

**Embedded  
Code Generation**

***DEMO MODEL***

## **Vector Control of an Induction Machine**

**Control of an induction machine with embedded code generation for TI C2000 MCUs**

Last updated in C2000 TSP 1.6.1

[www.plexim.com](http://www.plexim.com)

- ▶ Request a PLECS and PLECS Coder trial license
- ▶ Get the latest TI C2000 and RT Box Target Support Package
- ▶ Check the PLECS, RT Box and TI C2000 TSP documentation

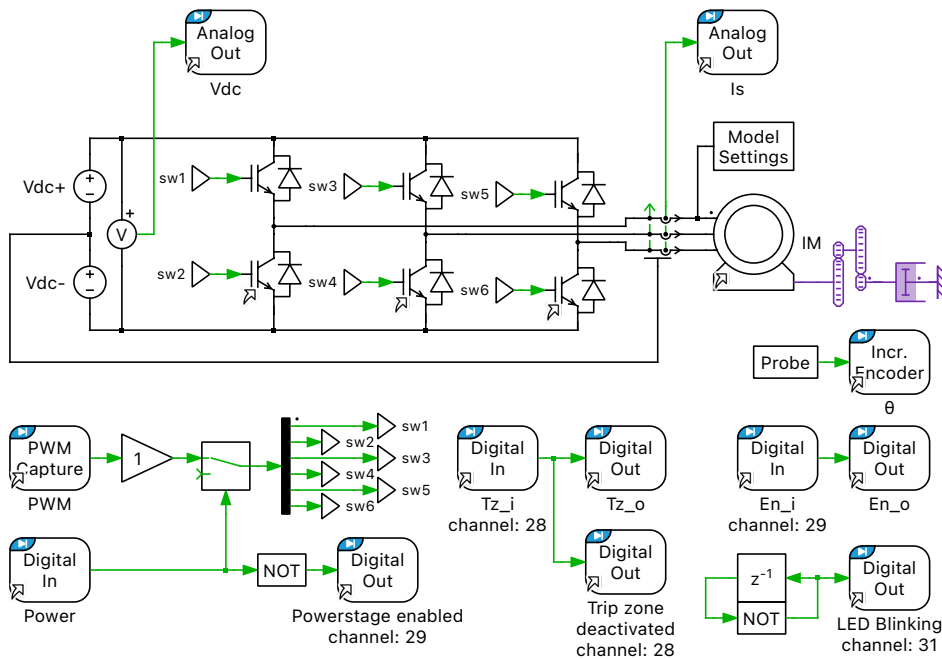
# 1 Overview

This demo model shows the simulation of an induction motor drive system, and provides an explanation of the typical workflow of the PLECS Embedded Coder, using Texas Instruments (TI) C2000 MCUs. Combined with a PLECS RT Box, the performance of the MCU can be verified directly.

**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\***



**Figure 1: Power circuit of the induction machine drive system**

## 2 Model

The demo model is composed of two subsystems: the “Plant” subsystem includes the power circuit and the “Controller” subsystem includes the control loop.

### 2.1 Power Circuit

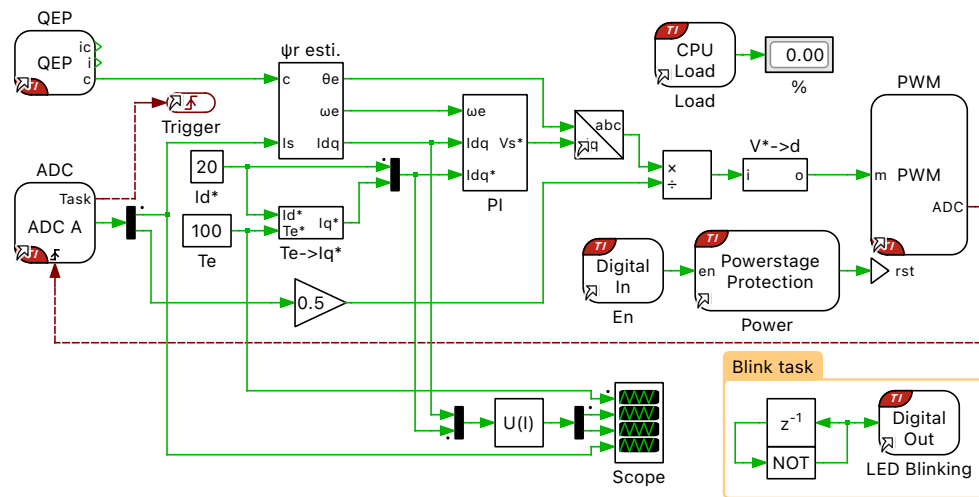
The power circuit includes an induction machine (IM) and a three-phase full bridge voltage-source inverter (VSI). The mechanical interface of the IM is loaded by a rotational damper block via a gear box to emulate the linear friction. The VSI is supplied by a DC voltage source of  $V_{dc} = 400\text{ V}$ , and is represented by three IGBT Half Bridge power modules.

The six pulse-width modulated (PWM) switching signals are brought into the subsystem using a PWM Capture block from the PLECS RT Box component library. The measurements of the DC voltage and the AC current are exported out of the subsystem via Analog Output ports. The rotor angular position and rotational speed are converted into digital orthogonal pulses by an Incremental Encoder block. The

encoder pulses can be accessed on the corresponding port of the “Plant” subsystem mask. Two Digital Output blocks forward the external digital input signals to two digital output pins of the RT Box. These pins should be connected with the GPIOs of the MCU to enable/disable the PWM outputs via software or trip zones. The mechanism of the MCU’s PWM enable/disable will be elaborated in Section 2.2.

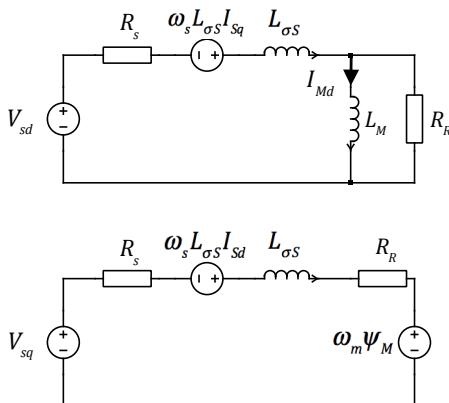
## 2.2 Controls

In the “Controller” subsystem, rotor-field oriented control is applied to the drive system. The basic structure is shown in Fig. 2, where the stator current is regulated in the dq frame.



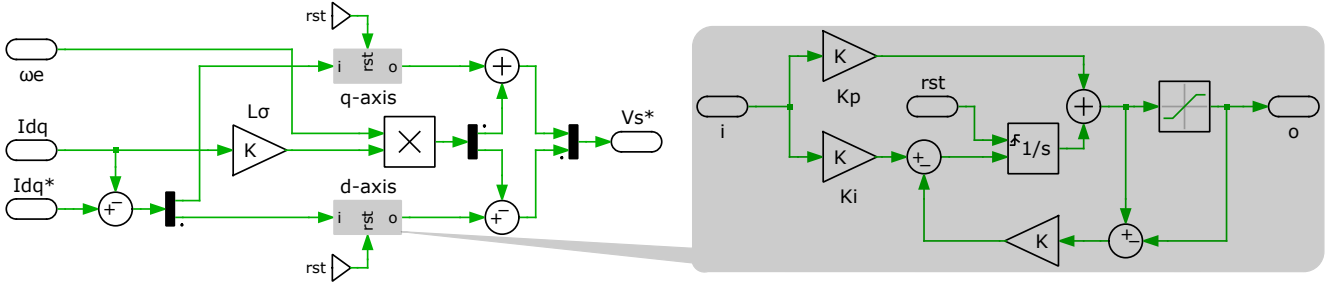
**Figure 2: Controller model of the induction machine drive system**

Fig. 3 shows the equivalent circuit of the induction machine in the dq frame, which rotates synchronously with the rotor flux. The values of  $L_M$ ,  $L_{\sigma S}$  and  $R_R$  are calculated from the original machine parameters, which can be found in the initialization commands of the model.



**Figure 3: Equivalent circuit of the induction machine in the dq frame**

The PI controllers for the d and q axis current are included in the subsystem “PI”, as shown in Fig. 4. The proportional and integral gains are designed following the “magnitude optimum” method [2][3].

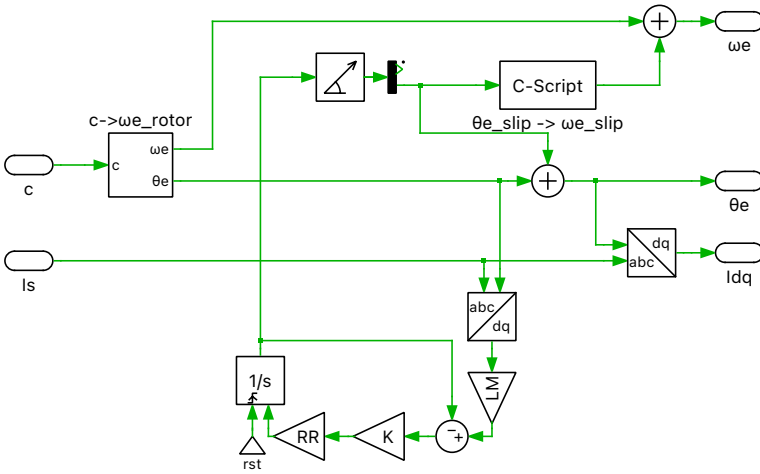


**Figure 4: PI controller in the dq frame**

To avoid the use of an embedded flux sensor, a magnetic flux estimator introduced on page 322 of [1] is employed in the subsystem “ $\Psi_r$  esti.”. Making use of the measured mechanical angular speed  $\omega_m$ , the stator current is transformed into the rotor rotational frame (“xy”) as  $\vec{I}_{s,xy}$ . The rotor flux  $d\vec{\Psi}_{r,xy}$  in the rotor rotational frame is governed by the differential equation given below:

$$\frac{d\vec{\Psi}_{r,xy}}{dt} = R_R \left( \frac{-d\vec{\Psi}_{r,xy}}{L_M} \right) + \vec{I}_{s,xy} \quad (1)$$

Following the differential equation,  $d\vec{\Psi}_{r,xy}$  can be calculated using  $\vec{I}_{s,xy}$  as an input. Using a “Rectangular to Polar” transformation block, we can process the “x” and “y” components of the rotor flux in the rotor rotational frame to yield the slip angular position. Summing up the slip angular position and the rotor electrical angular position, the rotational flux angle  $\theta$  is obtained.  $\theta$  is then used to transform the stator current from the “abc” frame to “dq” frame. The structure of the rotor flux estimator is demonstrated in Fig. 5.



**Figure 5: Structure of the rotor flux estimator**

### Configuring TI C2000 Target library components

The measurements of the DC-link voltage and the stator currents are introduced to the model environment from the ADC block of the TI C2000 Target component library. In order to convert the detected analog voltage into values with physical units to be used for the control algorithm, a scaling factor and an offset factor are provided to each channel, via the parameter window of the ADC block. The **ADC unit** and the **Analog input channel** parameters can be modified accordingly per available resources of different MCUs.

The mechanical angular speed of the rotor is obtained from the Quadrature Encoder Counter (QEP) block, which converts the orthogonal digital pulses into the rotational position-related counter value.

The **Maximum counter value** must be provided according to the configuration of the real encoder hardware, along with the index of the **GPIO numbers**.

The gate signals are generated by the PWM block. The **Carrier type**, **Carrier frequency** and **Dead time** parameters can be configured in an intuitive way from the **Main** tab of the PWM block parameters window. The input to the PWM block is obtained from the controller in the form of a duty cycle, in the range of [0,1]. Please note that each duty cycle signal is fed to both A and B outputs of an ePWM module, but with opposite polarity. From the **Events** tab of the PWM block parameter window, the **ADC Trigger** parameter is configured as Underflow. This makes the first ePWM module configured in the **PWM generator** of the **Main** tab generate a “start of conversion” impulse for the ADCs, whenever the carrier value reaches its minimum. With this configuration, an additional output port called “ADC” appears on the mask of the PWM block. This is connected to the input port called “Trigger” of the ADC block, via the red dashed signal wire.

The control task is executed after each ADC conversion. This is configured by connecting the output port called “Task” of the ADC block to a “Control Task Trigger” port from the TI C2000 Target component library.

The trip zone submodule is used to disable the powerstage and associated PWM blocks following a trip event. Trip events are detected when there is an active low condition on the trip zone GPIO inputs assigned in the **Protections** tab of the **Coder + Coder Options + Target** window. When a trip event is detected the Powerstage Protection module can take no action, activate a one-shot trip event, or activate a cycle-by-cycle trip event. In this model, the Powerstage Protection module will activate a one-shot trip event. This will latch the PWM output to the PWM safe state and can only be cleared by cycling the Powerstage Protection block from *disabled* to *enabled*.

### 3 Simulation

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 TI C2000 MCUs. The default I/O configuration of all the peripheral blocks (ADC, PWM etc.) supports the TI 280039C [4], TI 280049C [5], TI 28069 [6], TI 28379D [8], TI 28P650DK9 [9] LaunchPads, and the TI 28335 [11] and TI 28388D [13] controlCARD.

Additionally, the demo model allows for code generation for the TI 28377S [7] LaunchPad, as well as the TI 280039C [10] and TI 28379D [12] controlCARDS. To configure this, go to the **Model initialization commands** window from the **Simulation + Simulation Parameters... + Initializations** menu, and change the value of board\_type, to select the desired board. You must also configure the corresponding **Target** and **Board** type in the **Coder Options** window accordingly. Furthermore, the model of the plant can be deployed on the PLECS RT Box for a hardware-in-the-loop (HIL) test of the generated code. Follow the instructions below to upload the “Controller” subsystem to a TI 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 the appropriate target from the dropdown menu. Then under the **General** subtab, select the desired **Build type**.
- Then, to Build and program the MCU directly from PLECS, choose either Run from Flash or Run from RAM as the **Build configuration** to program the MCU either to flash memory or to RAM respectively, then select LaunchPad as the **Board** type, and click **Build**.

If programmed correctly, LED “D9” (or the LED corresponding to GPIO “DO\_DSP\_LED” listed in the model initialization commands) should blink.

For advanced users who are familiar with Code Composer Studio, there is an option to Generate code into CCS project. Locate the appropriate cg folder from the CCS project (refer to [14] for step-by-step instructions), enter its path into the **CCS project directory** field and click **Build**. The code of the “Controller” subsystem will be automatically generated. Then, proceed to build and debug the project as a normal CCS project.

**Note** If using the RT Box LaunchPad Interface board, make sure that the **RST** jumper is open throughout the simulation.

Prior to controlling a real hardware prototype of the 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 hardware-in-the-loop (HIL) test. A typical hardware configuration is shown in Fig. 6, where the evaluation kit, a TI 28069 LaunchPad (the red board), is connected to the RT Box via an RT Box LaunchPad Interface (the green board).



**Figure 6: Hardware setup of the HIL verification**

Follow the instructions below to run a real-time model on the RT Box. Before building on the RT Box, ensure that you have already built the "Controller" subsystem on the appropriate TI MCU.

- From the **System** tab of the **Coder + Coder options...** window, select "Plant" and **Build** it onto the RT Box.
- 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 LaunchPad Interface board should blink.

Within the "Plant" subsystem or the power circuit running on the RT Box, the simulated voltages and currents are proportionally converted into analog signals, and delivered through Analog Out connectors on the front panel of the RT Box. These analog signals are captured by the RT Box LaunchPad Interface board and routed to the ADC input pins of the TI LaunchPad. The MCU then processes these analog signals to generate PWM switching signals, which are delivered to the RT Box via the Digital In pins. Toggle the switch "DI-29" on the RT Box LaunchPad Interface board from "High" to "Low" and then back to "High" to reset the MCU and observe the real-time waveforms in the Scope of the "Plant" subsystem. When the power stage is enabled, the LED corresponding to "DO-29" of the LaunchPad interface board should turn on.

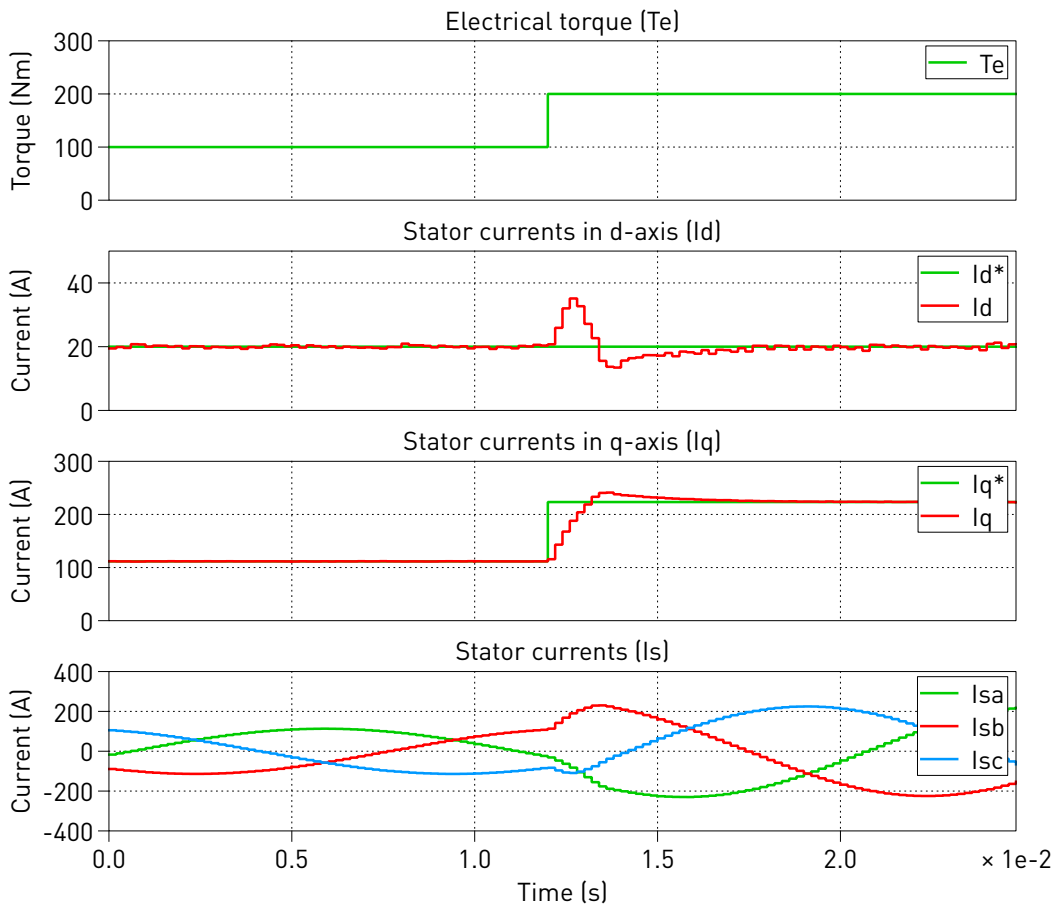
If switch "DI-28" on the RT Box LaunchPad Interface board is toggled to "Low", a trip-zone event is activated and disables the output of all the PWMs. When the trip-zone event is activated, the LED corresponding to "DO-28" of the LaunchPad interface board is turned off. In order to resume the system, toggle "DI-28" back to "High", turning the LED corresponding to "DO-28" on, then toggle switch "DI-29" from "High" to "Low", and then back to "High" to reset the MCU. The LED corresponding to "DO-29" should now turn on.

**Note** At this stage, verify that the LEDs corresponding to “DO-28” and “DO-29” on the RT Box LaunchPad Interface board are turned on.

In order to tune the parameters of the control program in the MCU and observe any intermediate values, follow the instructions below to connect to the external mode of the TI MCU.

- First, **Disconnect** the “Plant” subsystem from the **External Mode** of the PLECS RT Box, if connected.
- Then, from the **System** tab 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.

In this demo model, a step change to the reference torque can be configured by changing the value of the Constant block “Te”, on the fly, in real-time, since it has been added to the **Exceptions** list in the **Parameter Inlining** tab of the **Coder options...** window. The step response can be observed in real-time, as shown in Fig. 7, by setting up an appropriate **Trigger control** from the **External Mode** tab.



**Figure 7: Transient response to a step change of the torque reference in the controller in real-time**

## 4 Conclusion

This model demonstrates an induction machine drive system that supports embedded code generation for TI C2000 MCUs.

## References

- [1] R. De Doncker, D. Pulle and A. Veltman, “Advanced Electrical Drives”, Springer, 2011.
- [2] Konstantinos G. Papadopoulos, “PID Controller Tuning Using the Magnitude Optimum Criterion”, Springer, 2015.
- [3] Damir Vrancic, “Magnitude Optimum Techniques for PID Controllers”, 2012.
- [4] TI C2000 F280039C LaunchPad development kit,  
URL: <http://www.ti.com/tool/LAUNCHXL-F280039C>.
- [5] TI C2000 Piccolo MCU F280049C LaunchPad development kit,  
URL: <http://www.ti.com/tool/LAUNCHXL-F280049C>.
- [6] TI C2000 Piccolo MCU F28069M LaunchPad development kit,  
URL: <http://www.ti.com/tool/LAUNCHXL-F28069M>.
- [7] TI C2000 Delfino MCU F28377S LaunchPad development kit,  
URL: <https://www.ti.com/lit/pdf/sprui25>.
- [8] TI C2000 Delfino MCU F28379D LaunchPad development kit,  
URL: <http://www.ti.com/tool/LAUNCHXL-F28379D>.
- [9] TI C2000 F28P650DK9 LaunchPad development kit,  
URL: <https://www.ti.com/tool/LAUNCHXL-F28P65X>.
- [10] TI C2000 F280039C controlCARD evaluation module,  
URL: <https://www.ti.com/tool/TMDSCNCD280039C>.
- [11] TI C2000 TMS320F28335 controlCARD,  
URL: <http://www.ti.com/tool/TMDSCNCD28335>.
- [12] TI C2000 F28379D controlCARD development kit,  
URL: <https://www.ti.com/tool/TMDSCNCD28379D>.
- [13] TI C2000 F28388D controlCARD evaluation module,  
URL: <https://www.ti.com/tool/TMDSCNCD28388D>.
- [14] PLECS TI C2000 Target Support User Manual,  
URL: <https://www.plexim.com/download/documentation>.



## Revision History:

C2000 TSP 1.0	First release
C2000 TSP 1.3.1	Turned on Assertions in the three IGBT Half Bridge power modules
C2000 TSP 1.4.5	Updated the flux estimator block, powerstage protection parameters and web links
C2000 TSP 1.5.1	Added support for 28388D and 28379D controlCARD targets
C2000 TSP 1.6.1	Added support for 280039C LaunchPad and controlCARD targets, and auto-pin selection

## How to Contact Plexim:

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

### *Embedded Code Generation Demo Model*

© 2002–2023 by Plexim GmbH

The software PLECS described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from Plexim GmbH.

PLECS is a registered trademark of Plexim GmbH. MATLAB, Simulink and Simulink Coder are registered trademarks of The MathWorks, Inc. Other product or brand names are trademarks or registered trademarks of their respective holders.