STM32 Nucleo Pack FOC and 6-step motor control platform
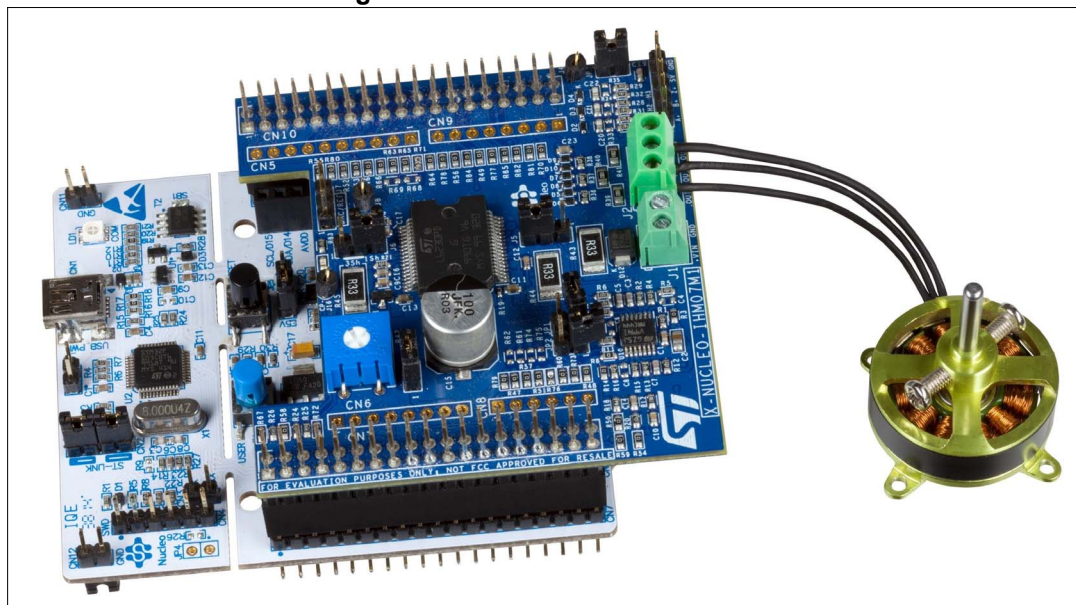for 3-phase low voltage motor

Introduction

The NUCLEO Pack (P-NUCLEO-IHM001) is a motor control kit based on X-NUCLEO-IHM07M1 and NUCLEO-F302R8 boards. The power board with ST L6230 DMOS driver, belonging to STPIN family, provides a motor control solution for low voltage 3-phase DC brushless motor with the addition of STM32 NUCLEO board through the ST morpho connector.

The driver used on power board is the L6230, a DMOS fully integrated driver for 3-phase brushless PMSM motor, assembled in PowerSO36 package, with overcurrent and thermal protection. The NUCLEO-F302R8 provides an affordable and flexible way for users to try out new ideas and build prototypes with STM32 MCU. It does not require any separate probe as it integrates the ST-LINK/V2-1 debugger and programmer.

This document describes the procedure to configure this P-NUCLEO-IHM001 kit to run the low voltage motor included in the package. The evaluation board is fully configurable and ready, to support different closed loop controls based on sensorless or sensored mode, and it is compatible with 3-shunt or 1-shunt current sense measuring. The *Figure 1* shows the motor control kit assembled.

Figure 1. P-NUCLEO-IHM001 Pack

1. Picture not contractual

Contents

List of tables

List of figures

1 Features

The information listed below shows the main board specification data and parameters set for the X-NUCLEO-IHM07M1 expansion board included in the P-NUCLEO-IHM001 kit:

- 3-phase driver board for BLDC/PMSM Motors
- Nominal operating voltage range from 8 V to 48 V DC
- 2.8 A output peak current (1.4 A RMS)[a]
- Operating frequency up to 100 kHz
- Non dissipative overcurrent detection and protection
- Cross conduction protection
- Thermal measuring and overheating protection
- Full compatible with ST 6-Step or ST FOC control algorithm
- Full support for sensorless and sensor mode
- 3-shunt and 1-shunt configurable jumpers for motor current sensing
- Hall / encoder motor sensor connector and circuit
- Debug connector for DAC, GPIOs, etc.
- Potentiometer available for speed regulation
- Fully populated board conception with test points
- User LED
- Compatible with STM32 Nucleo boards
- Equipped with ST morpho connector
- PCB type and size:
  - Material of PCB - FR-4
  - 4 layers layout
  - Copper thickness: 70 μm (external layer), 35 μm (internal layer)
  - Total dimensions of the expansion board: 70 mm x 66 mm

---

a. Device characteristic

2 Ordering information

To order the motor control kit based on X-NUCLEO-IHM07M1 and NUCLEO-F302R8, use the order code: P-NUCLEO-IHM001.

3 Basic user - Getting started

3.1 System architecture

A generic motor control system can be basically schematized as the arrangement of three main blocks (see *Figure 2*):

- **Control block**: its main task is to accept user commands and configuration parameters to drive a motor. The P-NUCLEO-IHM001 is based on NUCLEO-F302R8 board that provides all digital signals to perform the proper motor driving control algorithm (for instance 6-step or FOC).

- **Power block**: the X-NUCLEO-IHM07M1 is based on 3-phase inverter topology. The core of the power block embedded on board is the driver STSPIN L6230, which contains all the necessary active power and analog components to perform a low voltage PMSM motor control.

- **PMSM Motor:** low voltage 3-phase brushless motor.

Figure 2. System architecture
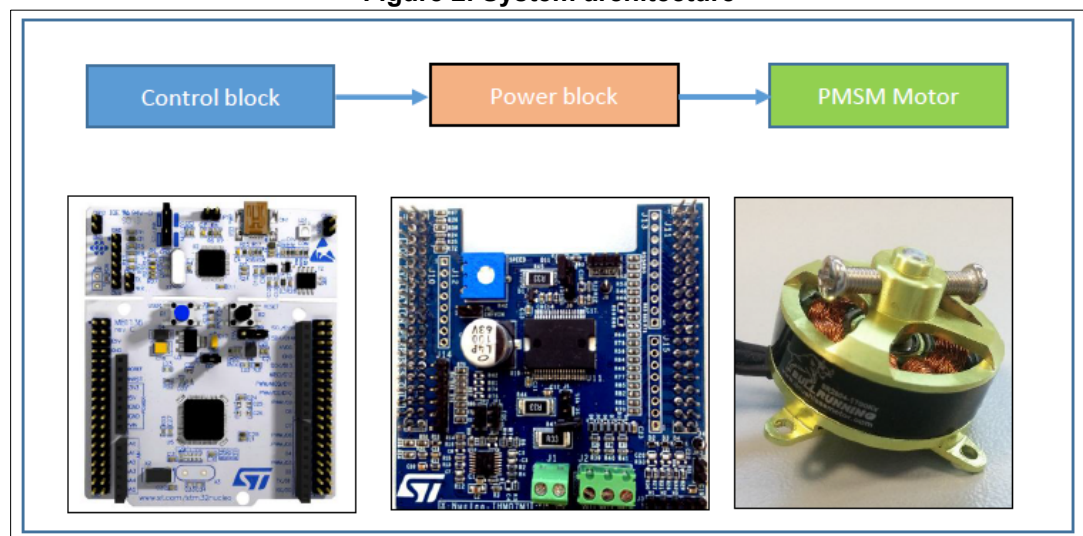
3.2 Building and run the motor control Nucleo Pack

The P-NUCLEO-IHM001 is a complete hardware development platform (**Power** & **Control** block **+ Motor**) for STM32 Nucleo ecosystem to evaluate a motor control solution for single motor.

For a regular board operating, follow the hardware configuration shown below:

1. The X-NUCLEO-IHM07M1 must be stacked on a NUCLEO-F302R8 board through the ST morpho connector. There is only one position allowed for this connection, in particular the two buttons on NUCLEO-F302R8 board (blue B1 and black B2) must be kept out, as shown in *Figure 3*.

Figure 3. X-NUCLEO-IHM07M1 and NUCLEO-F302R8 assembled
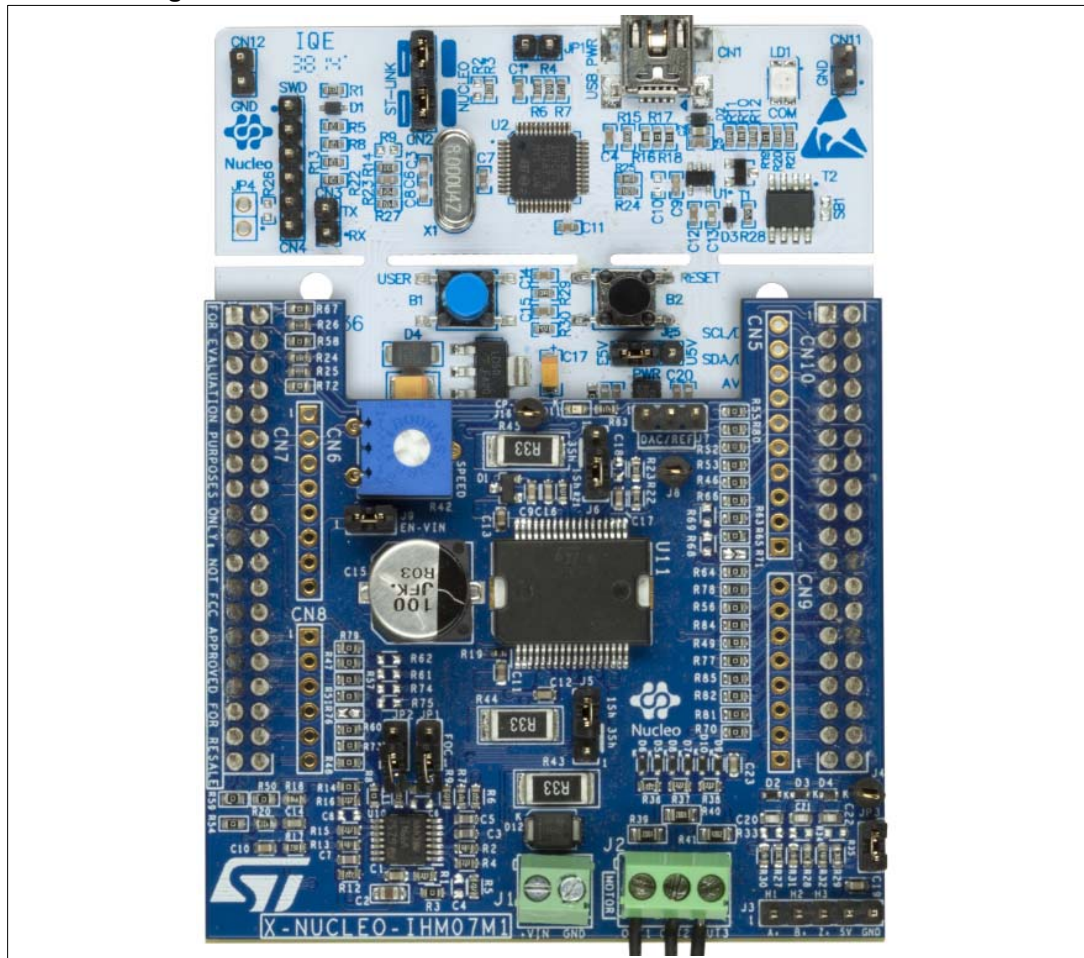
1. The interconnection between the X-NUCLEO-IHM07M1 with STM32 NUCLEO board has been designed for a full-compatibility with a lot of control board and no modification of solder bridges is required.

2. Connect the three motor wires U,V,W at J2 connector as shown in the *Figure 4: Motor connection with X-NUCLEO-IHM07M1*: it is mandatory to connect the white or yellow wire to OUT1, the black one to OUT2 and the red one to OUT3, to respect clockwise and counterclockwise motor rotation, according to the firmware implementation.

Figure 4. Motor connection with X-NUCLEO-IHM07M1
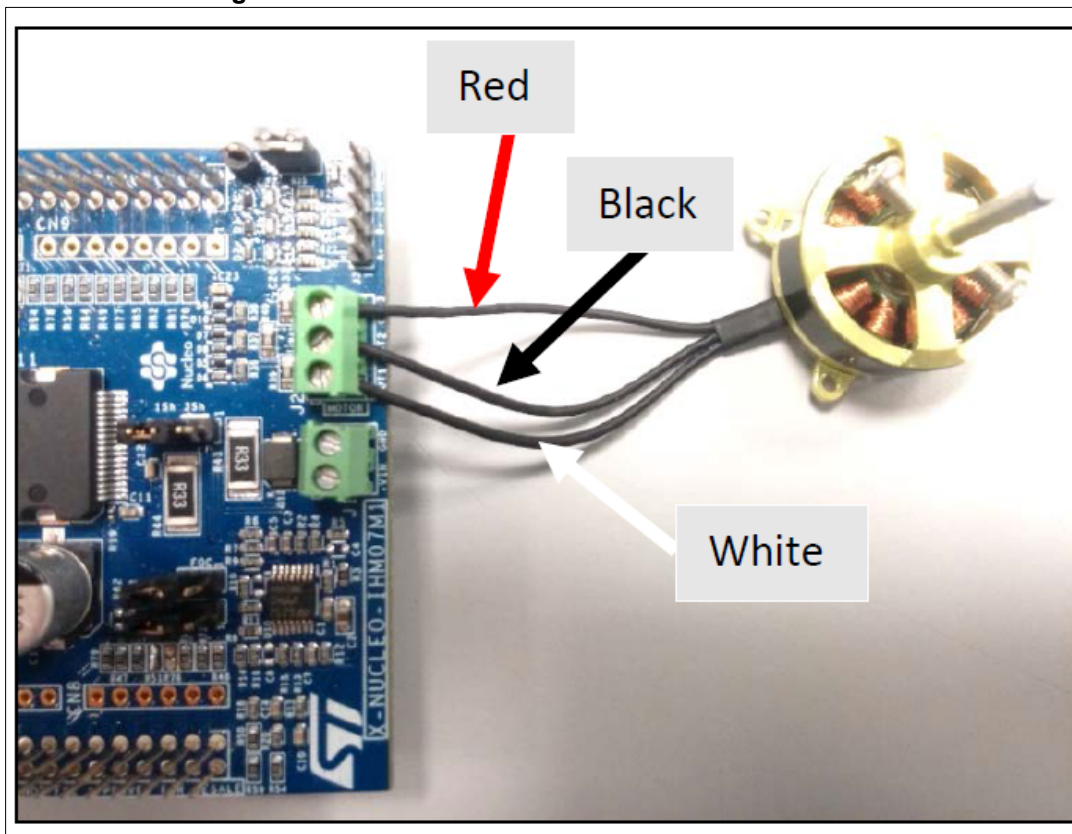
1. The three motor wires are identified by a colored line, marked on them. The line marked on OUT1 (refer to the white arrow in the above figure) can be white or yellow.

3. Select the jumper configuration on the power board to choose the desired control algorithm (6-step or FOC) as described below:

   a) On NUCLEO-F302R8 board, check jumper setting: JP1 open, JP5 (PWR) on E5V side, JP6 (IDD) closed.

   b) On X-NUCLEO-IHM07M1 expansion board:

   – Check jumper settings: J9[b] closed, JP3 closed

   – For 6-step control set jumper settings as: JP1 and JP2 open, J5&J6 on 1Sh side[b]

   – For FOC control set jumper settings as: JP1 and JP2 closed, J5&J6 on 3Sh side

---

b. It is important that supply voltage is powered off before control mode changing

4. Connect DC supply voltage on J1[(c)] connector and power-on (up to 12 V DC for BR2804 motor included in the Pack, as shown in *Figure 5: Power supply connection for X-NUCLEO-IHM07M1*).

Figure 5. Power supply connection for X-NUCLEO-IHM07M1
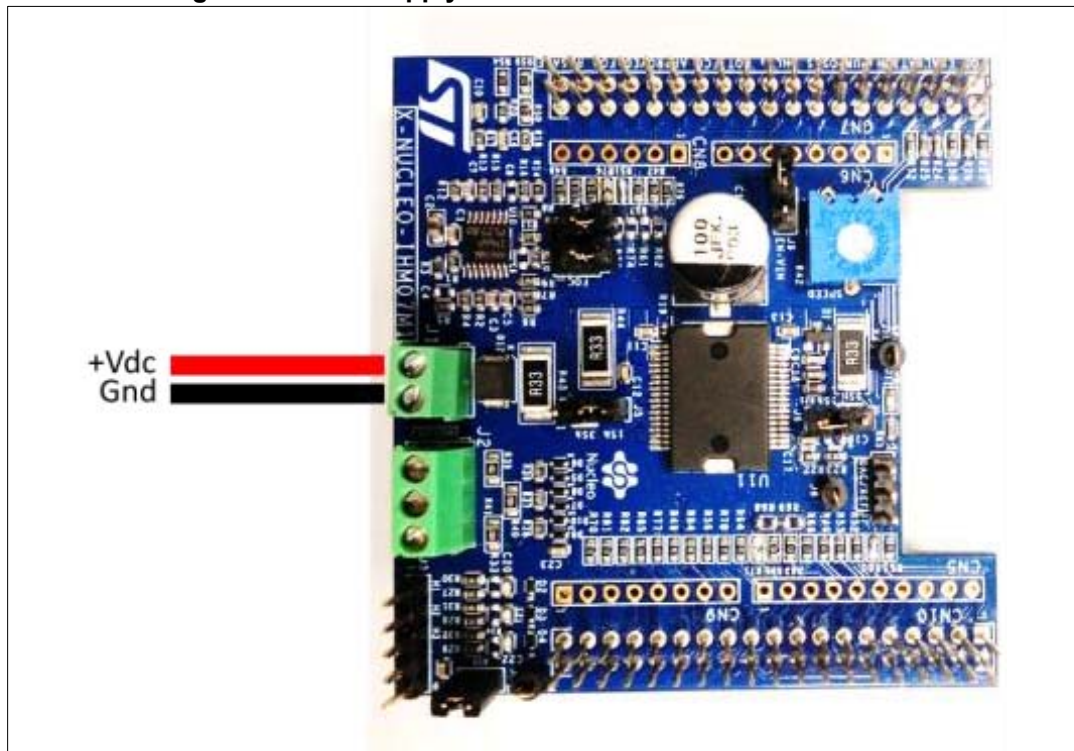
5. At power-on (or reset) led D11 on X-NUCLEO-IHM07M1 board starts to blink according to the control algorithm choice:
   – 2 times for FOC mode control
   – 4 times for 6-step mode control

After the confirmation of control algorithm selected, the system is ready to start.

6. Push the blue button on NUCLEO-F302R8 (B1) and the motor starts to spin.

7. Rotate the potentiometer on X-NUCLEO-IHM07M1 board in order to regulate the motor speed.

3.3 Hardware settings

The *Table 1* shows the jumper configuration on X-NUCLEO-IHM07M1 board (see also *Figure 6: X-NUCLEO-IHM07M1 – top layer with silk-screen* and *Figure 7: X-NUCLEO-IHM07M1 connectors view*). According to the jumpers selection, it is possible to choose the

---

c. For a different motor (>12V) it is mandatory to remove the jumper J9 on power board, before the power-on, to avoid to damage the NUCLEO-F302R8 board. To supply the STM32-NUCLEO from USB, the jumper JP5 must be connected between PIN 1 and PIN2. For further details on Nucleo settings refer to UM1724 available from ST web site *www.st.com*.

1-shunt or 3-shunt mode, the current sensing circuit offset level, the hall/encoder with pull-up or the external supply for NUCLEO-F302R8 board.

Table 1. Jumper settings

Jumper	Permitted configurations	Default condition
JP1[1]	Selection for FOC current sensing circuit. The default condition is for 6-Step control algorithm	OPEN
JP2[1]	Selection for FOC current sensing circuit. The default condition is for 6-Step control algorithm	OPEN
JP3	Selection for pull-up enabling in hall/encoder detection circuit	CLOSED
J9	Selection to supply the Nucleo board through the X-NUCLEO-IHM07M1[2]*	CLOSED
J5[3]	Selection for 1-shunt or 3-shunt configuration (single shunt by default)	2-3 CLOSED
J6[3]	Selection for 1-shunt or 3-shunt configuration (single shunt by default)	2-3 CLOSED
J7	Debug connector for DAC	OPEN

1. JP1 and JP2 selection between FOC or 6-step current sensing circuit. It must be both closed (FOC selection) or both open (6-step selection - default setting)

2. It is recommended to verify that power supply voltage is not higher than 12V dc, in order to avoid damaging on NUCLEO-F302R8 board. For further details on Nucleo settings refer to UM1724 available from ST web site *www.st.com*.

3. J5 and J6 must have both the same configuration. Both 1-2 for three shunt configuration. Both 2-3 for single shunt configuration. On the silkscreen the correct position for three and for single shunt is indicated. Also the default position is indicated.

The *Table 2* shows the main connector on X-NUCLEO-IHM07M1 board.

Table 2. Screw terminal table

Screw terminal	Function
J1	Motor power supply input (8V÷48V)
J2	3-phase motor connector (U,V,W)

The X-NUCLEO-IHM07M1 is based on ST morpho connector, male pin headers (CN7 and CN10) accessible on both sides of the board. They can be used to connect this power board to NUCLEO-F302R8 board. All signals and power pins for MCU are available on ST morpho connector. For further details refer to UM1724 document (section 5.12 ST morpho connector) available on website *www.st.com*.

Figure 6. X-NUCLEO-IHM07M1 – top layer with silk-screen
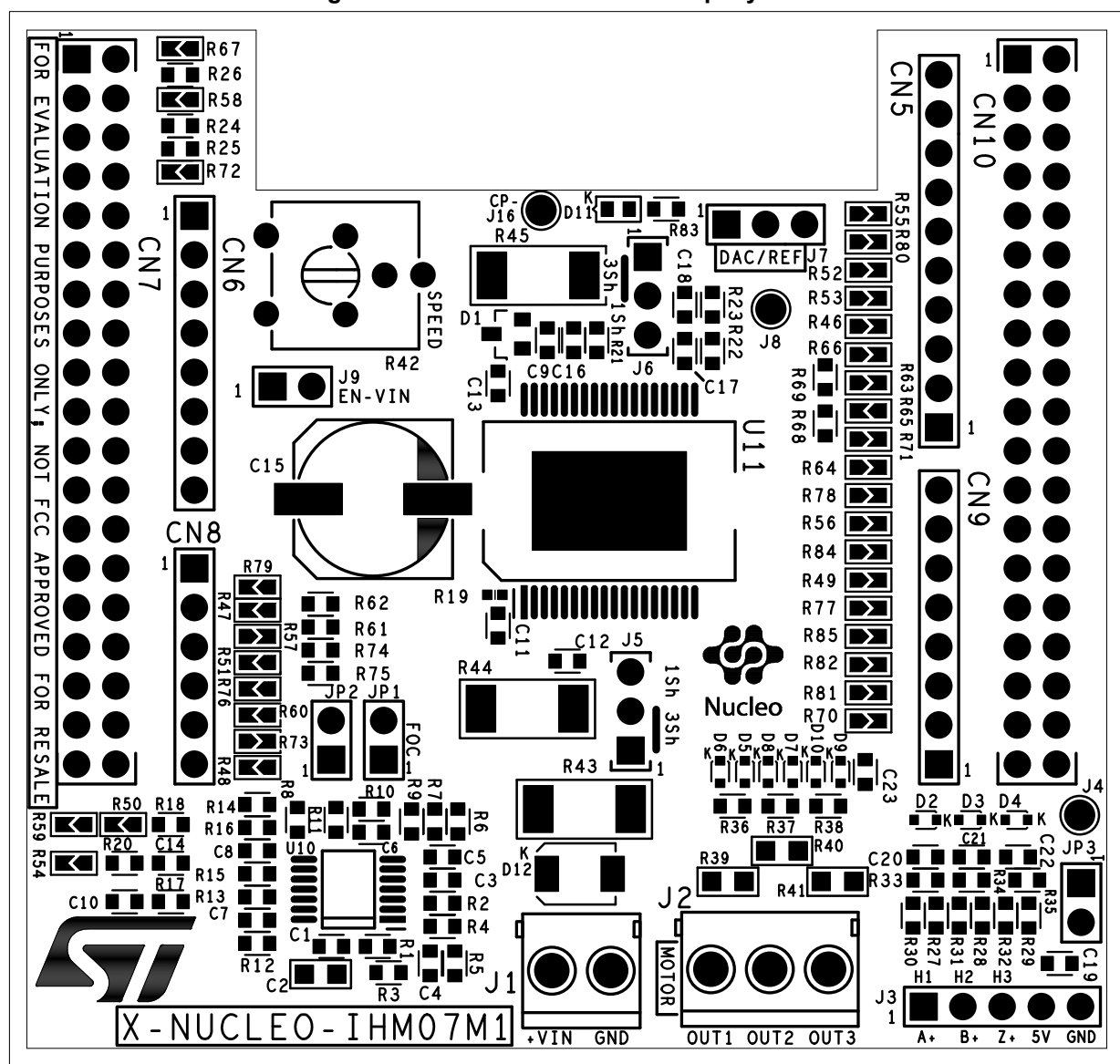
Table 3. Connectors description

Part reference	Description
CN7	ST morpho connector
CN6	Arduino UNO connector
CN8	Arduino UNO connector
U11	L6230 driver
U10	TSV994IPT op. amp.
J1	Power supply connector

Table 3. Connectors description (continued)

Part reference	Description
J9	Enable VIN supply voltage
JP1, JP2	Jumpers for FOC
SPEED	Potentiometer
CN10	ST morpho connector
CN5	Arduino UNO connector
CN9	Arduino UNO connector
J2	Motor connector
J3	Hall/Encoder sensor connector
J7	Debug connector
JP3	External pull-up for sensors
J5, J6	Current measure mode (1Sh/3Sh)
D11	LED status indicator

Figure 7. X-NUCLEO-IHM07M1 connectors view
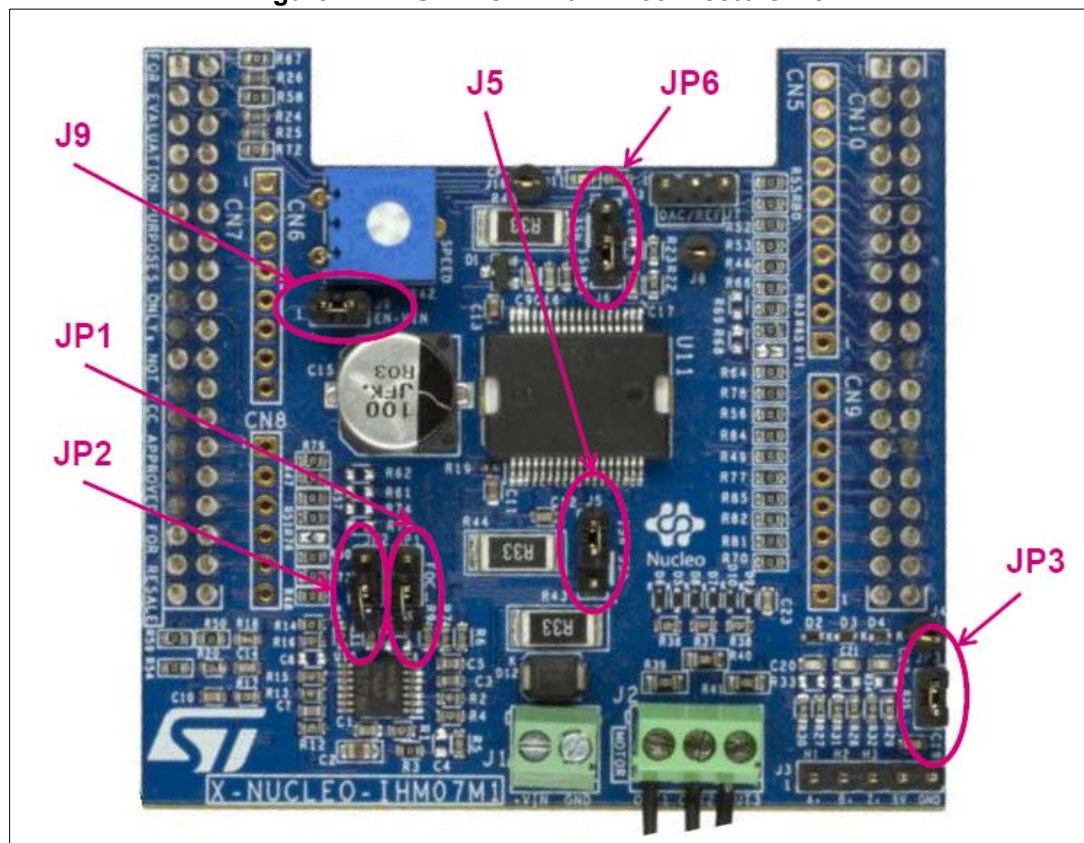
3.4 Upload the firmware example

The motor control Nucleo Pack example firmware is pre-loaded in NUCLEO-F302R8 board. As described in the previous section, it performs two different algorithms to run the motor, 6-step (trapezoidal control) or FOC (Field Oriented Control). This chapter describes the procedure to reload the firmware demonstration inside the NUCLEO-F302R8 board, in order to restart by default condition. There are two ways to do it: one is through ST-LINK tool (free download available from ST web site: *www.st.com*), and one with drag & drop function (suggested).

3.4.1 Drag & drop procedure:

1. Install ST-LINK drivers from *www.st.com* website.
2. On the NUCLEO-F302R8 board put JP5 jumper in U5V position.
3. Plug the NUCLEO-F302R8 Board to the host PC using a micro USB cable. If the ST-LINK driver is correctly installed, it will be recognized as an external memory device called "NUCLEO" or similar.
4. Take the binary file of the firmware demonstration (P-NUCLEO-IHM001.bin) and drag and drop the file into the "NUCLEO" device listed inside the disk drives list (this is showed by clicking the Start button of Windows OS interface), contained into X-CUBE-SPN7 firmware pack.
5. Wait until flashing is complete.

3.4.2 ST-LINK tool:

1. Open "ST-LINK tool".
2. Connect NUCLEO-F302R8 board to PC with a USB type A to Mini-B cable through USB connector (CN1) on NUCLEO-F302R8 board.
3. Make sure that the embedded ST-LINK/V2 is configured for in-system programming on NUCLEO-F302R8 board (both CN2 jumpers ON).
4. Use "P-NUCLEO-IHM001.bin" binary file to upload the code inside STM32, the window will appear as shown in *Figure 8*.

Figure 8. ST-LINK utility tool
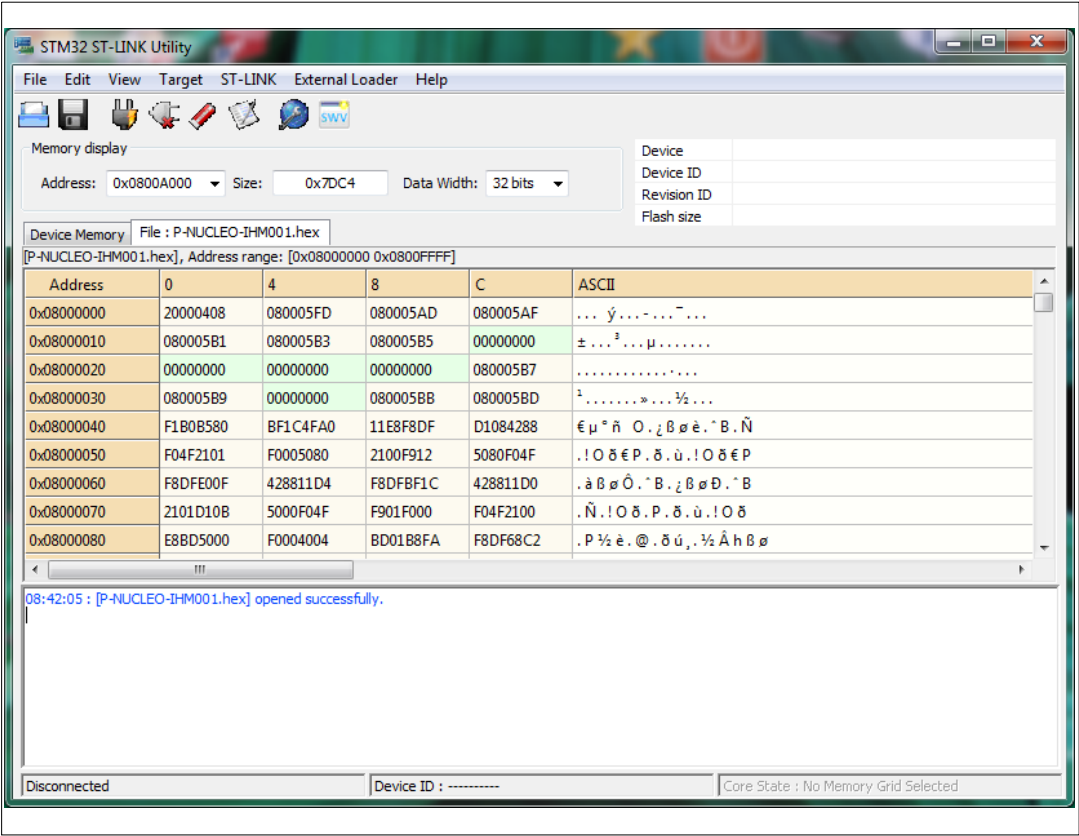
5.   Click on Target and Program buttons (see *Figure 9: ST-LINK utility tool programming environment*).

Figure 9. ST-LINK utility tool programming environment
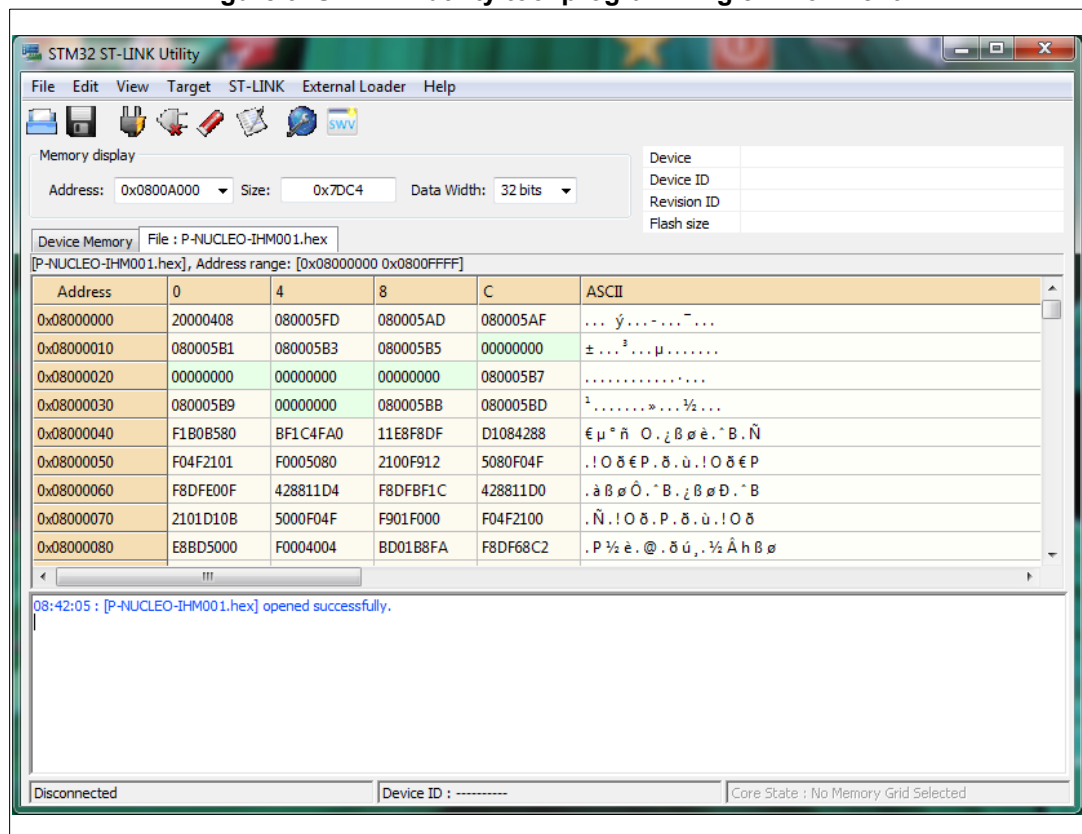
6. Click on Start and the firmware will be uploaded successfully.

4 Advanced user - 6-step and FOC control algorithm settings

4.1 6-step firmware based on X-CUBE-SPN7

4.1.1 Firmware architecture overview

The firmware example in X-CUBE-SPN7 is provided for three different IDE tools, in this case the IAR™ IDE workspace appears as shown in *Figure 10*.

Figure 10. 6-step firmware – Project workspace on IAR
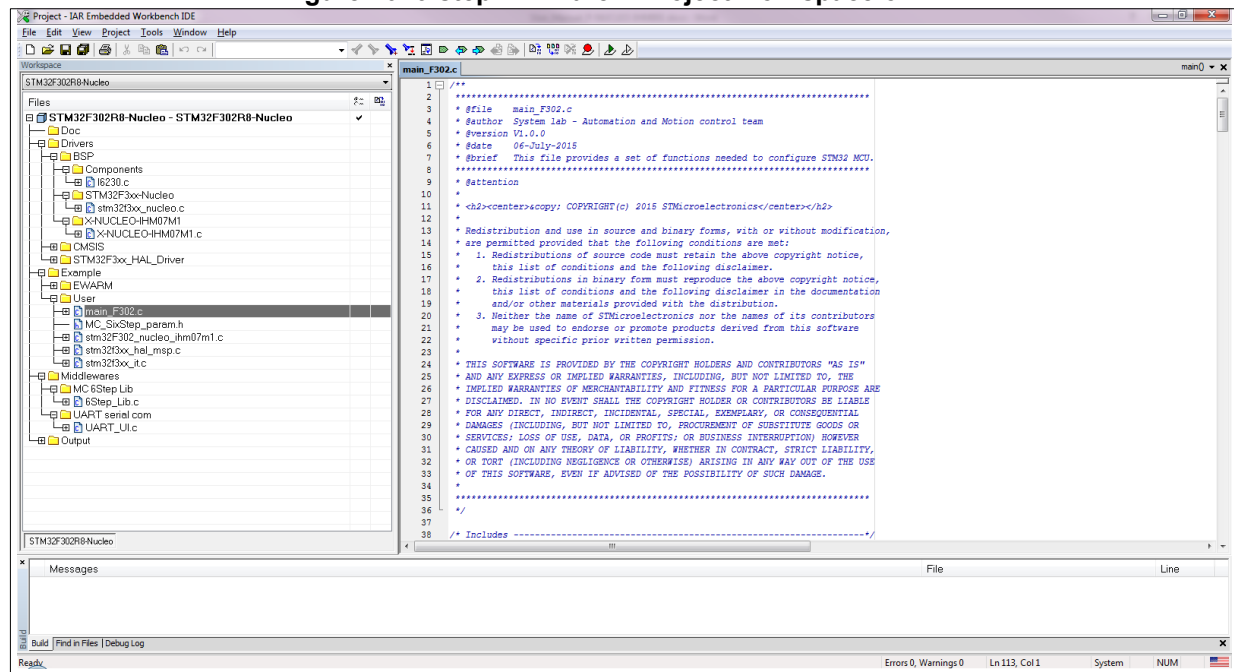
The firmware solution is built around three independent levels that can easily interact with each others as described in the below *Figure 11: Generic firmware architecture*:

Figure 11. Generic firmware architecture
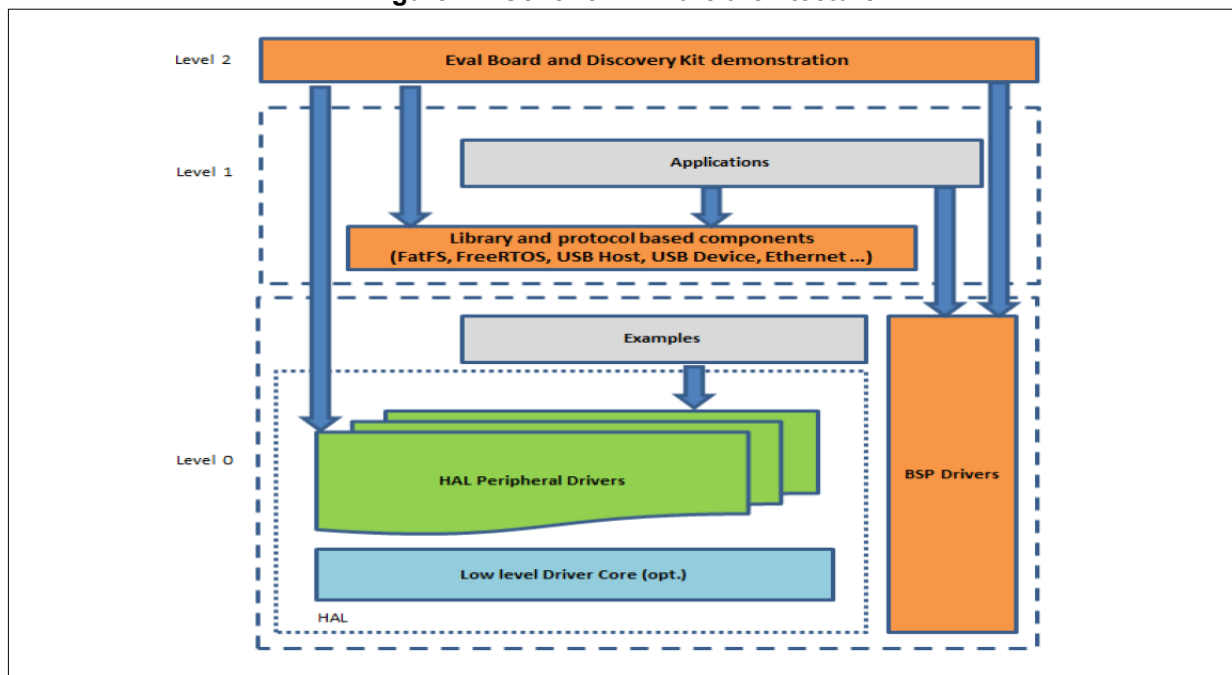
Level 0:

This level is divided into three sub-layers:

- **Board Support Package (BSP):** this layer offers a set of APIs relative to the hardware components in the hardware boards (Audio codec, IO expander, Touchscreen, SRAM driver, LCD drivers. etc…) and composed of two parts:

  – Component: is the driver relative to the external device on the board and not related to the STM32. The component driver provides specific APIs to the BSP driver external components and could be portable on any other board. In this case (X-NUCLEO-IHM07M1) the ST L6230 driver has been provided inside the firmware package.

  – BSP driver: it allows to link the component driver to a specific board and provides a set of friendly used APIs. The APIs naming rule is BSP_FUNCT_Action(): ex. BSP_LED_Init(),BSP_LED_On().

It is based on modular architecture, allowing to port it easily on any hardware by just implementing the low level routines.

- **Hardware Abstraction Layer (HAL):** this layer provides the low level drivers and the hardware interfacing methods to interact with the upper layers (application, libraries and stacks). It provides a generic, multi instance and functionalities oriented APIs which permit to offload the user application implementation, by providing ready to use process. For example, for the communication peripherals (I2S, UART…) it provides APIs allowing to initialize and configure the peripheral, manage data transfer based on polling, interrupt or DMA process, and manage communication errors that may raise during communication. The HAL drivers APIs are splitted in two categories: generic APIs which provides common and generic functions to all the STM32 series and

extension APIs, which provides specific and customized functions for a specific family or a specific part number.

- **Basic peripheral usage examples**: this layer encloses the examples built over the STM32 peripheral using only the HAL and BSP resources.

### Level 1:

This level is divided into two sub-layers:

- **Middleware components**: set of Libraries covering USB Host and device libraries, STemWin, FreeRTOS, FatFS, LwIP, and PolarSSL. Horizontal interactions between the components of this layer is done directly by calling the feature APIs, while the vertical interaction with the low level drivers is done through specific callbacks and static macros, implemented in the library system call interface. For example, the FatFs implements the disk I/O driver to access microSD drive or the USB Mass Storage Class. The middleware components provided with the X-CUBE-SPN7 package contain the core of the motor control algorithm: 6-step library (6Step_Lib.c/h) and interface files (stm32f302_ihm07m1.c/h). The interface file includes the map of STM32 MCU peripherals used (for instance, advanced TIMx, general TIMx, ADCx, DACx, UART etc.) to operate with the MC SixStep library. This file must be updated according with the modification done by the user through STM32CubeMX software, if channels or peripherals will be modified respect to the default configuration. At middleware level a serial communication based on UART with external PC terminal emulator has been included in X-CUBE-SPN7 package (see the UART_UI.c/h).

- **Examples based on the middleware components**: each middleware component comes with one or more examples (called also Applications), showing how to use it. Integration examples that use several middleware components are provided as well. This folder is created with STM32CubeMX software and it contains the main file for firmware initialization (peripherals, MC_6Step and UART communication). In addition a specific file (MC_SixStep_param.h) has been added to provide the complete list of parameters for MC-6Step library at application level. Inside the stm32fxxx_it.c file all interrupt handlers are defined and in particular it contains the starting point for UART communication.

Level 2:

This level is composed of a single layer which is global real-time and graphical demonstration based on the middleware service layer, the low level abstraction layer and the basic peripheral usage applications for board based functionalities.

The *Figure 12: X-CUBE-SPN7 software architecture* shows the firmware architecture of X-CUBE-SPN7 package, including also the HW components level.

Figure 12. X-CUBE-SPN7 software architecture
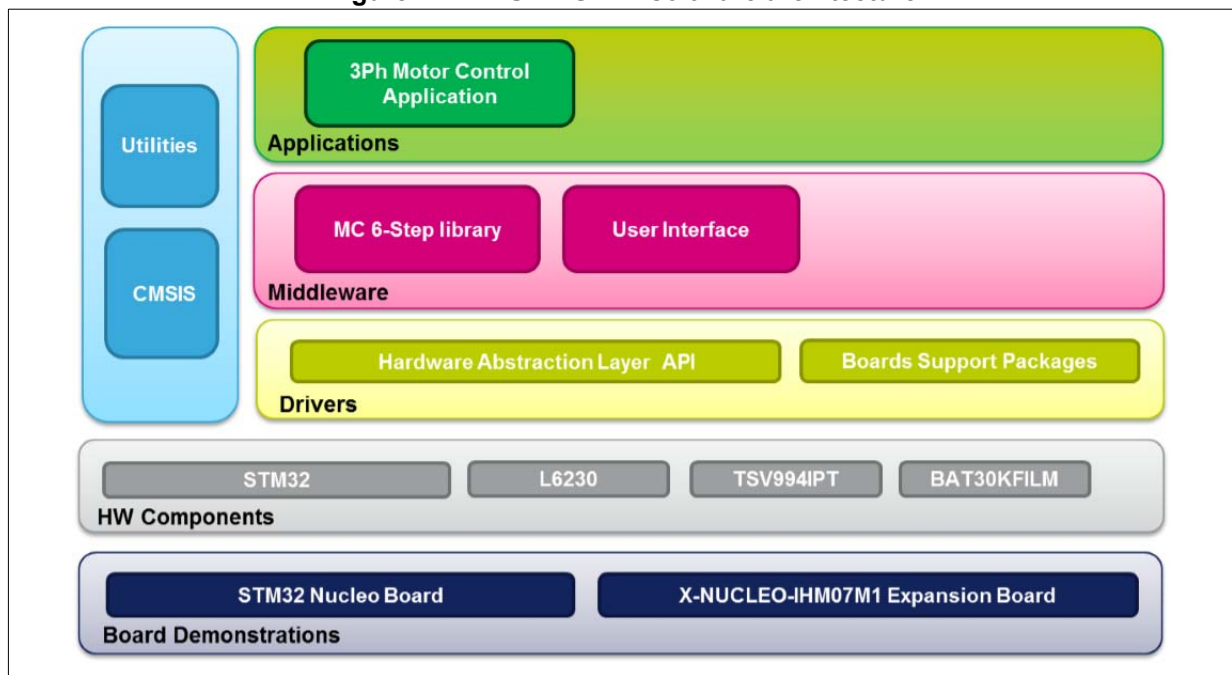
4.1.2 Firmware parameter settings to spin different BLDC motors

The firmware example provided for P-NUCLEO-IHM001 kit is tuned for low inductance/high speed motor (reference part: Bull-Running model BR2804-1700kV, 11.1Vdc, 5A, 7 pole pairs, 19000 MaxRPM speed). The X-CUBE-SPN7 firmware package is developed to simplify the way to spin a different kind of motors with only few changes. In this case, a header file (MC_SixStep_param.h) contained inside X-CUBE-SPN7, includes several parameters organized in two sections: basic and advanced. In the first section it is possible to change the main parameters: for instance, motor pole pairs, clockwise or counter clockwise motor direction, target speed or potentiometer selection. In the advanced section, it is possible to set the PI parameters, define the alignment time or the acceleration rate during startup, change the zero crossing threshold and a lot of parameters usefully for system fine tuning.

In case of different motor connected or different load condition, after the reset or power-on, the firmware is able to reduce the acceleration rate if startup fails and, at the next push button event (on NUCLEO-F302R8 board), a new value speed profile will be generated.

After this changes the firmware is ready to be recompiled with IDE tool and uploaded in NUCLEO-F302R8 board.

4.1.3 Inside the 6-step firmware

The main.c file contains the starting point of 6-step library for motor control, in particular the MC_SixStep_INIT() configures the basic structure of MC driver based on 6-step control algorithm and the header file (6Step_Lib.h) provides the connection between the application layer with this driver. The stm32f3xx_it.c file includes the entry point for UART communication and the handling code for BKIN interrupt. Inside the example folder the stm32F302_nucleo_ihm07m1.c contains all the MCU related functions, header files and the complete list of the peripherals used.

The core of 6-step algorithm is contained inside the middleware folder (6Step_Lib.c) and its header file has the complete list of API functions available for the user at application level, for instance `MC_StartMotor(),MC_StopMotor()` or `MC_SetSpeed(value).`

At the base of MC driver three main tasks run at different frequency and with different priority level, according to the specific function to cover, in particular:

1. *High frequency task*: it is dedicated for the high frequency function (advanced TIMx PWM generation, ADC reading) and it is managed at highest priority. This frequency is changeable through the STM32CubeMX software, i.e. TIM1_ARR.

2. *Medium frequency task*: it is dedicated for the medium frequency function (general TIMx for step timing) and it is managed at medium priority. This frequency is changeable through the STM32CubeMX software, i.e. TIM6_ARR (for NUCLEO-F302R8 board).

3. **Low frequency task**: it is dedicated for the low frequency function (SysTick timer for Speed Loop timing) and it is managed at lowest priority. The MC_SysTick_SixStep_MediumFrequencyTask() is called at SysTick frequency (1msec), while the speed loop function is managed by SPEED_LOOP_TIME (msec) defined in MC_SixStep_param.h file.

4.1.4 DAC settings for debug

For debug purpose it is possible to use the DAC peripheral and configure the 6-step library, in order to drive the signal. The function SET_DAC_value(dac_value) allows to convert the variable "dac_value" in 16-bit format (with no sign) in analog signal so it is possible to monitor for instance the motor speed (set by default) or the potentiometer value through an external oscilloscope attached at configured pin. By default PA4 pin is configured and it is accessible through the ST morpho connector and it is typically connected to DAC_CH1 (NUCLEO-F302R8 board). Other pin are available at J7 connector according with the NUCLEO-F302R8 board used. For pin modification remember to modify also the stm32F302_nucleo_ihm07m1.h file. The DAC peripheral is on by default but it is possible to disable it through the MC_SixStep_param.h.

4.2 ST FOC SDK – Configuration guide for P-NUCLEO-IHM001

The demonstration board supports also the ST FOC library and no hardware modification is needed to run the motor with this control algorithm. In this case it is only needed to configure the board for current sensing (1-shunt or 3-shunt mode) and select the JP1, JP2 jumpers, according to the jumper settings shown on *Table 1: Jumper settings*. It is also available the support of MC Workbench software through the USB cable used for the NUCLEO programming. In this case it is recommended to configure the FOC SDK for USART2 on PA2 and PA3 pin.

For further information about ST FOC SDK library refer to STSW-STM32100 document on ST website: *www.st.com.*

Electrical schematics

Figure 13. Current sense conditioning circuit

Figure 14. Sensors and shunt resistor circuit

VBUS SENSOR

HALL/ENCODER SENSOR

SPEED REGULATION

SHUNT RESISTOR

Figure 15. L6230 driver and BEMF detection circuit

L6230 DMOS driver
for three-phase brushless
DC motor

BEMF DETECTION - 6-STEP

Figure 16. STM32 MCU pin assignment

PWM

C10_14 PA11 R46 0
C10_28 R302 R49 0
C10_13 R53 PA6-BKIN DIAG/EN
C10_23 R56 PA8-UH IN1
C7_1 R58 PC10 EN1
C10_21 R64 PA9-VH IN2
C7_2 R67 PC11 EN2
C10_33 R70 PA10-WH IN3
C7_3 R72 PC12 EN3

Currents

C7_28 PA0 - PhA 0 R47 Curr_fdbk1
C7_36 PC1 - PhB 0 R48 Curr_fdbk2
C7_38 PC0 - PhC 0 R50 Curr_fdbk3
C7_30 PA1 - BUSV 0 R51 VBUS
C7_35 PC2 - Temp. Sens 0 R54 Temperature feedback

0 R57 C7_30
3V3

Vshunt_1 PA1 - EmbPGA R61 N.N. 4.99K R62 N.M.13K 1%
Vshunt_2 PA7 - EmbPGA R68 N.N. 4.99K R65 0 R69 N.M.13K 1% C10_15
C10_26 R302 R71 0-N.M. R73 0 C7_34
Vshunt_3 PB0 - EmbPGA R74 N.N. 4.99K R75 N.M.13K 1%

C10_12 PA12- TIM1 ETR R52 CPOUT / GPIO_BEMF
C10_1 PC9 - GPIO 0 R55
C7_37 PC3 - ADC 0 R59 BEMF1
C7_34 PB0 - ADC 0 R60 BEMF2
C10_15 PA7 - ADC 0 R63 BEMF3
C10_26 R302 R66

BEMF 6-step

HALL/ENCODER SENSOR

C7_17 R79 0 PA15 - Enc. A/H1 A+/H1
C10_31 R81 0 PB3 - Enc. B/H2 B+/H2
C10_25 R84 0 PB10 - Enc. Z/H3 Z+/H3

DAC/REF

C7_32 0-N.M. R76 PA4- DAC CURRENT REF
C10_27 0 R77 PB4-PWM
C10_11 0 R80 PA5 - DAC
C10_30 R302 R82 PA5/PB13
C10_29 0 R85 PB5-DAC PWM

1 2 3 J7 Stripline m. 1x3

J8 RING 1

C10_24 PB1 0 R78 SPEED
C10_22 PB2 R83 510 D11 RED

VIN+ J9
2 1 VIN C7_24
OPEN BY DEFAULT

If J9 ON:
1) 12VMAX ON J1
2) JP5 (Nucleo pin 2,3)
2) JP1 (Nucleo) Removed

STM32 NUCLEO PART USED

PC13 - START/STOP (B1 BUTTON)

RESET - MCU RST (B2 BUTTON)

CN7

1	C7_1	C7_2	2
3	C7_3	C7_4	4
5	C7_5	E5V C7_6	6
7	C7_7	C7_8	8
9	C7_9	N.C. C7_10	10
11	C7_11	C7_12	12
13	C7_13	C7_14	14
15	C7_15	C7_16	16
17	C7_17	C7_18	18
19	C7_19	C7_20	20
21	C7_21	C7_22	22
23	C7_23	C7_24	24
25	C7_25	C7_26	26
27	C7_27	C7_28	28
29	C7_29	C7_30	30
31	C7_31	C7_32	32
33	C7_33	C7_34	34
35	C7_35	C7_36	36
37	C7_37	C7_38	38

5V
C7_18
3V3

ARDUINO/MORPHO CONNECTOR

CN6
C6_1 N.C. 1
C6_2 +3V3 2
C6_3 3
C6_4 4
C6_5 5
C6_6 6
C6_7 7
C6_8 8

CN8
C8_1 1
C8_2 2
C8_3 3
C8_4 4
C8_5 5
C8_6 6

CN5
10 C5_10
9 C5_9
8 C5_8
7 C5_7
6 C5_6
5 C5_5
4 C5_4
3 C5_3
2 C5_2
1 C5_1

CN9
8 C9_8
7 C9_7
6 C9_6
5 C9_5
4 C9_4
3 C9_3
2 C9_2
1 C9_1

CN10

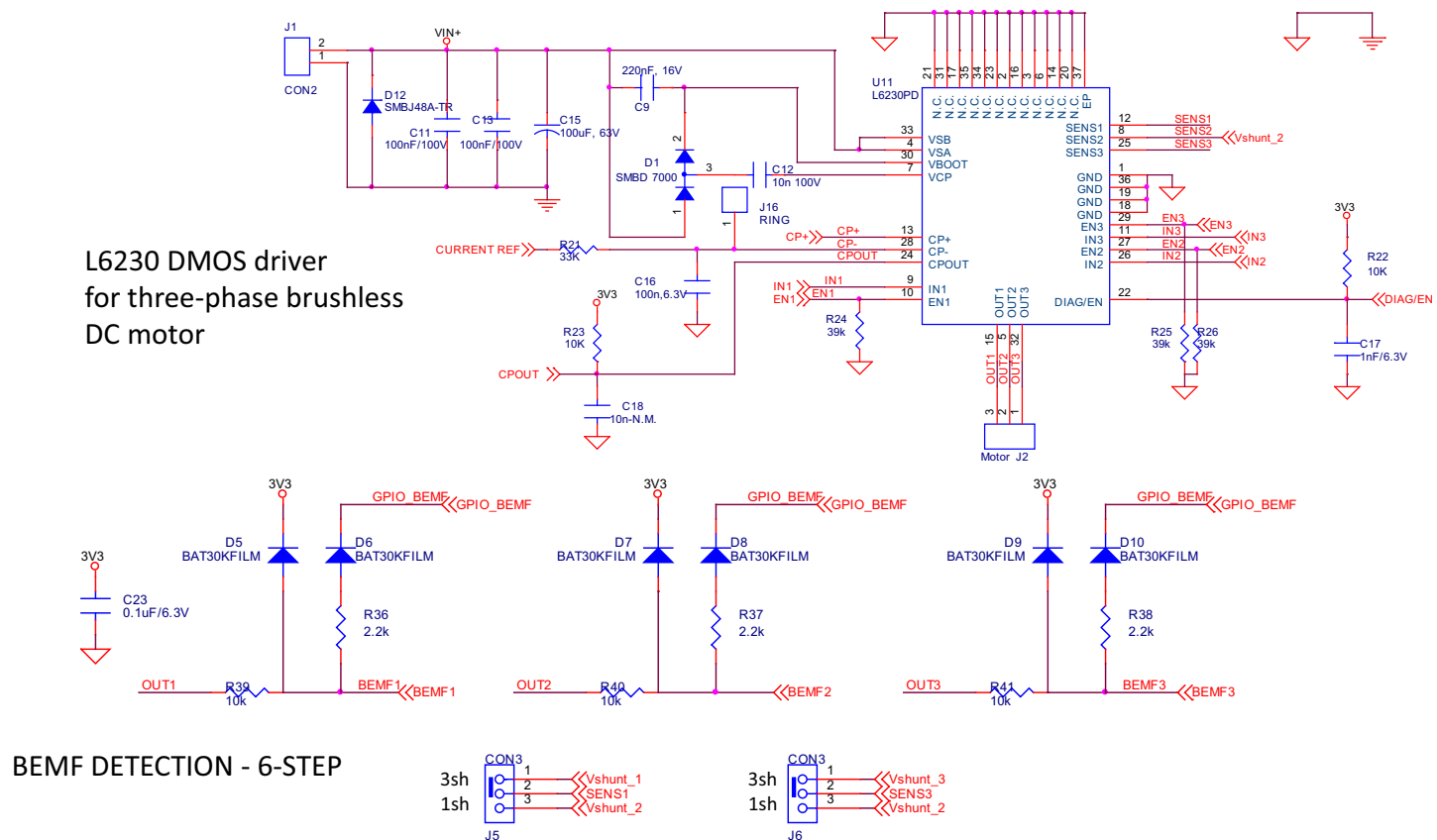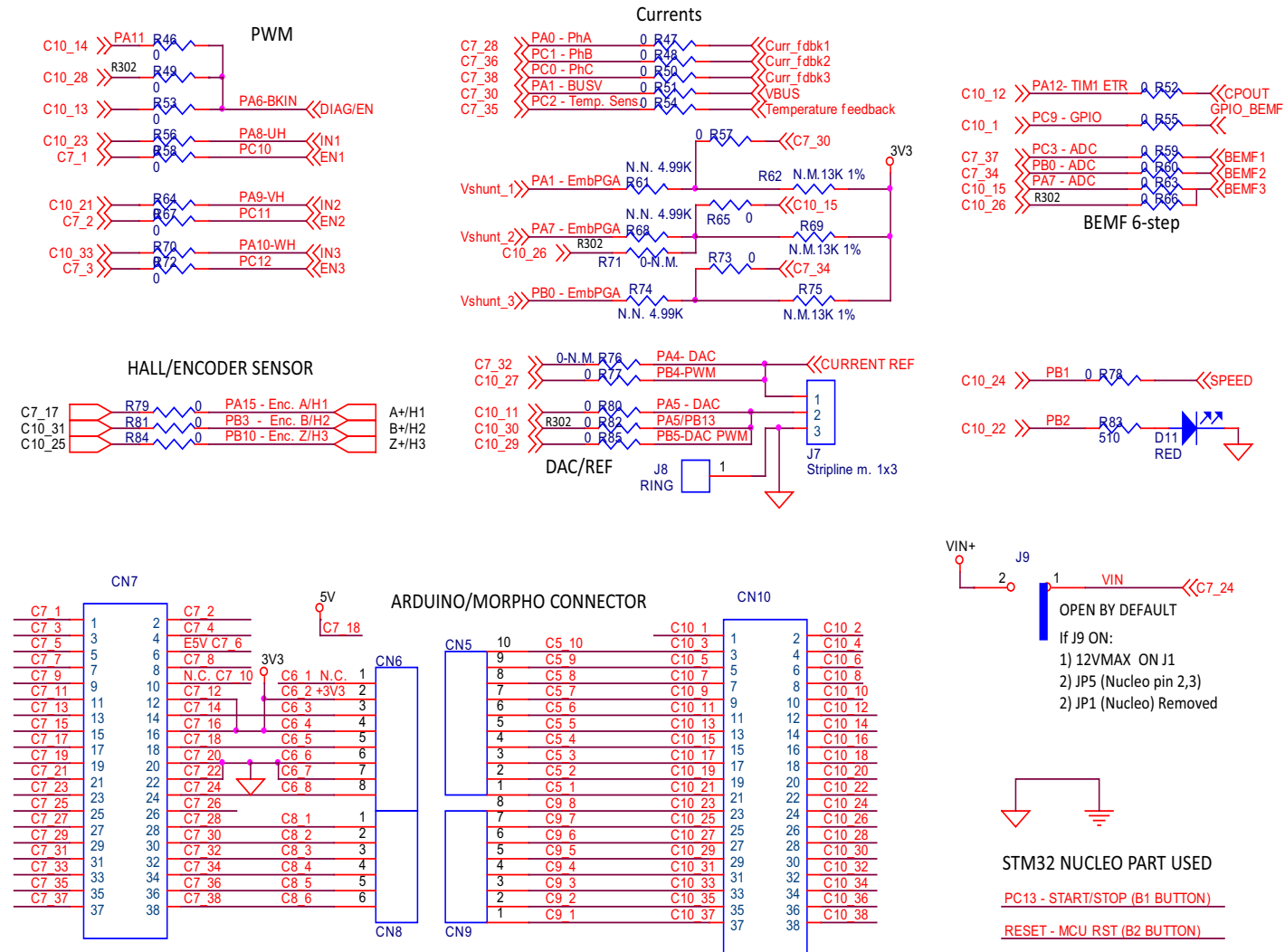| | C10_1 | C10_2 | |
1	C10_3	C10_4	2
3	C10_5	C10_6	4
5	C10_7	C10_8	6
7	C10_9	C10_10	8
9	C10_11	C10_12	10
11	C10_13	C10_14	12
13	C10_15	C10_16	14
15	C10_17	C10_18	16
17	C10_19	C10_20	18
19	C10_21	C10_22	20
21	C10_23	C10_24	22
23	C10_25	C10_26	24
25	C10_27	C10_28	26
27	C10_29	C10_30	28
29	C10_31	C10_32	30
31	C10_33	C10_34	32
33	C10_35	C10_36	34
35	C10_37	C10_38	36
37			38

Appendix A   Federal Communications Commission (FCC) and Industry Canada (IC) Compliance Statements

A.1   FCC Compliance Statement

A.1.1   Part 15.19

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

A.1.2   Part 15.105

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference's by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and the receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

A.1.3   Part 15.21

Any changes or modifications to this equipment not expressly approved by STMicroelectronics may cause harmful interference and void the user's authority to operate this equipment.

A.2   IC Compliance Statement

A.2.1   Compliance Statement

Industry Canada ICES-003 Compliance Label: *CAN ICES-3 (B)/NMB-3(B)*

A.2.2   Déclaration de conformité

Étiquette de conformité à la NMB-003 d'Industrie Canada: *CAN ICES-3 (B)/NMB-3(B)*

6 Revision History

Table 4. Document Revision History

Date	Version	Revision Details
22-Sep-2015	1	Initial version

IMPORTANT NOTICE – PLEASE READ CAREFULLY