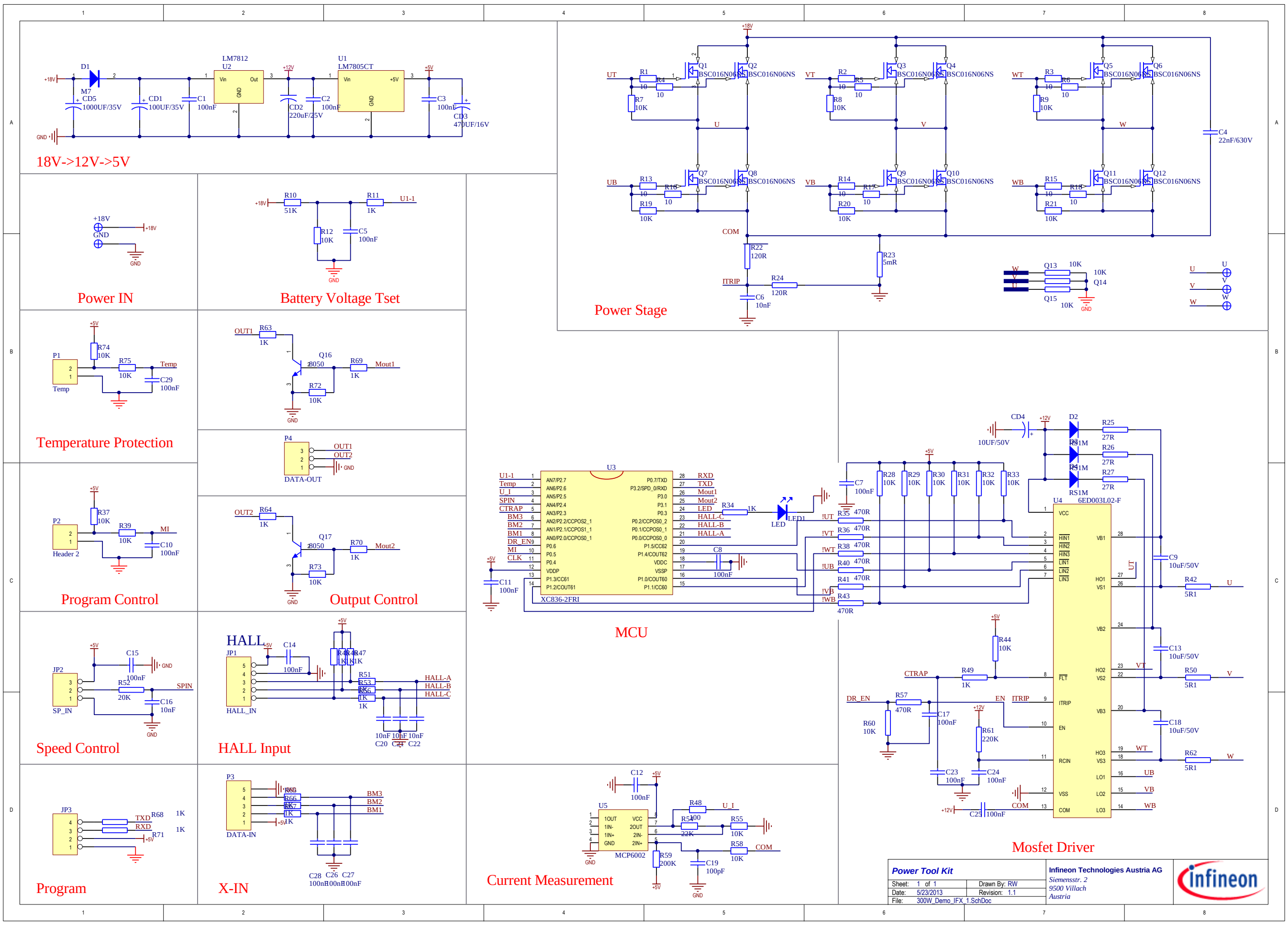


Figure 5.1: Schematic

5.2 Placement

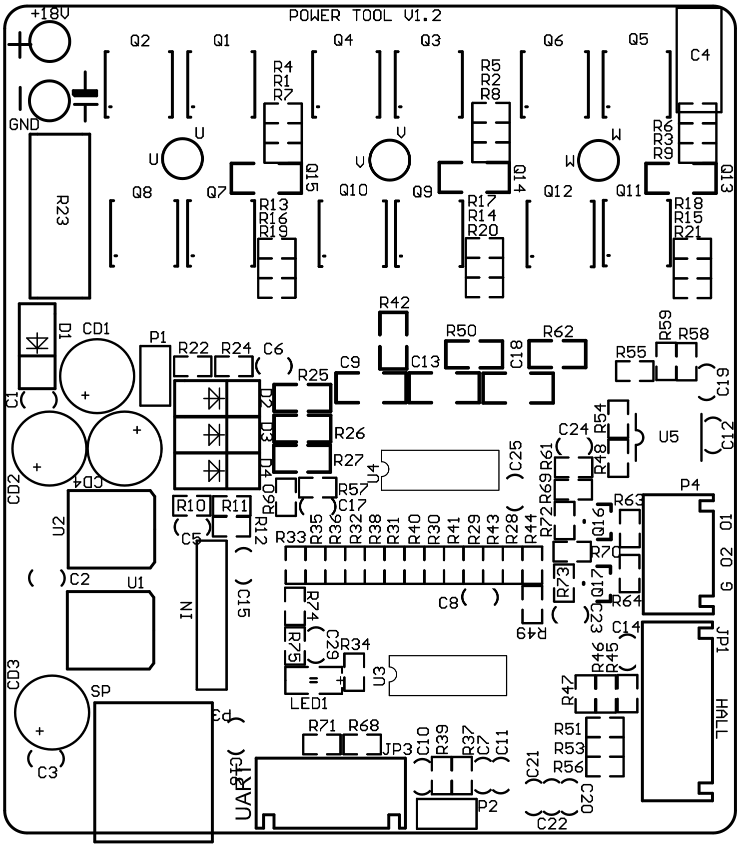


Figure 5.2: Placement

5.3 Bill of Material

Comment	Description	Designator	Footprint	Quantity
		+18V, GND, U, V, W	PIN-1	10
100nF	Capacitor	C1, C2, C3, C5, C7, C8, C10, C11, C12, C14, C15, C17, C23, C24, C25, C26, C27, C28, C29	C0603	38
22nF/630V	Capacitor	C4	RAD-0.2	2
10nF	Capacitor	C6, C16, C20, C21, C22	C0603	10
10uF/50V	Capacitor	C9, C13, C18	1206	6
100pF	Capacitor	C19	C0603	2
100uF/35V	Capacitor	CD1	6.3*11	2
220uF/25V	Capacitor	CD2	6.3*11	2
470uF/16V	Capacitor	CD3	6.3*11	2
10uF/50V	Capacitor	CD4	6.3*11	2
1000uF/35V	Capacitor	CD5	10*12	2
M7	Diode	D1	DO-214	2
RS1M	1 Amp General Purpose Rectifier	D2, D3, D4	DO-214	6
HALL_IN		JP1	XH2.54-5P	2
SP_IN		JP2	XH2.54-3P	2
	4 Pin Header	JP3	XH2.54-4P	2
LED		LED1	L0805	2
Temp	Header, 2-Pin	P1	HDR1X2	2
Header 2	Header, 2-Pin	P2	HDR1X2	2
DATA-IN	Header, 5-Pin	P3	XH2.54-5P	2
DATA-OUT		P4	XH2.54-3P	2
BSC016N06NS		Q1, Q2, Q3, Q4, Q5, Q6, Q7, Q8, Q9, Q10, Q11, Q12	TDSON-8	24
10K		Q13, Q14, Q15	1206	6
`8050	NPN General Purpose Amplifier	Q16, Q17	SOT-23-A	4
10R	Resistor	R1, R2, R3, R4, R5, R6, R13, R14, R15, R16, R17, R18	603	24
10K	Resistor	R7, R8, R9, R12, R19, R20, R21, R28, R29, R30, R31, R32, R33, R37, R39, R44, R55, R58, R60, R72, R73, R74, R75	603	46
51K	Resistor	R10	603	2
1K	Resistor	R11, R34, R45, R46, R47, R49, R51, R53, R56, R63, R64, R65, R66, R67, R68, R69, R70, R71	603	36
120R		R22, R24	603	4
5mR		R23	0.02/2W	2
27R		R25, R26, R27	805	6
470R		R35, R36, R38, R40, R41, R43, R57	603	14
5R1		R42, R50, R62	805	6
100	Resistor	R48	603	2
20K		R52	603	2
22K	Resistor	R54	603	2
200K	Resistor	R59	603	2
220K		R61	603	2
LM7805CT		U1	TO-252AA	2
LM7812		U2	TO-252AA	1
LM7815		U2	TO-252AA	1
XC836-2FRI		U3	TSSOP-28	2
6ED003L02-F		U4	TSSOP-28	2
MCP6002		U5	SO-8	2

Table 5.1: Bill of Material

6 Microprocessor Software Description

6.1 Overview

Motor control software task is to get information or commands from user and control the motor according to them.

The core of the software is motor control algorithm calculating duty cycle according to different control schemes. Fast motor control respond is obtained by reading analogue values like DC link current, voltage, etc. using A/D converter and calculating duty cycle calculation every second PWM period. Before running motor control algorithm, there is a sequence of operations that ensure safe motor start/stop (control parameters initialization, charging bootstrap capacitors, switching patterns configuration …). This sequence is controlled by the scheduler. User can control software behavior using serial communication.

Above described software function are distributed among different MCU events and main loop according to their priority of execution:

- Main function is the start up function initializing all the peripheral, activating other units and jumping into never endless loop with the lowest priority (Can be overrun by any other unit).
- Scheduler is split between main loop and timer T1 event because there are some parts of it that require higher priority. This event is triggered every 1ms.
- Serial communication has its own event which is triggered when the message is received. Sending messages is done in main loop periodically.
- Motor control algorithm is linked to timer T12 event (part of CCU6) executing it periodically every second PWM period.

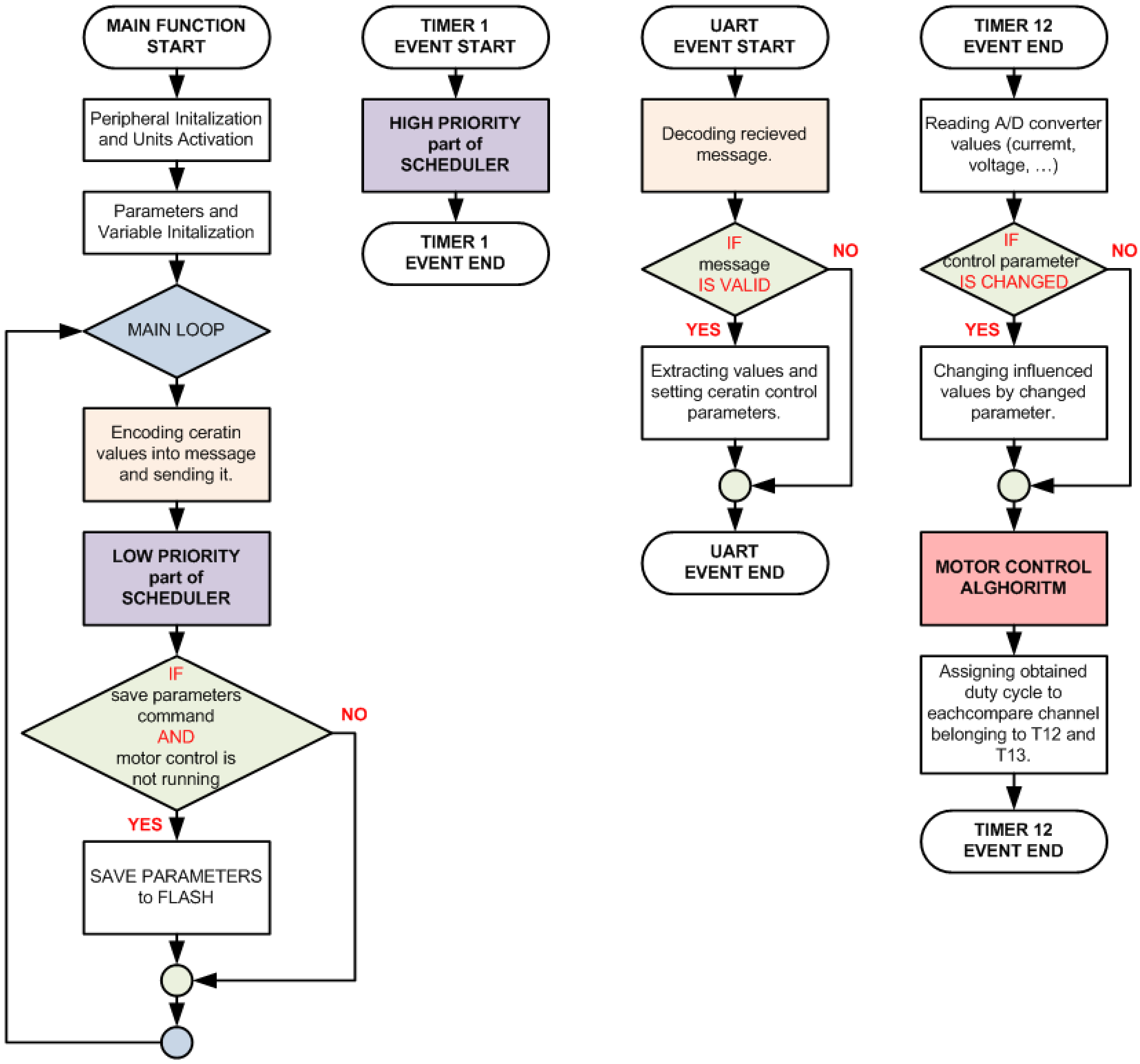


Figure 6.1: MCU Software Flow Chart

Table 6.1 represents a description of events and its priority starting from 1 as the highest priority. High priority tasks can interrupt low priority tasks. Tasks with the same priority have to wait for the first one started to be finished.

Priority	MCU unit, event, trigger	Description
1	Capture/Compare Unit 6 - Timer T12, Period Match Event, every 2nd PWM period	The CCU6 unit is made up of a Timer T12 Block with three compare channels and a Timer T13 Block. The T12 channels can jointly generate control signal patterns to drive AC-motors or inverters. T13 is used for current measurement synchronized with T12, its duty cycle and A/D converter in order to sense the current in the right moment. T12 event is executed every two PWM periods and is used to execute motor control algorithm and consequently to obtain and set T12 duty cycles for each compare channel.
2	Timer T1, Period Match Event, every 1ms	This timer is set to trigger event every 1 ms. Within this event high priority scheduler part of algorithm is executed.
2	UART, Message received event	This event is triggered by a received message from PC using serial communication.
Lowest	Main function, Main Loop	Main loop is a never endless loop that is running only when the other events are not running. Within this loop several tasks are executed: Sending messages to PC using serial communication. Low priority scheduler. Write parameters to MCU permanent memory (FLASH). Other

Table 6.1: MCU Software Events Priority Description

6.2 Motor Control Algorithm

The 300W Motor Control Application Kit was designed to run Brushless DC Motors (BLDCM). When rotating, BLDCM induces trapezoidal voltage waveform. Therefore a modulation is needed, which creates a square voltage waveform. This is called Block Commutation Control.

6.2.1 Unipolar Block Commutation

The trapezoidal commutation method is the simplest way to control BLDC motors and easy to implement the control aspects of it. For proper commutation and for motor rotation, the rotor position information is very crucial. Only with the help of rotor position information, the electronic switches in the inverter bridge will be switched correctly to ensure proper direction of current flow in respective coils. Three hall sensors are used in general as position sensor. Each hall sensor is typically placed 120° apart and senses the position of the rotor field. Position information is needed to keep the angle between the rotor and stator magnetic field between 60° and 120° in order to get the maximum torque.

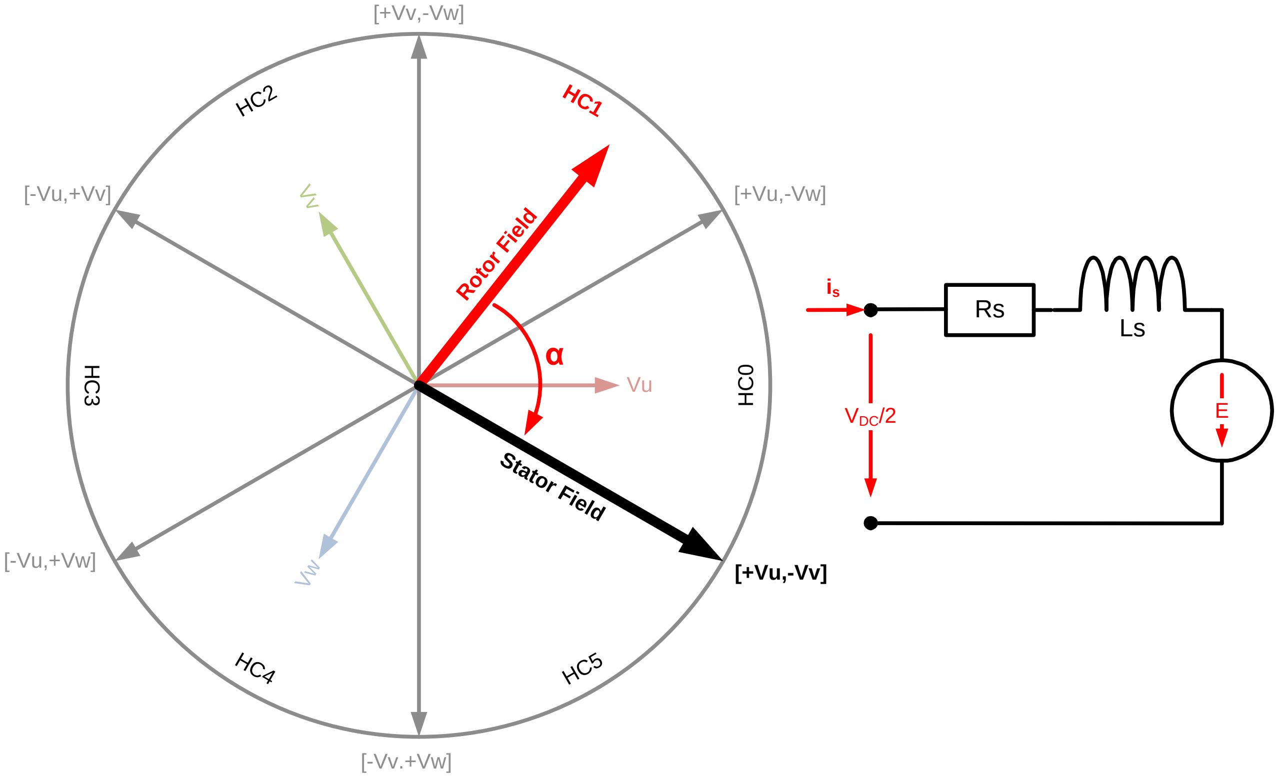


Figure 6.2: BLDC Motor Vector Diagram and phase circuit

The Inverter can switch among six different voltage vectors. Stator magnetic field is proportional to stator current. The current is the consequence of the difference between DC and BEMF voltage (1).

$$\frac{V_{DC}}{2} - E_{BEMF} = i_s \cdot R_s + L_s \cdot \frac{di_s}{dt} \qquad (1)$$

$$E_{BEMF} = k_E \cdot n \qquad (2)$$

Rotor field is produced by permanent magnets. Stator field is attracting rotor field with torque defined by stator field magnitude and the angle between them (3), therefore the rotor will move toward stator field.

$$T = k_T \cdot I_s \cdot \sin(\alpha) \qquad (3)$$

Hall combination (HC1) is triggering the right voltage vector (+Vu,-Vu) in order to achieve maximum torque possible. When angle between rotor and stator field is reduced to 60° and hall combination change to (HC0), the successive stator voltage vector (-Vv,+Vw) is activated and the angle is again 120°. This is called commutation because the current is commutating from one phase to another.

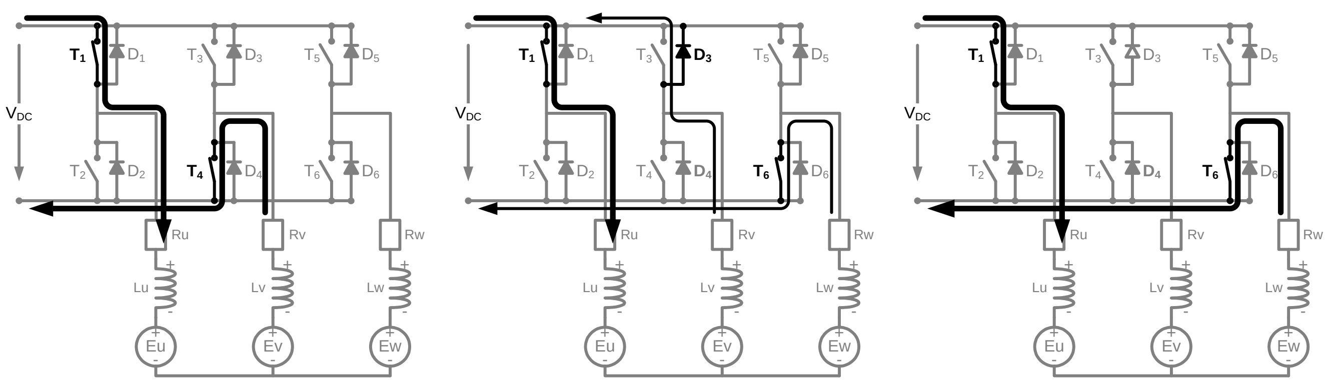


Figure 6.3: BLDC motor current commutation

Commutation table defines the right relationship between hall combinations (HCx) and voltage vectors (V1, V2) for both rotation directions.

Hall Combination	Voltage vector for different rotation direction	
	CCW	**CW**
HC0	+Vv , -Vw	-Vv , +Vw
HC1	+Vv , -Vu	-Vv , +Vu
HC2	+Vw , -Vu	-Vw , +Vu
HC3	+Vw , -Vv	-Vw , +Vv
HC4	+Vu , -Vv	-Vu , +Vv
HC5	+Vu , -Vw	-Vu , +Vw

Table 6.2: BLDC motor commutation table

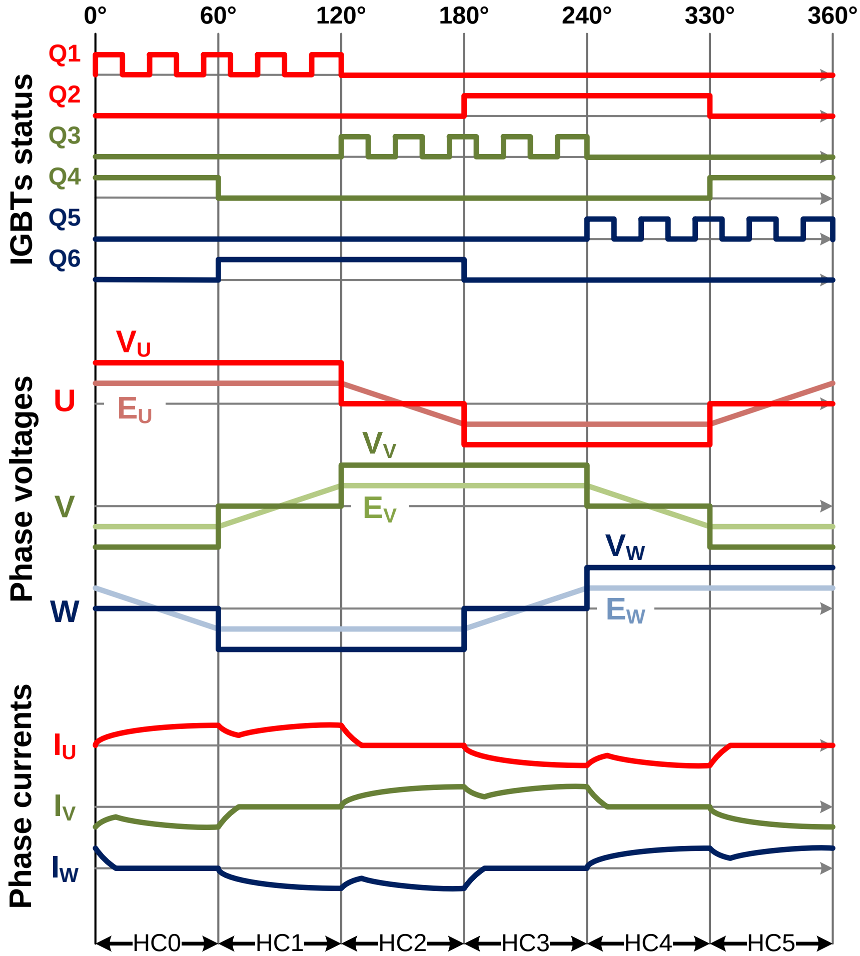


Figure 6.4: BLDC motor switching pattern

BLDC motor static characteristic is obtained combining equations (1), (2) and (3). Current transient (di/dt) and torque fluctuation sin ∝ are neglected.

$$n(V_{DC}, T) = \frac{V_{DC}}{2 \cdot k_E} - T \cdot \frac{R_s}{k_E \cdot k_T} \qquad (4)$$

Motor speed is a function of phase voltage and load torque. By increasing the voltage also speed will increase. Voltage control is done using PWM technique by changing duty cycle. PWM is applied only on one high side switch per time for 120°. Low side switch is switched ON for 120° too but without using PWM. This is calling unipolar switching.

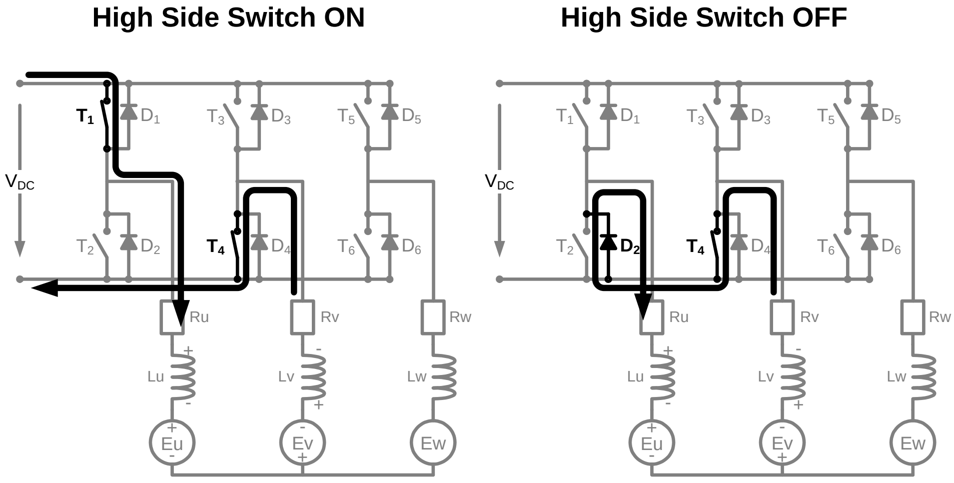


Figure 6.5: BLDC motor unipolar switching

As mentioned before the main control parameter is the duty cycle. This can be calculated selecting from three available motor control techniques (Figure 6.7).

1. **Voltage control** – phase voltage can be changed from 0% to 100% (VDC/2).
2. **Speed regulation** – phase voltage is regulated using PI controller. Input parameter is reference speed.
3. **Current (Torque) regulation** – phase voltage is regulated using PI controller. Input parameter is reference current.

The single shunt current measurement must be done when high side switch is OFF (Figure 6.5). This is the moment when the current is flowing through the shunt back into the power source. Therefore A/D converter must be synchronized with PWM and its duty cycle as visible on (Figure 6.6). T13 duty cycle is equal PWM period value (T12PER) and T13 period value is three times PWM period. Those values will assure that the current is sensed on right moment.

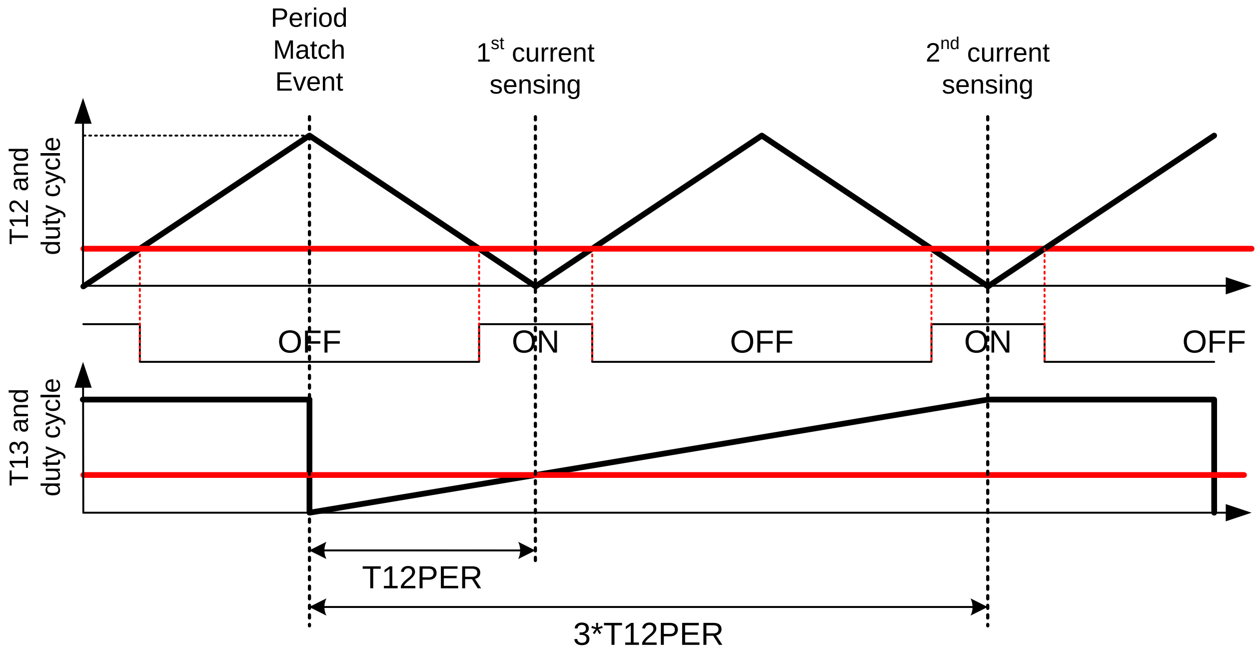


Figure 6.6: BLDC motor current measurement using a single shunt

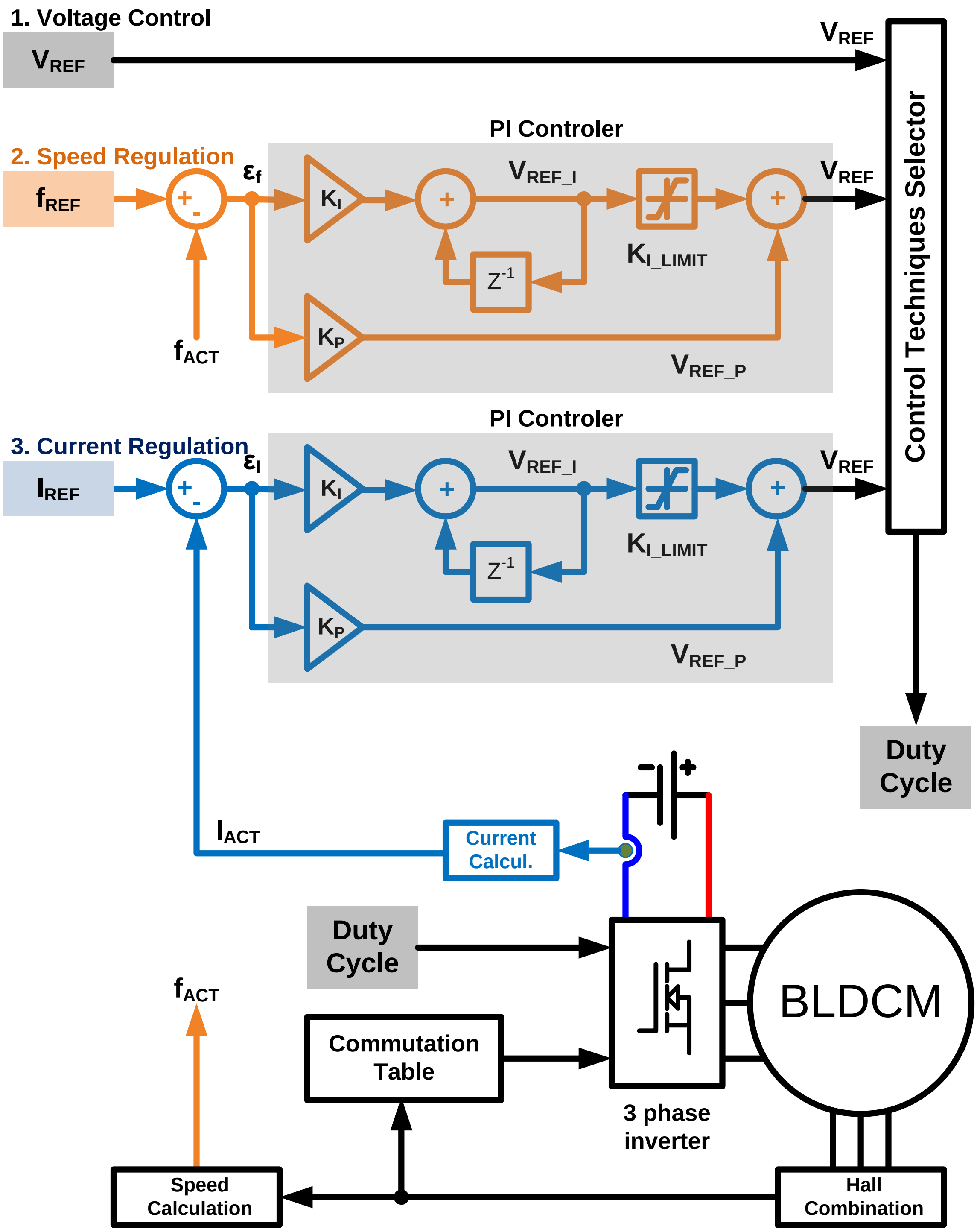


Figure 6.7: BLDC motor block commutation control techniques

6.3 Scheduler

There are some procedures that must be run before the main motor control algorithm is started. Those procedures ensure safe motor start up conditions. Scheduler is controlling procedures to be executed one after another sequentially.

The Scheduler is split in two parts: high and low priority. High priority scheduler is linked to a timer executing every 1ms. Low priority scheduler is linked to main loop executing when no other task is running. High priority scheduler is taking care of parameters and MCU register initialization, bootstrap capacitor charging and speed ramp generation. Low priority scheduler is controlling start/stop of the motor.

6.4 Serial Communication Protocol

UART settings are:

- 192000 Baud rate
- 8-bit data
- 1 start bit
- 1 stop bi

Communication protocol is a set of communication rules applied to serial communication messages. Each message is made out of 9 bytes. The last byte is a check sum of first 8 bytes. There are two different message types:

- Message with fixed parameters:

  Certain parameters are mapped to byte 1 - 7. This mapping is fixed and depends on byte 0 value. This message is sent both directions. MCU sent it periodically every 10ms. It can be also sent by user to MCU in order to control the motor real time.

Byte 0		**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**	**Byte 8**
<b7-b4>	<b3-b0>	<b7-b0>	<b7-b0>	<b15-b8>	<b7-b0>	<b15-b8>	<b7-b0>	<b15-b8>	<b7-b0>
ACM Software									
0xE	2		V_{DC}						Check Sum
	4	Mode	f_{REF}		f_{ACT}		SVM Angle		
	6	State	I_{REF}		I_{ACT}		V_{ACT}		
	8	Control Bits					PWM period		
	10	Status Bits							
BLDCM Software									
0xE	2		Temperature		Potentiometer		V_{DC}		Check Sum
	4	Mode	f_{REF}		f_{ACT}		Hall Combination		
	6	State	I_{REF}		I_{ACT}		V_{ACT}		
	8	Control Bits	Hall Index						
	10	Status Bits							

Table 6.3: Serial message with fixed parameters, MCU to PC

Byte 0		Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
<b7-b4>	<b3-b0>	<b7-b0>	<b7-b0>	<b15-b8>	<b7-b0>	<b15-b8>			<b7-b0>
0xE	0	Mode	Control Bits Set		Control Bits Clear				Check Sum
	2		f_{REF}						
	4		I_{REF}						
	6		V_{REF}						

Table 6.4: Serial message with fixed parameters, PC to MCU

- Message with indexed parameters:
  Parameters that can be saved into FLASH are indexed and are reachable by setting the correct index in bytes 2-3. Value can be read/write by setting or reading
  byte 4-5.

Byte 0				Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
<b7-b4>	<b3>	<b2>	<b1-b0>		<b7-b0>	<b15-b8>	<b7-b0>	<b15-b8>			<b7-b0>
0xF	1	0 = read 1 = write	0 = byte 1 = word		Index		(if byte) Value				Check Sum
							(if word) Value				

Table 6.5: Serial message with indexed parameters, PC to MCU

6.5 Parameters

Parameters are values that influence motor and its control behavior. There are two types of parameters: permanent and variable parameters. Permanent parameters are saved in a permanent MCU memory (FLASH). Each of them is identified and reachable by its index using communication message (**Error! Reference source not found.**). Variable parameters reflect the current state of the motor control or it can be controlled by them using communication message (Table 6.3). MCU receives recalculated parameter by Infineon PC user interface according to coefficient K in Table 6.7 and equation (5).

$$MCU\_Value = K \cdot PC\_Value \qquad (5)$$

Length	Name [Unit]	Min	Max	K	Description
Byte	Mode	0	4	1	Operational mode
Byte	Status	0	255	1	Status of the operational mode
Byte	Command Bits	0	255	1	Commands sent to MCU in order to activate some functionality
Byte	Status Bits	0	255	1	Statuses of the program that is running
Word	f_{REF} [Hz]	-300	300	10	Reference current frequency
Word	f_{ACT} [Hz]	-300	300	10	Actual current frequency
Word	I_{ACT} [A]	0	2		Actual DC link current
Word	I_{REF} [A]	0	2		Reference DC link current
Word	V_{REF} [%]	0	100	327.67	Reference DC phase voltage or duty cycle
Word	Hall Sensors	0	5	1	Hall sensors combination
Word	V_{DC} [V]	0	420		DC link voltage

Table 6.6: Variable parameters

Index	Length	Group.Name [Unit]	Def.	Min	Max	K	Description
0	Byte	Pwm.SwitchDelay [x21*ns*]	**63**	-128	127	1	Fine tuning of shunt current sensing moment.
1	Byte	Pwm.Frequency [*kHz*]	**15**	5	25	1	Switching frequency
2	Byte	Pwm.DeadTime [x21*ns*]	**20**	10	255	1	Dead time between half bridge devices
3	Word	I.Kp	**0**	0	32767	1	Proportional part of PI controller for current control loop
4	Word	I.Ki	**10**	0	32767	1	Integral part of PI controller for current control loop
7	Byte	I.Ki_Limit [%]	**100**	0	100	1.27	Integral part limit of PI controller for current control loop
8	Word	Freq.Max [Hz]	**300**	0	300	10	Absolute aloud maximum frequency
10	Word	Freq.Slew [Hz/s]	**4**	0.1	100	10	Ramp for the frequency when increasing or decreasing
12	Word	FreqI.Kp	**0**	0	32767	1	Proportional part of PI controller for speed control loop which output is reference current
14	Word	FreqI.Ki	**5**	0	32767	1	Integral part of PI controller for speed control loop which output is reference current
16	Byte	FreqI.Ki_Limit [%]	**100**	0	100	1.27	Integral part limit of PI controller for speed control loop which output is reference current
17	Word	FreqV.Kp	**0**	0	32767	1	Proportional part of PI controller for speed control loop which output is duty cycle
19	Word	FreqV.Ki	**5**	0	32767	1	Integral part of PI controller for speed control loop which output is duty cycle
21	Byte	FreqV.Ki_Limit [%]	**100**	0	100	1.27	Integral part limit of PI controller for speed control loop which output is duty cycle
22	Byte	Adc.Offset [x 21ns]	**62**	-128	127	1	Tuning the zero of current sensing
23	Word	Adc.TmotorMax	**32767**	0	65535	1	Maximum motor temperature before the motor switch off
25	Byte	Sys.Commands	**0**	0	255	1	Startup values of the commands
26	Byte	Sys.Modes	**2**	0	4	1	Startup mode

Table 6.7: Permanent parameters saved or can be saved to Flash