UM1945
User manual

STM32 Nucleo Packs - Motor Control FOC and 6-step solutions for three-phase, low-voltage and low-current motors

Introduction

The STM32 Nucleo Packs P-NUCLEO-IHM001 and P-NUCLEO-IHM002 are motor control kits 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 3-phase, low-voltage, DC brushless motor with the addition of the STM32 NUCLEO board through the ST morpho connector (see *Figure 1*). The P-NUCLEO-IHM002 comes with a power supply unit (see *Figure 2*).

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 the STM32 Nucleo Packs 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.

Figure 1. P-NUCLEO-IHM001 Pack

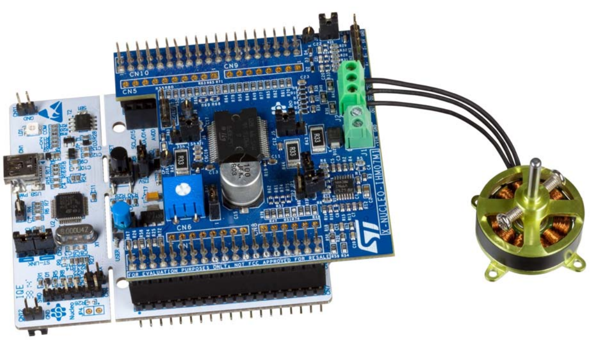


Figure 2. P-NUCLEO-IHM002 Pack

1. Pictures are 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 both Nucleo Packs:

- 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 sensored 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 connectors
- PCB type and size:
  - Material of PCB - FR-4
  - 4-layer layout
  - Copper thickness: 70 µm (external layer), 35 µm (internal layer)
  - Total dimensions of the expansion board: 70 mm x 66 mm

The information listed below shows the power supply electrical characteristics:

- Output characteristics:
  - Nominal voltage 12V
  - Nominal current 2A
  - Minimum voltage 11.4V
  - Maximum voltage: 12.6V
  - Load regulation +-5%
  - Ripple at nominal voltage: 50mVpp
- Input characteristics:
  - Nominal voltage: 100-240Vac
  - Variation voltage range: 90-264Vac
  - Nominal frequency: 50/60Hz
  - Variation frequency: 47-63Hz

a. Device characteristic.

2 Ordering information

To order the STM32 Nucleo Packs refer to *Table 1*.

Table 1. Ordering information

Order code	Board	DC Power Supply
P-NUCLEO-IHM001	X-NUCLEO-IHM07M1, NUCLEO-F302R8	Not provided
P-NUCLEO-IHM002	X-NUCLEO-IHM07M1, NUCLEO-F302R8	Provided (12 V, 2 A)

3 Getting started as basic user

3.1 System architecture

A generic motor control system as the P-Nucleo-IHM001 and the P-Nucleo-IHM002 can be basically schematized as the arrangement of three main blocks (see *Figure 3*):

- **Control block**: its main task is to accept user commands and configuration parameters to drive a motor. The P-Nucleo-IHM001 and the P-NUCLEO-IHM002 Packs are based on the 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.

The P-Nucleo-IHM002 has a fourth block (see *Figure 4*):

- **DC Power supply unit**: it provides the power for the Pack (12 V, 2 A).

Figure 3. P-NUCLEO-IHM001 Pack main blocks

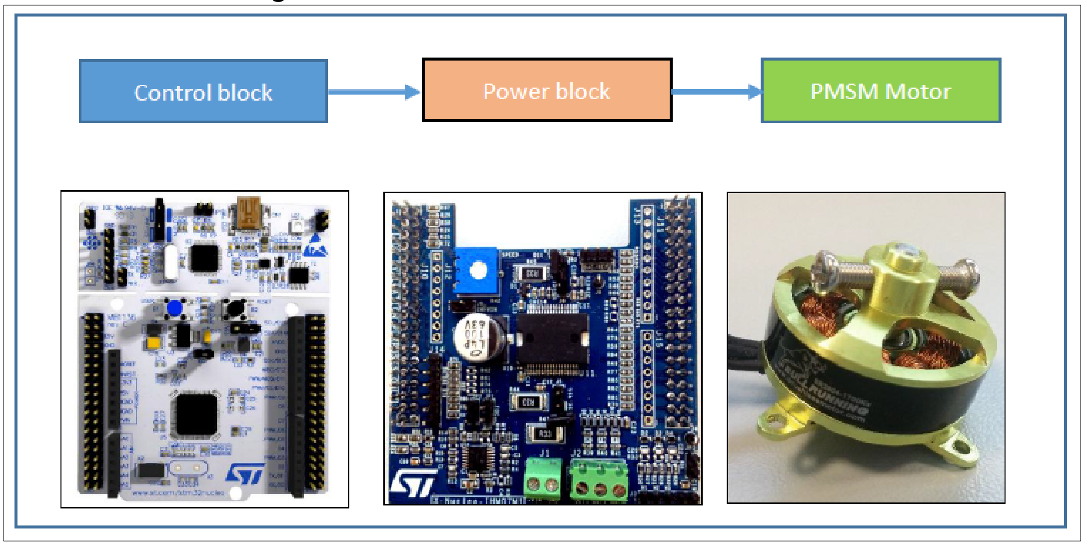
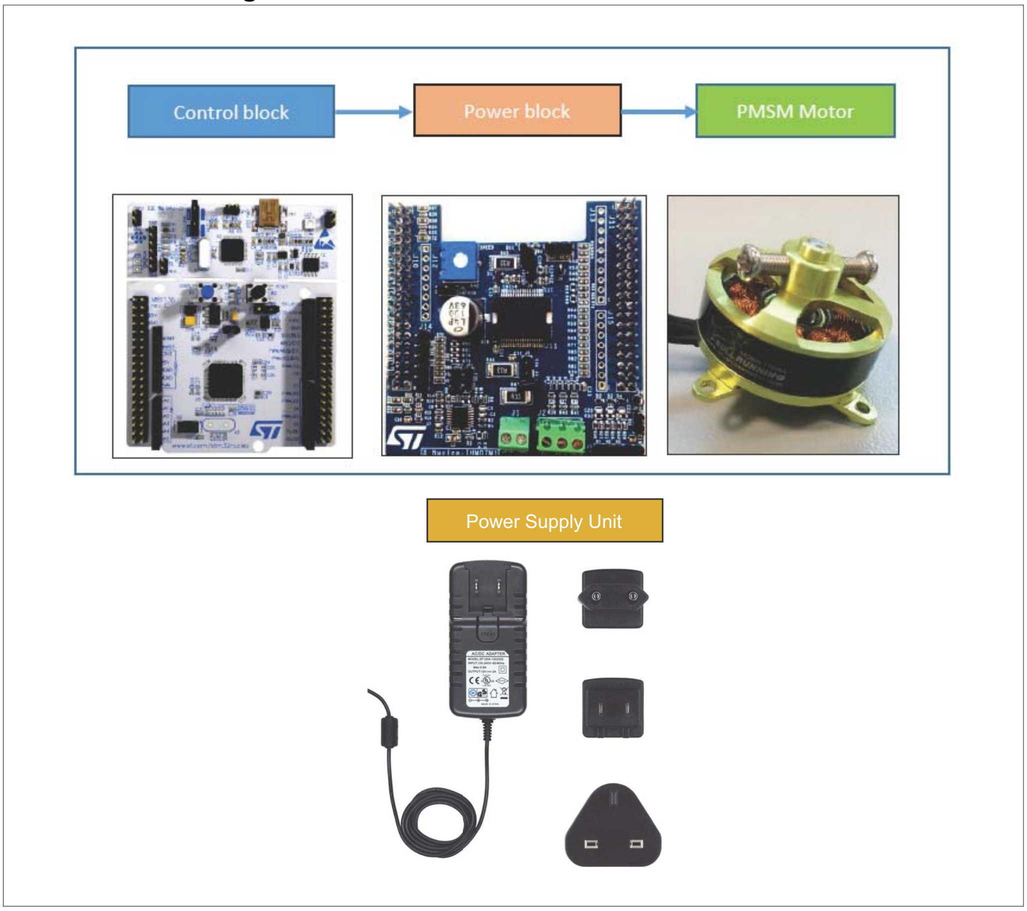


Figure 4. P-NUCLEO-IHM002 Pack main blocks



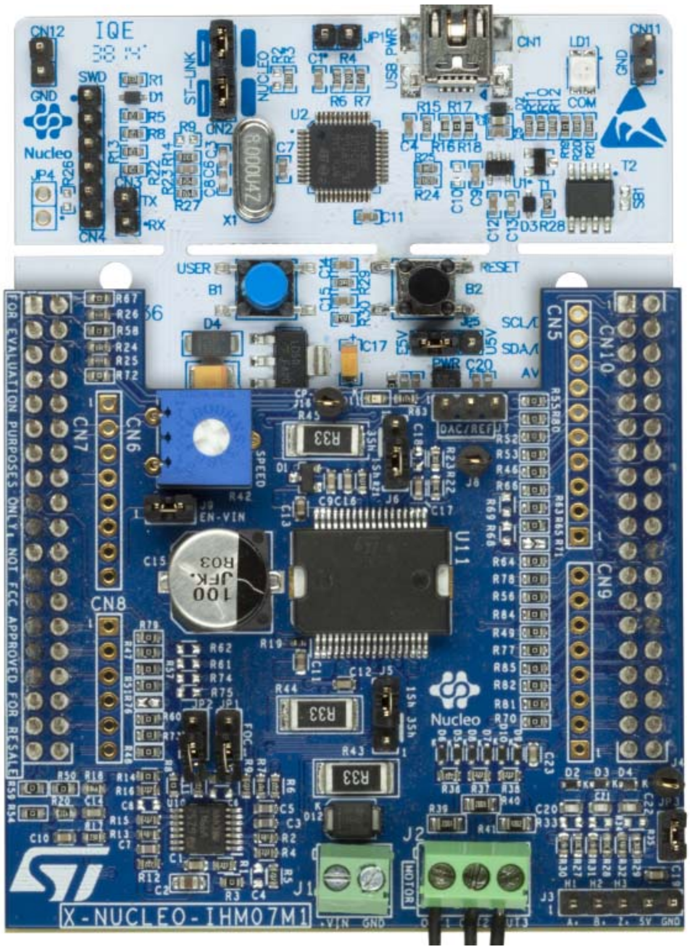
3.2 Building and run the motor control for the STM32 Nucleo Packs

The Nucleo Packs are complete hardware development platforms for the STM32 Nucleo ecosystem to evaluate a motor control solution with a single motor.

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

1. The X-NUCLEO-IHM07M1 must be stacked on the NUCLEO-F302R8 board through the ST morpho connectors. There is only one position allowed for this connection, in particular the two buttons on the NUCLEO-F302R8 board (blue button B1 and black button B2) must be kept out, as shown in *Figure 5*.

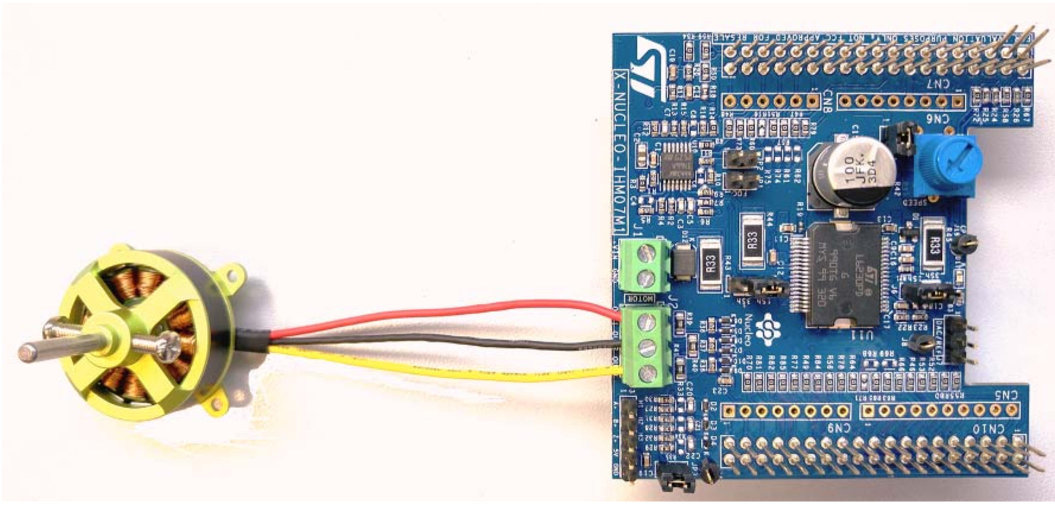
Figure 5. X-NUCLEO-IHM07M1 and NUCLEO-F302R8 assembled



1. The interconnection between the X-NUCLEO-IHM07M1 and the STM32 NUCLEO boards 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 6: Motor connection with X-NUCLEO-IHM07M1*: it is mandatory to connect the yellow wire to OUT3, the black one to OUT2 and the red one to OUT1, to respect clockwise and counterclockwise motor rotation, according to the firmware implementation.

Figure 6. Motor connection with X-NUCLEO-IHM07M1

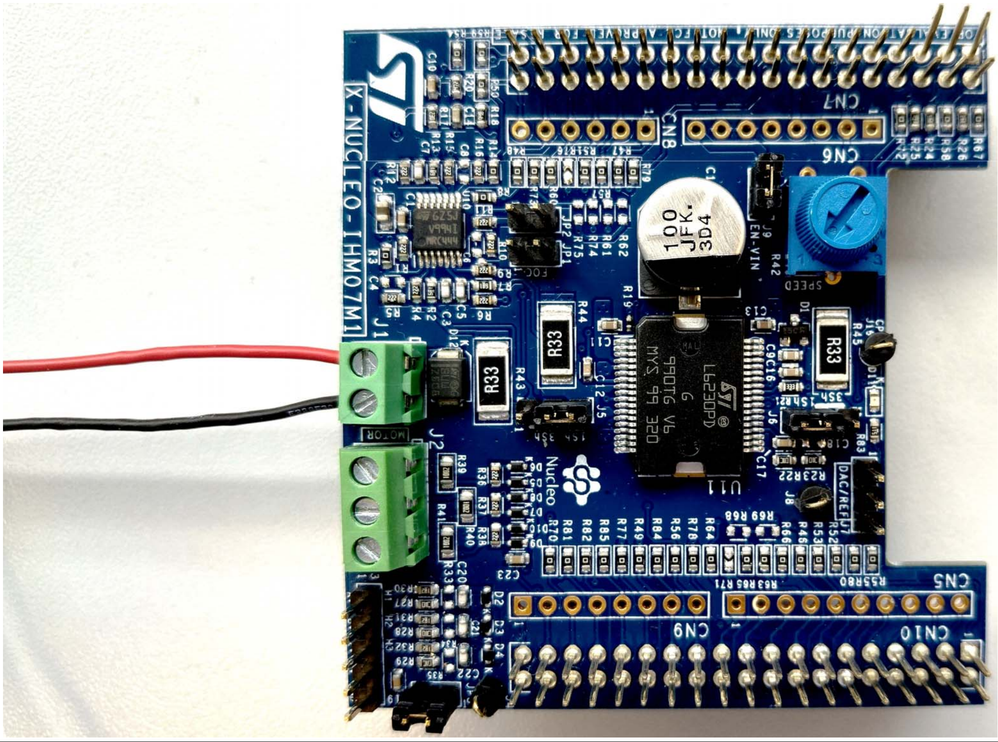


3. Select the jumper configuration on the power board to choose the desired control algorithm (6-step or FOC) as described below:
   a) On the NUCLEO-F302R8 board, check the jumper settings: JP1 open, JP5 (PWR) on E5V side, JP6 (IDD) closed.
   b) On X-NUCLEO-IHM07M1 expansion board:
   – Check jumper 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
4. Connect the DC power supply (use the power supply provided with the board or an equivalent one) on J1[c] connector and power-on (up to 12 V DC for BR2804 motor included in the P-NUCLEO-IHM002 Pack, as shown in *Figure 7: Power supply connection for X-NUCLEO-IHM07M1*).

b. Supply voltage has to be off before changing the control mode.

c. For a different motor (> 12V) it is mandatory to remove the jumper J9 on power board, before the power-on, to avoid damaging the NUCLEO-F302R8 board. To supply the STM32-NUCLEO from USB, the jumper JP5 must be connected between Pin 1 and Pin 2. For further details on Nucleo settings refer to "*STM32 Nucleo-64 boards*" User manual (UM1724) available at the *www.st.com* website.

Figure 7. Power supply connection for X-NUCLEO-IHM07M1



1. The red cable is VCC, the black one is GND.

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

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

6. Press the blue button on NUCLEO-F302R8 (B1) and the motor starts spinning.
7. Rotate the potentiometer on X-NUCLEO-IHM07M1 board to regulate the motor speed.

3.3 Hardware settings

The *Table 2* shows the jumper configuration on X-NUCLEO-IHM07M1 board (see also *Figure 8: X-NUCLEO-IHM07M1 – top layer with silk-screen* and *Figure 9: X-NUCLEO-IHM07M1 connectors*). According to the jumper selection, it is possible to choose the 1-shunt or 3-shunt mode, the current sensing circuit offset level, the hall/encoder with pull-up or the external supply for the NUCLEO-F302R8 board.

Table 2. Jumper settings

Jumper	Permitted configuration	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. They 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, to not damage the NUCLEO-F302R8 board. For further details on Nucleo settings refer to *"STM32 Nucleo-64 boards"* User manual (UM1724) available from the *www.st.com* website.
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/single-shunt is indicated. Also the default position is indicated.

The *Table 3* shows the main connector on the X-NUCLEO-IHM07M1 board.

Table 3. 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 stacked on ST morpho connectors, male pin headers (CN7 and CN10) accessible on both sides of the board. They can be used to connect this power board to the NUCLEO-F302R8 board. All signals and power pins for MCU are available on the ST morpho connectors. For further details refer to the section 5.12 ST morpho connector of the "STM32 Nucleo-64 boards" User manual (UM1724) available on the *www.st.com* website.

Figure 8. X-NUCLEO-IHM07M1 – top layer with silk-screen

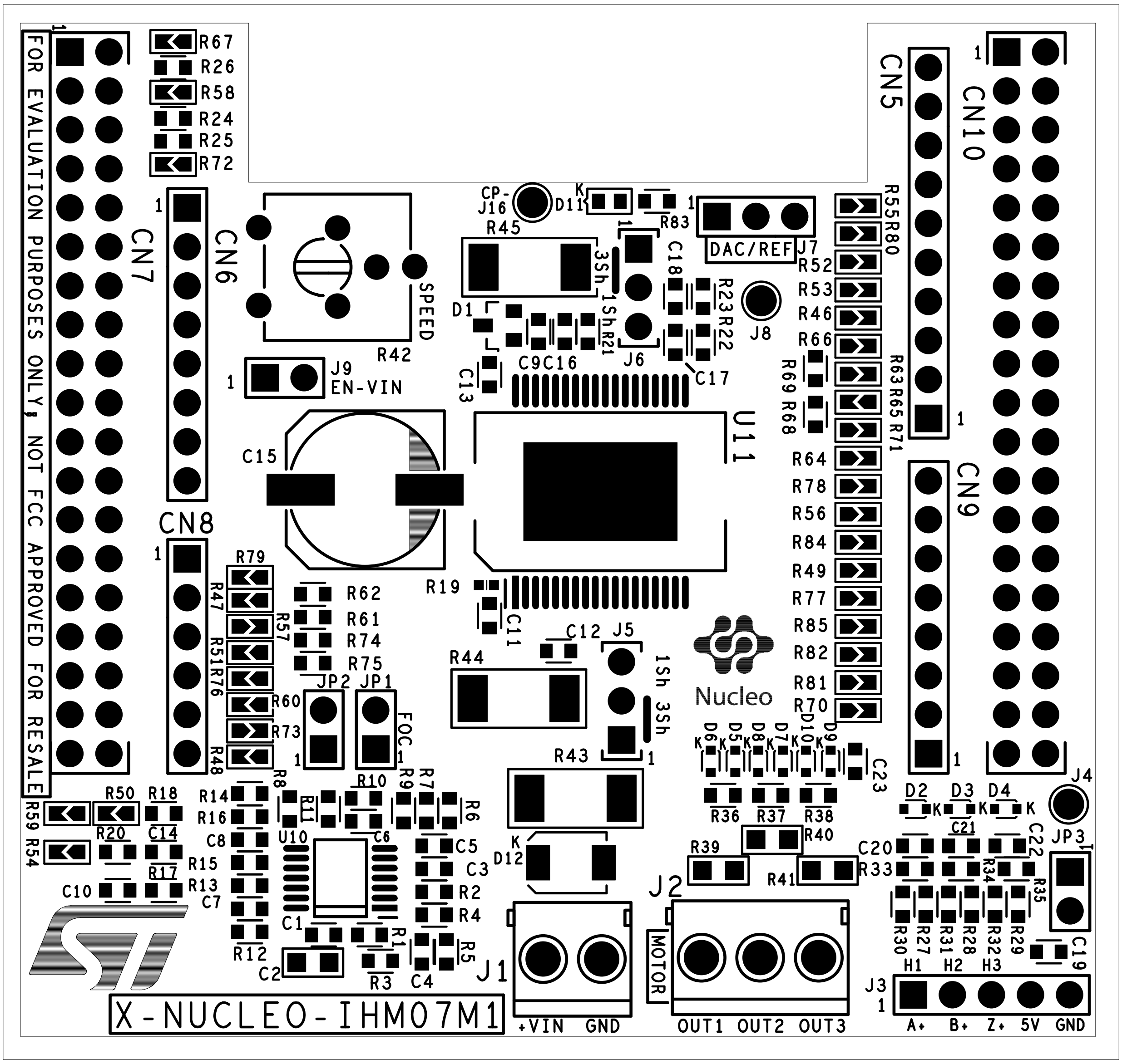


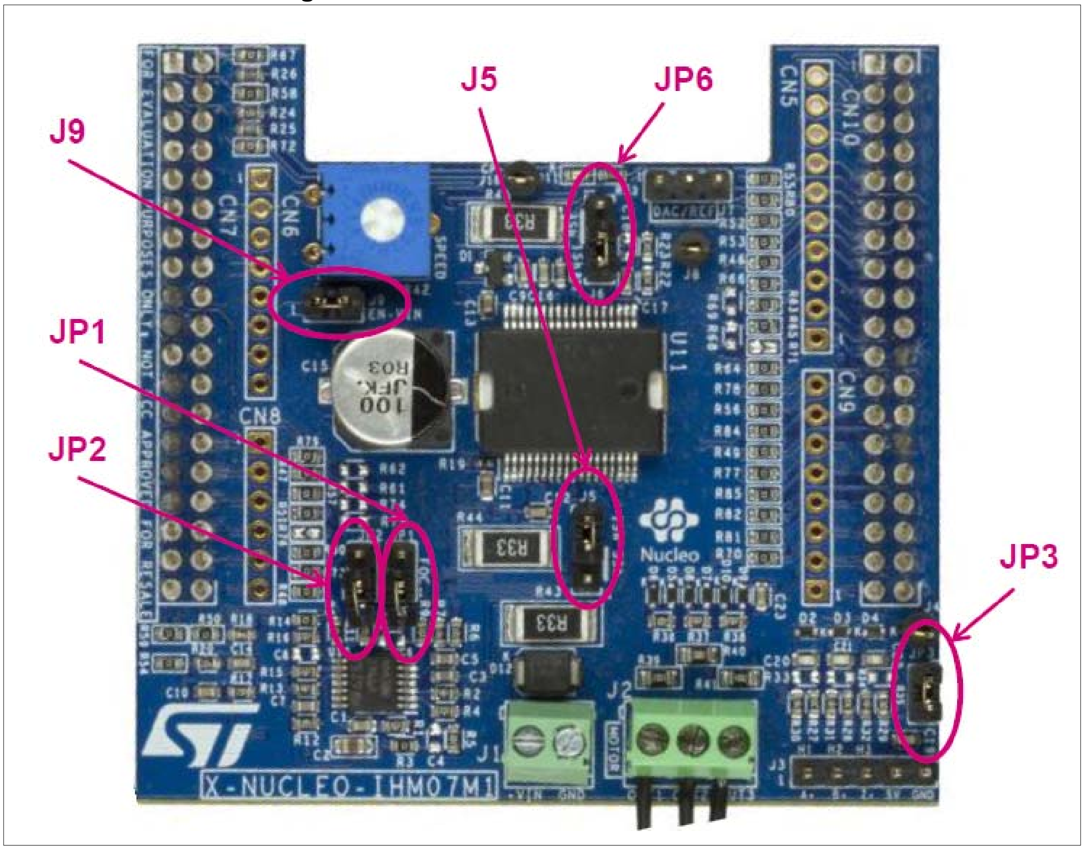
Table 4. Connector descriptions

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 4. Connector descriptions (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 9. X-NUCLEO-IHM07M1 connectors



3.4 Upload the firmware example

The example for the motor-control firmware of the Nucleo Packs is pre-loaded in the NUCLEO-F302R8 board. As described in the previous *Section 3.3: Hardware settings*, 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, to restart by the default condition. There are two ways to do it:

- Drag and drop procedure (suggested), see *Section 3.4.1*
- Through ST-LINK tool (free download available from the ST web site: *www.st.com*), see *Section 3.4.2*

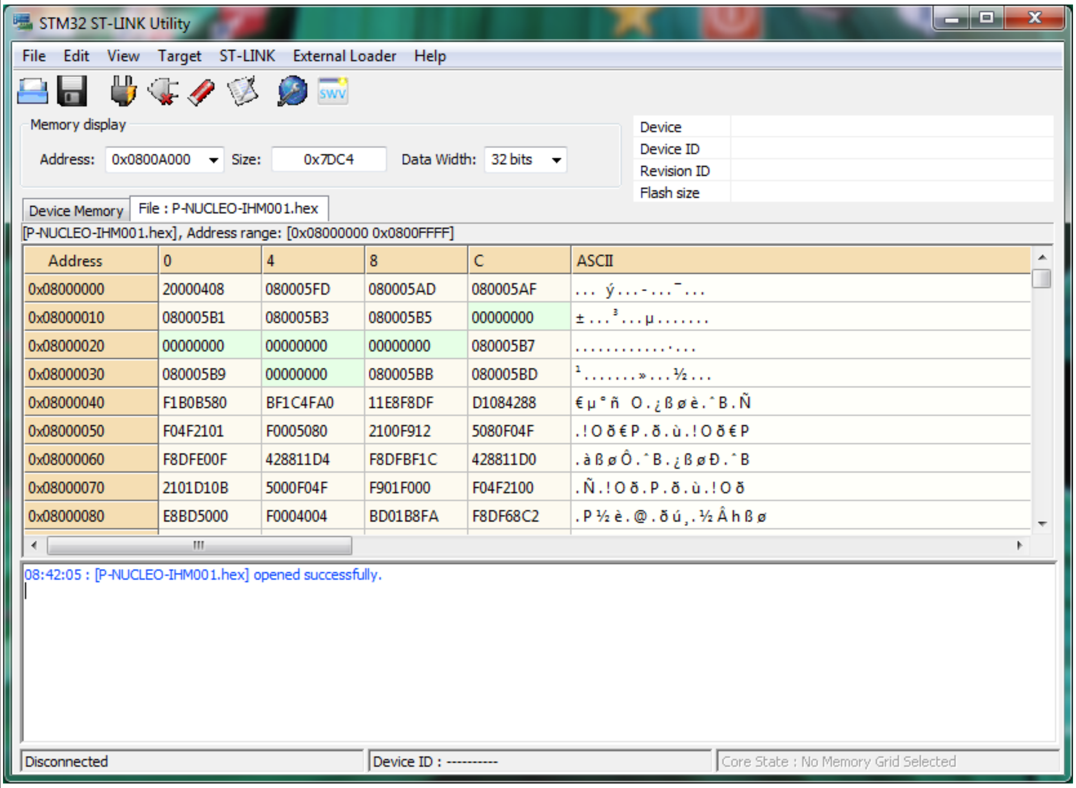
3.4.1 Drag and drop procedure

1. Install ST-LINK drivers from the *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 is 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 list of the disk drives (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

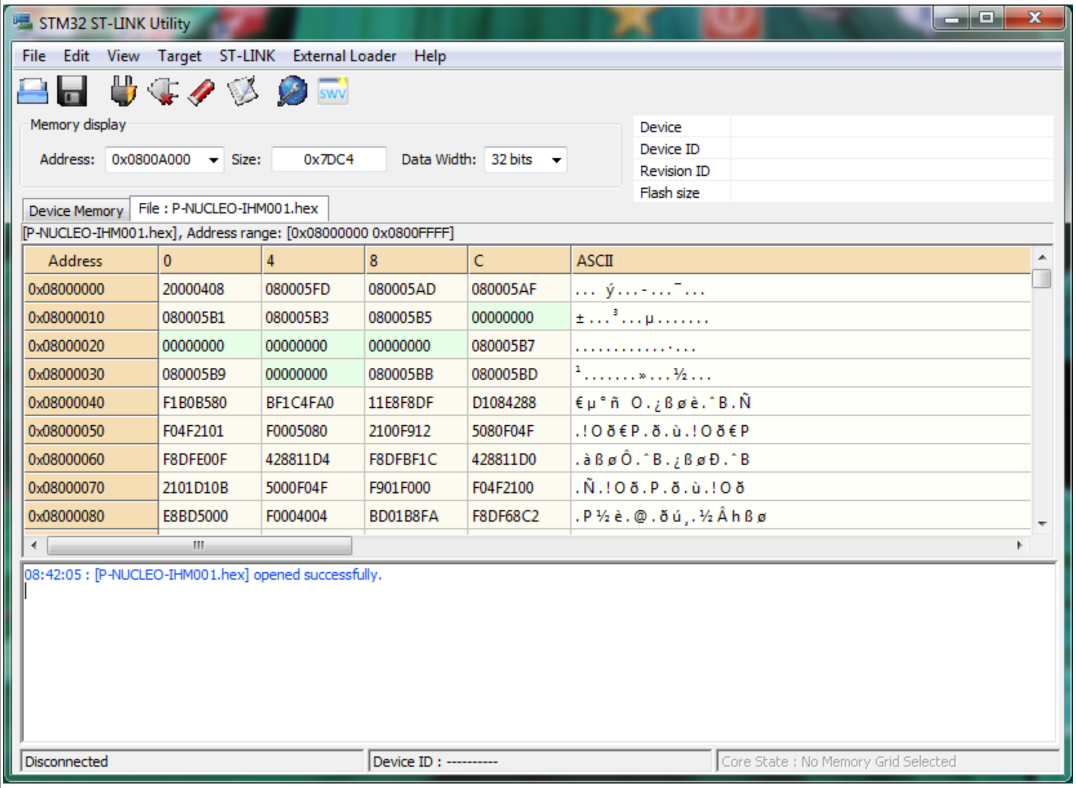
1. Open "ST-LINK tool".
2. Connect the NUCLEO-F302R8 board to the PC with a USB Type-A to Mini-B cable through the USB connector (CN1) on the NUCLEO-F302R8 board.
3. Make sure that the embedded ST-LINK/V2 is configured for in-system programming on the NUCLEO-F302R8 board (both CN2 jumpers ON).
4. Use “P-NUCLEO-IHM001.bin“ binary file to upload the code inside STM32, the window will appear as shown in *Figure 10*.

Figure 10. ST-LINK utility tool



5. Click on Target and Program buttons (see *Figure 11: ST-LINK programming environment*).

Figure 11. ST-LINK programming environment



6. Click on Start to upload the firmware.

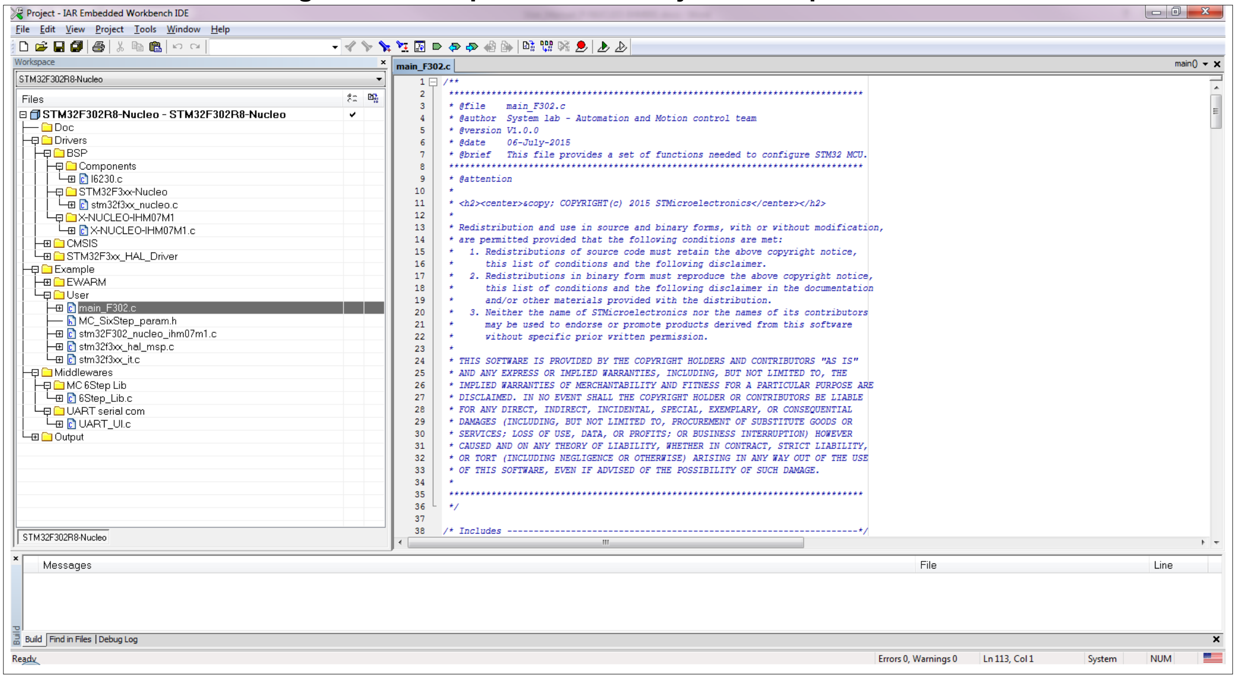
4 6-step and FOC control algorithm settings - advanced user

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

4.1.1 Firmware architecture overview

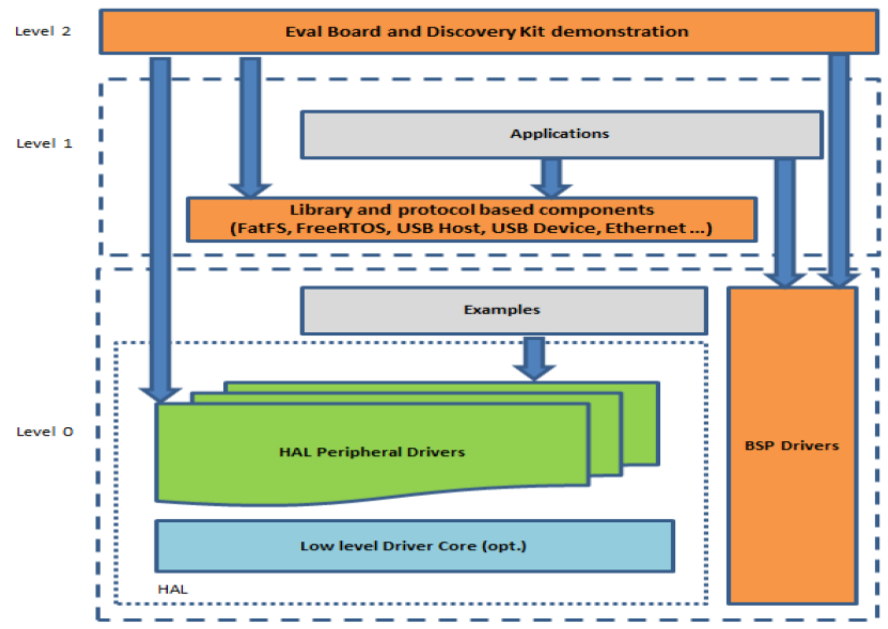
The firmware example in X-CUBE-SPN7 is provided for three different IDE tools, in this case the IAR™ IDE workspace appears as shown in *Figure 12*.

Figure 12. 6-step firmware – Project workspace on IAR



The firmware solution is built around three independent levels that easily interact as described in the *Figure 13: Architecture of a generic firmware* below:

Figure 13. Architecture of a generic firmware



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 it is composed of two parts:
  - Component: is the driver of the external device on the board and not related to the STM32. The component driver provides specific APIs for the BSP driver of external components and it is portable on any 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 functionality-oriented APIs which permit to offload the implementation of the user application, by providing ready-to-use process. For example, for the communication peripherals (I^2S, UART…) it provides APIs allowing the initialization and configuration of the peripheral, the management of the data transfer based on polling, the interrupt or DMA process, and the management of the communication errors that may raise during the communication. The HAL drivers APIs are splitted in two categories: generic APIs which provide common and generic functions to all the STM32 Series and extension APIs, which provide specific and customized functions for a specific family or a specific part number.

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

Level 1:

This level is divided into two sub-layers:

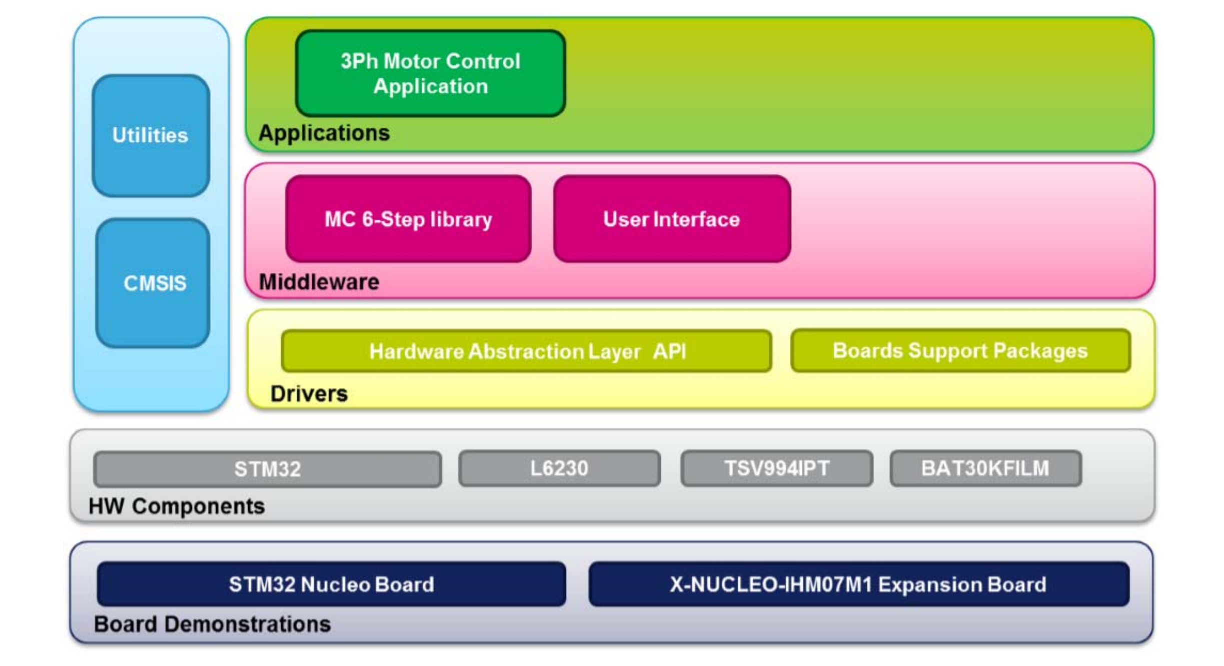
- **Middleware components**: set of libraries covering USB host and device libraries, STemWin, FreeRTOS, FatFS, LwIP, and PolarSSL. Horizontal interactions between the components of this layer is done directly by calling the feature APIs, while the vertical interaction with the low-level drivers is done 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 the microSD driver 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 the 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 the STM32CubeMX software (i.e. if channels or peripherals are modified respect to the default configuration). At middleware level a serial communication based on UART with external PC terminal emulator has been included into the 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 the 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 the 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 a 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 14: X-CUBE-SPN7 firmware architecture* shows the firmware architecture of the X-CUBE-SPN7 package, also including the hardware-component level.

Figure 14. X-CUBE-SPN7 firmware architecture



4.1.2 Firmware parameter settings to spin different BLDC motors

The firmware example provided for the STM32 Nucleo Packs 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 the X-CUBE-SPN7 package, 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, to define the alignment time or the acceleration rate during startup, to change the zero-crossing threshold and a lot of parameters useful to fine-tune the system.

In case of a 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-speed-profile value will be generated.

After this changes the firmware is ready to be recompiled with the IDE tool and uploaded in the 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 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 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 for the low-frequency function (SysTick timer for Speed Loop timing) and it is managed at the 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, 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 the 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 the STM32 Nucleo Packs

The demonstration board supports also the ST FOC library and no hardware modification is required to run the motor with this control algorithm. In this case the board must be configured for the current sensing (1-shunt or 3-shunt mode) selecting the JP1, JP2 jumpers, according to the jumper settings shown in *Table 2: 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 *“STM32 PMSM FOC Software Development Kit”* Databrief (DB2187) at the *www.st.com* website.

5 Electrical schematics

Figure 15. Current sense conditioning circuit

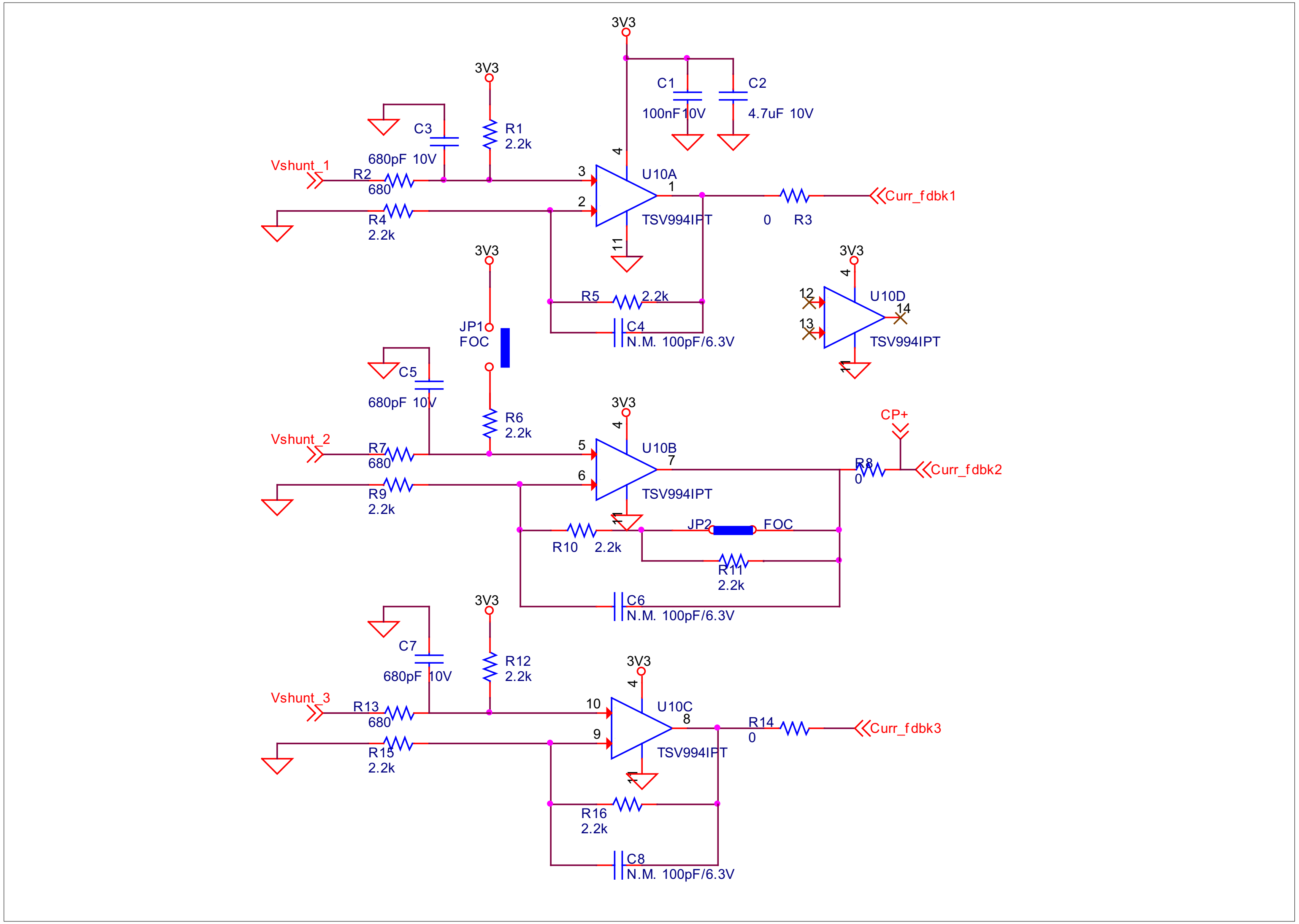


Figure 16. Sensors and shunt resistor circuit

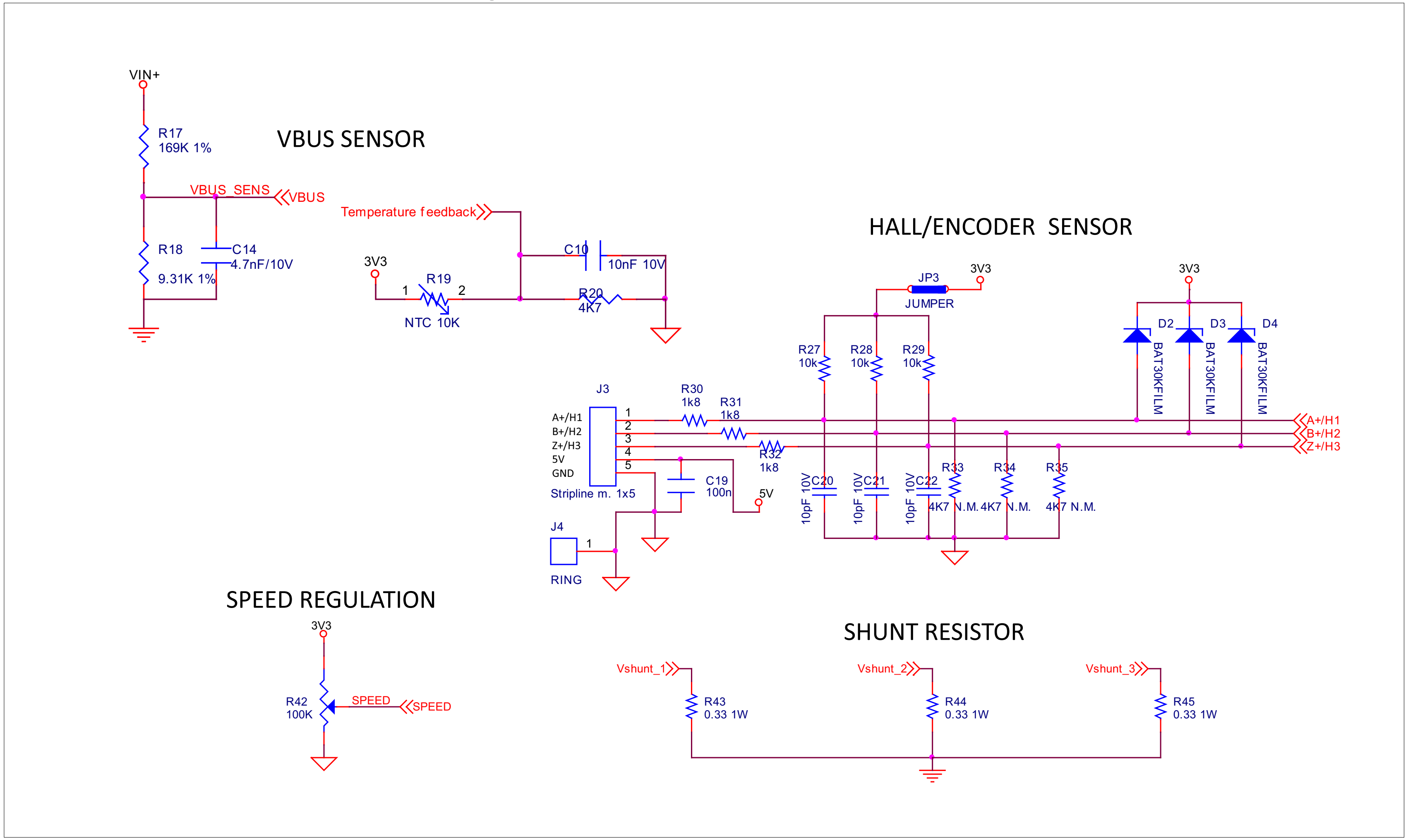


Figure 17. L6230 driver and BEMF detection circuit

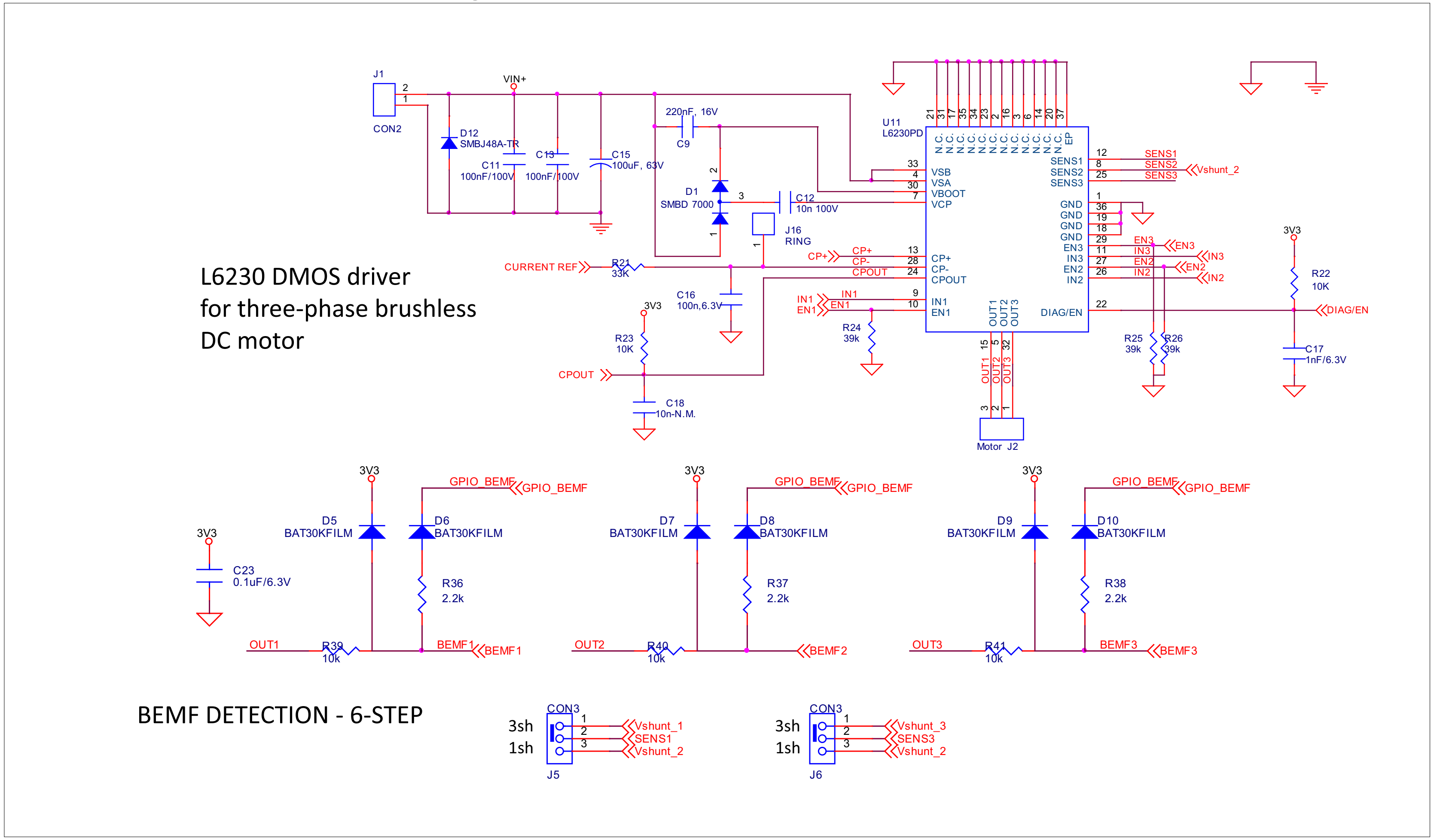
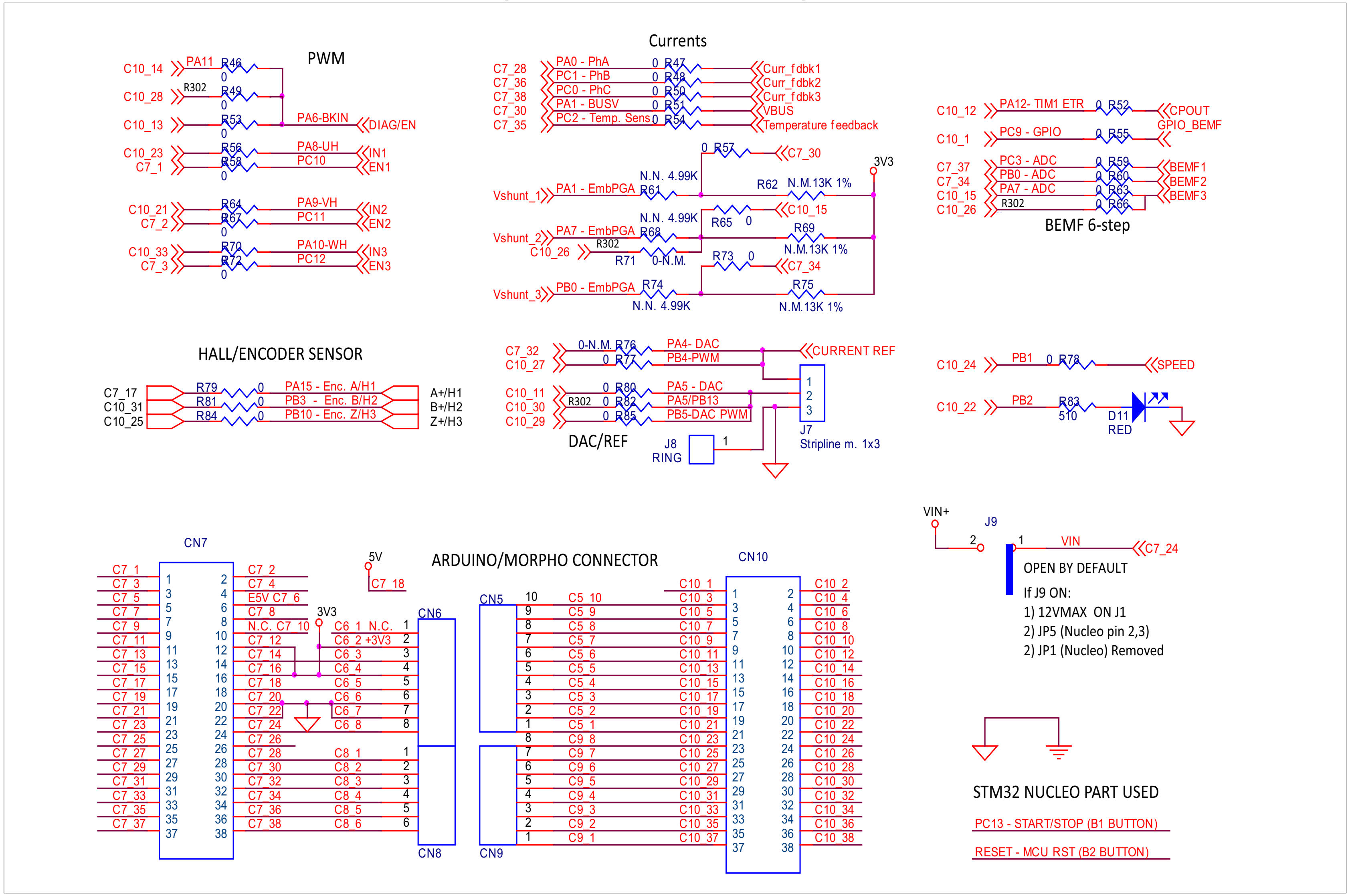


Figure 18. STM32 MCU pin assignment



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 5. Document revision history

Date	Revision	Changes
22-Sep-2015	1	Initial version
01-Sep-2016	2	Updated *Introduction*, *Section 3.1: System architecture*, *Section 3.2: Building and run the motor control for the STM32 Nucleo Packs* to introduce the power supply unit.