YROTATE-IT- S5D9

Rotate it! – Motor Control S5D9

UM-YROTATE-IT-S5D9
Rev. 1.0
March 23, 2018

Introduction

The Renesas Motor Control Kit called YROTATE-IT-S5D9, is based on the 32-bit ARM Cortex M4 Renesas Synergy S5D9.

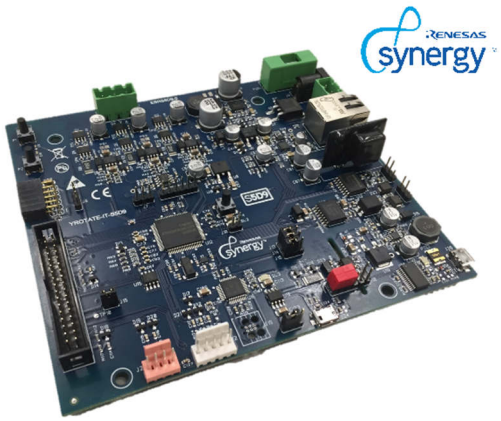
The kit enables engineers to easily test and evaluate the performance of the S5D9 in a laboratory environment when driving any 3-phase Permanent Magnet Synchronous Motor (e.g. AC Brushless Motor) using an advanced sensorless Field Oriented Control algorithm. Typical applications for this type of solution are compressors, air conditioning, fans, air extractors, pumps, home appliances inverters and industrial drives.

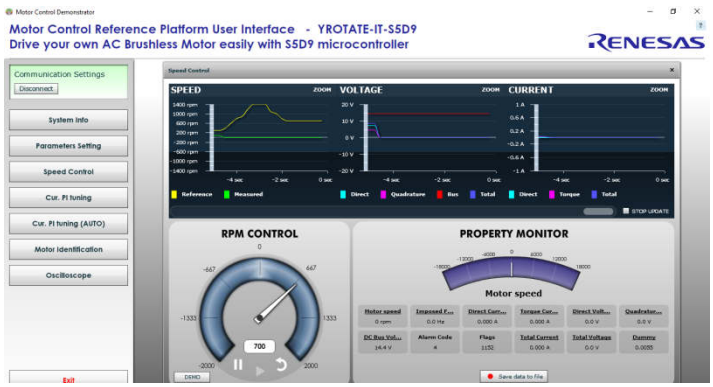
The phase current measurement is done via three shunts which offers a cost optimized solution, avoiding the need for an expensive current sensor or hall sensor. A single shunt current reading method is also available to ensure an even more compacter bill of material.

The powerful user-friendly PC Graphical User Interface (GUI) gives real time access to key motor performance parameters and provides a unique motor auto-tuning facility. Furthermore, it becomes also possible to select the best switching frequency and control frequency (e.g. control loop) to adapt the control dynamics suitable to the application requirements.

The hardware is designed for easy access to key system test points and for the ability to use the internal debugger based on R7FS1247 MCU. Although the board is normally powered directly from the USB port of a Host PC, connectors are provided to utilise external power supplies where required.

The YROTATE-IT-S5D9 is an ideal tool to check out all the key performance parameters of your selected motor, before





Target Device: **S5D9 Microcontrollers Family**

Contents

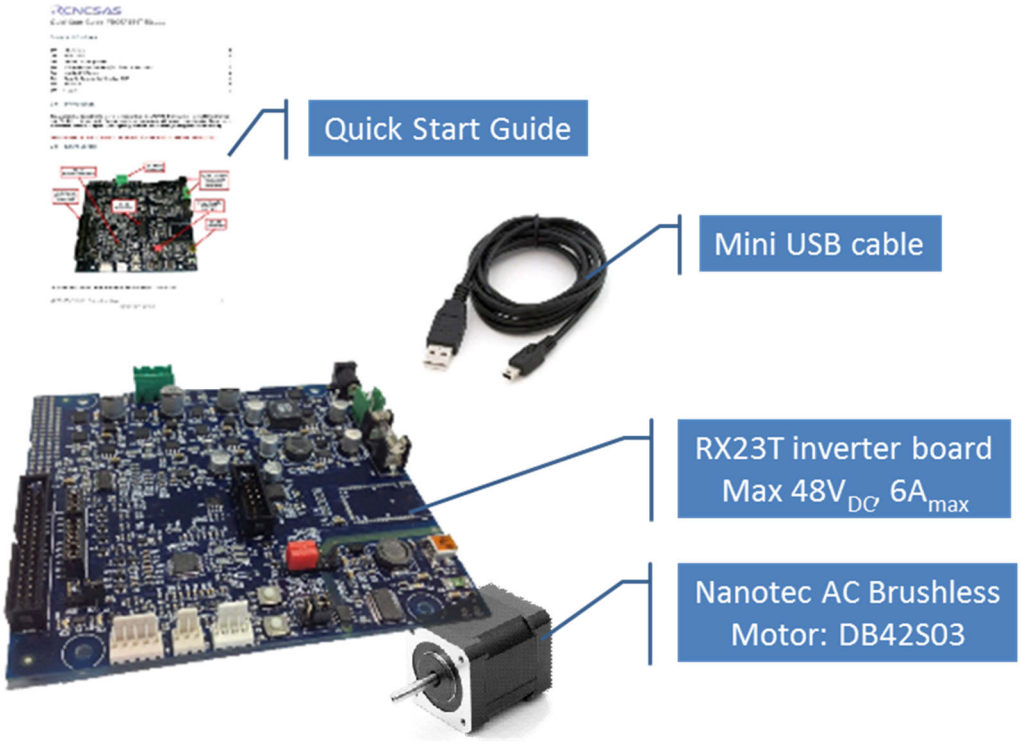
1. Specifications & Hardware overview 3
2. Connectors description 7
3. Power supply selection 8
4. LEDs functions description 10
5. Test points for debugging 11
6. Jumpers description 12
7. Internal power stage description 13
8. Selection of Current reading methods 16
9. RS485 and CAN COMMUNICATION circuit 17
10. ETHERNET CIRCUIT 18
11. USB COMMUNICATION 19
12. Power supply circuits: step-up, step-down 20
13. Interface to an external power stage 21
14. Single shunt current reading 25
15. Current reading timing in three shunts and single shunt configurations 26
16. Microcontroller S5D9 short overview 27
17. Permanent Magnets Brushless Motor model 29
18. Sensorless Field Oriented Control algorithm 34
19. Software description 35
20. Reference system transformations in details 44
21. Rotor position estimation 45
22. EEPROM parameters: detailed description 48
23. Communication protocol between the MCU and the PC GUI 49
24. Inverter Kit – Bill of Material 53
25. Inverter Kit – PCB topography 58
26. Inverter Kit – Board picture in high resolution 59

1. Specifications & Hardware overview

ITEM	SPECIFICATIONS
TYPE OF MOTORS SUPPORTED	3-phase Permanent Magnet Synchronous (PMSM, PMAC, BLAC) 3-phase Brushless DC (BLDC)
KIT MOTOR PARTNAME	NANOTEC, DB42S03, 24V_{DC}, Nominal Speed: 4000 RPM
KIT MAX INPUT RANGE	External power supply from: 20V_{DC} to 48$V_{DC,}$ 6A_{peak}
TRANSISTOR USED	Renesas Mosfets: RJK0654DPB, 60V, 30A
POWER SUPPLY OPTION	Either USB connection or external supply: up to 48V_{DC}
CURRENT DETECTION	One or three shunts configuration (100mΩ)
USB IC USED ON THE BOARD	FT232R - USB UART IC from FDTI, 76.6KBd communication speed
MICROCONTROLLER	Synergy S5D9
MCU PERFORMANCE	ARM Cortex M4, 120MHz
KEY FEATURES	Floating Point Unit, 3-phase inverter Timer 12-bit A/D Converter, fast on-chip Comparators, Port Output Enable
MCU EMBEDDED FIRMWARE	Sensorless vector control algorithm (Field Oriented Control)
SWITCHING FREQUENCY	4KHz to 64KHz, 16KHz by default (PWM frequency)
CONTROL LOOP FREQUENCY (SAMPLING FREQUENCY)	4KHz to 16KHz, 16KHz by default
CONTROL LOOP TIMING	50µs including debug features and auto-tuning/self-identification 40µs including only the sensorless vector control algorithm
ENVIRONMENT STANDARDS	RoHS compliant including China regulations WEEE, RoHS, CE

The inverter kit YROTATE-IT-S5D9 is a single inverter board, based on the 32-bit Synergy series Microcontroller S5D9 and includes a low-voltage MOSFETs power stage and a communication stage. The PCB is a four layers board and ensure the management of Permanent Magnet Motors up to $48V_{DC}$ and up to $6A_{max}$.

Please find below the content of the YROTATE-IT-S5D9 kit:

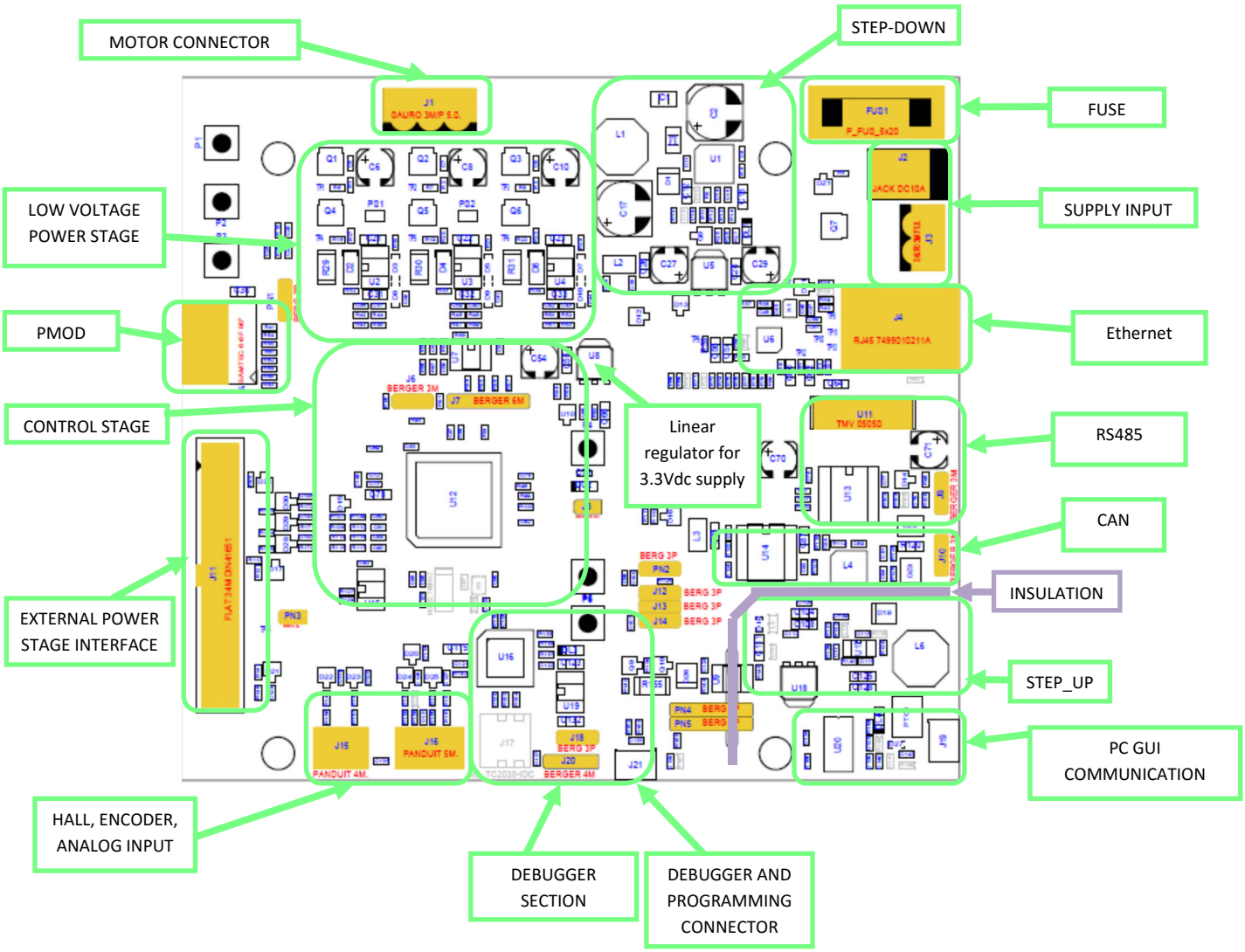


To obtain the maximum flexibility, the inverter reference kit includes:

- A complete 3-phase inverter on-board with a low voltage motor, so it becomes easy to test the powerful sensorless algorithm running on the Renesas 32-bit S5D9 microcontroller (e.g. MCU)
- An insulated USB communication with the PC offering communication speed up to 76.6KBd
- An in-circuit debugger MCU is on-board, to enable easy debugging and programming of the S5D9. Basically, no external JTAG external debugger tool is needed.
- An insulated RS485 Hardware circuit is available, the default software doesn't include the code
- An insulated CAN hardware circuit is available but not managed by the default software
- An ETHERNET hardware circuit and connector is available but not managed in the default software
- A PMOD connector is available to connect Wireless modules (not provided)
- Connectors for hall sensors and encoder connections. Both encoder and hall sensors are not managed in the sensorless software but they can be supported under request.
- The kit offers the full compatibility with the existing external power stages: high voltage one: 1.5KW at $230V_{AC}$, the high current one: up to $60V_{DC}$, $60A_{DC}$.

To achieve these aims, three different DC-DC converters are used:

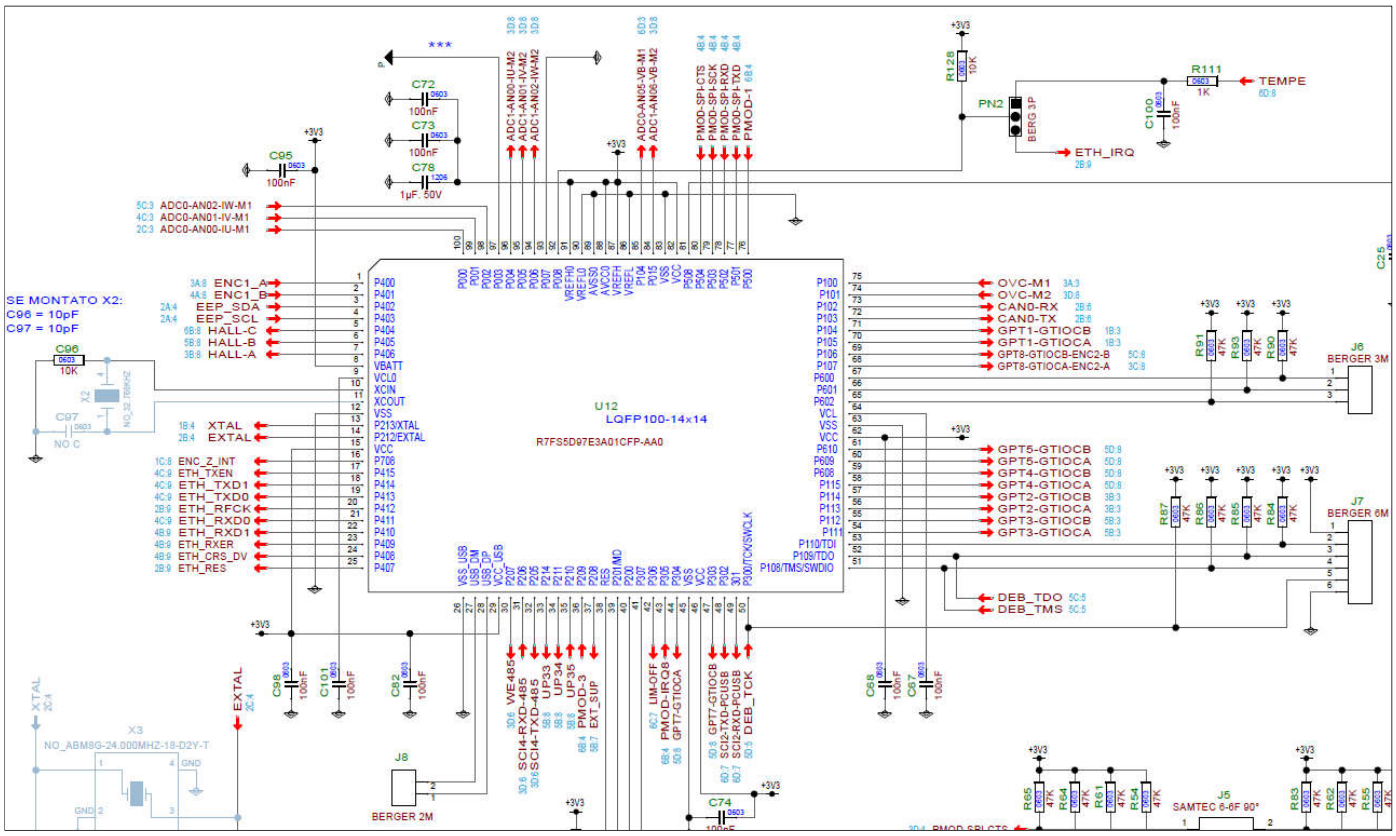
1. A step-up DC-DC converter to increase the voltage from the USB standard (5V) up to 13.5V_{DC}
2. A step-down converter and a low dropout linear converter, from the DC bus first to 15V and then to the CPU supply voltage: 3.3V, please find below the PCB overview.



The full schematics and CAD design files (e.g. Gerber) of the inverter kit are available on the website: www.renesas.eu/motorcontrol , in the section related to the YROTATE-IT-S5D9 development kit.

In the YROTATE-IT-S5D9 kit, the S5D9 in a 100-pin package was selected to ensure the management of inverter, external communications, three shunts, the EEPROM communication, three voltages phases, the over-current detection, the Bus voltage monitoring, etc.

The picture below is showing the detailed I/O pins assignment of the 100-pin S5D9 to manage the complete kit.

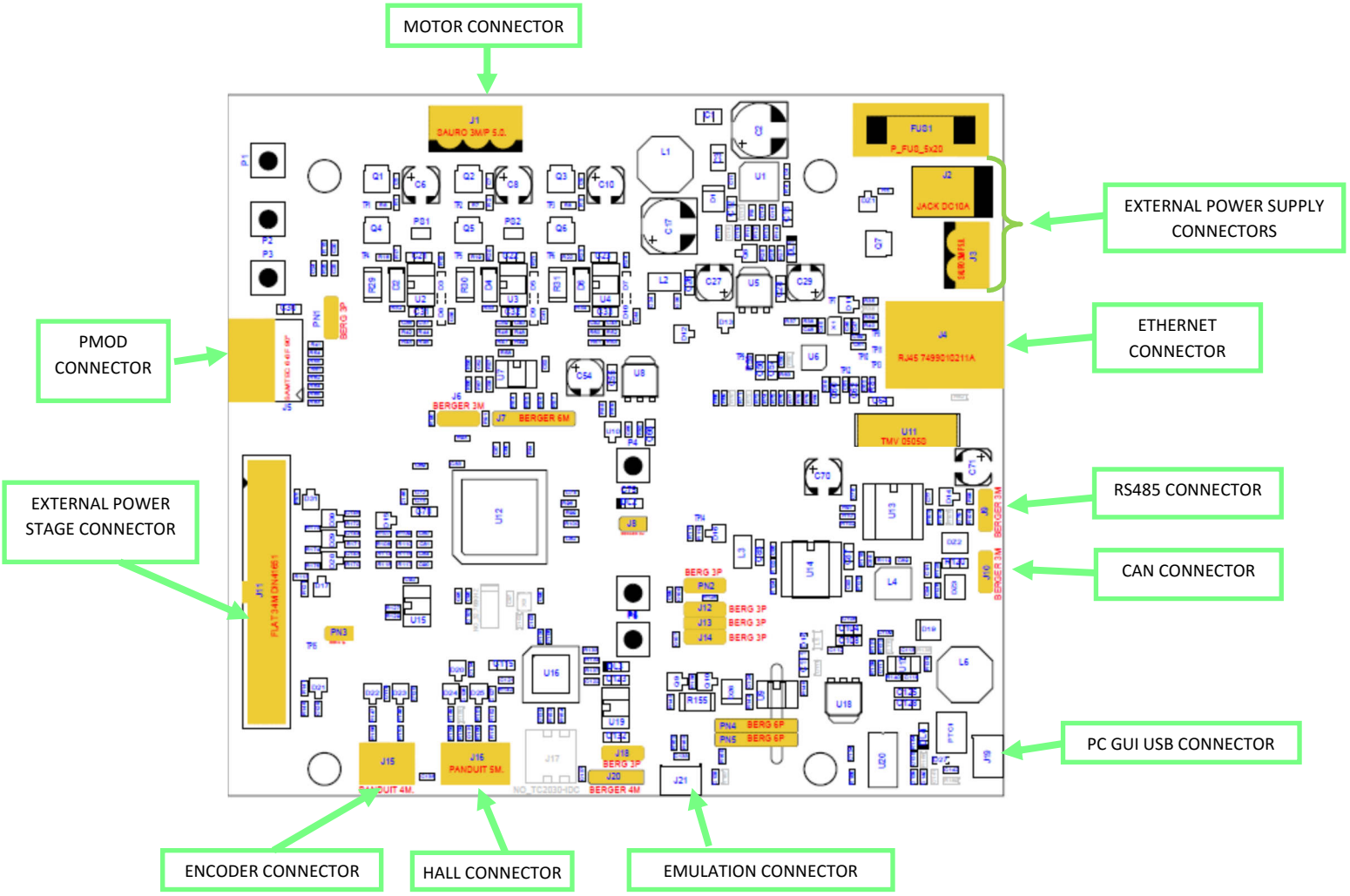


2. Connectors description

As in the following figure, you can find the position and the description of the connectors present on the board. Please refer to the board schematics for the full description of the connectors.

The J21 connector is used for the programming and the debugging of the software running on the S5D9. It can be connected to the E2 studio development environments.

The external power stage connector is compatible with the power stages, designed for Renesas inverter kits, which are able, the first one to drive 230V_{AC} motor up to 1.5KW, and the second one up to 60V_{DC}, 60A_{DC}. The schematics and Gerber file of the power stage are available on the website: www.renesas.eu/motorcontrol



3. Power supply selection

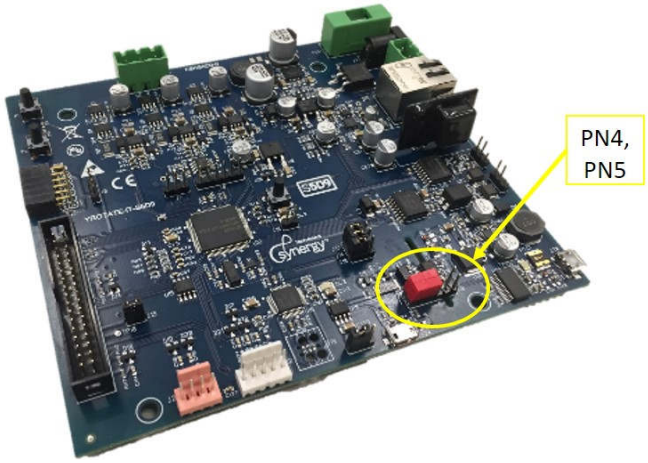
As stated before, there are two ways to supply the power to the board.

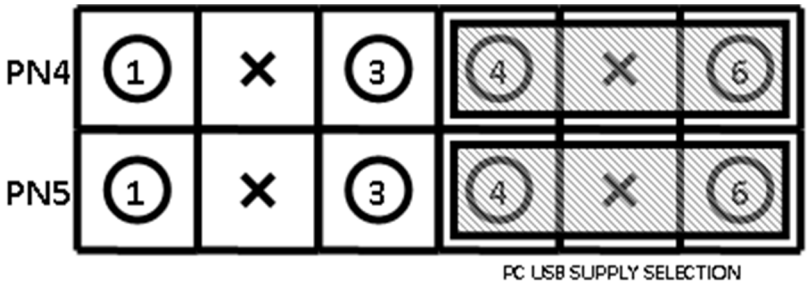
1. The first possibility is to use directly the PC USB supply delivered with the kit. In this case the maximum current that can be delivered to the motor is limited by the USB current capabilities.

2. The second possibility is to use an external voltage DC power supply to supply the board.

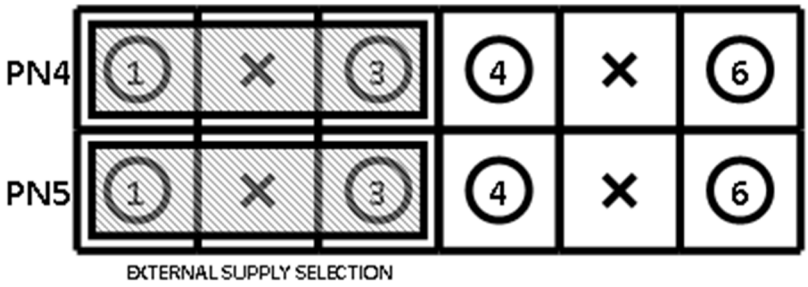
The recommended external power supply voltage is between **20V$_{DC}$** to **48V$_{DC}$**. In this case the communication stage is insulated from the inverter.

The selection between the two possibilities is made through two jumpers: **PN4** and **PN5**. Please find below the description:





PC USB SUPPLY SELECTION



EXTERNAL SUPPLY SELECTION

1) The first jumper configuration connects the USB ground to the inverter ground and the output of the step-up converter to the inverter DC link.

   Please notice that in this case there is no galvanic insulation between the device connected to the USB and the board.

2) The second jumper configuration connects the external power supply ground to the inverter ground and the external + V_{DC} to the inverter DC link. The USB circuit will be insulated.

Please find below the configuration of the S5D9 kit using the **internal** MOSFETs power stage.

YROTATE-IT-S5D9 powered via USB port



YROTATE-IT-S5D9 powered via external power

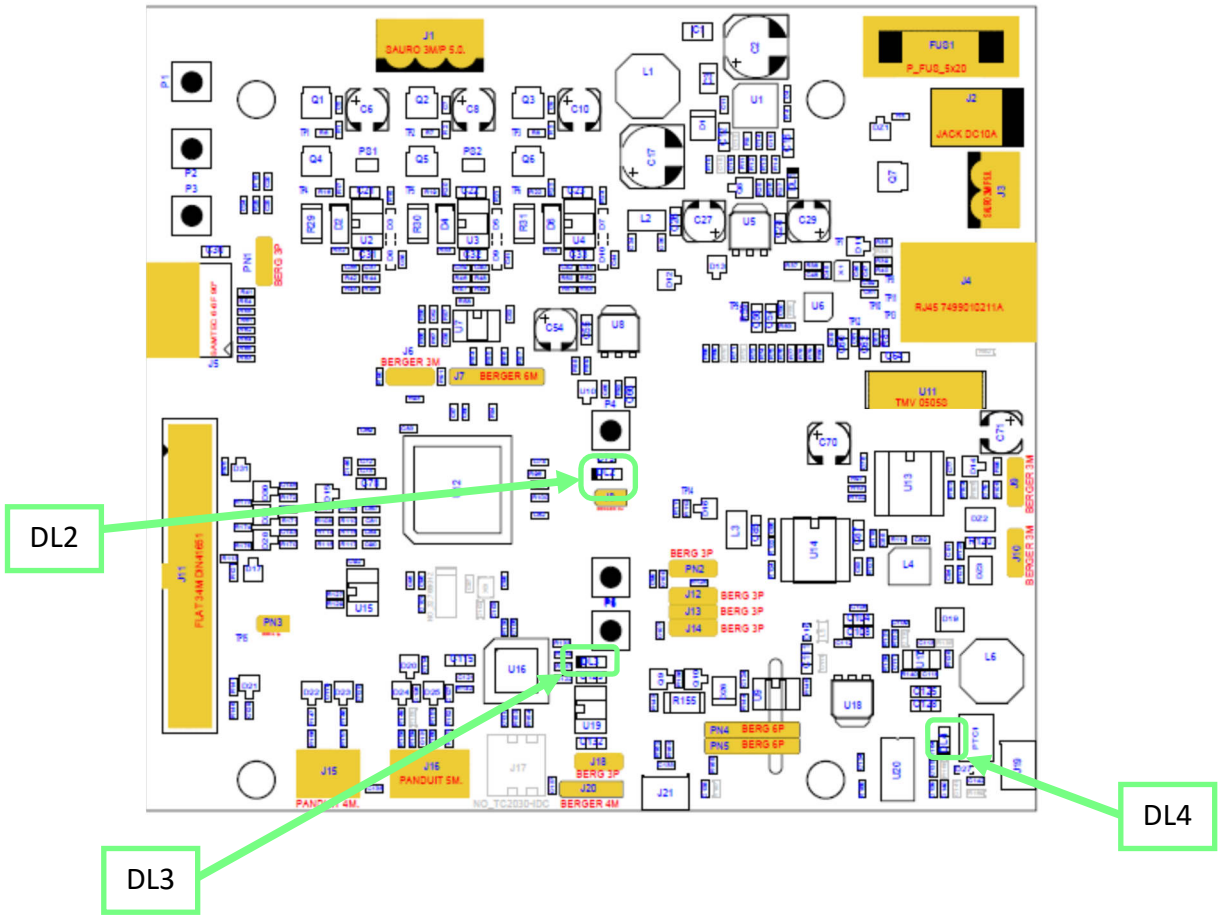


Important Note : Caution on Isolation Voltage

The second Jumper configuration is intended to isolate a small and electrically safe GND gap between the user system and user-accessible circuitry. In any case isolation voltage must be maintained within SELV limits i.e. less than 40VAC, or 60VDC.
The insulated part cannot be treated as safety isolation system. The part could be expected to function correctly at higher voltage across the isolation barrier; but then the circuitry on both sides of the barrier must be regarded as operating at an unsafe voltage and further isolation/insulation systems must form a barrier between these circuits and any user-accessible circuitry according to safety standard requirements.

4. LEDs functions description

Three LEDs available on the board are directly connected to the hardware and allow the user to understand the board status. Please refer to the LED map for the following indications:

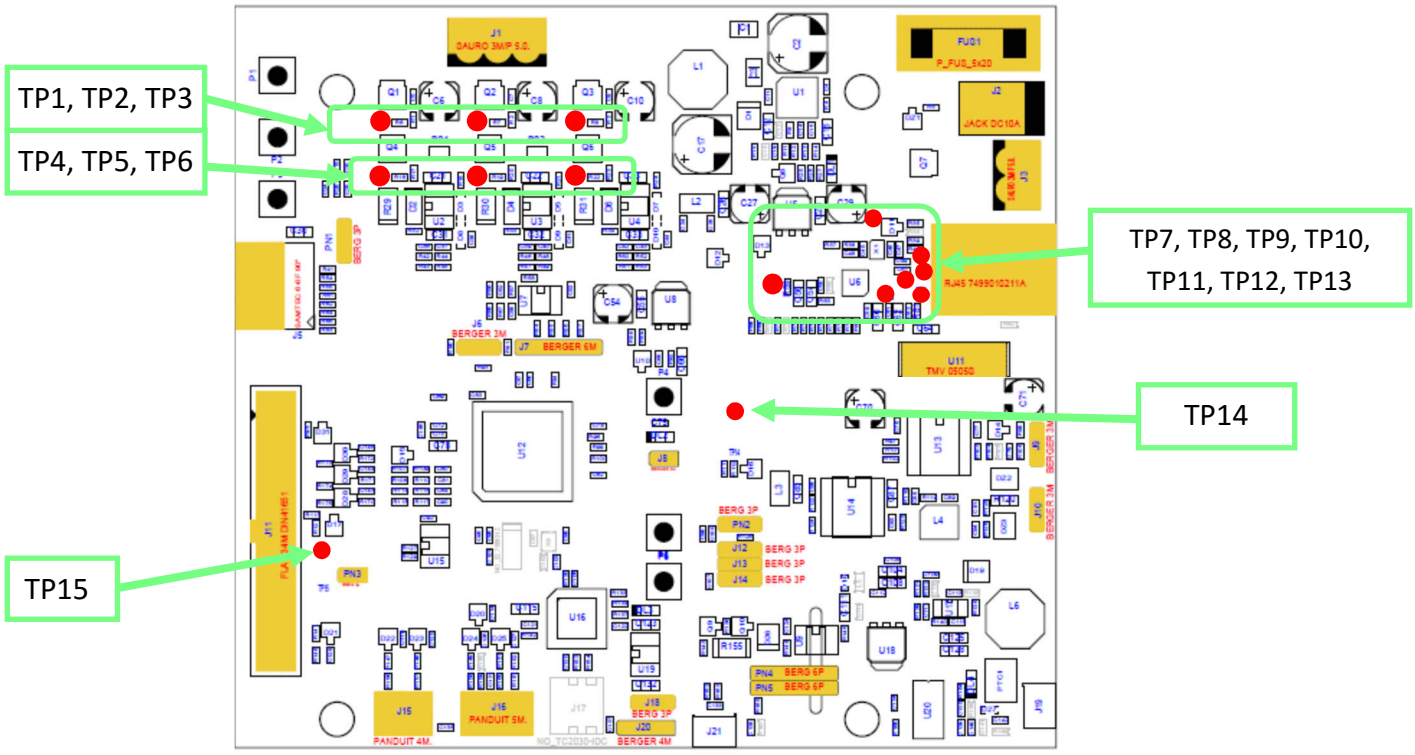


Other LEDs in the board are driven via software, in particular:

- DL1 is connected to the output of the 15 step-down DC-DC converters and indicates the presence of the internal 15V supply.
- DL2 is blinking slowly if the control section of the S5D9 microcontroller is running normally. In case of hardware or software alarms, the LED DL3 is blinking quickly.
- DL3 is connected to the debugging MCU (U16).
- DL4 is the USB communication indicator and blinks when there are data exchanges between the PC and the board.

5. Test points for debugging

Several specific test points are available on the board to visualize with the oscilloscope the behaviour of some internal analog signals.

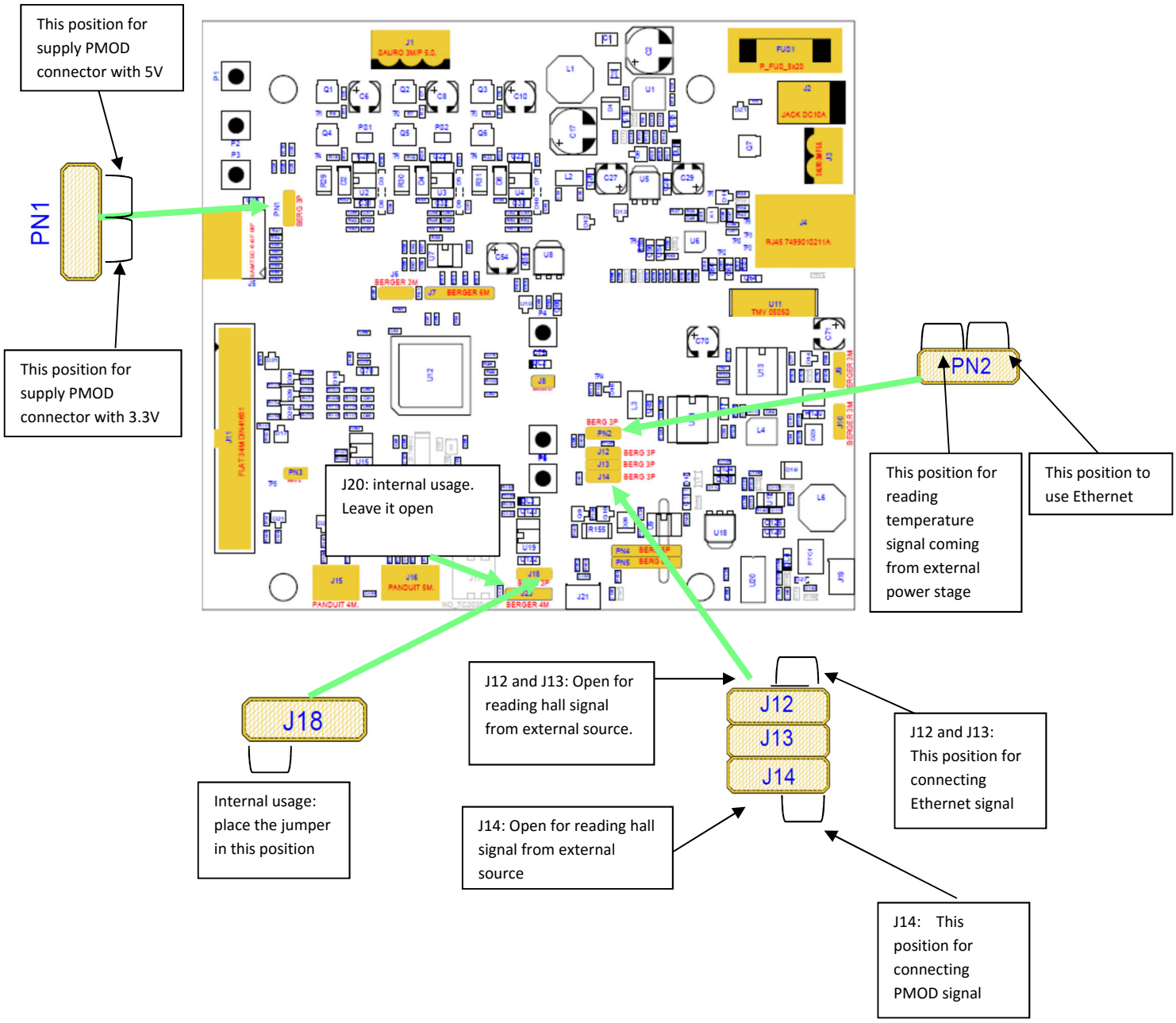


Please find below the description of the test points:

- **TP1**, **TP2**, and **TP3**: connected to the three inverter outputs (sources of the higher switches)
- **TP4**, **TP5**, and **TP6**: connected to the sources of the lower switches of the inverter
- **TP7, TP8, TP9, TP10, TP11, TP12, TP13**: connected to some interesting node of the ETHERNET circuit
- **TP14** connected to the temperature signal in case of connection with external power stage.
- **TP15** connected to the 5V of the external power stage (in case of connection with external power stage)

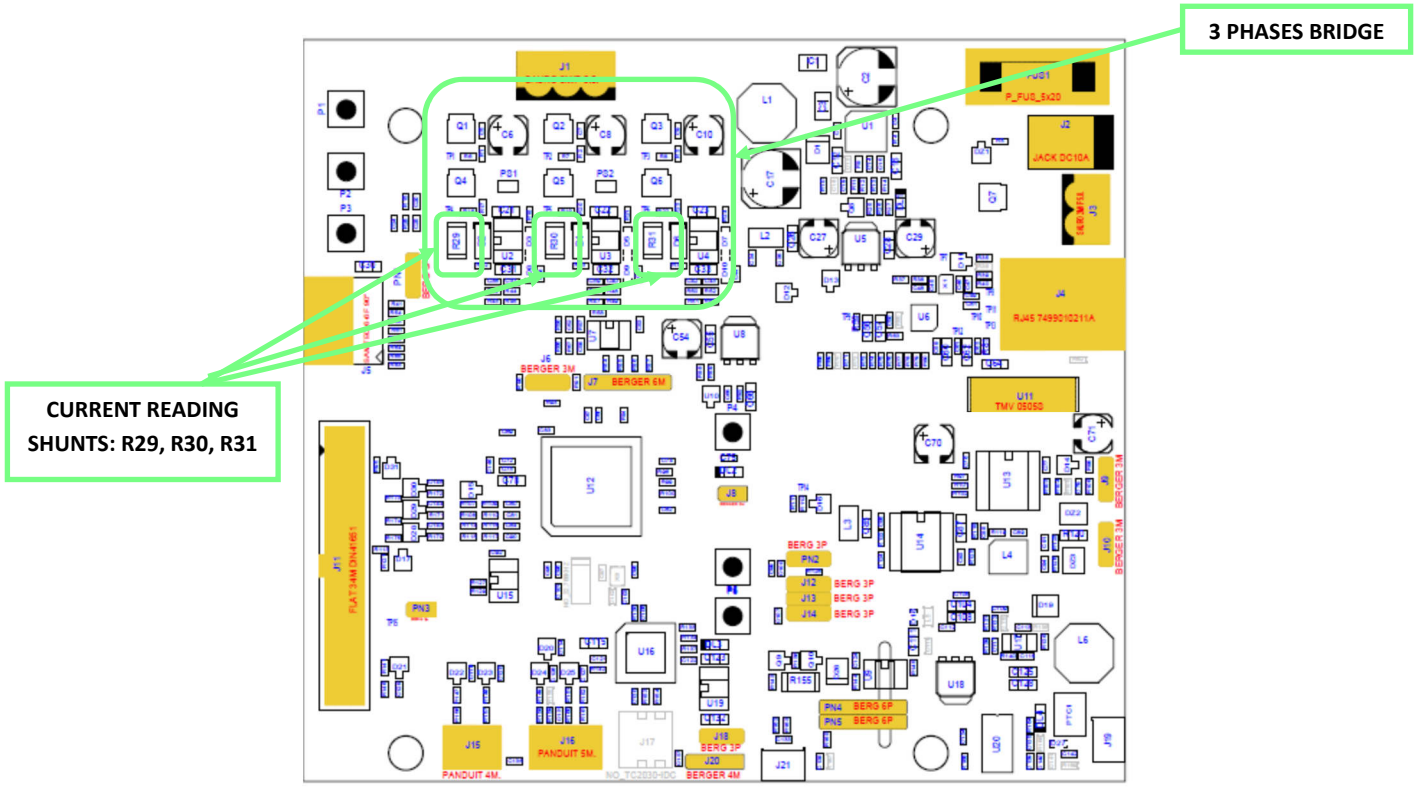
6. Jumpers description

Please find below the description of each jumper. For more details, please refer to the board schematics.

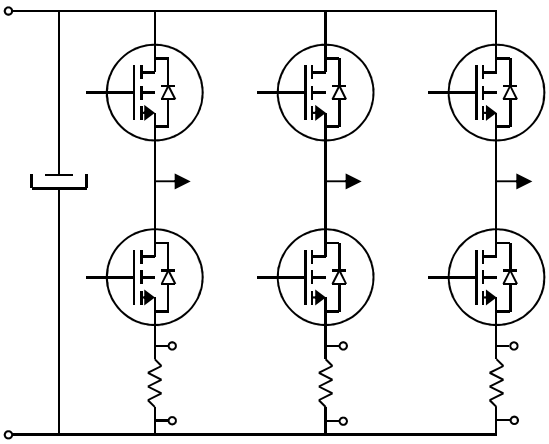


7. Internal power stage description

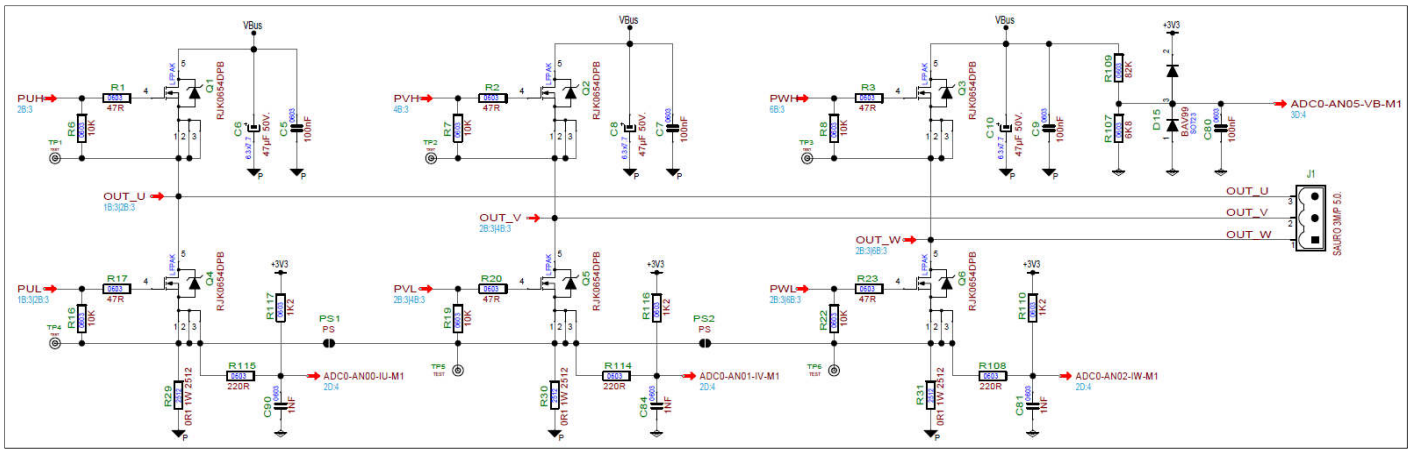
The power stage is a complete 3-phase bridge composed with discrete low voltage and high current MOSFETs. The MOSFETs are the Renesas **RJK0654DPB** n-channel power MOSFETs. Please refer to the data-sheet available on the Renesas website: www.renesas.eu for the switches characteristics and to the board schematics for the details on the driving circuit. The maximum current is **30A**, and the maximum voltage is **60V$_{DC}$**.



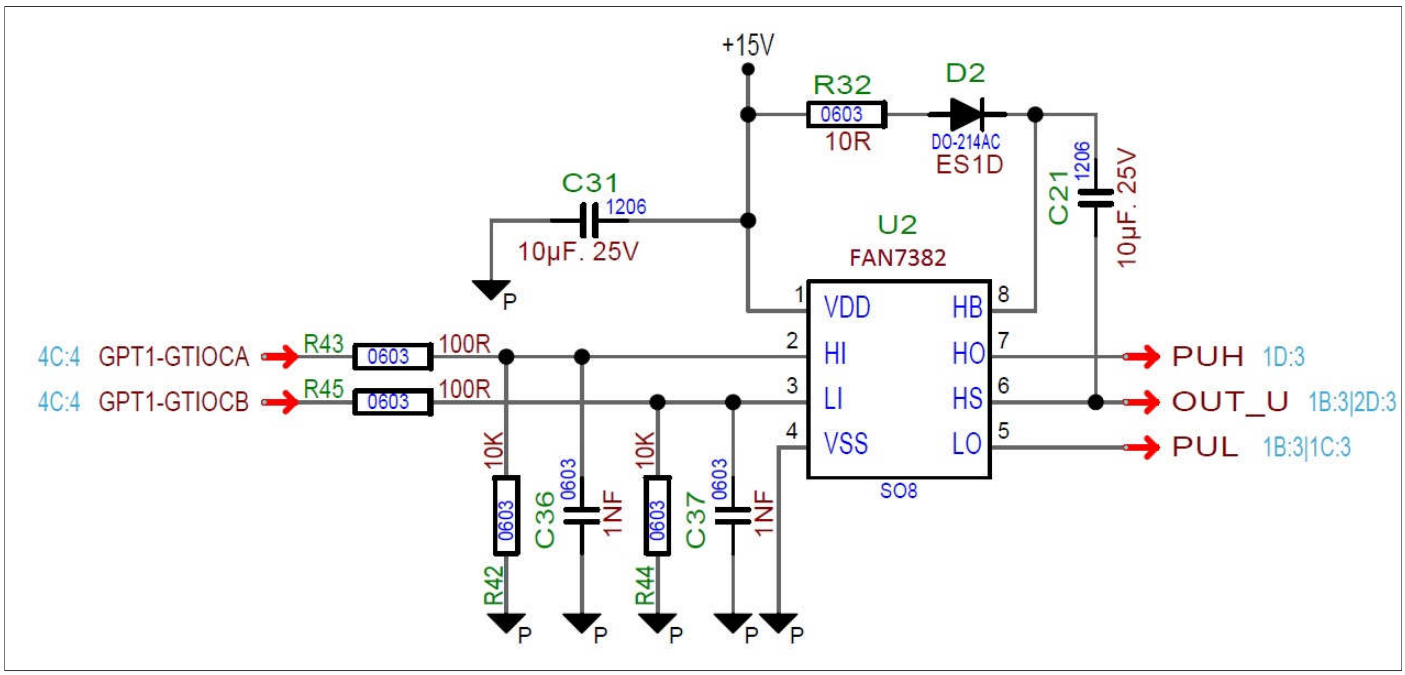
The inverter has the classical schema with the three shunts on the lower arms, with the possibility to use a single shunt by removing two of them.



Please find below the schematics extract showing the draying in mode details.

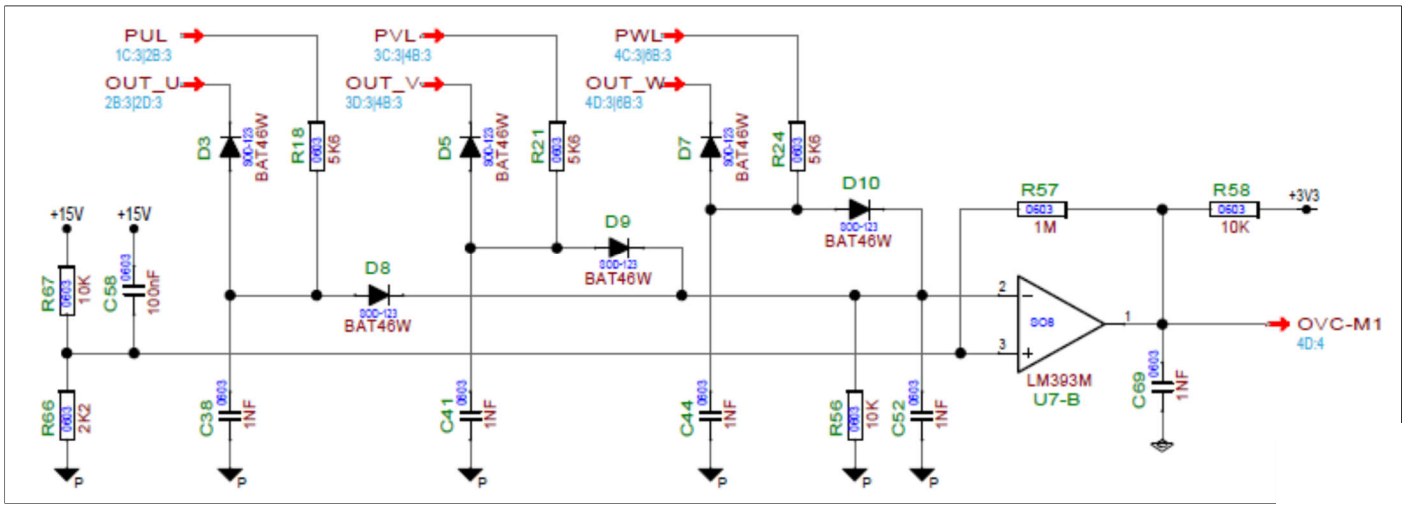


Furthermore, the three driving circuits for the six low-voltage MOSFETS **RJK0654DPB** are described below.



Furthermore, the signals from the gate driver circuits called “PUL, PVL, PWL” are used to manage the risk of over-current in the board. The signals are directly linked to the S5D9 Microcontroller module called the Port Output Enable (e.g. POE) to stop the PWM signal in hardware and protect the 3-phase power stage.

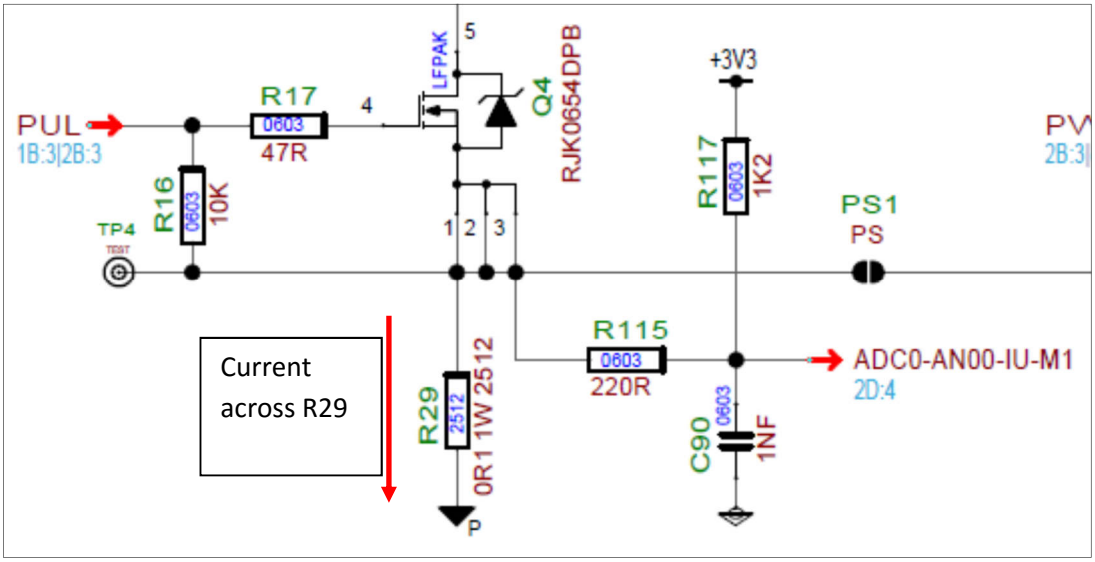
In case of over-current, the POE module will set in hardware the six PWM signals in high impedance mode to ensure safe-fail mechanism. Please find below the extract of the schematics regarding the over-current management.


PUL
1C:3|2B:3
PVL
3C:3|4B:3
PWL
4C:3|6B:3
OUT_U
2B:3|2D:3
OUT_V
3D:3|4B:3
OUT_W
4D:3|6B:3
D3
BAT46W
R18
5K6
D5
BAT46W
R21
5K6
D7
BAT46W
R24
5K6
D10
BAT46W
R57
1M
R58
10K
+3V3
+15V
+15V
R67
10K
C58
100nF
D8
BAT46W
D9
BAT46W
OVC-M1
4D:4
LM393M
U7-B
R66
2K2
C38
1NF
C41
1NF
C44
1NF
R56
10K
C52
1NF
C69
1NF

8. Selection of Current reading methods

The reading of the motor current by using internal operational amplifiers has been implemented as shown here below:

Please find below more details about the circuit used to read the current flowing through the shunts.

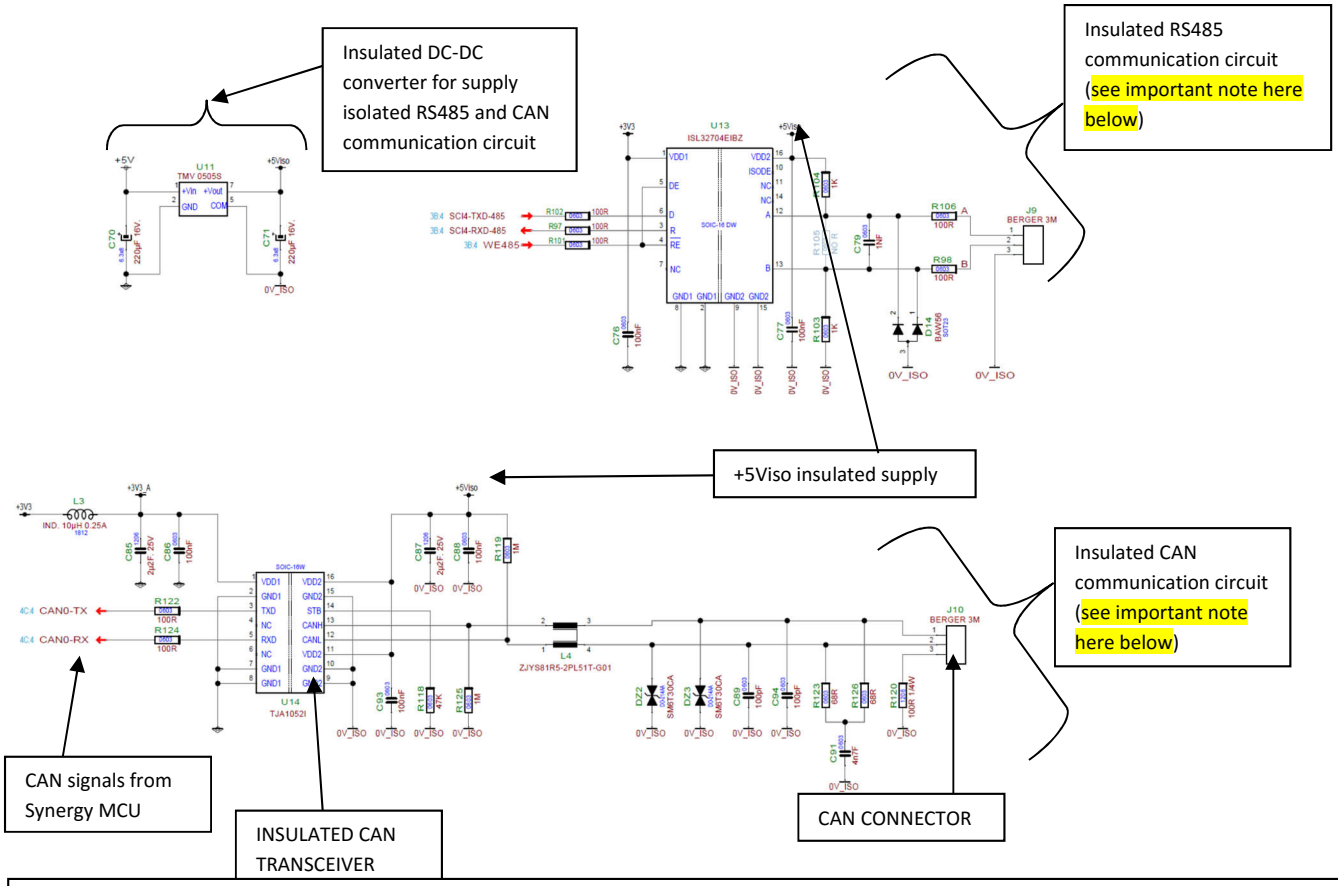


The current flowing across the shunt R29 produces a voltage which can be read using the circuit made with R115, C90, R117. The output voltage will be connected directly to the MCU A/D Converter Unit.

The internal Operational Amplifier Circuits of the S5D9 MCU are used to manage the signal.

By default, three shunts are used to measure the current flowing into the motor, but there is also the option to use a single shunt configuration.

9. RS485 and CAN COMMUNICATION circuit

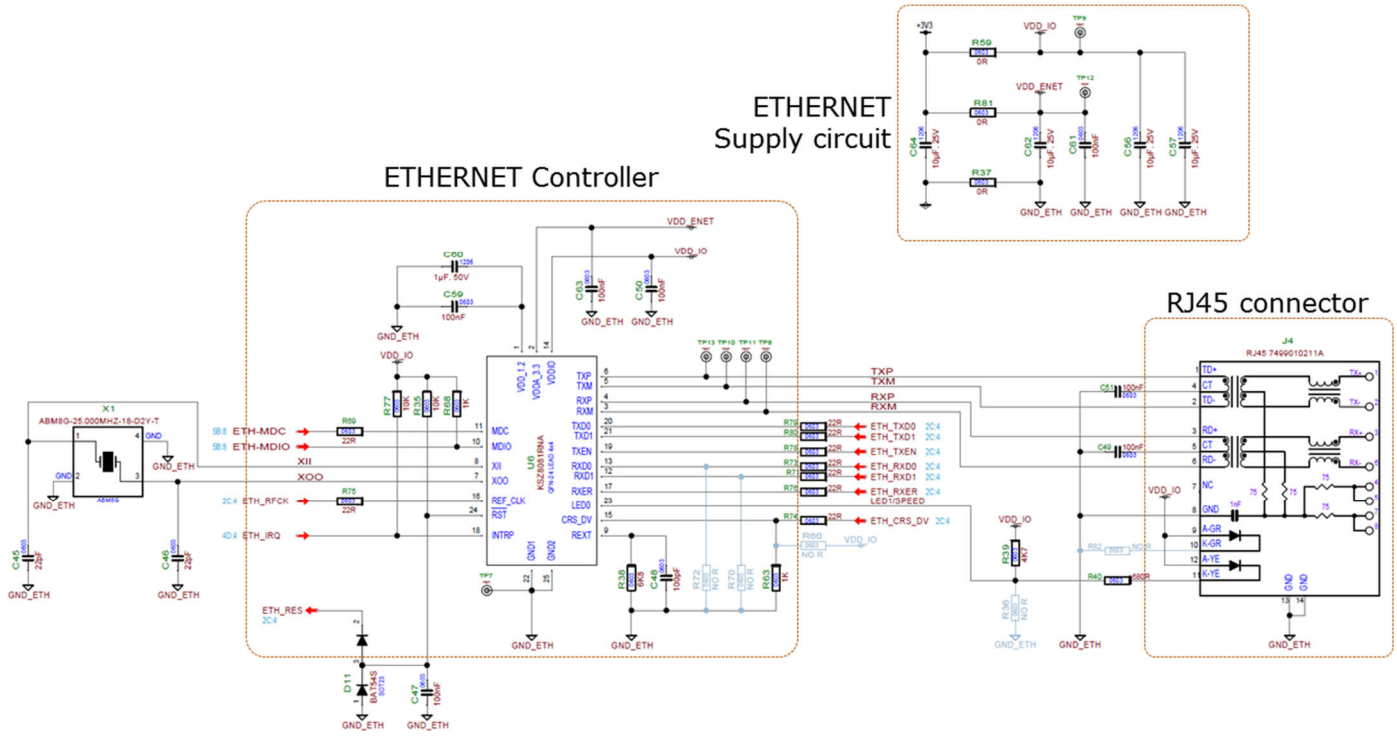


Important Note : Caution on Isolation Voltage

This product is intended to isolate a small and electrically safe GND gap between the user system and user-accessible circuitry. In any case isolation voltage must be maintained within SELV limits i.e. less than 40VAC, or 60VDC.
The insulated part cannot be treated as safety isolation system. The part could be expected to function correctly at higher voltage across the isolation barrier; but then the circuitry on both sides of the barrier must be regarded as operating at an unsafe voltage and further isolation/insulation systems must form a barrier between these circuits and any user-accessible circuitry according to safety standard requirements.

10. ETHERNET CIRCUIT

Please find below the details about the ETHERNET controller circuit and connector available on the YROTATE-IT-S5D9 inverter kit.

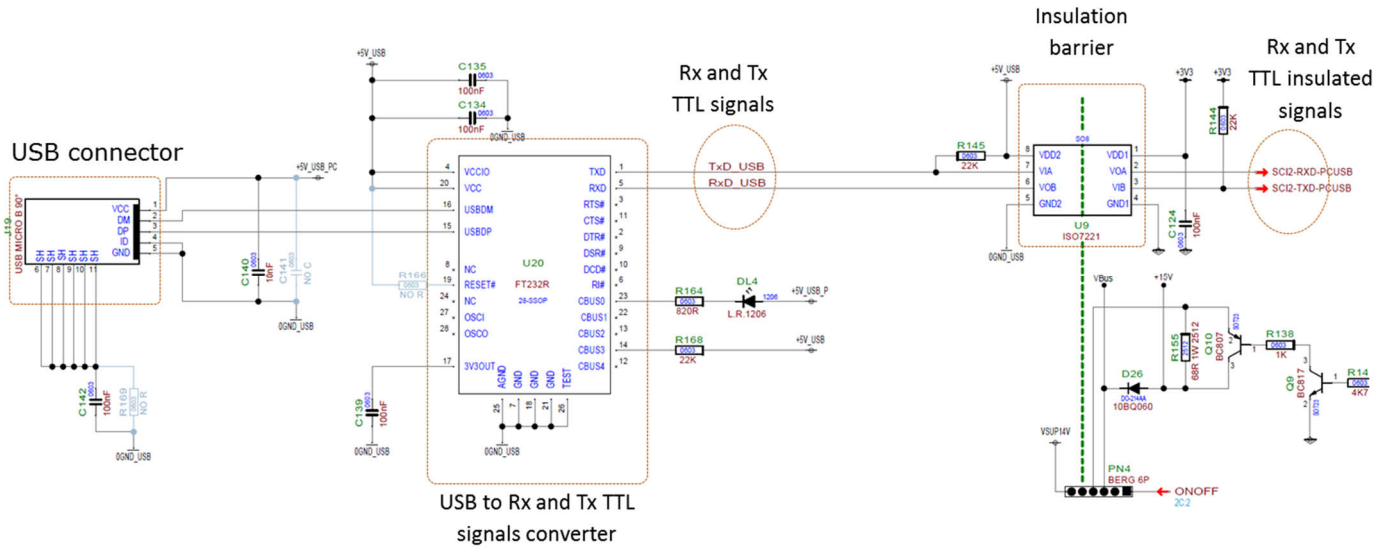


11. USB COMMUNICATION

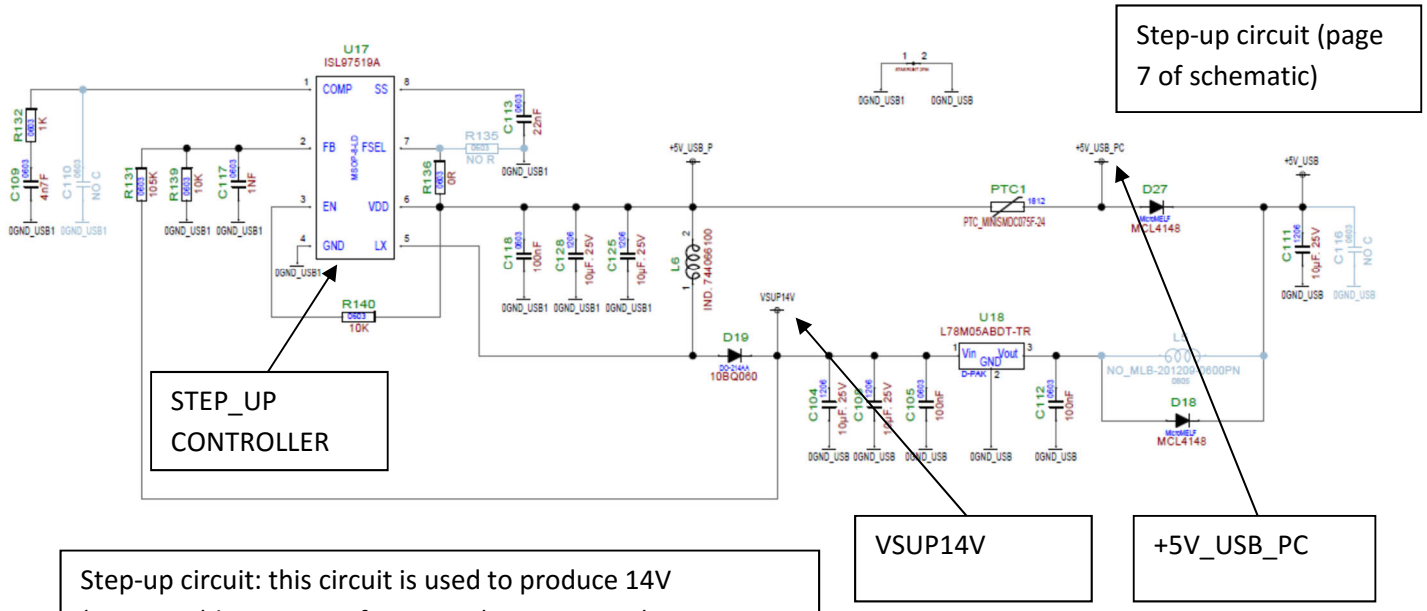
Important Note : Caution on Isolation Voltage

The isolation barrier is intended to isolate a small and electrically safe GND gap between the user system and user-accessible circuitry. In any case isolation voltage must be maintained within SELV limits i.e. less than 40VAC, or 60VDC.

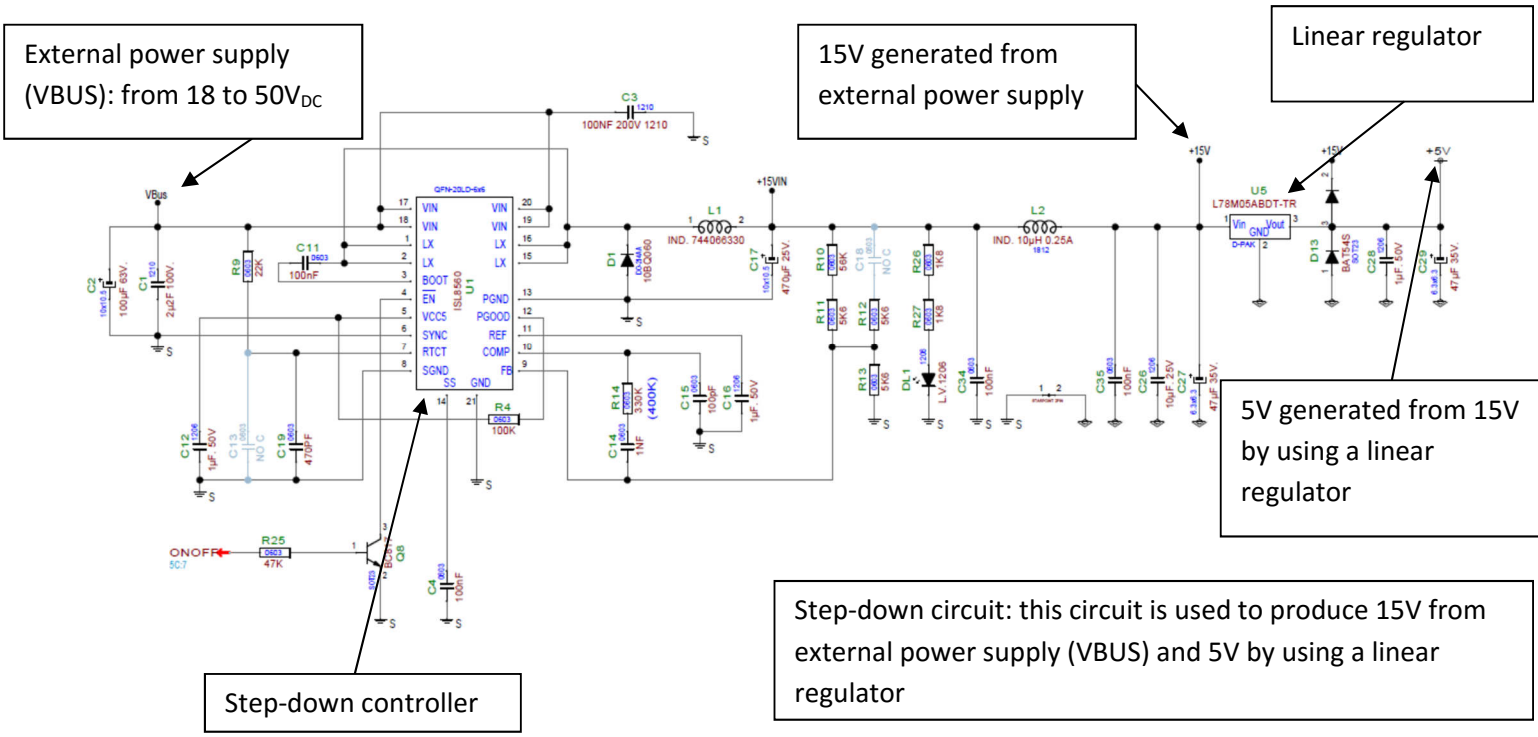
The insulated part cannot be treated as safety isolation system. The part could be expected to function correctly at higher voltage across the isolation barrier; but then the circuitry on both sides of the barrier must be regarded as operating at an unsafe voltage and further isolation/insulation systems must form a barrier between these circuits and any user-accessible circuitry according to safety standard requirements.



12. Power supply circuits: step-up, step-down



Step-up circuit: this circuit is used to produce 14V (VSUP14V) by using 5V from USB (+5V_USB_PC).



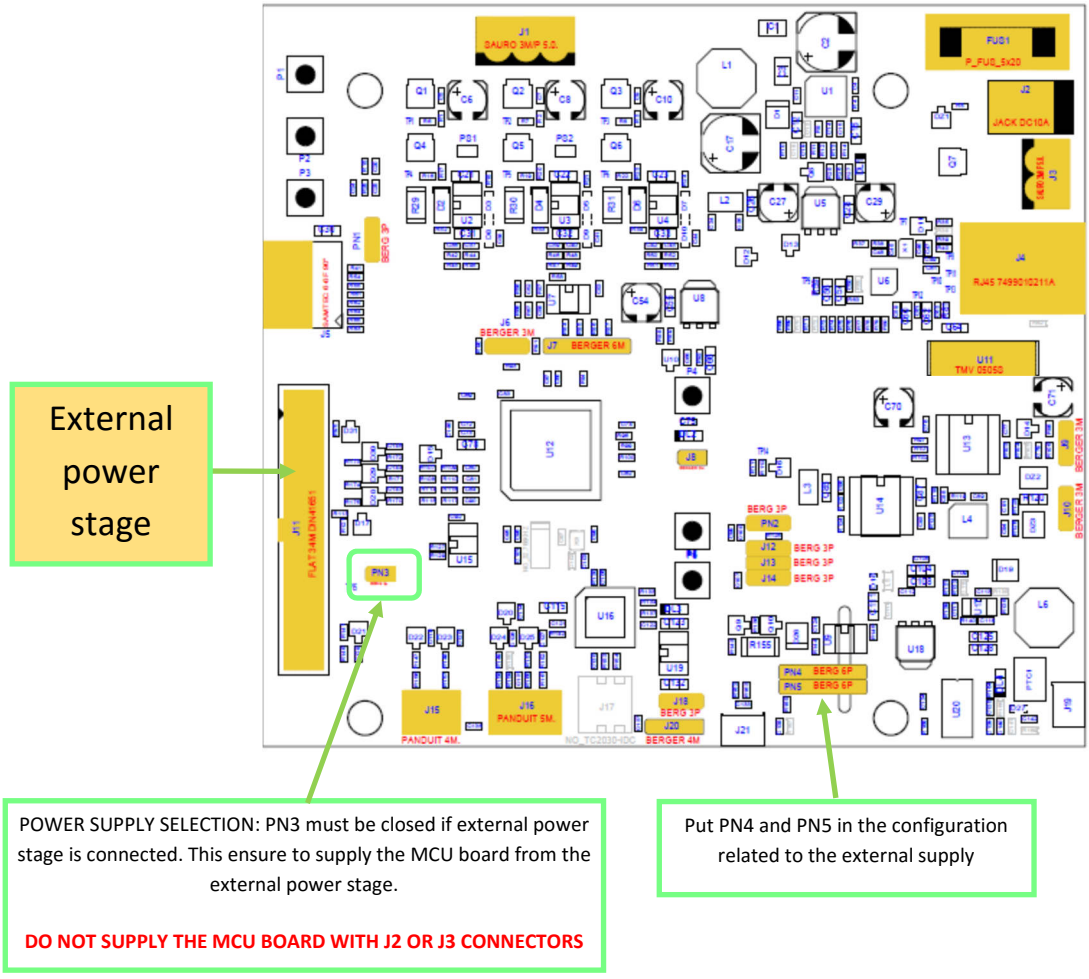
Step-down circuit: this circuit is used to produce 15V from external power supply (VBUS) and 5V by using a linear regulator

13. Interface to an external power stage

Since internal power stage allows only the management of low voltage motors, an interface with an external power stage has been developed.

The selection between the internal power stage and the external power stages is ensured by jumpers PN3, PN4, PN5 via hardware (to ensure the supply voltage to the MCU board), but it is mandatory to load the correct firmware for external power stage management

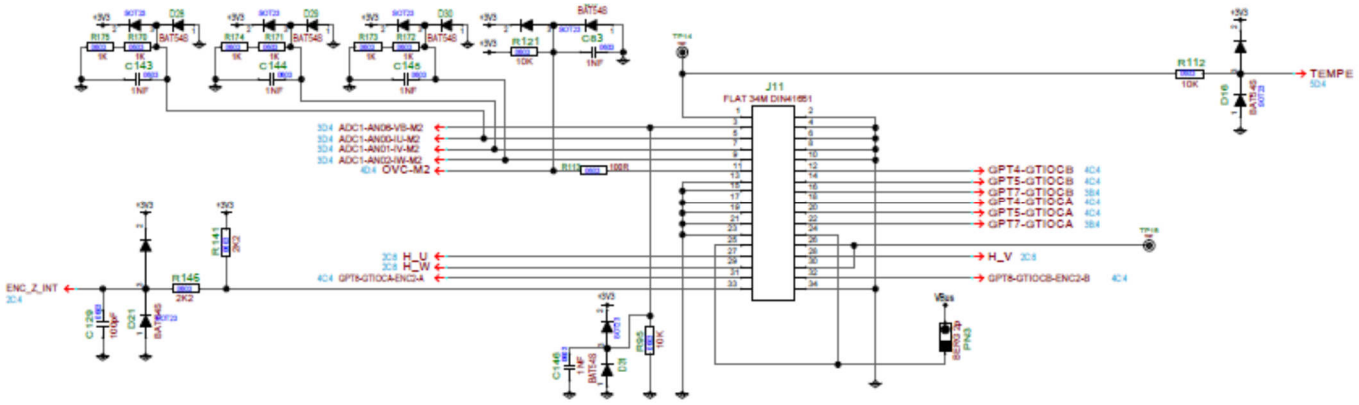
The S5D9 Microcontroller has two complementary PWM sections. One is dedicated to the internal power stage and the other is dedicated to the external one (so the appropriate firmware must be loaded). It is a safe way to ensure that the right voltage and current signals are active.



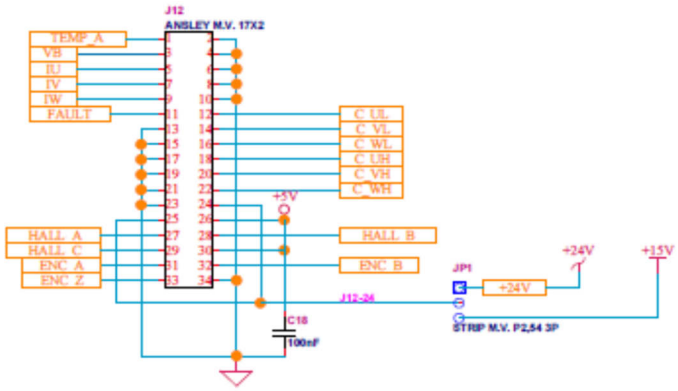
Please be careful to take into account the following precautions:

1. PN3, chose the configuration with the pins 1 and 2 shorted, when external power supply board is used (as specified in the previous page).

Please find below the drawing of the interface connector.



For a comparison, find below the drawing of the corresponding connector in the 1.5KW, 230V_{AC} external power stage (E6108A).



If using a different external power stage, please keep present the following notes:

a) The PWM drive signals are directly connected to the microcontroller output pins, and there is no pull-up or pull-down resistor connected, so the polarization has to be done in the power stage. In case of alarm, the microcontroller output pins can be placed in high impedance state, so the external polarization is necessary. These output commands are logic level signals, with limited current output capability, so an external driver is probably required. A further line is connected to the microcontroller: it is the external alarm signal, connected to the POE input pin; this pin is polarized with a 10K pull-up toward the logic supply.

b) The analog measurement signals from power stage, in particular the current readings and the DC link voltage reading are clamped (with diodes from logic GND and to logic V_{cc}) and weakly filtered, then directly connected to the A/D converter input pins of the microcontroller, so the external power stage has to take care of the gain and the offset of these signals.

c) The ground connection is always active, and it represents the reference for all the interface signals.

In the next figure a simple example regarding how the power board connections have to be arranged, is presented.

In this schematic it is supposed that the power board has its own supply for the power module (+15V); +15 to +24V supply from power board is also used to supply the microcontroller (thanks to jumper PN3 in microcontroller board).

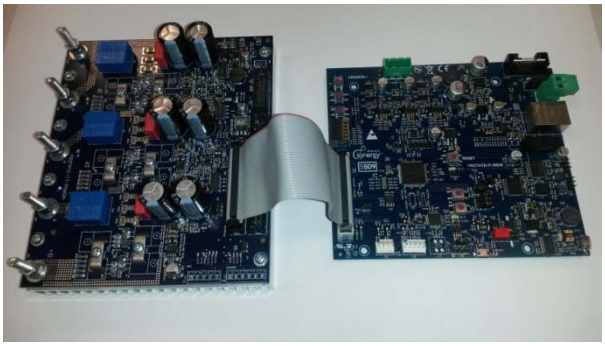
Please refer to the complete schematics for further details that are available on the website: www.renesas.eu/motorcontrol.

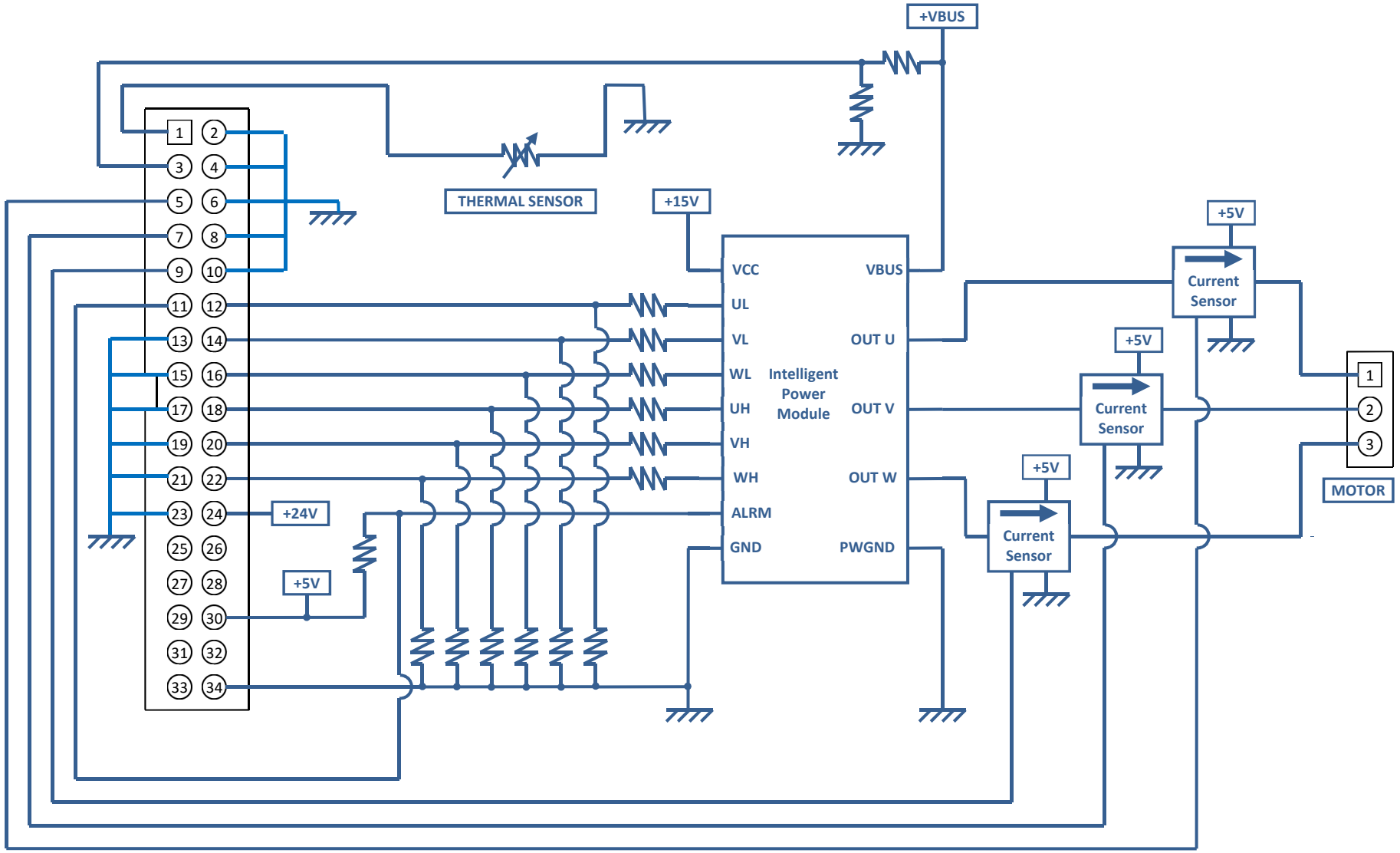
Two power stages are currently available under request, were successfully tested with the YROTATE-IT-S5D9 kit.

On the left-hand side, the power stage can manage up to $300V_{DC}$ for 10A and the right-hand side $60V_{DC}$ for 60A.

The full documentation of the power stage are available to provide more details.






+VBUS
THERMAL SENSOR
+15V
VCC
UL
VL
WL
UH
VH
WH
ALRM
GND
VBUS
OUT U
OUT V
OUT W
PWGND
Intelligent Power Module
+24V
+5V
+5V
Current Sensor
+5V
Current Sensor
+5V
Current Sensor
MOTOR

14. Single shunt current reading

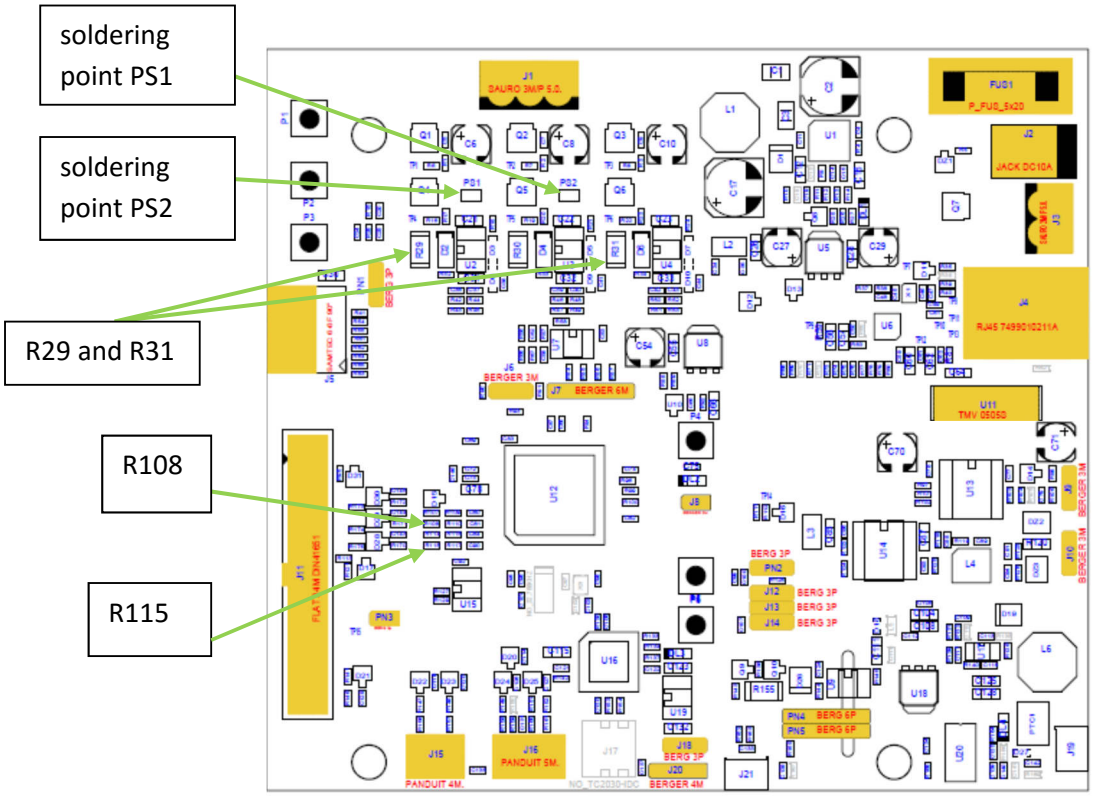
While the normal configuration of the board and the standard software are based on three shunts current reading, we also offer the possibility to configure the board for single shunt current reading.

Some hardware modifications are required and a different software version has to be program into the S5D9 flash memory.

The required hardware modifications are the following (please refer to the board schematics):

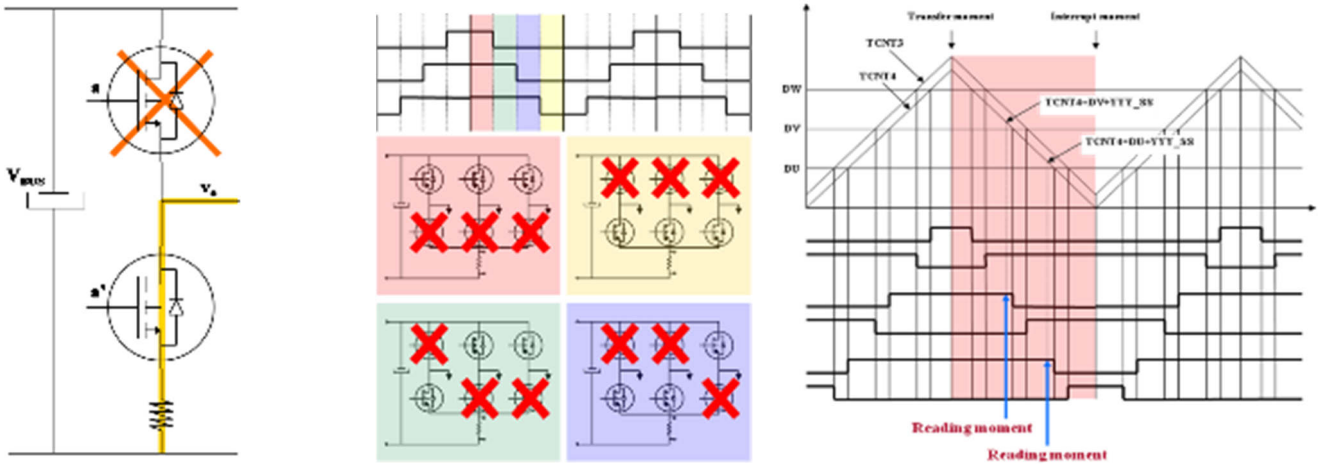
- Remove from the board the shunts R29 and R31 (they are the shunts related to the phases U and W)
- Remove R115 and R108
- Close the soldering points PS1 and PS2 (those soldering points put the three inverter harms in common, below the lower switches and above the shunts)

  The components involved in the modifications are indicated in the figure below.



15. Current reading timing in three shunts and single shunt configurations

The figures below show the different situations related to the two configurations. The first figure is related to three shunts current reading, the other are related to the single shunt current reading.



Three shunts configuration

In the three shunts configuration the current in one shunt is equal to the corresponding phase current when the corresponding lower switch is ON.

The most suitable moment to read the current in this configuration is at the trough of the PWM.

By default the YROTATE-IT-S5D9 kit is delivered in the three shunts configuration.

Single shunt configuration

In the single shunt configuration, only when one or two of the lower switches are ON the current through the shunt is related with the phase current.

When only one of the lower switches is ON, the current in the shunt is equal to the current of the corresponding inverter phase.

When two of the lower switches are ON, the current in the shunt is equal to the sum of the currents of the corresponding phases that is it is minus the current of the third phase.

Important Note:

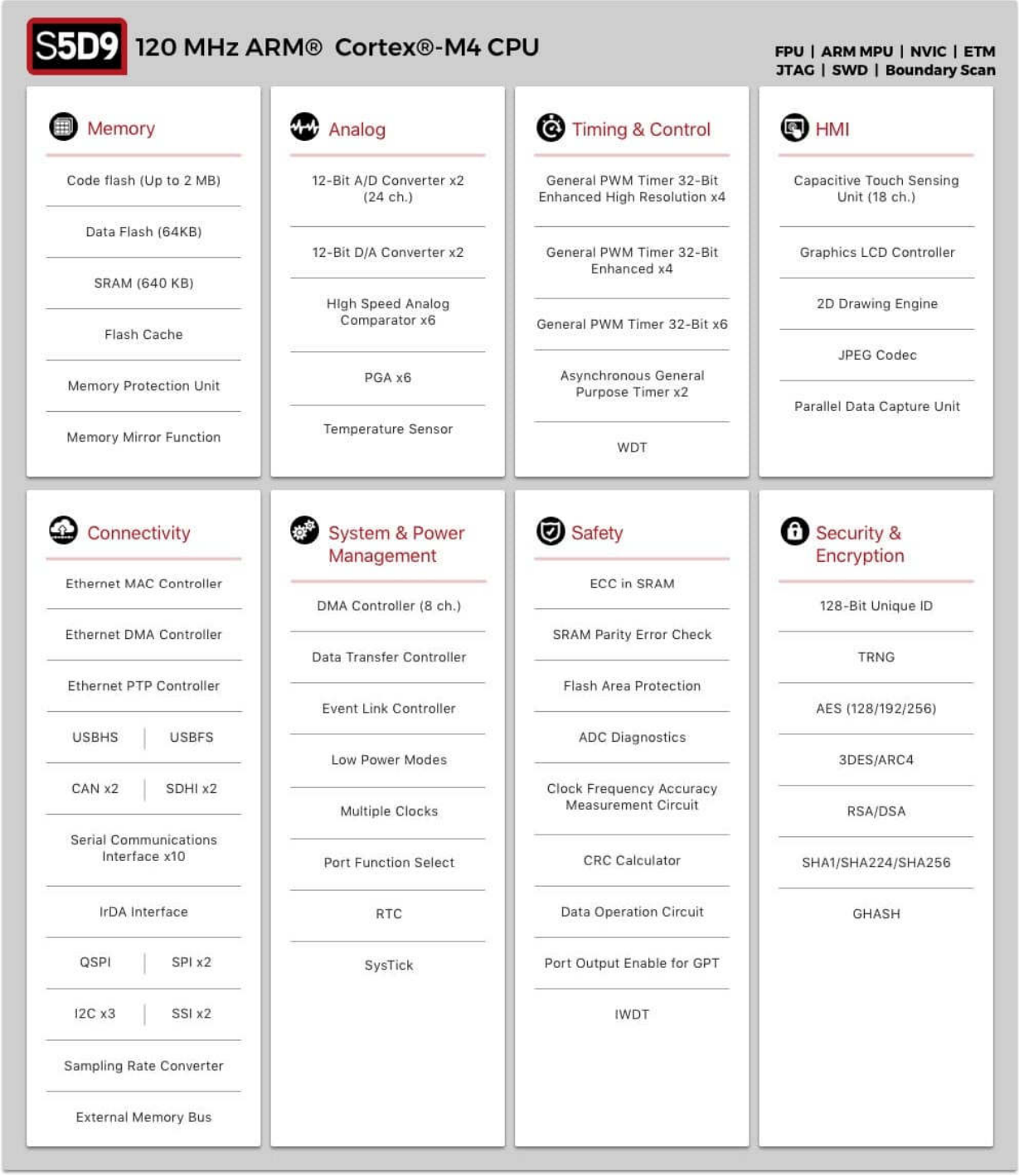
The software project available on the website: www.renesas.eu/motorcontrol is designed under e²studio environment and only for three shunts configuration.

16. Microcontroller S5D9 short overview

The S5D9 Group is 32-bit microcontroller and suited for single inverter control and has a built-in FPU (floating-point processing unit) that enables it to easily program complex inverter control algorithms. This helps to greatly reduce the man-hours required for software development and maintenance. Below are the main specifications:

BASIC INFORMATION			
Family	Synergy MCU	12-Bit D/A Converter (ch)	2
Series	S5 Series	High-Speed Analog Comparator (ch)	6
Group	S5D9	Low-Power Analog Comparator (ch)	0
CPU	M4	PGA (ch)	6
Max. Freq (MHz)	120	OPAMP (ch)	0
Code Flash (KB)	2048	Temperature Sensor (ch)	1
Data Flash (KB)	64	Ethernet	YES
SRAM (KB)	640	Ethernet MAC Controller (port)	1
Pin Count	100	Ether PTP Controller	YES
I/O Count	76	USBFS	YES
Operating Voltage Range (V)	2.7 - 3.6	CAN (ch)	2
Package Type	LQFP	SD/MMC Host Interface	YES
Operating Temperature Range (degC)	-40 to 105	SD/MMC Host Interface (ch)	2
External Memory Bus	YES	Serial Communi- cation Interface (ch)	10
External Memory Bus (bit)	8	IrDA	1
General PWM Timer 32-Bit Enhanced High Resolution (ch)	4	QSPI	YES
General PWM Timer 32-Bit Enhanced (ch)	4	SPI (ch)	2
General PWM Timer 32-Bit (ch)	5	I2C (ch)	2
General PWM Timer 16-Bit (ch)	0	Serial Sound Interface	YES
Asynchronous General purpose Timer	2	Serial Sound Interface (ch)	1
Watchdog Timer	1	Graphics LCD Controller	YES
Independent Watchdog Timer	1	Segment LCD Controller	No
DMA Controller (ch)	8	Parallel Data Capture Unit	YES
Data Transfer Controller	1	Capacitive Touch Sensing Unit (ch)	12
RTC	1	Safety	YES
14-Bit A/D Converter (unit)	0	Security and Encryption	YES
12-Bit A/D Converter (unit)	2	Security and Encryption (Remarks)	128-bit Unique ID TRNG AES(128/192/256) 3DES/ARC4 RSA/DSA SHA1/SHA224/SHA256 GHASH
12-Bit A/D Converter (ch)	19	Production Status	Mass Production

Please find below the S5D9 microcontroller block diagram.

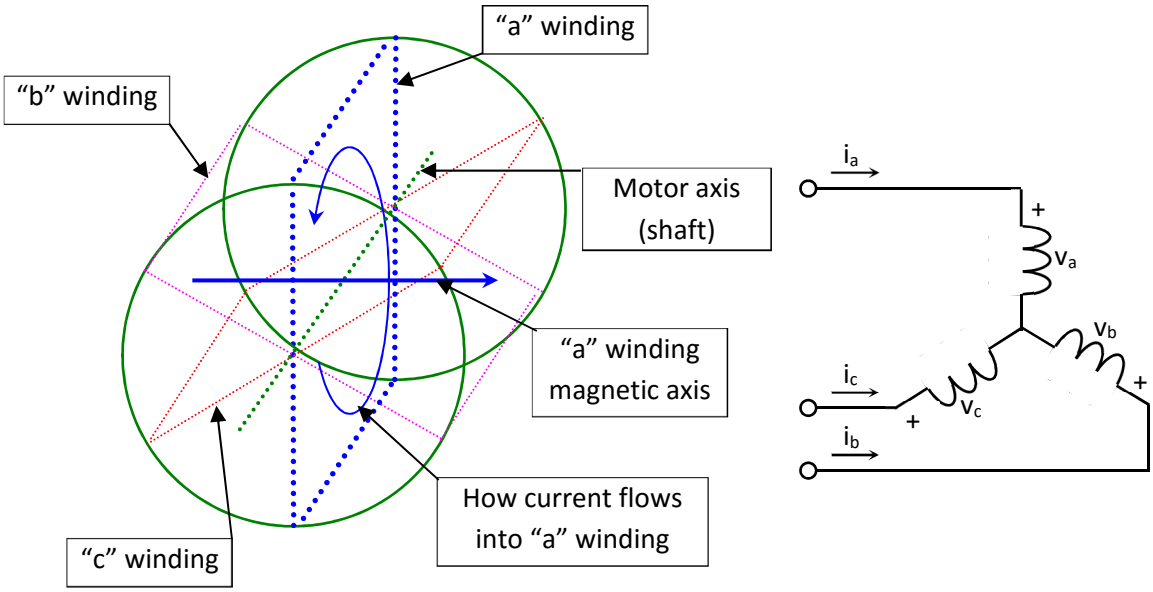


17. Permanent Magnets Brushless Motor model

The synchronous permanent magnets motor (sinusoidal brushless motor) is widely used in the industry. More and more home appliance makers are now using such brushless motor, mainly because of the intrinsic motor efficiency.

The permanent magnet motor is made with few components:

1. A *stator* formed by stacking sheared metal plates where internally the copper wiring is wound, constructing the stator winding
2. A *rotor* in which permanent magnets are fixed
3. Two covers with ball bearings that keep together the stator and the rotor; the rotor is free to rotate inside the stator



The working principle is quite simple: if we supply the motor with a three-phase system of sinusoidal voltages, at constant frequency, in the stator windings flow sinusoidal currents, which create a rotating magnetic field.

The permanent magnets in the rotor tend to stay aligned with the rotating field, so the rotor rotates at synchronous speed.

The main challenge in driving this type of motor is to know the rotor position in real-time, so mainly implementation are using a position sensor or a speed sensor.

In our implementation, the system is using either one or three shunts to detect the rotor position in real-time.

Let's analyse the motor from a mathematic point of view.

If we apply three voltages $v_a(t)$, $v_b(t)$, $v_c(t)$ to the stator windings, the relations between phase voltages and currents are:

$$v_a = R_S i_a + \frac{d\lambda_a}{dt}$$

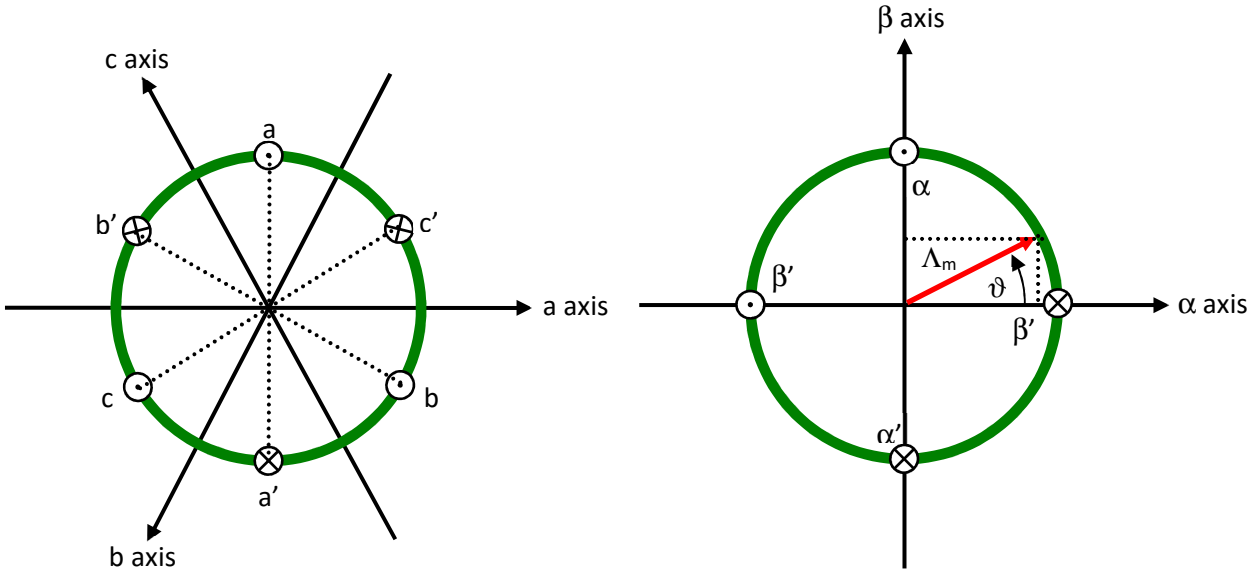
$$v_b = R_S i_b + \frac{d\lambda_b}{dt}$$

$$v_c = R_S i_c + \frac{d\lambda_c}{dt}$$

- λ_i is the magnetic flux linkage with the i-th stator winding

- R_S is the stator phase resistance (the resistance of one of the stator windings)

The magnetic flux linkages λ_i are composed by two items, one due to the stator currents, one to the permanent magnets.



Real axes (a, b, c) and equivalent ones (α, β); a fixed amplitude vector can be completely determined by its position respect the (α, β) system (angle ϑ)

The permanent magnet creates a magnetic field that is constant in amplitude and fixed in position in respect to the rotor. This magnetic field can be represented by vector Λ_m whose position in respect to the stator is determined by the angle ϑ between the vector direction and the stator reference frame.

The contribution of the permanent magnets in the flux linkages depends on the relative position of the rotor and the stator represented by the mechanical-electric angle ϑ.

It is, in every axis, the projection of the constant flux vector Λ_m in the direction of the axis:

$$\lambda_a = Li_a + \Lambda_m \cos(\vartheta)$$
$$\lambda_b = Li_b + \Lambda_m \cos(\vartheta - 2\pi/3)$$
$$\lambda_c = Li_c + \Lambda_m \cos(\vartheta - 4\pi/3)$$

Supposing that the rotor is rotating at constant speed ω (that is: ϑ(t) = ωt) the flux linkages derivatives can be calculated, and we obtain:

$$v_a = R_S i_a + L\frac{di_a}{dt} - \omega\Lambda_m \sin(\vartheta)$$
$$v_b = R_S i_b + L\frac{di_b}{dt} - \omega\Lambda_m \sin(\vartheta - 2\pi/3)$$
$$v_c = R_S i_b + L\frac{di_b}{dt} - \omega\Lambda_m \sin(\vartheta - 4\pi/3)$$

A “three phases system” may be represented by an equivalent “two phases system”. So the by using specific transformations, our three equations system is equivalent to a two equations system. It is basically a mathematical representation in a new reference coordinates system.

In the two phases (α,β) fixed system the above equations become:

$$v_\alpha = R_S i_\alpha + \frac{d\lambda_\alpha}{dt}$$
$$v_\beta = R_S i_\beta + \frac{d\lambda_\beta}{dt}$$

For the magnetic field equations, we got:

$$\lambda_\alpha = Li_\alpha + \lambda_{\alpha m} = Li_\alpha + \Lambda_m \cos(\vartheta)$$
$$\lambda_\beta = Li_\beta + \lambda_{\beta m} = Li_\beta + \Lambda_m \sin(\vartheta)$$

After performing the derivation:

$$\frac{d\lambda_\alpha}{dt} = L\frac{di_\alpha}{dt} - \omega\Lambda_m \sin(\vartheta) = L\frac{di_\alpha}{dt} - \omega\lambda_{\beta m}$$
$$\frac{d\lambda_\beta}{dt} = L\frac{di_\beta}{dt} + \omega\Lambda_m \cos(\vartheta) = L\frac{di_\beta}{dt} + \omega\lambda_{\alpha m}$$

Finally, we obtain for the voltages in (α,β) system:

$$v_\alpha = R_S i_\alpha + L\frac{di_\alpha}{dt} - \omega\lambda_{\beta m}$$

$$v_\beta = R_S i_\beta + L\frac{di_\beta}{dt} + \omega\lambda_{\alpha m}$$

A second reference frame is used to represent the equations as the frame is turning at the rotor speed. So the “d” axis is chosen in the direction of the magnetic vector Λ_m, and with the “q” axis orthogonal to the “d” axis. The new reference system is (d, q).

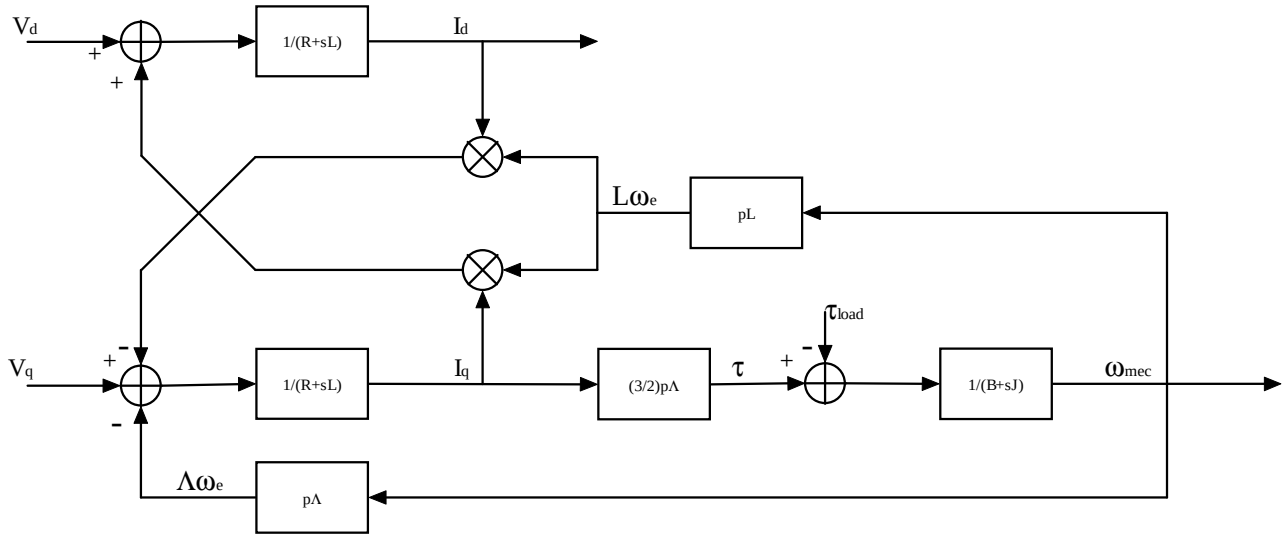
The reference frame transformations from the (α,β) system to the (d, q) system depends on the instantaneous position angle ϑ

So we obtain two inter-dependant equations in the (d, q) system:

$$v_d = R_S i_d + L\frac{di_d}{dt} - \omega L i_q$$

$$v_q = R_S i_q + L\frac{di_q}{dt} + \omega L i_d + \omega\Lambda_m$$

These two equations represent the mathematical motor model.



A control algorithm which wants to produce determined currents in the (d, q) system must impose voltages given from the formulas above.

This is ensured by closed loop PI control on both axis “d” & “q” (Proportional Integral).

Since there is a mutual influence between the two axes, decoupling terms can be used.

In the block scheme the mechanic part is included, where “p” is the number of pole pairs, while “B” represents friction, “J” the inertia, “τ_{load}“ the load torque and “τ” the motor torque.

$$\tau = \frac{3}{2} \times p \times \Lambda$$

The angular speed ω is represented in the scheme as ω_e to distinguish the electrical speed from the mechanical one.

Let's now consider the equations we have seen in (α,β) system:

$$v_\alpha = R_S i_\alpha + \frac{d\lambda_\alpha}{dt}$$
$$v_\beta = R_S i_\beta + \frac{d\lambda_\beta}{dt}$$

These equations show that magnetic flux can be obtained from applied voltages and measured currents simply by integration:

$$\lambda_\alpha = \lambda_{\alpha 0} + \int_0^t (v_\alpha - R_S i_\alpha) dt$$
$$\lambda_\beta = \lambda_{\beta 0} + \int_0^t (v_\beta - R_S i_\beta) d$$

Furthermore:

$$\Lambda_m \cos(\vartheta) = \lambda_\alpha - L i_\alpha$$
$$\Lambda_m \sin(\vartheta) = \lambda_\beta - L i_\beta$$

If the synchronous inductance L is small, the current terms can be neglected, if not they have to be considered. In general:

$$x = \Lambda_m \cos(\vartheta) = \lambda_\alpha - L i_\alpha = \lambda_{\alpha 0} + \int_0^t (v_\alpha - R_S i_\alpha) dt - L i_\alpha$$
$$y = \Lambda_m \sin(\vartheta) = \lambda_\beta - L i_\beta = \lambda_{\beta 0} + \int_0^t (v_\beta - R_S i_\beta) dt - L i_\beta$$

So in the (α,β) system phase we obtain from the flux components:

$$\vartheta = \arctan(x/y)$$

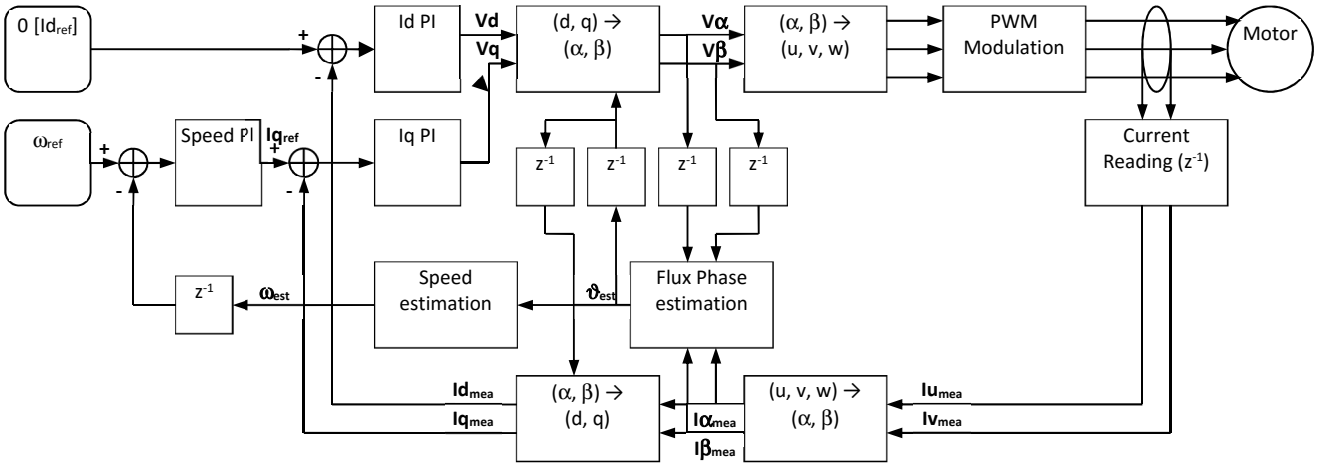
The system speed ω can be obtained as the derivative of the angle ϑ.

$$\omega = \frac{d}{dt} \vartheta(t)$$

Based on this, a sensorless control algorithm was developed to give the imposed phase voltages, to measure phase currents, to estimate the angular position ϑ and finally the system speed.

18. Sensorless Field Oriented Control algorithm

Please, find below the sensorless vector control algorithm block diagram.



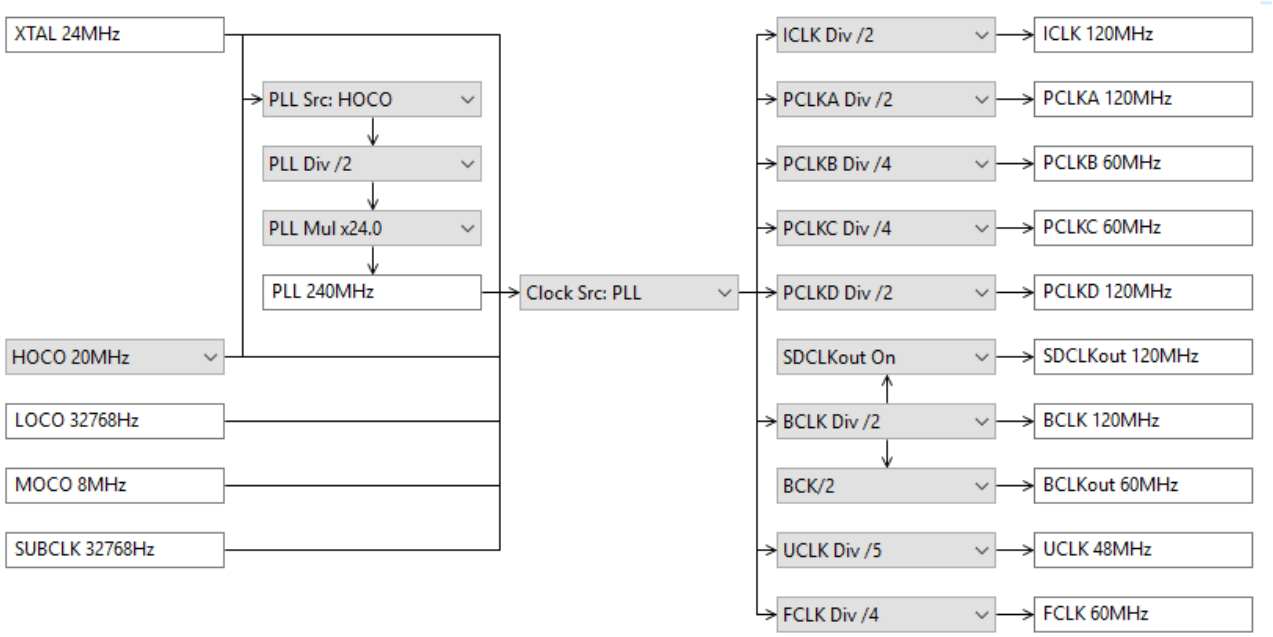
The main difference between the three shunts configuration and the single shunt one is in the “Current Reading” block, the rest of the algorithm remains the same in principle, even if the blocks order has been adjusted.

19. Software description

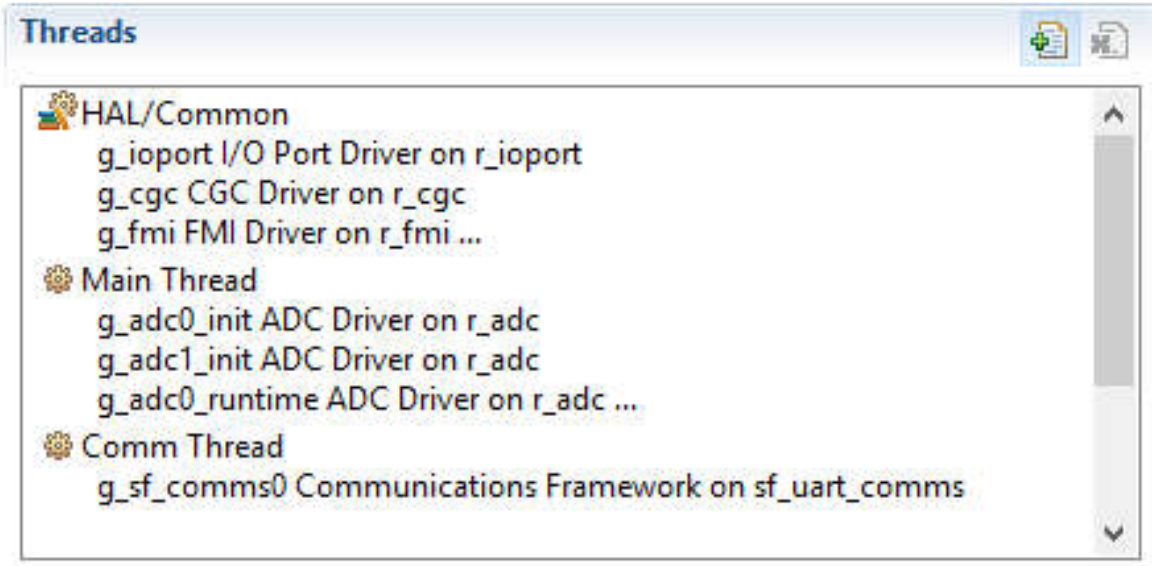
The software delivered in the YROTATE-IT-S5D9 kit, previously described, works on the Synergy S5D9 microcontroller.

Using Synergy the project development is in some way different than with other controllers, and the great part of the trivial work is made automatically by the IDE or is made very immediate through specific configuration editors. In particular all the pin configuration is simply managed with a friendly interface etc.

Let's consider as an example the clock configuration window, which allows the configuration of the clock settings with a simple graphic interface:

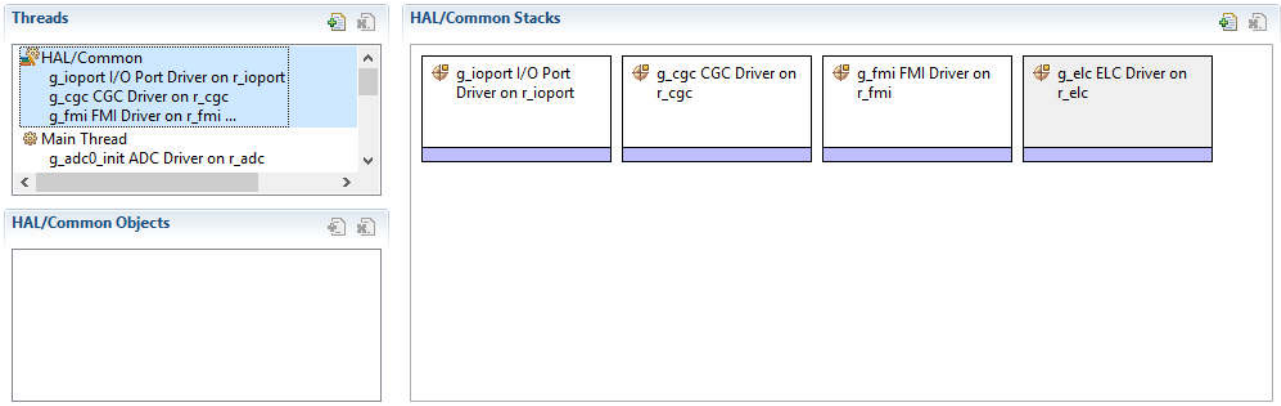


The demo project uses the operating system ThreadX, which takes care of all the tasks management. The developer should only configure the tasks required by its application. In particular the threads organization for the motor control project is the following.

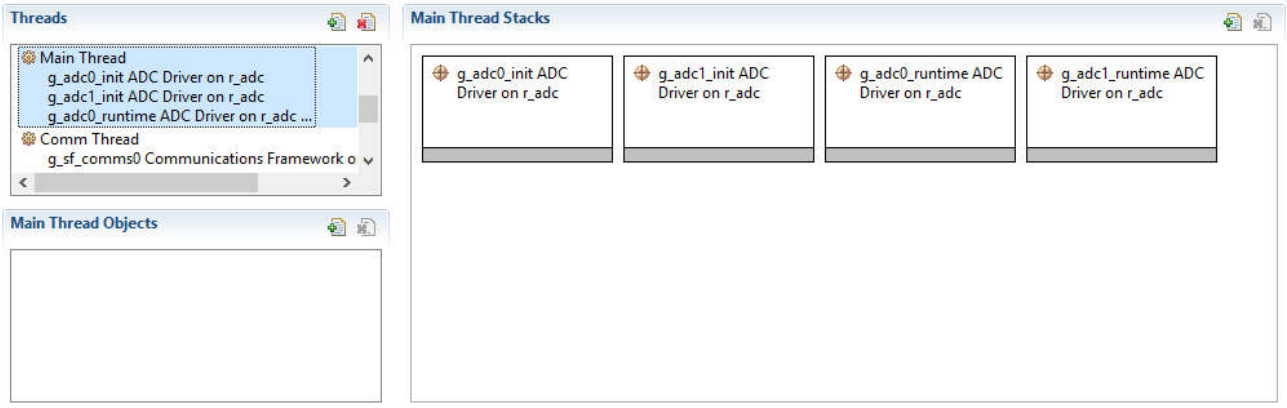


As shown in the configuration window, the projects are articulated into three main threads:

- HAL/Common; automatically generated, it includes I/O port driver, clock generator configuration, etc.

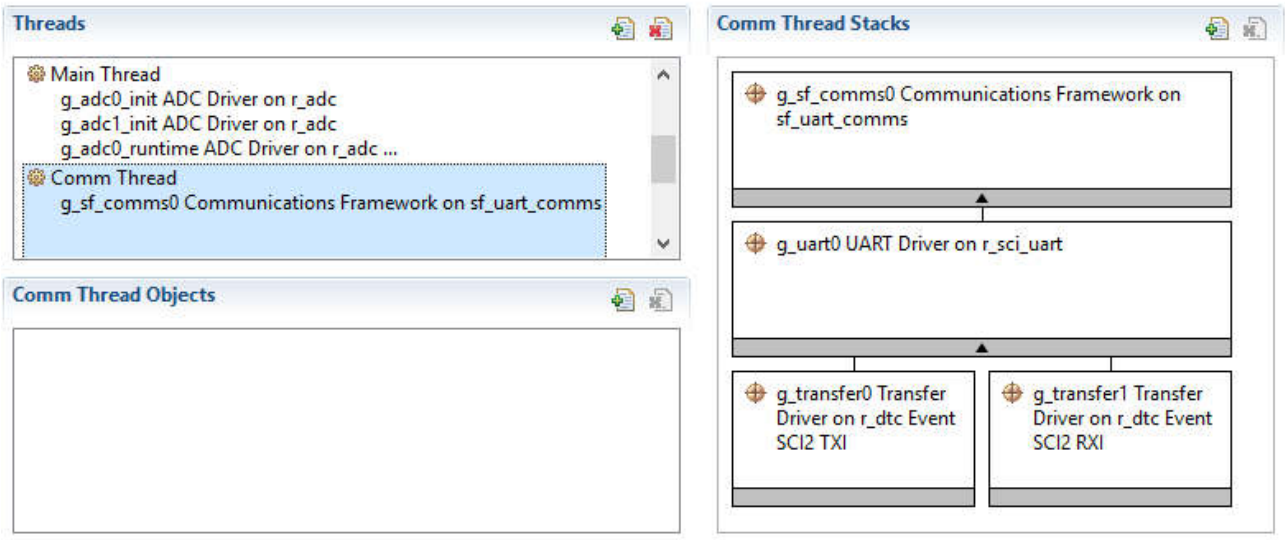


- Main Thread; it includes four instances of ADC driver, two based on peripheral 0 and two based on peripheral 1



- g_ADC0_init; used to detect the current reading circuit offset of the internal power stage at startup;
- g_ADC0_runtime; used to make the current (and other analog quantities) readings, related to the internal power stage, during motor control and to generate the motor control interrupt; the periodic conversion is launched by the PWM (at its trough) through the ELC.
- g_ADC1_init; used to detect the current reading circuit offset of the external power stage at startup;
- g_ADC1_runtime; used to make the current (and other analog quantities) readings, related to the external power stage, during motor control and to generate the motor control interrupt; the periodic conversion is launched by the PWM (at its trough) through the ELC.

- Comm Thread; it includes the communication framework on UART to manage the serial communication with the GUI.



The threads are automatically managed by the operating system.

Pls. find the configuration of each driver, framework etc. on the property window of the Synergy perspective.

19.1 Source Files organization

In a synergy project part many files are generated automatically and should not be directly modified by the developer. The part in which the developer can directly put his hands in placed in the <*src*> directory, and in our project it is organized as follows:

<*src*>

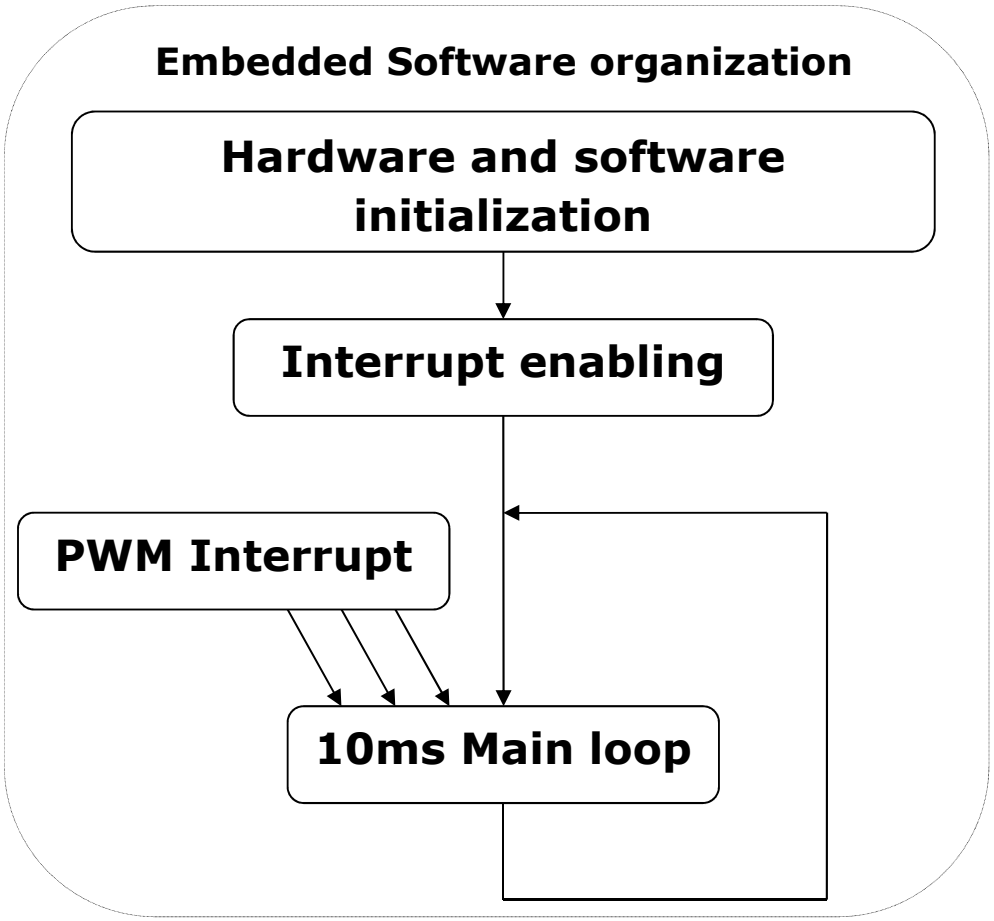
- *hal_entry.c* (**h**ardware **a**bstraction **l**evel entry): here the user can put his own "low level" code; it is not used in our case.
- *main_thread_entry.c*: here we put the "high level" startup, the first initializations and the main management loop.
- *comm_thread_entry.c*: here we put the GUI communication management code.
- <*mc_timer*>: in this directory it is placed the driver for the three-phases complementary pwm, based on GPT peripheral.
    - *mc_timer_api.h*: here the developer can find the api functions to be used to work with the three-phases complementary pwm.
    - *gpt_pwm_defs.c*: here the data structures related to the pwm on GPT can be found.
    - *gpt_pwm_defs.h*
    - *gpt_pwm.c*: driver implementation.
    - *gpt_pwm.h*

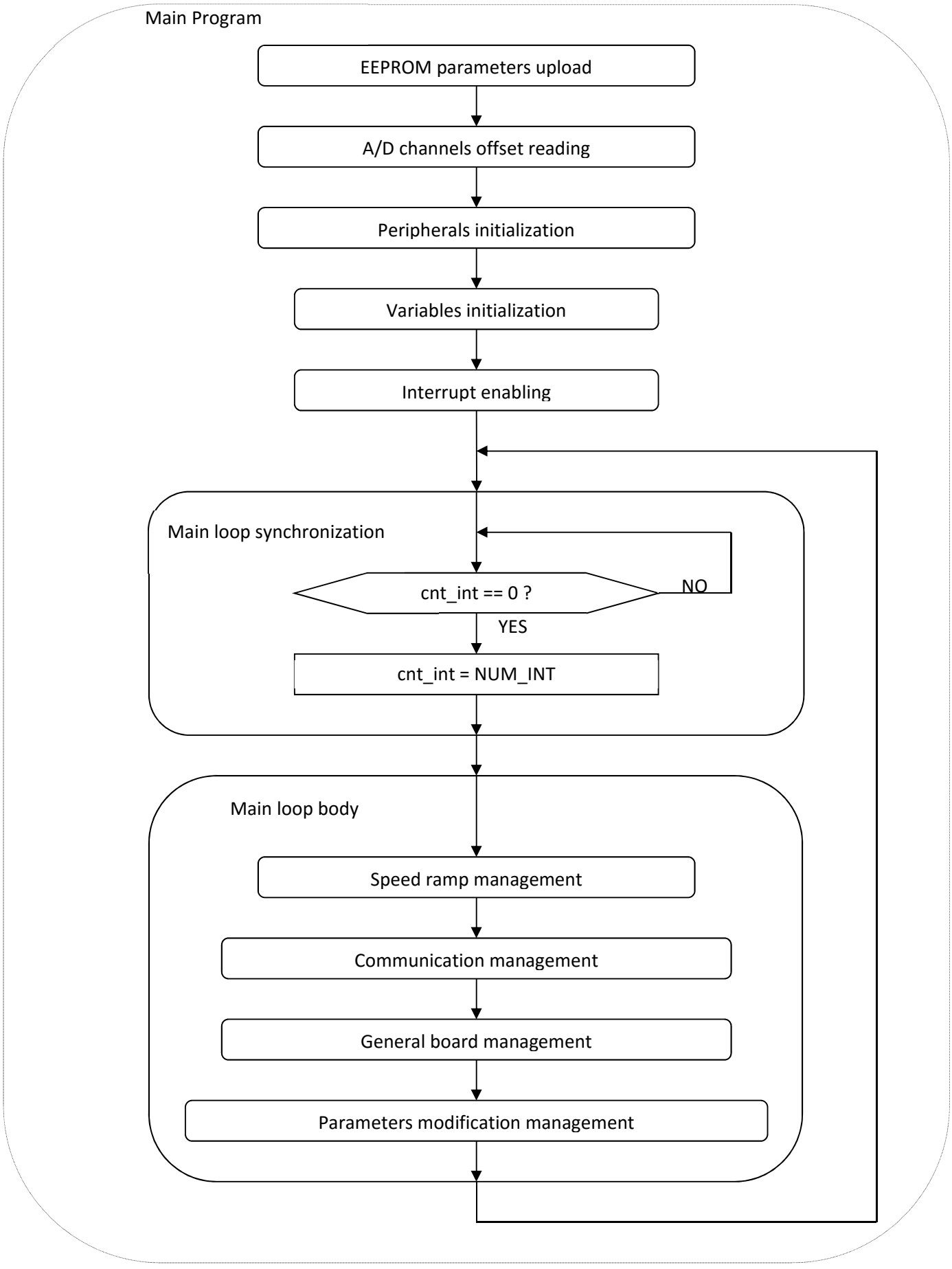
  - *gpt_pwm_private_api.h*
  - *hw_gpt_common.h*
  - *hw_gpt_private.h*
  - *r_gpt.h*: general structures for GPT management.
  - *r_timer_api.h*: general apis for timers.
- <*motor_control*>: in this directory we placed the files directly involved in motor-control management.
  - *typedefine.h*: definition of custom types used in the project (included MISRA types).
  - *mask.h*: definition of general utility bitmasks.
  - *globdef.h*: definition of general utility macro, referred to peripherals, I/O etc.
  - *pws_EBR_S5D9_0.h*: hardware related defines, referred to the on board low voltage powers stage (her are placed for example the gains of the current reading circuit and other hardware-related constants).
  - *pws_EBR_E6140.h*: hardware related defines, referred to E6140 external power stage (it is similar to the previous one).
  - *customize.h*: in this files are placed many defines which can be used by the developer to enable/disable some program features, and/or some constants which have not been included in the eeprom configuration parameters, so their modification requires a re-compilation of the project.
  - *const_def.h*: definition of the basic constants used in the project.
  - *gui_defs.h*: here the data-structure used to communicate with the GUI is defined; you can find here all the measurements that the program sends to the GUI and all the commands that the GUI sends to the program.
  - *parameters.h*: here the data-structure used for the eeprom parameters management is defined.
  - *par_defines.h*: here are placed the macro which are used in the previous file; the developer can modify the maximum and minimum values of all the configuration parameters in this file.
  - *defpar.h*: here the macro referred to the default values of the parameters are placed; here the developer can modify the default values of the parameters; this file can be automatically generated by the GUI.
  - *ges_eqp.c*: this file contains the code needed to manage the eeprom which contains the configuration parameters.
  - *i_ges_eqp.h*: interface definitions for the use of the functions included in *ges_eqp.c*.
  - *userif.c*: this file contains the code which manages the communication protocol with the GUI.
  - *i_userif.h*: interface definitions for the use of the functions contained in *userif.c*.
  - *motorcontrol.c*: in this file the function directly related to motor-control are included, for example the main control interrupt and so on.
  - *i_motorcontrol.h*: interface definitions for the use of the functions contained in *motorcontrol.c*.
  - <*library*>: in this directory is placed the motor-control library.
    - *mcrplibf.c*: motor control math library, which includes transformations, angle-management functions, observer/estimator etc.
    - *i_mcrplibf.h*: interface definitions for the use of the functions contained in *mcrplibf.c*.
    - *PI_tun.c*: current PI tuning fuctions.
    - *i_PI_tun.h*: interface definitions for the use of the functions contained in *PI_tun.c*.
    - *mot_ident.c*: motor identification functions.
    - *i_mot_ident.h*: interface definitions for the use of the functions contained in *mot_ident.c*.
- <synergy_gen>: in this directory are placed the synergy generated files, which should not be modified by the developer.

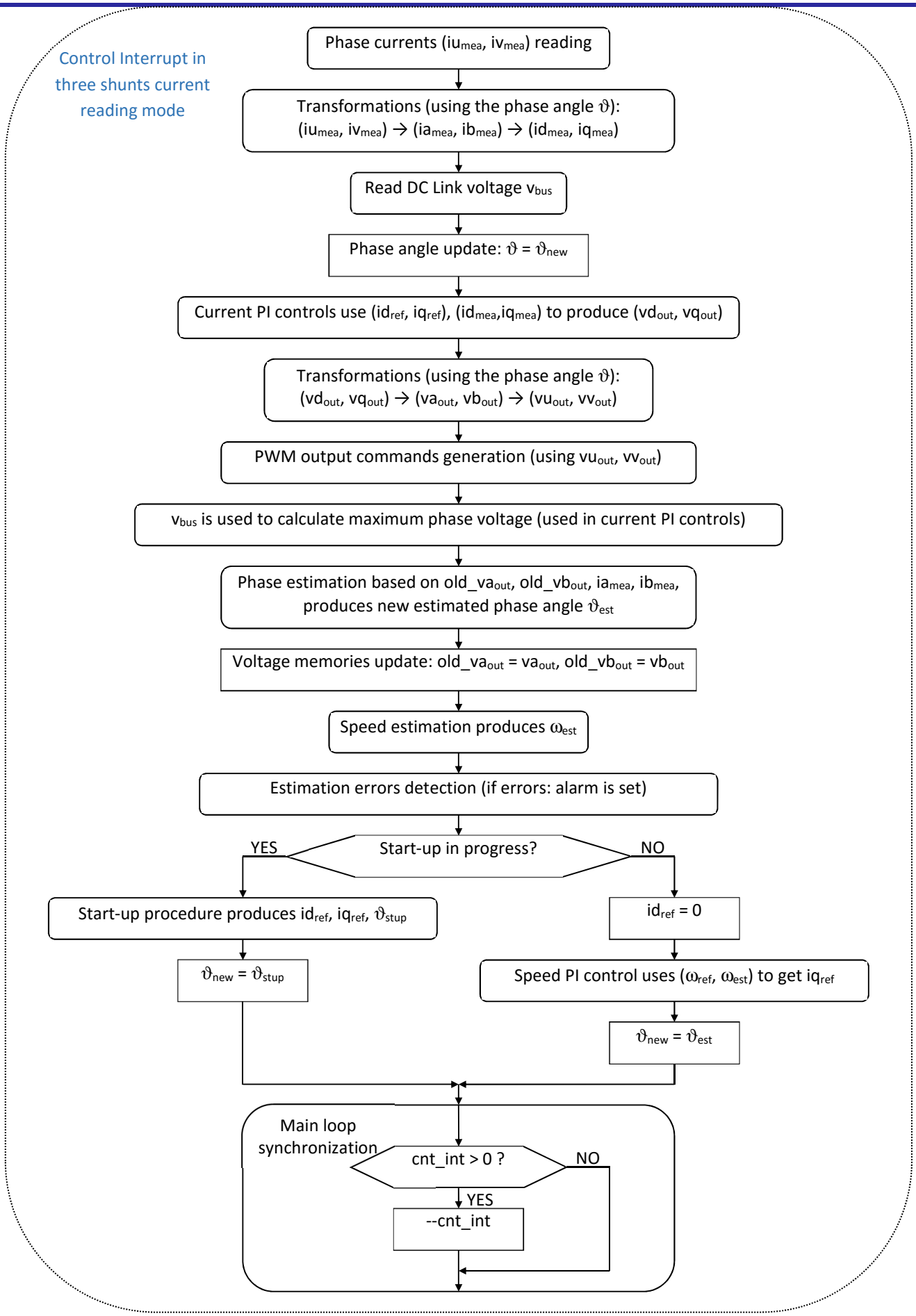
19.2 Program flux

At startup the Synergy software takes care of all the necessary initializations, as configured by the developer with the Synergy Configuration manager. Then the operating system (ThreadX) calls the threads in the proper moments. This process is synchronized with PWM timer management by means of a simple mechanism: through ELC, the PWM timer starts an ADC conversion on every PWM trough; when the conversion is finished an end-of-conversion interrupt is generated.

In the callback function the motor control (current loops, observer etc.) is managed, and also a synchronization timer to release the main loop is managed. The main thread at every wake-up check the counter and if it is the case (and this is true every 10ms) then performs one complete cycle. In this way the operating system becomes completely transparent to the user. So we can examine the program flux considering the three following graphs.




Main Program
EEPROM parameters upload
A/D channels offset reading
Peripherals initialization
Variables initialization
Interrupt enabling
Main loop synchronization
cnt_int == 0 ?
NO
YES
cnt_int = NUM_INT
Main loop body
Speed ramp management
Communication management
General board management
Parameters modification management


Control Interrupt in three shunts current reading mode
Phase currents (iumea, ivmea) reading
Transformations (using the phase angle ϑ): (iumea, ivmea) → (iamea, ibmea) → (idmea, iqmea)
Read DC Link voltage vbus
Phase angle update: ϑ = ϑnew
Current PI controls use (idref, iqref), (idmea,iqmea) to produce (vdout, vqout)
Transformations (using the phase angle ϑ): (vdout, vqout) → (vaout, vbout) → (vuout, vvout)
PWM output commands generation (using vuout, vvout)
vbus is used to calculate maximum phase voltage (used in current PI controls)
Phase estimation based on old_vaout, old_vbout, iamea, ibmea, produces new estimated phase angle ϑest
Voltage memories update: old_vaout = vaout, old_vbout = vbout
Speed estimation produces ωest
Estimation errors detection (if errors: alarm is set)
YES
Start-up in progress?
NO
Start-up procedure produces idref, iqref, ϑstup
idref = 0
ϑnew = ϑstup
Speed PI control uses (ωref, ωest) to get iqref
ϑnew = ϑest
Main loop synchronization
cnt_int > 0 ?
NO
YES
--cnt_int

The interrupt frequency is linked to the PWM frequency (since the interrupt is generated by the end of an A/D conversion launched at PWM trough), and can be modified with parameters.

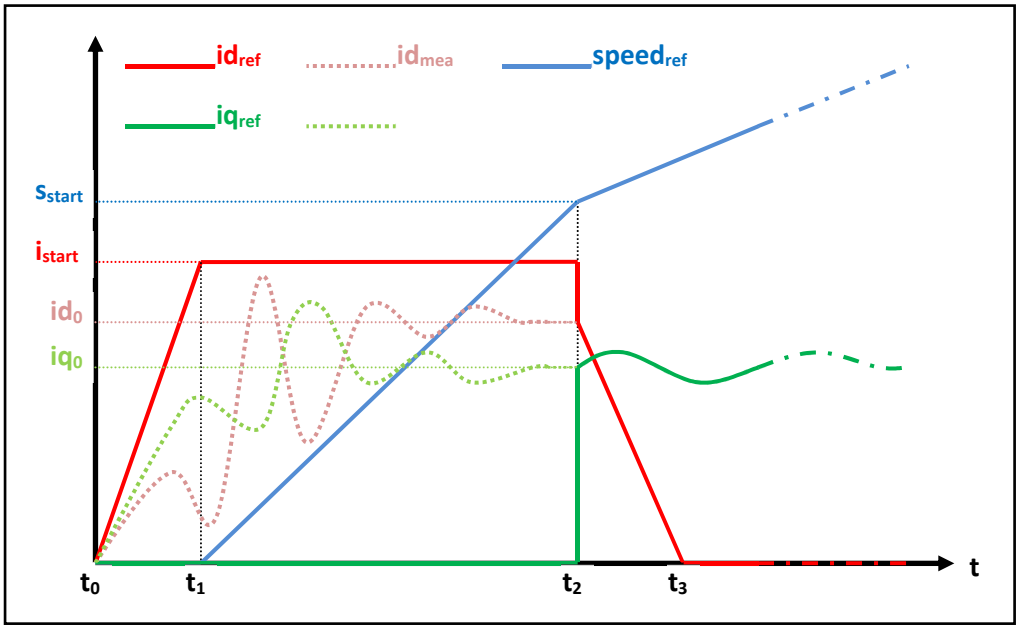
In particular. it is possible to choose (inside certain limits) the sampling frequency and the ratio between the PWM frequency and the sampling frequency (in fact using the interrupt skipping function the user can have a sampling frequency which is a submultiple of the PWM frequency, for example when a particularly high PWM freq. is required).

19.3 Start-up procedure – Embedded software

When the motor is in stand-still, the phase of the permanent magnet flux vector cannot be detected with the used algorithm. So an appropriate start-up procedure has to be applied.

The idea is to move the motor in feed-forward (with higher current than that required to win the load), till a speed at which the estimation algorithm can work. Then the system can be aligned to the estimated phase, and the current can be reduced to the strictly necessary quantity.

The following graph illustrates the strategy used (the suffix "$_{ref}$" stands for *reference*, the suffix "$_{mea}$" stands for *measured*).



Referring to the graph, the start-up procedure (in case of three shunts current reading) is described below.

a) At the beginning t_0, the system phase is unknown. No current is imposed to the motor; the system phase is arbitrarily decided to be ϑ_a=0. All the references: id_{ref}, iq_{ref} and $speed_{ref}$ are set to zero.

b) From the moment t_0, while the iq_{ref} and the $speed_{ref}$ are maintained to zero, id_{ref} is increased with a ramp till the value i_{start} is reached at the moment t_1.

The references are referred to an arbitrary (d_a, q_a) system based on the arbitrary phase ϑ_a. From this moment, the phase estimation algorithm begins to be performed, and the estimated phase ϑ_{est} is used to calculate the components of the measured current, referred to the (d, q) system based on the estimated phase, id_{mea} and iq_{mea}. The components of the current referred to the arbitrary (d_a, q_a) system are controlled to follow the references by the current PI

controllers. On the other hand, since the phase ϑ_{est} is still not correctly estimated, id_{mea} and iq_{mea} have no physical meaning. Even if they are not shown in the graph, the applied voltages are subjected to the same treatment (vd_{mea} and vq_{mea} are calculated in the algorithm).

c) At $t = t_1$, while iq_{ref} is maintained to zero and id_{ref} is maintained to its value i_{start}, $speed_{ref}$ is increased with a ramp till the value s_{start} is reached at the $t = t_2$. The system phase $\vartheta_a(t)$ is obtained simply by integration of $speed_{ref}$; in the meanwhile, the phase estimation algorithm begins to align with the real system phase. Furthermore id_{mea} and iq_{mea} begin to be similar to the real flux and torque components of the current. The real components are supposed to be id_0 and iq_0 (those values are obtained applying a low-pass filter to id_{mea} and iq_{mea}).

The interval (t_2-t_1) is the start-up time, and it is supposed to be large enough to allow the estimation algorithm to reach the complete alignment with the real phase of the system.

d) At $t = t_2$, the phase estimation process is supposed to be aligned. At this point a reference system change is performed: from the arbitrary (d_a, q_a) reference to the (d, q) reference based on the estimated phase ϑ_{est}.

The current references are changed to the values id_0 and iq_0, and all the PI controllers are initialized with these new values. The speed PI integral memory is initialized with the value iq_0, while the current PI integral memories are initialized with the analogous voltage values vd_0 and vq_0, obtained from vd_{mea} and vq_{mea}.

e) After $t > t_2$, the normal control is performed, based on the estimated phase ϑ_{est}; the speed reference is increased with the classical ramp; the id current reference is decreased with a ramp, till it reaches the value zero at the moment t_3; then it is maintained to zero; the iq current reference is obtained as output of the speed PI controller.

20. Reference system transformations in details

Find below the detailed equations used for the coordinates transformations in the embedded software for the RX23T microcontroller.

$$g_\alpha = \frac{2}{3}(g_u - \frac{1}{2}g_v - \frac{1}{2}g_w) = g_a$$
$$g_\beta = \frac{2}{3}(\frac{\sqrt{3}}{2}g_v - \frac{\sqrt{3}}{2}g_w) = \frac{1}{\sqrt{3}}(g_v - g_w) = \frac{1}{\sqrt{3}}(g_u + 2g_v)$$

(u, v, w) → (α, β)

$$g_u = g_\alpha$$
$$g_v = -\frac{1}{2}g_\alpha + \frac{\sqrt{3}}{2}g_\beta = (-g_\alpha + \sqrt{3}g_\beta)/2$$
$$g_w = -\frac{1}{2}g_\alpha - \frac{\sqrt{3}}{2}g_\beta = (-g_\alpha - \sqrt{3}g_\beta)/2$$

(α, β) → (u, v, w)

$$g_d = g_\alpha \cos(\vartheta) + g_\beta \sin(\vartheta)$$
$$g_q = -g_\alpha \sin(\vartheta) + g_\beta \cos(\vartheta)$$

(α, β) → (d, q)

$$g_\alpha = g_d \cos(\vartheta) - g_q \sin(\vartheta)$$
$$g_\beta = g_d \sin(\vartheta) + g_q \cos(\vartheta)$$

(d, q) → (α, β)

$$\left\{\begin{matrix} v_u = V\cos(\omega t + \varphi_0) \\ v_v = V\cos(\omega t + \varphi_0 - 2\pi/3) \\ v_w = V\cos(\omega t + \varphi_0 - 4\pi/3) \end{matrix}\right\} \leftrightarrow \left\{\begin{matrix} v_\alpha = V\cos(\omega t + \varphi_0) \\ v_\beta = V\sin(\omega t + \varphi_0) \end{matrix}\right\} \leftrightarrow \left\{\begin{matrix} v_d = V\cos(\varphi_0) \\ v_q = V\sin(\varphi_0) \end{matrix}\right\}$$

21. Rotor position estimation

The rotor position estimation method which has been chosen is the direct integration of the back EMF. Such method is enable by default in the RX23T inverter kit.

Please find below the fundamental equations:

$$x = \Lambda_m \cos(\vartheta) = \lambda_\alpha - Li_\alpha = \lambda_{\alpha 0} + \int_0^t (v_\alpha - R_S i_\alpha)dt - Li_\alpha$$

$$y = \Lambda_m \sin(\vartheta) = \lambda_\beta - Li_\beta = \lambda_{\beta 0} + \int_0^t (v_\beta - R_S i_\beta)dt - Li_\beta$$

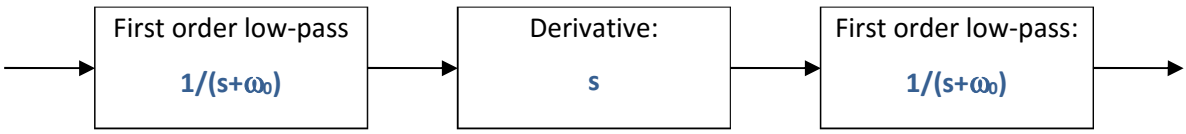
$$\vartheta = \arctan(x/y)$$

$$\omega = \frac{d}{dt}\vartheta(t)$$

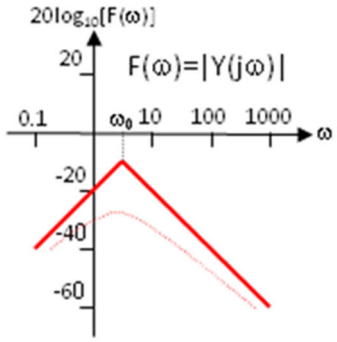
The challenges in this approach are the calculation of the integrals which is well known as a problematic issue in a numeric context, and the choice of the initial conditions, which are not known in general. There are two possibilities to overcome these difficulties:

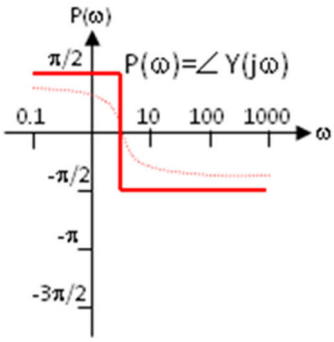
1. To use a so-called “approximated integration”, which means that instead of using an integral (1/s), a special transfer function is chosen, which is very similar to the integral in certain conditions.
2. To correct the result of the integration with a sort of feedback signal, obtained combining the estimated phase with the real flux amplitude, known as a parameter of the system.

In the **1st case**, we choose an integral approximation function which has a limited memory of the errors and with a zero DC gain. The goal is to reject any low frequency component, preventing the result to diverge, and automatically forgetting the errors (noise, etc.). This is obtained by combining a low-pass filter with a high-pass filter, as in the following scheme:



$$Y(s) = \frac{s}{(s+\omega_0)^2}$$





It is evident the relationship between Y(s) and the integral I(s)=1/s for s=jω, when ω>>ω0 .

That's why, a specific parameter related to the estimator can be tuned via the PC GUI. It is the parameter n°18 "App. FE Time Constant" in "ms" which is the Filter time constant used in the approximate integration flux estimation method. Please find below more technical details about it.

Regarding the Filter time constant, the BEMF integral is approximated with a function composed by two low pass filters and a derivative. In discrete time domain, the expression for the low pass filter is:

$$y(n)=((k-1)/k)*y(n-1)+(1/k)*x(n)$$

Where (k*Tc) is the so called time constant of the filter and Tc is the sampling period

In the new software release for RX23T: the parameter to be specified is the time constant on the filter in ms, which is much more flexible and also more immediate to understand.

In fact the cut-off frequency of the filter is directly:

1/(2*pi*time_const[s]) = 1000/(6.28*parameter_value[ms]).

Internally k is calculated as 1000*parameter_value[ms]/Tc, and the filter is implemented performing a division, so there's no restrictions on the value and the sampling frequency is now variable.

For example, if the parameter_value = 32, so the time_constant[s]=0,032 and the cutoff_freq=4,97Hz, internal k=0,032*sampling_freq.

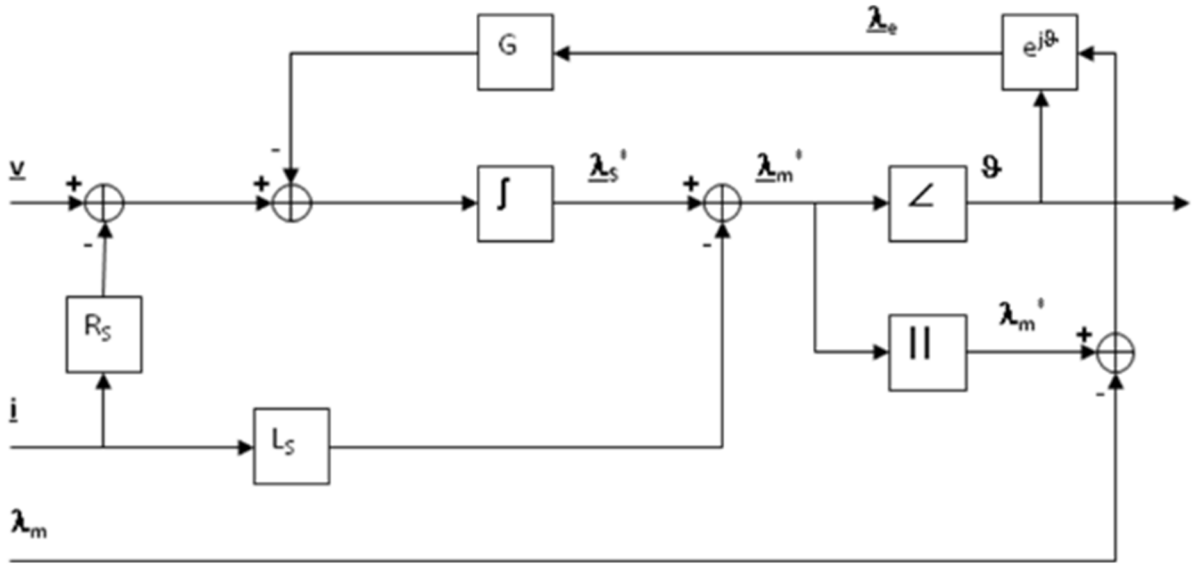
That's why, at 8KHz, please find below some calculation about the possible values to be used to tune the estimators:

Para. 18 in ms App. FE Time Constant	Cut-off Frequency
128	1.25Hz
64	2.5Hz
32	5Hz
16	10Hz

Obviously high values of the cut-off frequency produce poor behaviour at low speed and good behaviour at high speed, and vice-versa.

In the **2nd case**, to prevent the integral to diverge, and the errors related to wrong initial conditions are rejected, by the correcting action of the feedback.

The block scheme of the exact BEMF integration method for flux position estimation is the following:



The inputs of the system are the imposed voltage vector V and the measured current vector I. The motor phase resistance R_s, the synchronous inductance L_s and the permanent magnet flux amplitude λ_m are known as parameters and motor dependant.

The integral operation is corrected with a signal obtained modulating accordingly with the estimated phase the error between the estimated flux amplitude and the amplitude of the permanent magnets flux.

The gain of this correction is indicated with G. It is this feedback which avoids the integral divergence due to the errors or offsets. The higher G is, the higher is the relationship between the estimated amplitude and the theoretical one, but the larger can be the induced phase error.

The choice of G is a trade-off, in order to guarantee that the integral remains close to its theoretical value, but free enough to estimate the correct system phase.

In the **default embedded software** delivered on the YROTATE-IT-RX23T kit, this **second** strategy is selected. The choice to test the first one is left to the user thanks to the setting of the macros in the source code. Such modifications required a compilation of the embedded software. Within the software module “customize.h”, please uncomment the define APP_INT to enable the approximation method.

```
// Flux estimation method selection
//#define APP_INT     // uncomment this if you prefer approximated integration
```

22. EEPROM parameters: detailed description

Many Control parameters are included in a table which is saved in EEPROM and they can be modified by the user in runtime thanks to the GUI. The modified values are saved into the EEPROM, and will be maintained when the board is switched off.

Please find below the software parameters list including their descriptions. All the parameters are in FLOAT format to ensure a large resolution.

Parameter number	Short name	Description
0	SEL_OP	Default parameters setting, used only to perform special operations, like default parameter set re-loading, or to enable special (debug) working modes
1	RPM_MIN	Set the Minimum Speed in RPM
2	RPM_MAX	Set the Maximum Speed in RPM
3	R_ACC	Set the acceleration [RPM/s]
4	R_DEC	Set the deceleration [RPM/s]
5	C_POLI	Set the number of polar couples
6	I_START	Set the start-up current (peak) [Ampere]; used to specify the peak phase current value to be used during the start-up
7	I_MAX	Set the maximum phase current (peak) [Ampere]
8	R_STA	Set the stator resistance [Ohm]
9	L_SYN	Set the synchronous inductance [Henry]
10	PM_FLX	Set the permanent magnets flux linked to each stator phase winding [Weber]; this value is only used when the exact integration flux estimation algorithm is selected
11	KP_CUR	Set the Current loop Proportional coefficient: KP[Ohm]
12	KI_CUR	Set the Current loop Integral coefficient: KI[Ohm/sec]
13	KP_VEL	Set the Speed loop Proportional coefficient: KP[Amp*sec/rad]
14	KI_VEL	Set the Speed loop Integral coefficient: KI[Amp/rad]
15	FB_GAIN	Set the flux amplitude feedback gain; this value is only used when the exact integration flux estimation algorithm is selected
16	PHA_OFF	Set the phase offset [deg]; it is used to add a phase offset to the phase estimation, to reach better alignment
17	ST_TIM	Set the Start-up acceleration time [sec]
18	FL_FTAU	Set the flux estimation filter time constant TAU[sec]; this value is only used when the approximated integration flux estimation method is selected
19	SAM_FRE	Set the sampling frequency [Hz] of the control loop
20	F_RATIO	Set the ratio between the PWM frequency and sampling frequency, e.g. if 8000 is set in the parameter #19 and 2 in the parameter #20, the PWM frequency is 16KHz

23. Communication protocol between the MCU and the PC GUI

The communication between the demo board and the PC-GUI is based on a simple proprietary master-slave protocol, where the demo board is the slave. The master sends a request frame, where the requested operation is indicated, and the salve answers.
The maximum frame length is 255 bytes, so the number of data should be limited consequently. The opcodes refer to three data structures in microcontroller memory:

- ram tables (read table and write table); these tables are used to exchange the main measurement values and the references/commands to the board; all the data are 4bytes wide, but someone are long integers, some other are floating point values.
- EEPROM parameters (parameters vector, minimum values vector, default values vector, maximum values vector) ; all the data are floating point values except for the first parameter which is an unsigned long integer.
- measurement samples vector; it is only used to transfer the results of some tests (step answer) or for the oscilloscope window; all the data are signed short integers (2bytes).

The addresses refer to the data structures specified in the opcode.
The protocol is defined as follows:

Operation codes:

'**c**'=0x63	Check Request (master)
'**e**'=0x65	Check Answer (slave) (new programs with floating point parameters)
'**k**'=0x6B	Read Measurement Word (to read a word -2bytes- in measurement samples vector)
'**l**'=0x6C	Read Long (to read a longword -4bytes- in ram reading table)
'**L**'=0x4C	Write Long (to write a longword in ram writing table)
'**p**'=0x70	Read Long 1 (to read a longword in parameters vector)
'**P**'=0x50	Write Long 1 (to write a longword in parameters vector)
'**Y**'=0x59	Read Long 2 (to read a longword in parameters minimum values vector)
'**Z**'=0x5A	Read Long 3 (to read a longword in parameters default values vector)
'**J**'=0x4A	Read Long 4 (to read a longword in parameters maximum values vector)

Master frame:

l i s o a n D1 .. Dm k

l	frame total length, including this byte and the checksum (1 byte)
i	master string identification ('?'=0x3F)
s	station address (1 byte); only used in multi-slaves communication, in our case it is always 0
o	operation code (1 byte); used to decide which operation is required
a	data address (1 byte); it gives the data start location
n	data number (1 byte); it says how many data are involved in the operation
Dx	x-th data byte (1 byte)
k	checksum (1 byte)

Possible master frames:

'l'=5	**'?' 0 'c' k**	comm. check request

'l'=7	'?' 0 'k' a n k	word data read request
'l'=7	'?' 0 'l' a n k	long data read request
'l'=7+4n	'?' 0 'L' a n D13 D12 D11 D10 .. Dn3 .. Dn0 k	long data write request

Slave frame:

l i s o a n D1 .. Dm k

l	frame total length, including this byte and the checksum (1 byte)
i	slave string identification ('!'=0x21 for OK answer, '#'=0x23 for NOK answer)
s	station address (1 byte); only used in multi-slaves communication, in our case it is always 0
o	operation code (1 byte); to confirm the required operation
a	data address (1 byte); data start location
n	data number (1 byte)
Dx	x-th data byte (1 byte)
k	checksum (1 byte)

Possible slave frames:

'l'=5	'#' 0 o k	nok answer
'l'=5	'!' 0 'e' k	comm. check answer
'l'=7+2n	'!' 0 'k' a n D11 D10 .. Dn1 Dn0 k	word data read answer
'l'=7+4n	'!' 0 'l' a n D13 D12 D11 D10 .. Dn3 .. Dn0 k	long data read answer
'l'=7	'!' 0 'L' a n k	long data write answer

Examples:

PC request of reading 5 longword from the read ram table, starting from the third one (UIF_R.ram_tab[2], .., UIF_R.ram_tab[6]):

0	07	Number of bytes in the frame
1	3F	Master string indicator "?"
2	00	Station address (it is always 0)
3	6C	long reading operation "l"
4	02	data start address
5	05	number of data
6	k	checksum (it is calculated with a specific algorithm)

Board answer:

0	1B	Number of bytes in the frame (1Bh=27dec)
1	21	Slave string indicator "!"
2	00	Station address (it is always 0)
3	6C	long reading operation "l"
4	01	data start address
5	05	number of data
6	x	MSB of the MSW of first data (UIF_R.ram_tab[2]=UIF_R.var.fre)
7	x	LSB of the MSW of first data

8	x	MSB of the LSW of first data
9	x	LSB of the LSW of first data
6	x	MSB of the MSW of second data (UIF_R.ram_tab[3]=UIF_R.var.id)
7	x	LSB of the MSW of second data
8	x	MSB of the LSW of second data
9	x	LSB of the LSW of second data
6	x	MSB of the MSW of third data (UIF_R.ram_tab[4]=UIF_R.var.iq)
7	x	LSB of the MSW of third data
8	x	MSB of the LSW of third data
9	x	LSB of the LSW of third data
6	x	MSB of the MSW of fourth data (UIF_R.ram_tab[5]=UIF_R.var.vd)
7	x	LSB of the MSW of fourth data
8	x	MSB of the LSW of fourth data
9	x	LSB of the LSW of fourth data
6	x	MSB of the MSW of fifth data (UIF_R.ram_tab[6]=UIF_R.var.vq)
7	x	LSB of the MSW of fifth data
8	x	MSB of the LSW of fifth data
9	x	LSB of the LSW of fifth data
38	k	checksum

Checksum calculation

the checksum calculation is made with a simple function, referred to a constant table:

```
const uint8_t
	crc_tab[256] = // DOW CRC table
	{
		  0,  94, 188, 226,  97,  63, 221, 131, 194, 156, 126,  32, 163, 253,  31,  65,
		157, 195,  33, 127, 252, 162,  64,  30,  95,   1, 227, 189,  62,  96, 130, 220,
		 35, 125, 159, 193,  66,  28, 254, 160, 225, 191,  93,   3, 128, 222,  60,  98,
		190, 224,   2,  92, 223, 129,  99,  61, 124,  34, 192, 158,  29,  67, 161, 255,
		 70,  24, 250, 164,  39, 121, 155, 197, 132, 218,  56, 102, 229, 187,  89,   7,
		219, 133, 103,  57, 186, 228,   6,  88,  25,  71, 165, 251, 120,  38, 196, 154,
		101,  59, 217, 135,   4,  90, 184, 230, 167, 249,  27,  69, 198, 152, 122,  36,
		248, 166,  68,  26, 153, 199,  37, 123,  58, 100, 134, 216,  91,   5, 231, 185,
		140, 210,  48, 110, 237, 179,  81,  15,  78,  16, 242, 172,  47, 113, 147, 205,
		 17,  79, 173, 243, 112,  46, 204, 146, 211, 141, 111,  49, 178, 236,  14,  80,
		175, 241,  19,  77, 206, 144, 114,  44, 109,  51, 209, 143,  12,  82, 176, 238,
		 50, 108, 142, 208,  83,  13, 239, 177, 240, 174,  76,  18, 145, 207,  45, 115,
		202, 148, 118,  40, 171, 245,  23,  73,   8,  86, 180, 234, 105,  55, 213, 139,
		 87,   9, 235, 181,  54, 104, 138, 212, 149, 203,  41, 119, 244, 170,  72,  22,
		233, 183,  85,  11, 136, 214,  52, 106,  43, 117, 151, 201,  74,  20, 246, 168,
		116,  42, 200, 150,  21,  75, 169, 247, 182, 232,  10,  84, 215, 137, 107,  53
	};

/*****************************************************************************
Function:	cal_crc
Description:	DOW CRC calculation
Input:		the number of bytes to be taken into account, the vector
		containing these bytes
Output:		CRC
Note:		it uses the DOW CRC calculation table defined before
*****************************************************************************/
static uint8_t cal_crc(uint8_t u8n, uint8_t *u8v)
{
	uint8_t u8i, u8cks = 0;

	for(u8i = 0; u8i < u8n; u8i++)
	{
		u8cks = crc_tab[u8cks ^ u8v[u8i]];
	}
	return(u8cks);
}
```

Ram tables definition

Pls. find below the definition of the ram tables (they can be found in "gui_defs.h").

```
typedef union
{
	uint32_t		ram_tab[N_RAM_READ];
	struct
	{
		float32_t	rif;	//  0	internal speed reference [rpm]
		float32_t	rpm;	//  1	measured speed [rpm]
		float32_t	fre;	//  2	imposed frequency [Hz]
		float32_t	id;	//  3	direct current [A]
		float32_t	iq;	//  4	quadrature current [A]
		float32_t	vd;	//  5	direct voltage [V]
		float32_t	vq;	//  6	quadrature voltage [V]
		float32_t	vb;	//  7	bus voltage [V]
		uint32_t	all;	//  8	alarm
		uint32_t	flg;	//  9	flags
		float32_t	toti;	// 10	current vector amplitude [V]
		float32_t	totv;	// 11	voltage vector amplitude [V]
		float32_t	du0;	// 12
		float32_t	du1;	// 13
		float32_t	du2;	// 14
		float32_t	du3;	// 15
		uint32_t	mod;	// 16	mode
		float32_t	rs;	// 17	stator resistance [Ohm]
		float32_t	ls;	// 18	synchronous inductance [Hen]
		float32_t	fl;	// 19	permanent magnet flux [Web]
		float32_t	kpi;	// 20	current loop kp [Ohm]
		float32_t	kii;	// 21	current loop ki (referred to cont. time) [Ohm/sec]
		float32_t	pwmf;	// 22	pwm frequency [Hz]
		float32_t	sf;	// 23	sampling frequency [Hz]
		uint32_t	ena;	// 24	extra features enable flags
		float32_t	du4;	// 25
		float32_t	du5;	// 26
		float32_t	du6;	// 27
		float32_t	du7;	// 28
		float32_t	du8;	// 29
		float32_t	du9;	// 30
		float32_t	du10;	// 31
	}	var;
}	UIF_R_t;

typedef union
{
	uint32_t		ram_tab[N_RAM_WRITE];
	struct
	{
		uint32_t	trg;	//  0	trigger
		uint32_t	mod;	//  1	mode
		float32_t	rif;	//  2	speed reference [rpm]
		float32_t	cra;	//  3	current ratio [%]
		uint32_t	sel;	//  4	variable and time scale selection
		float32_t	du0;	//  5
		float32_t	du1;	//  6
		float32_t	du2;	//  7
	}	var;
}	UIF_W_t;
```

24. Inverter Kit – Bill of Material

Please find below the complete Bill of Material of the inverter kit YROTATE-IT-S5D9

Design File:	7EBRS5D9_2_Rev-0_PM161117
Design Title:	YROTATE-IT-S5D9
Selected Variant:	Based on RENESAS SYNERGY S5D9

BILL OF MATERIAL

Quantity	*Part Name*	*Description*	*Component Names*
6	0R	RESIST.SMD 0603 "0R 1%"	R37, R59, R81, R136, R142, R159
3	0R1 1W 2512	RESIST.SMD 2512 "0R1" 1% 200V 100PPM/°C	R29, R30, R31
8	1µF. 50V	COND. SMD 1µF 50V Y5V 1206 CL31F105ZBFNNNE	C12, C16, C28, C30, C55, C60, C66, C78
18	10µF. 25V	COND. SMD 10µF 25V X7R (Package 1206)	C21, C22, C23, C26, C31, C32, C33, C56, C57, C62, C64, C104, C108, C111, C123, C125, C128, C132
1	100µF 63V.	COND. SMD ELETTR. 100µF 63V 105°C 10x10.5 UUX1J101MNL1GS Nichicon	C2
1	100K	RESIST.SMD 0603 "100K 1%"	R4
51	100nF	COND. SMD 50V 0603 X7R PHY223858619812 "100nF"	C4, C5, C7, C9, C11, C20, C25, C34, C35, C47, C49, C50, C51, C53, C58, C59, C61, C63, C65, C67, C68, C72, C73, C74, C76, C77, C80, C82, C86, C88, C92, C93, C95, C98, C100, C101, C105, C106, C112, C118, C121, C122, C124, C130, C131, C134, C135, C136, C138, C139, C142
1	100NF 200V 1210	C.SMD 100nF 200V 1210 X7R	C3
7	100pF	COND. SMD 50V 0603 X7R WLS0603N101J500LT "100pF"	C15, C48, C89, C94, C119, C120, C129
14	100R	RESIST.SMD 0603 "100R 1%"	R43, R45, R47, R49, R51, R53, R97, R98, R101, R102, R106, R113, R122, R124
1	100R 1/4W	RESIST.SMD 1206 "100R 1/4W 1206"	R120
1	105K	R.SMD 105K 0603 1%	R131
3	10BQ060	Schottky, 60 V, 1 A, Singolo, DO-214AA, 2 Pin, 490 mV	D1, D19, D26

35	10K	RESIST.SMD 0603 "10K 1%"	C96, C103, R6, R7, R8, R15, R16, R19, R22, R28, R35, R42, R44, R46, R48, R50, R52, R56, R58, R67, R77, R95, R100, R112, R121, R127, R128, R129, R139, R140, R149, R151, R152, R153, R165
3	10nF	COND. SMD 50V 0603 X7R WLS0603B103K500CT "10nF"	C24, C107, C140
5	10R	RESIST.SMD 0603 "10R 1%"	R32, R33, R34, R162, R163
20	1K	RESIST.SMD 0603 "1K 1%"	R63, R68, R89, R94, R99, R103, R104, R111, R130, R132, R133, R134, R138, R154, R170, R171, R172, R173, R174, R175
3	1K2	RESIST.SMD 0603 "1K2 1%"	R110, R116, R117
2	1K8	RESIST.SMD 0603 "1K8 1%"	R26, R27
3	1M	RESIST.SMD 0603 "1M 1%"	R57, R119, R125
25	1NF	COND. SMD 50V 0603 X7R PHY223858615623 "1nF"	C14, C36, C37, C38, C39, C40, C41, C42, C43, C44, C52, C69, C79, C81, C83, C84, C90, C114, C117, C126, C127, C143, C144, C145, C146
1	2µ2F 100V.	C.SMD 2.2µF 100V 1210 X7R	C1
3	2µ2F. 25V	COND. SMD 2µ2F 25V X7R (Package 1206)	C85, C87, C115
2	220µF 16V.	COND. SMD ELETTR. 220µF 16V 6.3x8 NIC NACE	C70, C71
3	220R	RESIST.SMD 0603 "220R 1%"	R108, R114, R115
5	22K	RESIST.SMD 0603 "22K 1%"	R5, R9, R144, R145, R168
1	22nF	C.SMD 22nF 50V 0603 X7R	C113
3	22pF	COND. SMD 50V 0603 X7R WLS0603N220J500LT "22pF"	C45, C46, C133
9	22R	RESIST.SMD 0603 "22R 1%"	R69, R71, R73, R74, R75, R76, R78, R79, R80
11	2K2	RESIST.SMD 0603 "2K2 1%"	R66, R92, R141, R146, R147, R148, R156, R157, R158, R160, R161
1	330K	RESIST.SMD 0603 "330K 1%"	R14
1	3K3	RESIST.SMD 0603 "3K3 1%"	R88
3	47µF 35V.	COND. SMD ELETTR. 47µF 35V 6.3x6.3	C27, C29, C54
3	47µF 50V.	COND. SMD ELETTR. 47µF 50V UUX-CR 6.3x8	C6, C8, C10

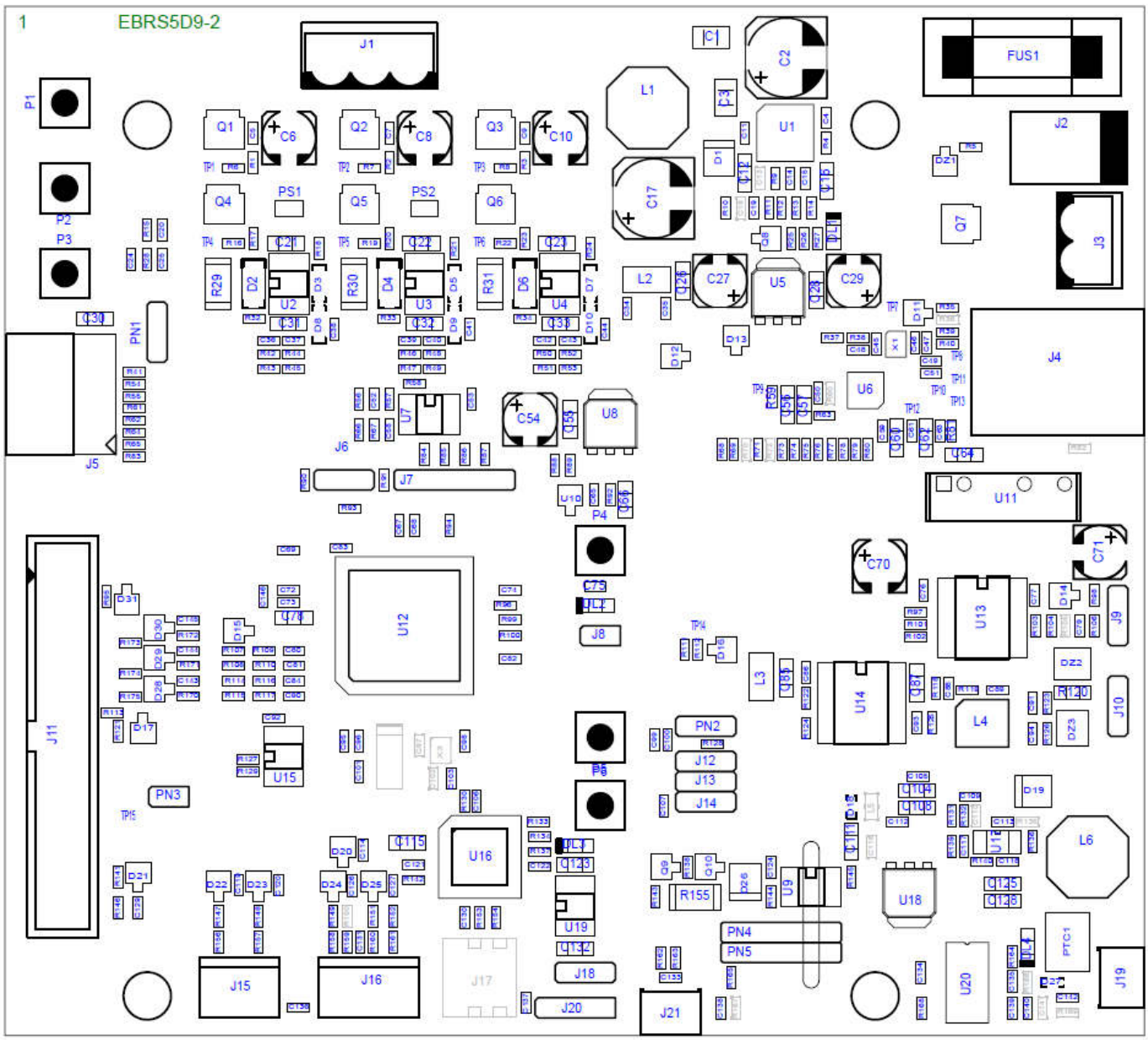
1	470µF 25V.	COND. SMD ELETTR. 470µF 25V 10x10.5 NIC NACE	C17
3	470NF 16V	COND. SMD 16V 0603 X7R WLS0603F474M160CT "470nF"	C75, C99, C137
1	470PF	COND. SMD 25V 0603 COG VJ0603D471KXXAJ "470pF"	C19
17	47K	RESIST.SMD 0603 "47K 1%"	R25, R41, R54, R55, R61, R62, R64, R65, R83, R84, R85, R86, R87, R90, R91, R93, R118
6	47R	RESIST.SMD 0603 "47R 1%"	R1, R2, R3, R17, R20, R23
2	4K7	RESIST.SMD 0603 "4K7 1%"	R39, R143
2	4n7F	COND. SMD 50V 0603 X7R "4n7F"	C91, C109
1	56K	RESIST.SMD 0603 "56K 1%"	R10
6	5K6	RESIST.SMD 0603 "5K6 1%"	R11, R12, R13, R18, R21, R24
3	680R	RESIST.SMD 0603 "680R 1%"	R40, R96, R137
2	68R	R.SMD 68R 0603 1%	R123, R126
1	68R 1W 2512	RESIST.SMD 2512 "68R" 1% 200V 200PPM/°C	R155
2	6K8	RESIST.SMD 0603 "6K8 1%"	R38, R107
1	820R	RESIST.SMD 0603 "820R 1%"	R164
1	82K	RESIST.SMD 0603 "82K 1%"	R109
1	ABM8G-25.000MHZ-18-D2Y-T	ABM8G-25.000MHZ-18-D2Y-T - Cristallo 25MHz SMD 3,2mm x 2,5mm 30ppm 18pF 20ppm Serie ABM8G	X1
1	AT24C02C-SSHM	C.I.SMD AT24C02C-SSHM E2PROM SO8	U15
6	BAT46W	D.BAT46W 150mA 100V SCHOTTKY SOD-123	D3, D5, D7, D8, D9, D10
15	BAT54S	D.SMD BAT54S 200mA 30V SCHOTTKY SOT-23	D11, D12, D13, D16, D17, D20, D21, D22, D23, D24, D25, D28, D29, D30, D31
1	BAV99	D. SMD SOT-23	D15
1	BAW56	D. SMD SOT-23	D14
1	BC807	TRANS. SMD BC807 SOT-23 PHABC807	Q10
2	BC817	TRANS. SMD BC817 SOT-23 PHABC817	Q8, Q9
1	BERG 2p	CONN. BERGHER 40P.M.VERT.WS1X40/90	PN3
6	BERG 3P	CONN. BERGHER 40P.M.VERT.WS1X40/90	J12, J13, J14, J18, PN1, PN2
2	BERG 6P	CONN. BERGHER 40P.M.VERT.WS1X40/90	PN4, PN5
1	BERGER 2M	STRIP 40P.M.VERT.WS1X40	J8
3	BERGER 3M	STRIP 40P.M.VERT.WS1X40	J6, J9, J10
1	BERGER 4M	STRIP 40P.M.VERT.WS1X40	J20
1	BERGER 6M	STRIP 40P.M.VERT.WS1X40	J7
1	BZX84C15	D. ZENER 15V 350mW SOT-23	DZ1
3	ES1D	D. SMD SMA_DO-214AC ES1D 1A 200V Trr 35ns	D2, D4, D6
2	Fiducial	Fiducial per montaggio SMT e coll. SPEA	AFID1, AFID2
1	FLAT 34M DIN41651	CONN. DIN41651 34vie M (Distr.143-65-398)	J11

1	FT232R	FT232R USB UART IC	U20
2	IND. 10µH 0.25A	IND.10µH 0.25A CM453232-100 BOUR 1.6ohm	L2, L3
1	IND. 744066100	10µH 30% 3.6A Idc Wurth WE-TPC Series	L6
1	IND. 744066330	33µH 30% 2.1A Idc Wurth WE-TPC Series	L1
1	ISL32704EIBZ	ISL32704EIBZ Isolated RS-485 Transceiver	U13
3	FAN7382	High- and Low-Side Gate Driver IC	U2, U3, U4
1	ISL8560	DC/DC Power Switching Regulator	U1
1	ISL97519A	ISL97519A PWM Step-Up Regulator	U17
1	ISO7221	ISO7221 Dual Channel Digital Isolators	U9
1	JACK DC10A	CONN. JACK 5A 2,1mm DC-10A 5A 12VDC (F-224959)	J2
1	KF33BD-TR	C.I. KF33BD-TR 3.3V 0.5A SO8	U19
1	KSZ8081RNA	KSZ8081RNA 10BASE-T/100BASE-TX PHY RMII	U6
3	L.R.1206	LED SMD ROSSO 1206	DL2, DL3, DL4
1	L.V.1206	LED.SMD VE 1206 HSMG-C150 15mcd GREEN	DL1
2	L78M05ABDT-TR	L78M05ABDT-TR IC,REG.TENSIONE +5.0V,0.5A,-40°C,DPAK,7805	U5, U18
1	LM317MDTX	C.I. FSCLM317MDTX 0,5A SMD D-PAK	U8
1	LM393M	C.I. SMD LM393M	U7
2	MCL4148	D. SMD MICRO-MELF MCL4148	D18, D27
1	P_FUS_5x20	PORTAF. C.S. 5x20 OMEGA C1024	FUS1
1	PANDUIT 4M.	CONN. PAND.M. 4V PAHLSS100-4	J15
1	PANDUIT 5M.	CONN. PAND.MAS. PAHLSS100-5	J16
2	PS	PUNTO DI SALDATURA	PS1, PS2
1	PTC_MINISMDC075F-24	PTC LITTELFUSE MINISMDC075F/24-2 (1812)	PTC1
6	PULS_KSEM31G LFS	KSEM31G LFS - Interruttore Tattile, Non Illuminato, 32 V, 50 mA, 1.5 N, SMD	P1, P2, P3, P4, P5, P6
1	R7FS1247x3A01CFL	R7FS1247x3A01CFL 32-bit ARM® Cortex®-M0+ microcontroller	U16
1	R7FS5D97E3A01CFP-AA0	R7FS5D97E3A01CFP-AA0 Synergy Microcontrollers S5 Series	U12
1	RJ45 7499010211A	Lan Ethernet Wurth Elektronik, perdita inserzione - 1.4dB, 1 porte, 13.5 x 15.88 x 21.84mm	J4
7	RJK0654DPB	MOS. RJK0654DPB LFPAK 60V 30A Rds=6.5m	Q1, Q2, Q3, Q4, Q5, Q6, Q7
1	SAMTEC 6-6F 90°	SSW-106-02-G-D-RA 2.54mm 12cont 90° Presa Serie SSW	J5
1	SAURO 2M/P 5.0.	C.SAURO CIM/OP-02 M.POL.02 p.5.0	J3
1	SAURO 3M/P 5.0.	C.SAURO CIM/OP-03 M.POL.03 p.5.0	J1
2	SM6T30CA	TRANSIL SM6T30CA Bidirect. SMD bobina	DZ2, DZ3
3	STARPOINT 2PIN	Start Point da 2 connessioni	ASTP1, ASTP2, ASTP3
15	TEST	TESTPOIN	TP1, TP2, TP3, TP4, TP5, TP6, TP7, TP8, TP9, TP10, TP11, TP12, TP13, TP14, TP15
1	TJA1052I	Galvanically isolated high-speed CAN transceiver	U14
1	TL431CDBZR	PROGRAMMABLE VOLTAGE REFERENCE TL431CDBZR SOT-23-3	U10

1	TMV 0505S	C.I. DC/DC ISO TMV0505S TRACO-POWER	U11
2	USB MICRO B 90°	MICRO USB JST UB-MC5BR3-SD204-4S-1-TB NMP tipo B femmina 90° SMT	J19, J21
1	ZJYS81R5-2PL51T-G01	IND.COMMON ZJYS81R5-2PL51T-G01 TDK	L4

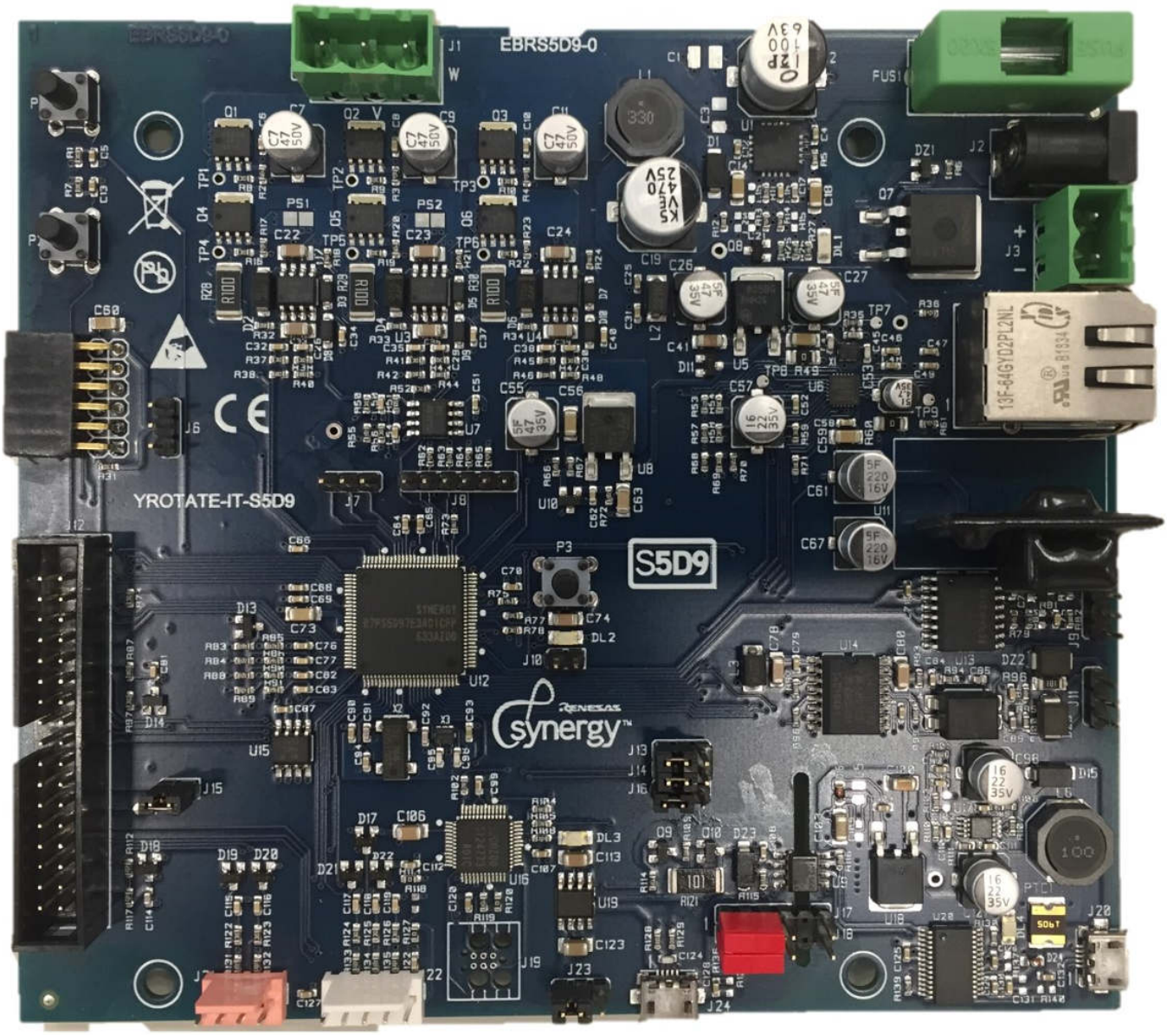
25. Inverter Kit – PCB topography

Please find below the placement of each components on the PCB of the kit YROTATE-IT-S5D9



26. Inverter Kit – Board picture in high resolution

Please find below the kit YROTATE-IT-S5D9, top view.



Revision History

		Description	
Rev.	**Date**	**Page**	**Summary**
0.80	November 20, 2017		First Edition
1.00	March 23, 2018		Chapter 10 to 26 are reworked

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins
   Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.
   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.
2. Processing at Power-on
   The state of the product is undefined at the moment when power is supplied.
   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.
3. Prohibition of Access to Reserved Addresses
   Access to reserved addresses is prohibited.
   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.
4. Clock Signals
   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.
   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.
5. Differences between Products
   Before changing from one product to another, i.e. to a product with a different type number, confirm that the change will not lead to problems.
   — The characteristics of an MPU or MCU in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.