Embedded
Code Generation

*DEMO MODEL*

# Sensorless Field-Oriented Control of PMSM

## Using the IHM03 Motor Control Pack Hardware

Last updated in STM32 TSP 1.4.1

# 1 Overview

This STM32 demo model features a drive system with a three-phase brushless DC motor (BLDC), with sinusoidal back-emf. The following sections describe the implementation of the power stage and controls in detail. Below are the highlights of this demo model.

- It provides an explanation of the typical workflow of the PLECS Coder for embedded targets, using an STM32G4x or STM32F3x microcontroller (MCU) from STMicroelectronics.
- Shows the implementation of a rotor position and speed observer for sensorless field oriented control (FOC) strategy.
- Implements a closed-loop cascaded controller with inner d and q axis current controller and outer speed controller.
- The model is split into two distinct subsystems called “Plant” and “Controller”. This allows the same model to be used for –
  - Offline simulation in PLECS
  - Hardware-in-the-loop (HIL) testing with the PLECS RT Box
  - Rapid control prototyping (RCP) with the P-NUCLEO-IHM03 motor control pack from STMicro-electronics [3].

The following sections provide a description of the model and instructions on how to simulate the model, and deploy the control code to the STM32 target.

## 1.1 Requirements

In order to run this model you will need:

- PLECS Blockset or Standalone 4.7.4 or newer
- PLECS Coder
- STM32 Target Support Package 1.4.1 or newer
- RT Box Target Support Package

For RCP:

- 1 P-NUCLEO-IHM03 STM32 Motor Control Nucleo Pack

For HIL:

- 1 PLECS RT Box
- 1 NUCLEO-G431RB STM32 Nucleo board
- 1 RT Box NUCLEO Interface board

The Plant model can be executed on all versions of the RT Box. Alternatively, the controller can also be executed on a NUCLEO-G474RE or a NUCLEO-F303RE board.

---

**Note** This model contains model initialization commands that are accessible from:
*PLECS Standalone:* The menu **Simulation + Simulation Parameters... + Initializations**
*PLECS Blockset:* Right click in the **Simulink model window + Model Properties + Callbacks + InitFcn***

---

# 2 Model

The top level schematic contains two separate subsystems representing the controller and the plant models, as shown in Fig. 1. Both subsystems are enabled for code generation from the **Subsystem + Execution settings...** context menu. This step is necessary to generate the model code for a subsystem via the PLECS Coder.

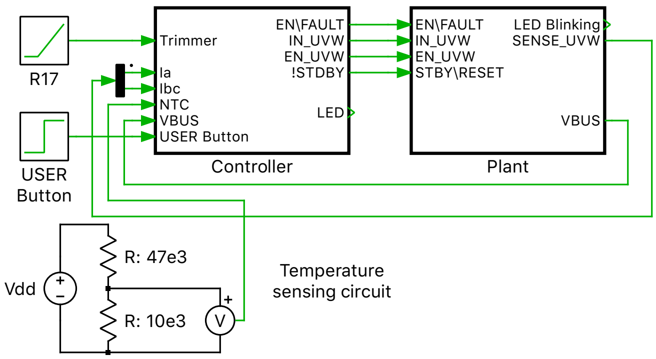


**Figure 1: Top level schematic of the demo model**

## 2.1 Plant

The power circuit, as shown in Fig. 2, includes a permanent magnet synchronous machine (PMSM) and a three-phase full bridge voltage source inverter (VSI), and is supplied by a DC source voltage ($V_i$) of $V_{dc}$ = 12 V. The VSI is composed of three MOSFET Half Bridge power module components.

The subsystem labeled "STSPIN830" models a basic functionality of the STSPIN830 three-phase motor driver [4].

The six pulse-width modulated (PWM) switching signals are obtained from the PWM Capture block of the PLECS RT Box library. Further details on the power module components and the sub-cycle averaging of PWM signals are described in [6]. The DC input voltage and the AC output stator current measurements are connected to Analog Out blocks from the PLECS RT Box library. The discretization step size of the plant subsystem is set to $3\,\mu s$.

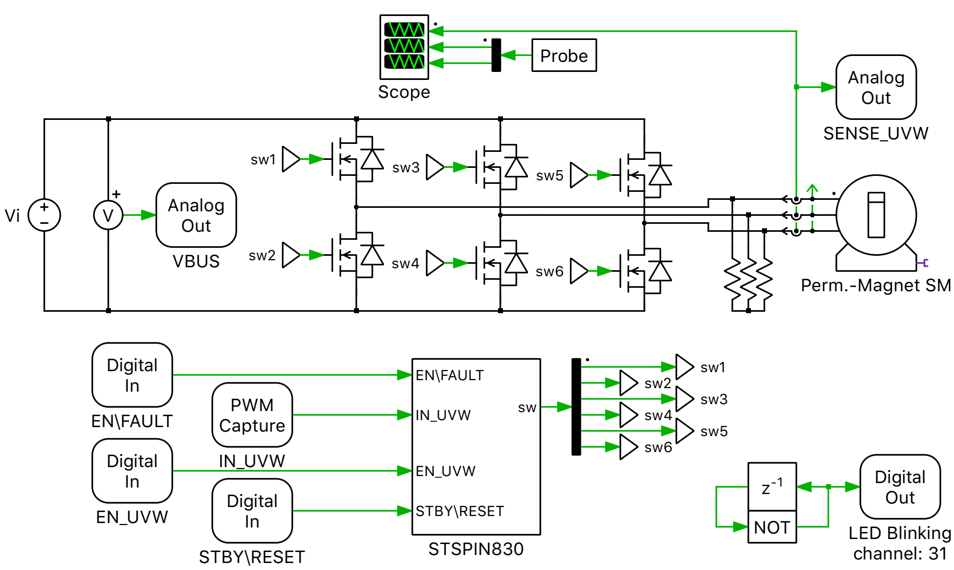


**Figure 2: Power circuit of the PMSM drive system**

### Machine Parameters

The machine parameters are listed in the table below, which were taken partly from the data estimated on the real hardware by using the motor profiler software provided by STMicroelectronics [5].

**Specifications of the GBM2804H-100T motor**

| | |
|---|---|
| Number of pole pairs | 7 |
| Maximum Torque (Nm) | 0.981 |
| Maximum Speed (rpm) | 2180 |
| Stator Resistance (Ω) | 4.7 |
| Stator Inductance (mH) | 0.96 |
| Inertia (nNms$^2$) | 497 |
| Friction (nNms) | 755 |
| Back EMF Voltage (V/krpm) | 4.85 |

## 2.2 Controller

In the controller subsystem, shown in Fig. 3, the three stator currents are measured by Analog In (Triggered) blocks from the STM32 Target component library. They represent injected conversions and the start of conversion is triggered by an external event, in this case a PWM underflow event. The DC-Bus voltage is monitored by using a regular ADC channel. A regular ADC channel is continuously converted in the background without any intervention of the CPU. The same type of ADC is used to sense the temperature on the PCB and the speed reference provided by the trimmer R17 mounted on the X-NUCLEO-IHM16M1 board.

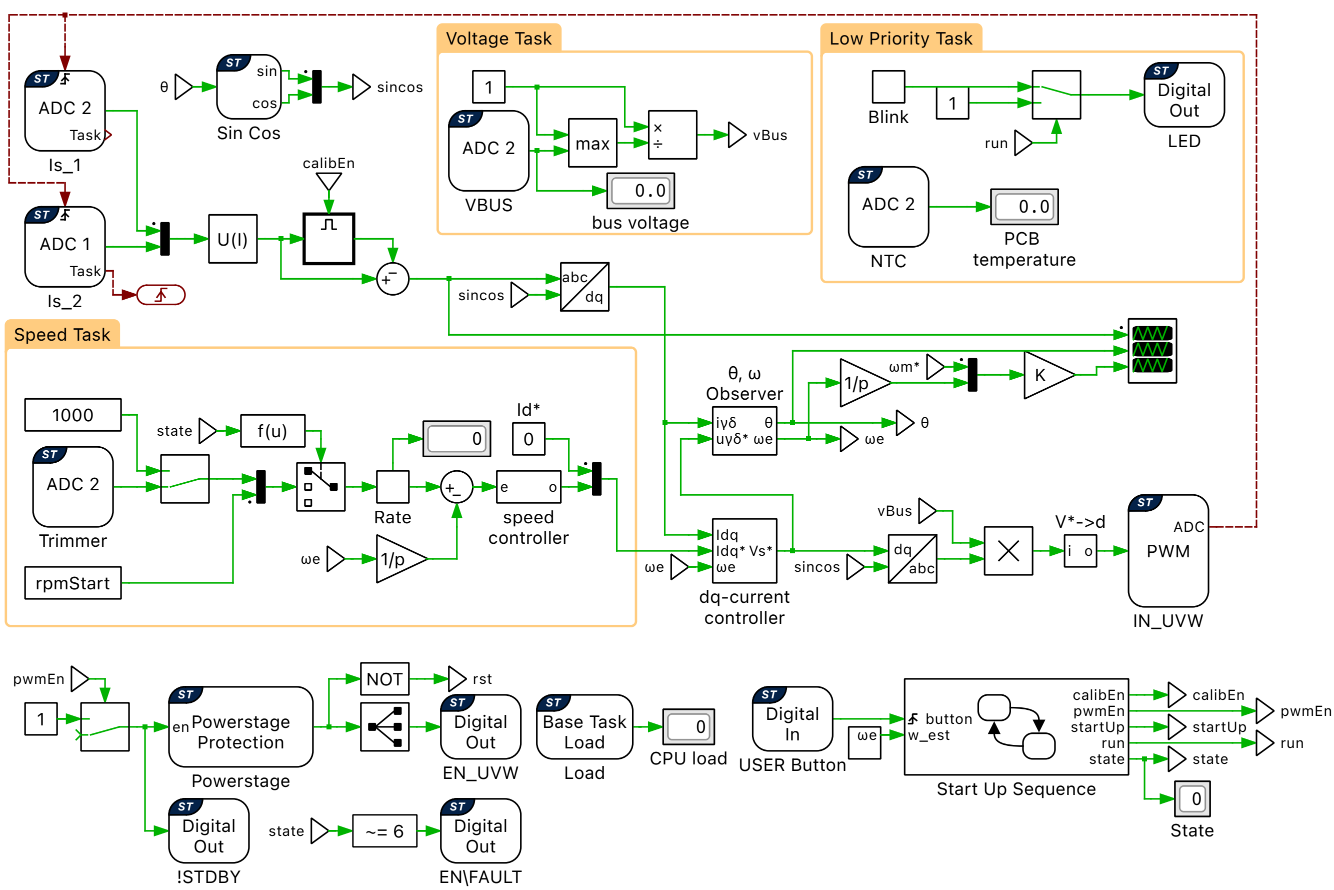


**Figure 3: Controller model of the sensorless rotor-field oriented control**

Rotor-field oriented control is applied to the drive system and the basic structure is shown in Fig 3. The stator currents are regulated in the dq rotating frame. The speed and rotor position are calculated by the subsystem labeled “Observer”. An outer control loop, which is a speed controller, is implemented, where a speed reference in RPM is manually set either via a constant block, or via the trimmer R17 on the X-NUCLEO-IHM16M1 board. The speed reference via the trimmer can be varied between 200 and 1500 RPM. Rotating the trimmer in a clockwise direction decreases the speed reference.

**Note** The complete controller can be enabled or disabled via the blue “User” push-button on the NUCLEO board.

Voltage control variables in the dq rotating frame are derived from Eq. 1, when $L_{\mathrm{sd}} = L_{\mathrm{sq}}$.

$$\begin{bmatrix} u_{\mathrm{d}} \\ u_{\mathrm{q}} \end{bmatrix} = \begin{bmatrix} R_{\mathrm{s}} & -\omega L_{\mathrm{sq}} \\ \omega L_{\mathrm{sd}} & R_{\mathrm{s}} \end{bmatrix} \cdot \begin{bmatrix} i_{\mathrm{d}} \\ i_{\mathrm{q}} \end{bmatrix} + \begin{bmatrix} L_{\mathrm{sd}} & 0 \\ 0 & L_{\mathrm{sq}} \end{bmatrix} \cdot \begin{bmatrix} \frac{d}{dt} i_{\mathrm{d}} \\ \frac{d}{dt} i_{\mathrm{q}} \end{bmatrix} + \begin{bmatrix} 0 \\ \omega \psi_{\mathrm{f}} \end{bmatrix} \tag{1}$$

## PI current controller

The d and q axis currents are controlled by two separate PI controllers, and are included in the subsystem labeled “dq-current controller”, shown in Fig. 4. The proportional and integral gains are designed to achieve a given bandwidth and phase-margin.

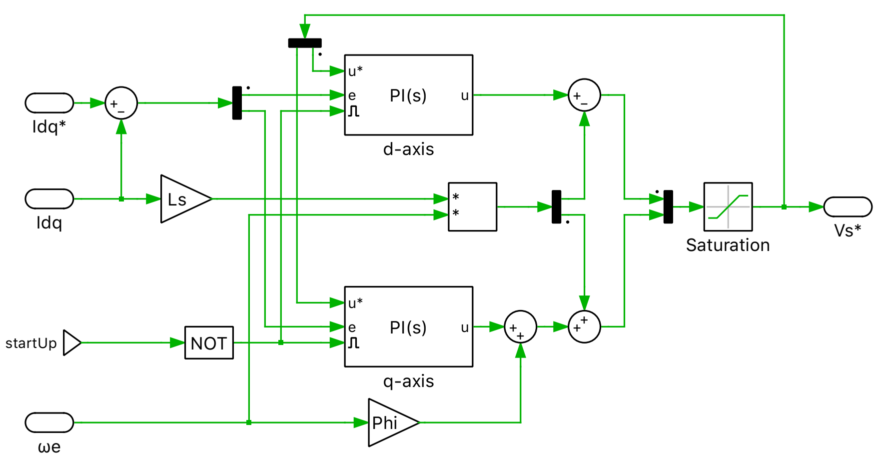


**Figure 4: PI controller in dq frame**

## Rotor position and speed observer

The subsystem labeled “Observer” is shown in Fig. 5. This is developed according to [1], and is conceptually explained below. This observer can be utilized by any type of synchronous motor such as a surface PMSM (non-salient $L_{\mathrm{d}} = L_{\mathrm{q}}$), interior PMSM (with saliency $L_{\mathrm{d}} < L_{\mathrm{q}}$) and synchronous reluctance motor.

In the observer model as seen in Fig. 5, the estimated rotating $\gamma\delta$ frame is used, which differs from the dq reference frame with respect to the position error $\theta_{\mathrm{e}}$. The position error $\theta_{\mathrm{e}}$ is obtained from the extended electromotive force (EMF), estimated by the observer. This position error $\theta_{\mathrm{e}}$ is used to estimate the rotor position and speed.

Fig. 6 shows the conceptual block diagram of the least-order observer for estimating $e_{\gamma}$; $g$ is the observer gain, and is set to $600$. The $\delta$-axis component of $e_{\delta}$ is estimated in the same way as $e_{\gamma}$. The blue dotted frame in Fig. 6 outlines how the observer is embedded in the overall model.

All integrators in the observer can be reset by a reset signal.

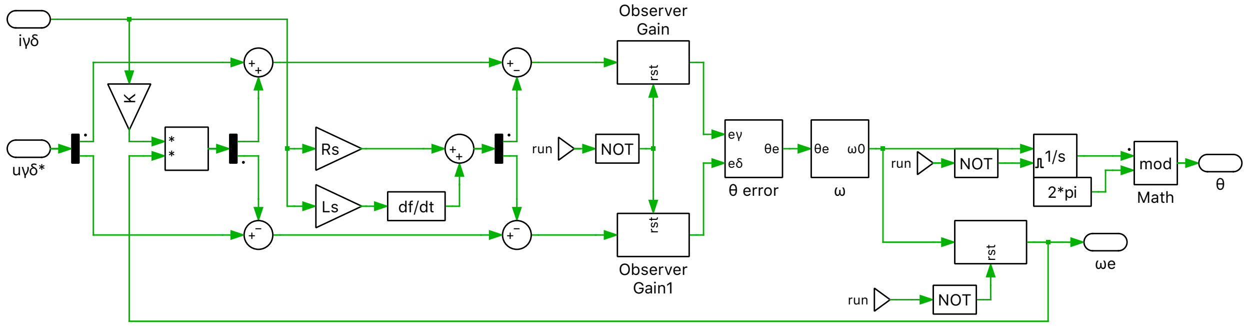


**Figure 5: Schematic of the rotor position and the speed observer**

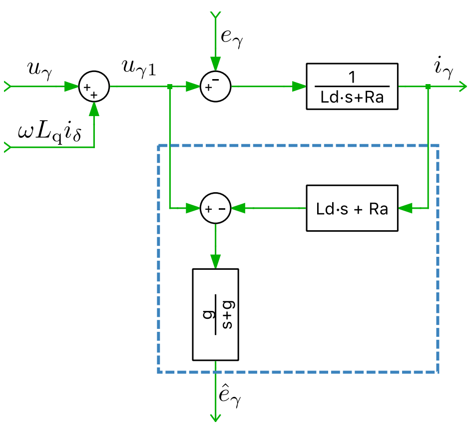


**Figure 6: Equivalent block diagram of the least-order observer**

## Speed Control

An *outer* speed control loop controls the drive speed. The time constants of the *outer* control loop is dictated by the mechanical time constant, which is larger in comparison to the *inner* current control loop.

The proportional-integral speed controller can be described by Eq. 2.

$$\frac{T_e^*}{\Delta\omega_m} = \left(K_p + \frac{K_i}{s}\right) \tag{2}$$

where, $T_{e^*}$ is the reference torque, $\Delta\omega_m = \omega_m^* - \omega_m$, $K_p$ is the proportional gain and $K_i$ is the integral gain.

The manipulated variable of the speed controller, which is the reference torque, $T_e^*$, is the setpoint for the *inner* current controller loop. Therefore, a conversion from mechanical torque, $T_e^*$, to current setpoint, $i_q^*$, is done using Eq. 3.

$$T_e^* = \frac{3}{2}\Psi_m i_q^* p \tag{3}$$

where, $\Psi_m$ is the flux linkage of the permanent magnets and $p$ is the number of rotor permanent magnet pole pairs.

## Multi-tasking code

The PLECS Coder and the STM32 Target Support Package allow the user to generate multi-tasking code for the STM32 family of MCUs. Multi-tasking code unlocks processing power for controls regulating multiple system outputs with dynamics on a range of time-scales. In this model there is a cascaded control scheme with a fast inner current control loop, and a slower outer speed loop. Multi-tasking code is well suited for this type of control scheme.

Multi-tasking code generation is configured in the **Scheduling** tab of the **Coder + Coder options...** dialog. By changing the **Tasking mode** to `multi-tasking` and the **Task configuration** to `specify`, the sample time for each task can be configured. The base sample time is always equal to the **Discretization step size**. The **Sample time** setting for lower priority tasks must be an integer multiple of the base sample time. Up to 15 slower lower priority tasks that execute at different rates can be specified, preserving processor time for the fastest, highest priority task in the application. For further information, refer to the "Code Generation" section in the PLECS User Manual [7].

Blocks in the PLECS schematic are assigned to lower priority tasks using the Task library component. In this model three lower priority tasks are defined in addition to the Base Task, as seen in Fig. 3. The table below lists the sample times of the lower priority tasks, with respect to the Base Task. The execution rate of the Base Task is 10 kHz, which is half the switching frequency (fsw).

| Task name | Sample time |
|---|---|
| Base Task | 1/fsw |
| Voltage Task | 10/fsw |
| Speed Task | 20/fsw |
| Low Priority Task | 10000/fsw |

### Start-up Sequence

A state-machine is implemented to handle the start-up process of the drive. The state-machine waits in the Idle state, until the blue “User” push-button on the NUCLEO-board is pushed. Once this button is pushed, it triggers the transition to the OffsetCalib, or offset calibration state. In this state, the offset due to the shunt current measurements is calibrated during a fixed period of time. After this calibration time is expired, the start-up is initiated. When the the blue “User” push-button is pushed again for a second time, the state machine goes back to the Idle state and the PWM signals are disabled.

# 3 Simulation

## 3.1 PLECS Offline Simulation Results

Run the model as provided from **Simulation + Start**, to observe the results from the offline simulation. Fig. 7 shows the results from the Scope in the “Controller” subsystem.

At 0.05 s, the start-up process is initiated. First, the static measurement offset of the current ADC channels is removed. At 0.05 s the current reference is ramped up. Once the motor position is aligned, the motor starts spinning with a speed reference of 500 RPM. When the estimated speed reaches this speed reference, the motor starts tracking the final speed reference value of 1200 RPM provided by a constant block. The rate of change of the speed reference is limited to 2000 RPM/s.

## 3.2 Configuring the STM32 Target

In addition to running a simulation of this demo model in offline mode on a computer, the “Controller” subsystem can be directly converted into target specific code for the STM32G431RB, STM32G474RE or STM32F303RE NUCLEO board. The model is configured by default for a STM32G431RB NUCELO board, but other targets are supported. Choose the desired **Target** in the **Coder Options** window. The I/O configuration are automatically adapted according to the selected target. This is accomplished in the **Model initialization commands** window of **Simulation Parameters... + Initializations** tab

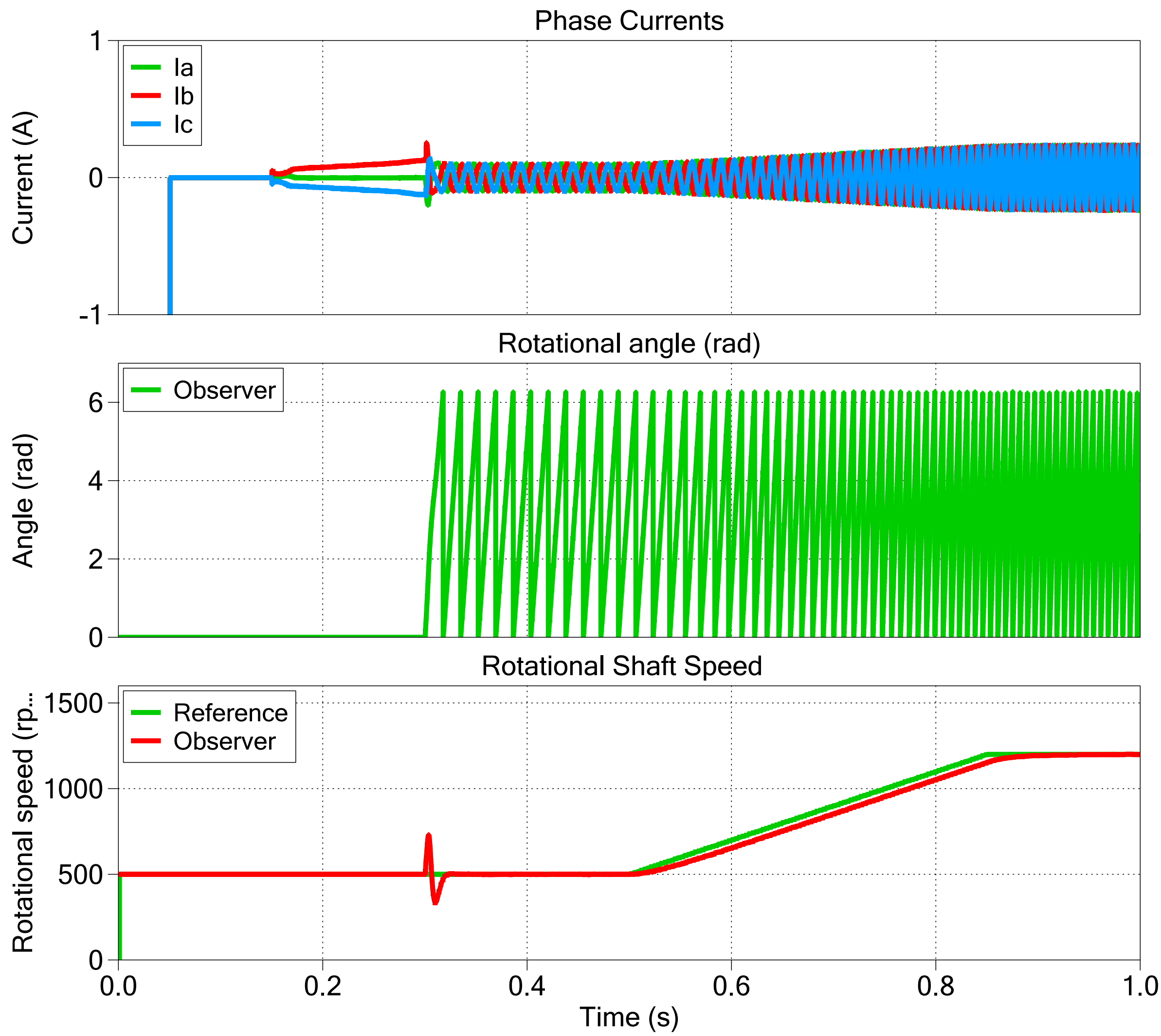


**Figure 7: Offline simulation result of the PMSM starting-up with a speed reference of 500 RPM and ramping up to 1200 RPM at 0.5 s**

from the **Simulation** menu. Optionally, the "Plant" subsystem can be deployed on the PLECS RT Box to perform a hardware-in-the-loop (HIL) test of the generated code. The controller is also configured to work with the NUCLEO-IHM03 experimental kit offered by ST.

Follow the instructions below to upload the "Controller" subsystem to a STM32 MCU.

- Connect the MCU to the host computer through a USB cable.
- From the **System** tab of the **Coder + Coder options...** window, select "Controller".
- Next, from the **Target** tab, select `STM32G4x` or `STMF3x` from the dropdown menu. Then under the **General** sub-tab, select `G431RB`, `G474RE` or `F303RE`.
- To deploy the MCU target directly from PLECS, uncheck the `Generate code only` parameter, choose the desired **Programming interface** from the dropdown menu. The default programming interface is `OpenOCD`.
- Then click **Build**.

If programmed correctly, the green LED "LD2" on the NUCLEO board should blink.

**Note** Verify the jumper configuration on the NUCLEO board: JP5 on 5V_STLK, JP8 in position [1–2], JP6 closed.

## 3.3 Configuring the PLECS RT Box

Prior to controlling a real power stage with the programmed MCU, it is highly recommended to first verify the behavior of the controller using a PLECS RT Box and perform a HIL test. A typical hardware configuration is shown in Fig. 8, where the NUCLEO board (the light blue board) is connected to the RT Box via an RT Box NUCLEO Interface board (the green board).

Follow the instructions below to run a real-time model on the RT Box. The “Plant” subsystem is executable on all the RT Box platforms.

- From the **System** tab of the **Coder + Coder options...** window, select “Plant”. Click the **Target** tab and select a target device. Then click **Build** to deploy the model to the target RT Box. Since the RT box pinning depends on the microcontroller, make sure the microcontroller target in the Controller system is selected and saved first before generating the code for the RT Box Plant system.
- Once the model is uploaded, from the **External Mode** tab of the **Coder options...** window, **Connect** to the RT Box and **Activate autotriggering** to observe the test results in real time.

If programmed correctly, the LED corresponding to “DO-31” of the RT Box NUCLEO Interface board should blink.


**Figure 8: Hardware setup for the HIL verification**

## 3.4 Executing a Closed-Loop HIL Test

Enable the MCU by pushing the blue “User” push-button on the NUCLEO board. This initiates the start-up sequence and PWM signal generation. If the start-up is successful, the speed controller will automatically track the speed reference and the green LED “LD2” on the NUCLEO board turns solid ON. Observe the real-time waveforms in the Scope of the “Plant” subsystem. From the **Model initialization commands** window of **Simulation Parameters... + Initializations** tab from the **Simulation** menu, change the value of `application` to “rt_box”. Make sure that the speed reference source in the “Controller” subsystem is connected to the constant block. The speed reference value of the constant block can be changed on the fly, in real-time, since the component has been added to the “Exceptions” list found in the **Parameter Inlining** tab of the **Coder options...** window, prior to building the model. To observe any intermediate values calculated on the MCU and to change the speed reference in the constant block, follow the instructions below to connect to the external mode of the STM32 MCU.

- First, **Disconnect** the “Plant” subsystem from the **External Mode** of the PLECS RT Box, if connected.
- From the **System** menu on the left hand side of the **Coder + Coder options...** window, select “Controller”.
- Next, from the **External Mode** tab, select the appropriate **Target device** and click **Connect**.
- Then, **Activate autotriggering** to observe the test results in the “Controller” subsystem Scope.

Then, connect to the external mode of the RT Box again following the instructions provided in Section 3.3.

To stop the motor from spinning, push the blue “User” push-button on the NUCLEO board once again. This will stop the PWM signal generation and the controller will enter an idle state, waiting for the next start-up sequence to be initiated by the user.

## 3.5 Executing a Closed-Loop RCP Test

Finally, the generated control code running on the NUCLEO board can also be used to spin the motor provided with the NUCLEO-IHM03 experimental kit offered by ST. From the **Model initialization commands** window of **Simulation Parameters... + Initializations** tab from the **Simulation** menu, change the value of `application` to “rcp”. Follow the instructions below to setup the hardware to test the generated control code with the NUCLEO motor-control pack.

- The X-NUCLEO-IHM16M1 must be stacked on the NUCLEO board through the CN7 and CN10 ST morpho connectors. Mind the direction of placement of the IHM16M1 board. The two buttons on the NUCLEO board must still be accessible.
- Connect the three motor wires on the CN1 connector.
- Check the jumper settings on the X-NUCLEO-IHM16M1 board: J5 and J6 closed, J2 closed on position [2–3] and J3 closed on position [1–2].
- Connect the 12 V DC power supply (use the power supply provided with the pack or an equivalent one) on CN1 or J4.

In the end, your setup should look similar to Fig. 9.

Then program your MCU via the PLECS Coder following the instructions given in Section 3.2. If programmed correctly, the green LED “LD2” on the NUCLEO board should blink.

Enable the MCU by pushing the blue “User” push-button on the NUCLEO board. This initiates the start-up sequence and PWM signal generation. If the start-up is successful, the speed controller will automatically track the speed reference and the green LED “LD2” on the NUCLEO board turns solid ON.

To observe any intermediate values calculated on the MCU and to change the speed reference source, follow the instructions below to connect to the external mode of the STM32 MCU.

- From the **System** menu on the left hand side of the **Coder + Coder options...** window, select “Controller”.
- Next, from the **External Mode** tab, select the appropriate **Target device** and click **Connect**.
- Then, **Activate autotriggering** to observe the test results in the “Controller” subsystem Scope.

Rotate the potentiometer on the X-NUCLEO-IHM16M1 board to change the motor speed reference. For this to work, the speed reference source in the “Controller” subsystem must be connected to the ADC “Trimmer” signal.

To stop the motor from spinning, push the blue “User” push-button on the NUCLEO board once again. This will stop the PWM signal generation and the controller will enter an idle state, waiting for the next start-up sequence to be initiated by the user.

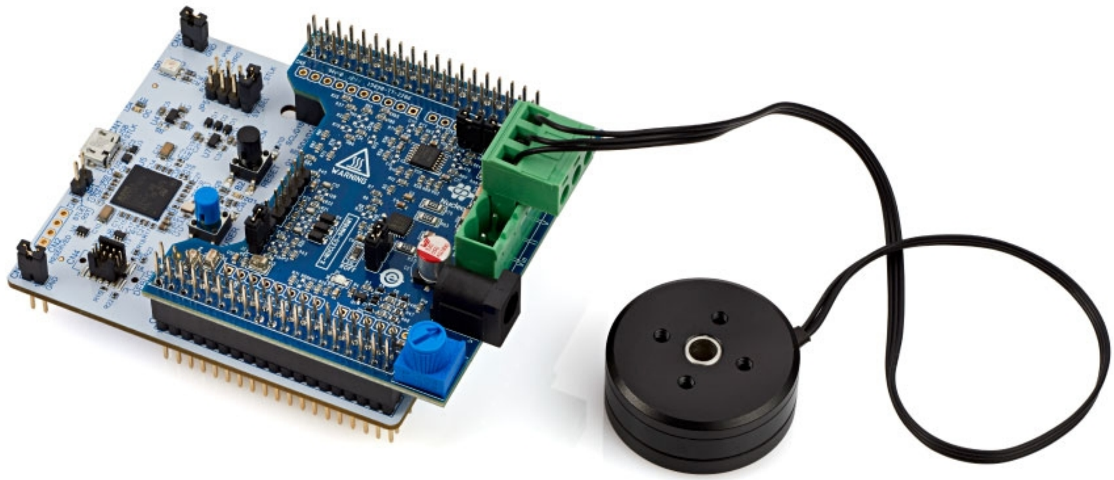

**Figure 9: Hardware setup for the RCP verification**

# 4 Conclusion

This model demonstrated the sensorless FOC control application of a PMSM drive system using the NUCLEO-IHM03 experimental kit. It provided three modes of simulation with step-by-step instructions.

- Offline simulation in PLECS
- Hardware-in-the-loop (HIL) testing with the PLECS RT Box
- Rapid control prototyping (RCP) with the P-NUCLEO-IHM03 motor control pack from STMicroelectronics [3]

# References


[1] S. Morimoto, K. Kawamoto, M. Sanada and Y. Takeda, "Sensorless control strategy for salient-pole PMSM based on extended EMF in rotating reference frame", IEEE Trans. Ind. Appl., vol. 38, no. 4, pp. 1054-1061, Jul./Aug. 2002.

[2] R. De Doncker, D. Pulle and A. Veltman, “Advanced electrical drives”, Springer, 2011.

[3] Motor Control Nucleo Pack with NUCLEO-G431RB and X-NUCLEO-IHM16M1,
URL: https://www.st.com/en/evaluation-tools/p-nucleo-ihm03.html

[4] STSPIN830 Compact and versatile three-phase and three-sense BLDC motor driver,
URL: https://www.st.com/en/motor-drivers/stspin830.html

[5] STM32 Motor Control Software Development Kit (MCSDK),
URL: https://www.st.com/en/embedded-software/x-cube-mcsdk.html

[6] J. Allmeling, and N. Felderer, “Sub cycle average models with integrated diodes for real-time simulation of power converters,” IEEE Southern Power Electronics Conference (SPEC), 2017.

[7] PLECS User Manual,
URL: https://www.plexim.com/sites/default/files/plecsmanual.pdf.

## Revision History:

| | |
|---|---|
| STM32 TSP 1.0.1 | First release |
| STM32 TSP 1.4.1 | Initialization commands |

## How to Contact Plexim:


| | | |
|---|---|---|
| ☎ | +41 44 533 51 00 | Phone |
| | +41 44 533 51 01 | Fax |
| ✉ | Plexim GmbH<br>Technoparkstrasse 1<br>8005 Zurich<br>Switzerland | Mail |
| @ | `info@plexim.com` | Email |
| | `http://www.plexim.com` | Web |


*Embedded Code Generation Demo Model*