BLDC Motor Operation with S32K144 EVB in MatLab Simulink Environment

Step 1

Go to mathworks.com and download MatLab version R2018b

Step 2

Ensure that S32 Design Studio has been downloaded, and your environment variables are set up. In addition, ensure that the **NXP_MBDToolbox_S32K1xx** version 4.0.0 has been installed, and is properly licensed.

These steps were given in lab 1.

Step 3

Get the following items from your instructor:

- S32K144 EVB

  Jumpers J107 should be set to 2 and 3

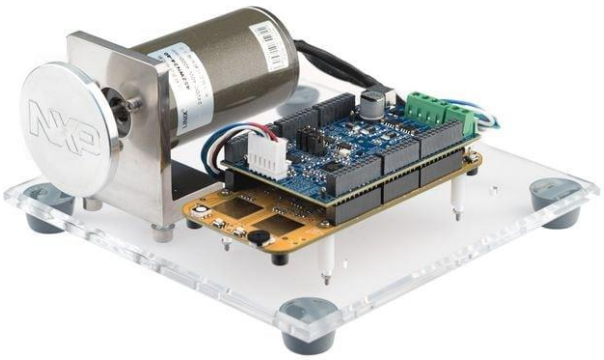
  Jumpers J104 should be set to 2 and 3

- Linix Motor

- Motor GD DevKit

Jumpers should not be altered from default settings

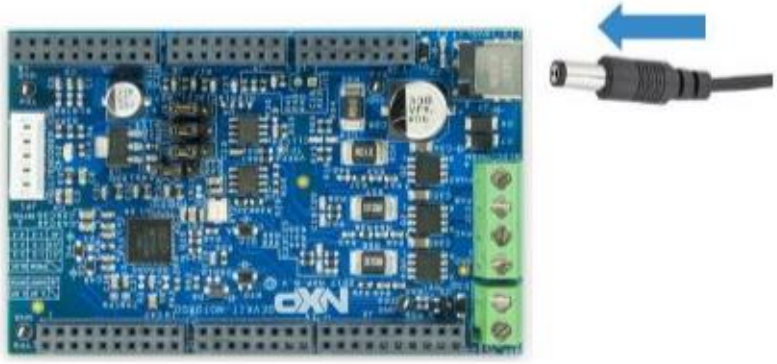
Complete System:



- PHA – White
- PHB – Blue
- PHC – Green

- Power on the DEVKIT-MOTOTORGD (not S332V234 EVB) via J7 connector. **USE 12 Volt DC.** Do not exceed 18 V DC. Do not use the 24 V DC.

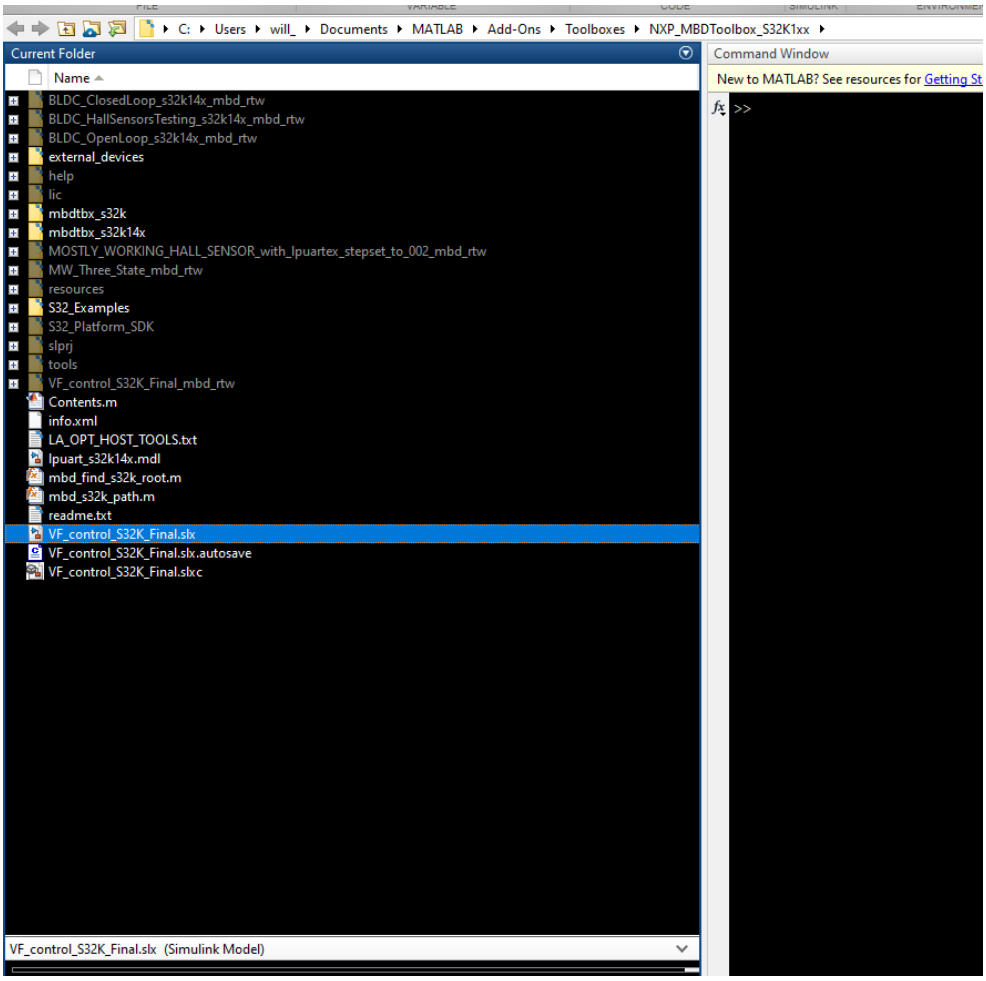
  The DEVKIT-MOTORGD also powers the S32K144 base board.



Step 4

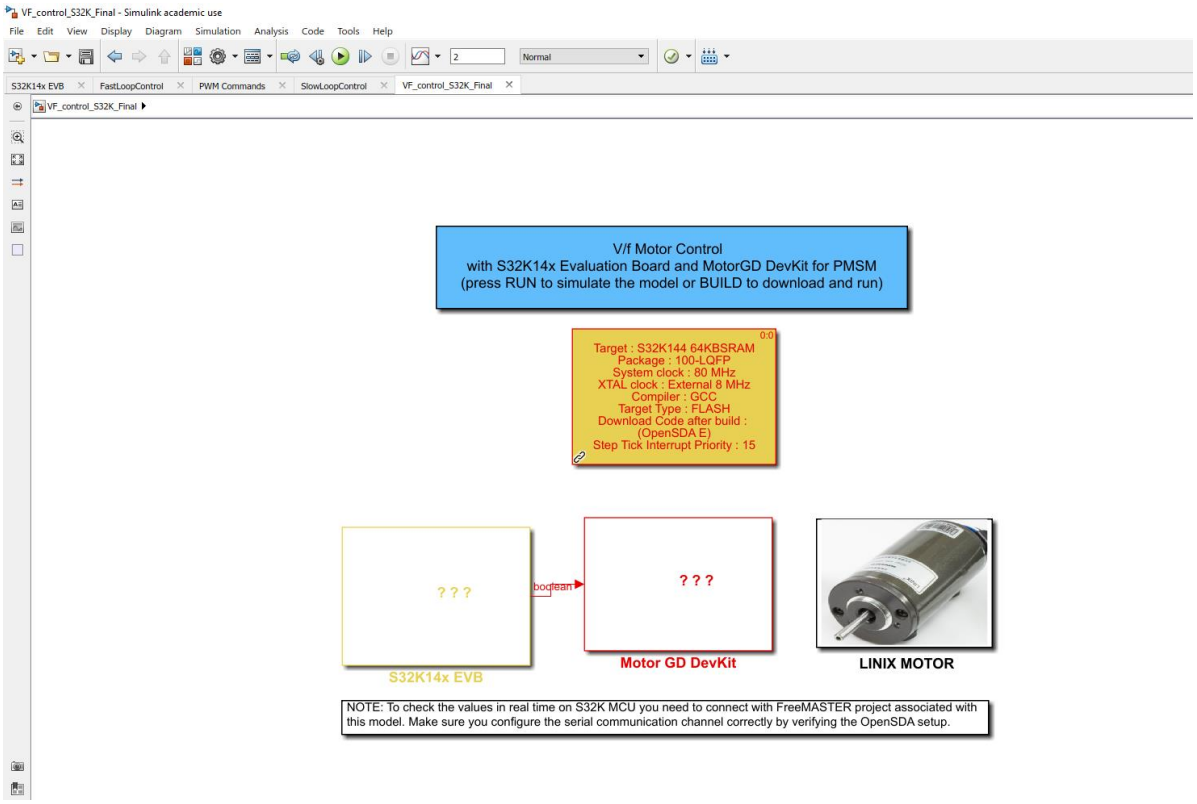
- Download vf_lab_7_software Zip file from canvas and place Vf_control_Final.slx into MatLab workspace:

C:\Users\Your_Name\Documents\MATLAB\Add-Ons\Toolboxes\NXP_MBDToolbox_S32K1xx

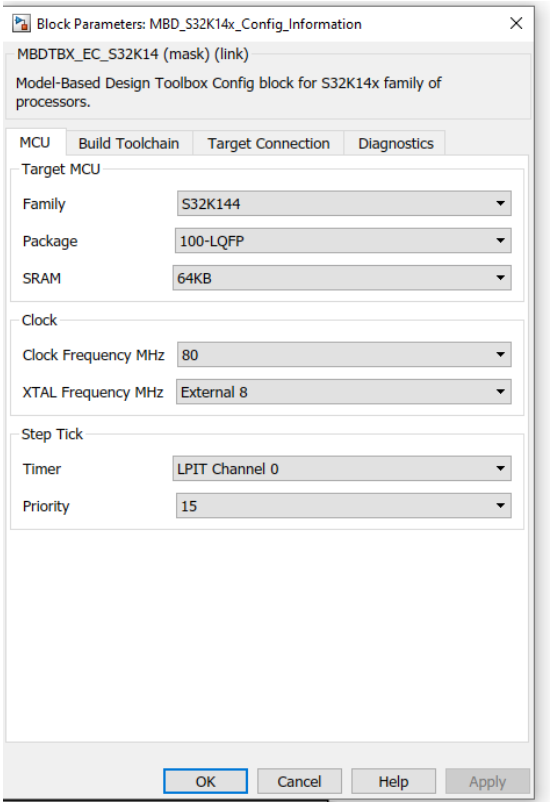


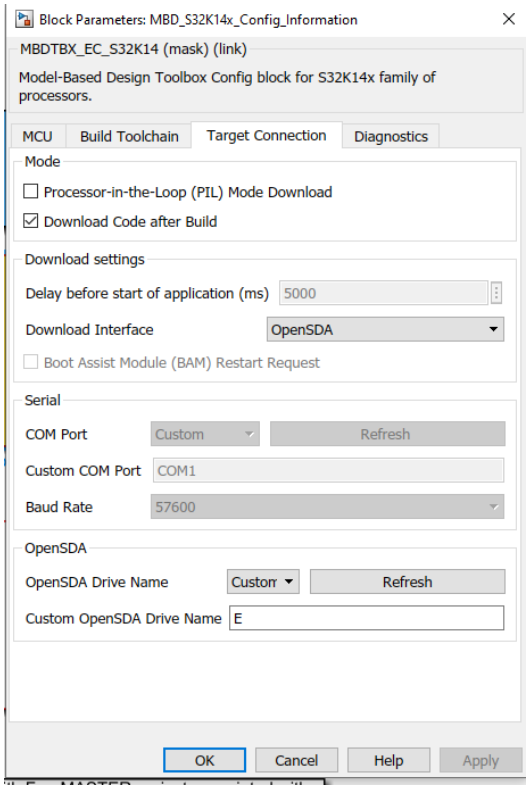
Step 5

- Double click on Vf_control_Final.slx in order to open the file in Simulink

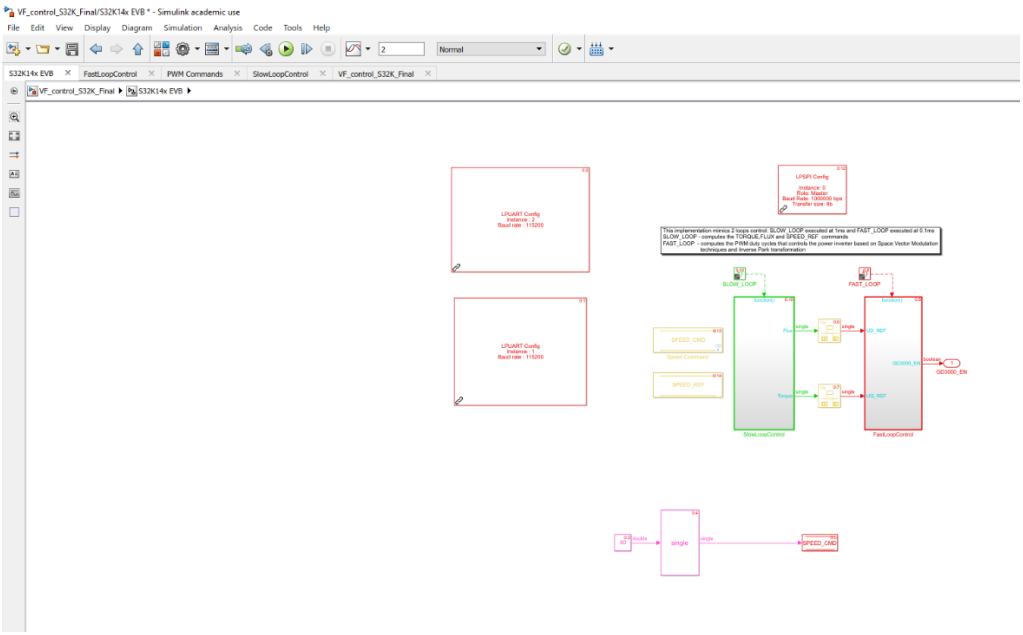


- Double click on target and ensure the following options are selected:

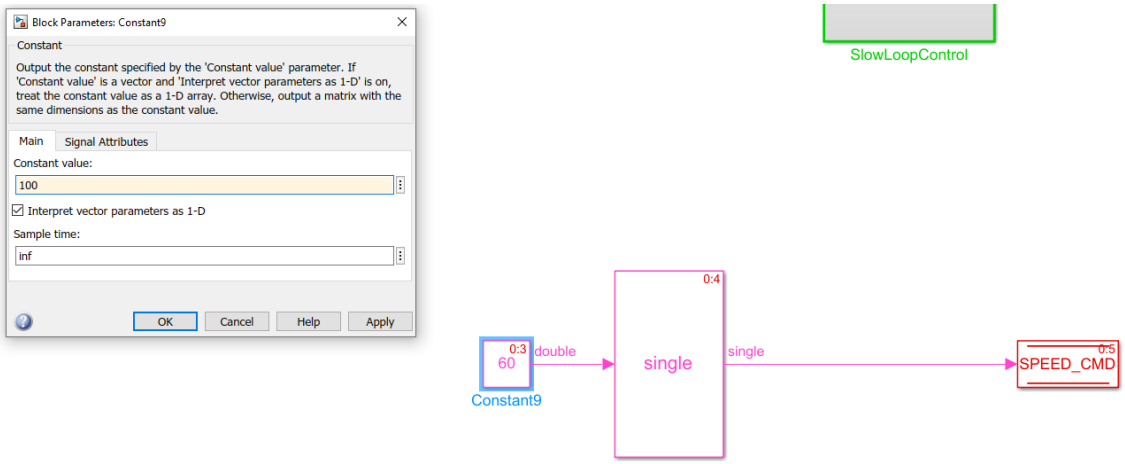




- Double click on S32k14x EVB

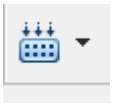


- The following page is displayed
- In order to change the reference speed of the motor, the constant can be changed by double clicking on the constant and altering its value like so:

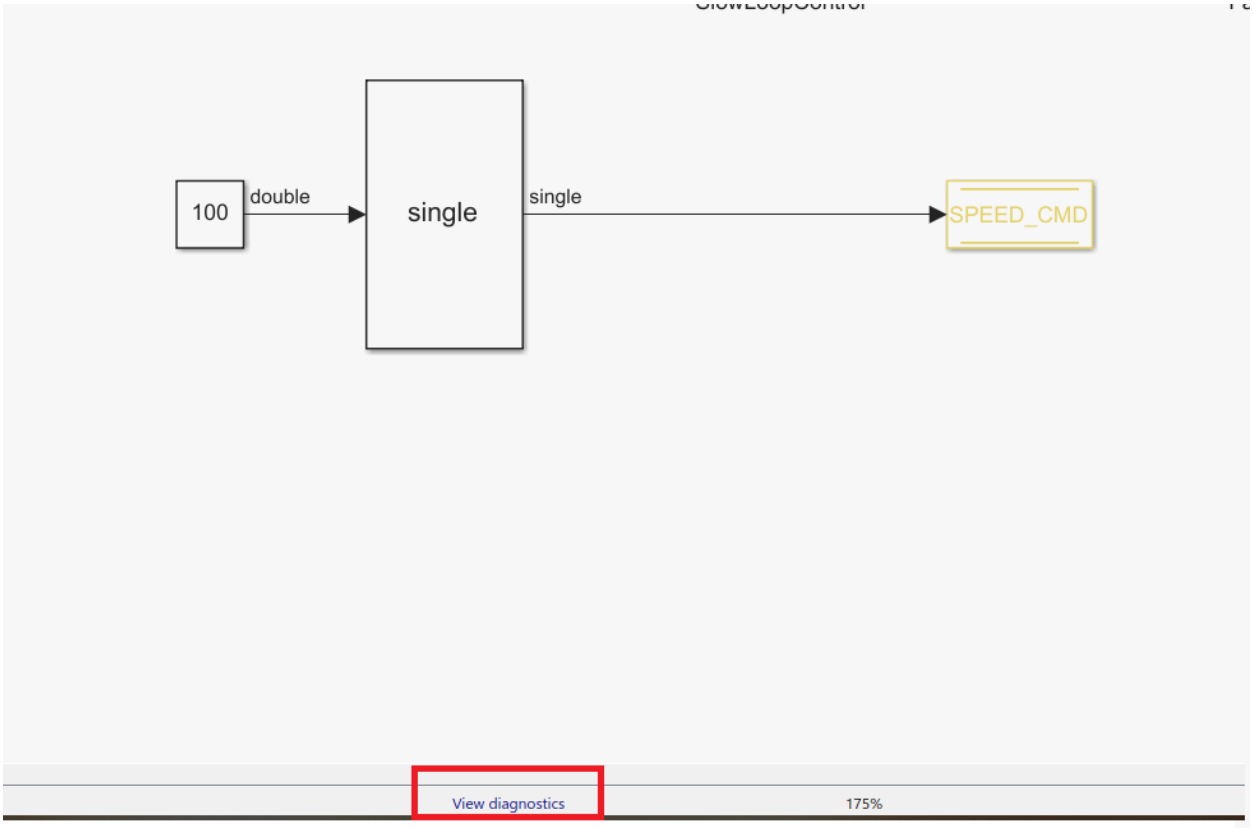


Step 6

- Click on the build model icon in order to send the code to the board

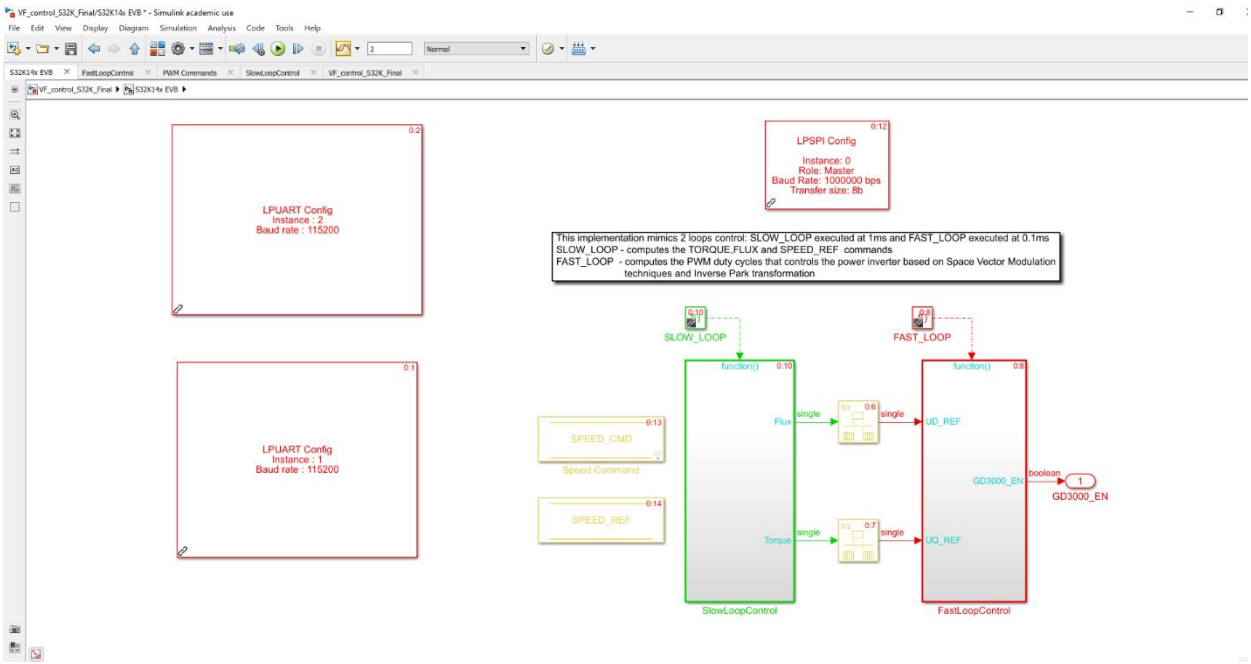


- Open Diagnostic viewer to see compilation process and logs

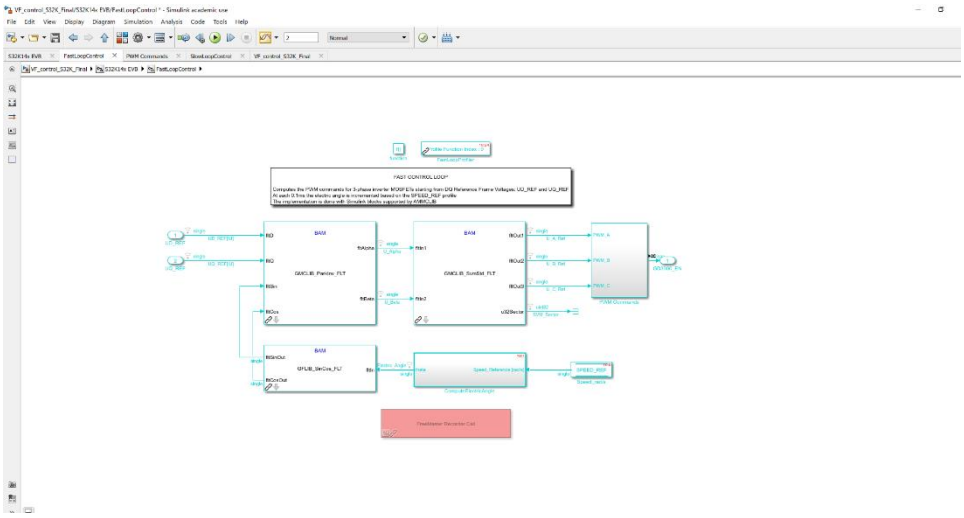


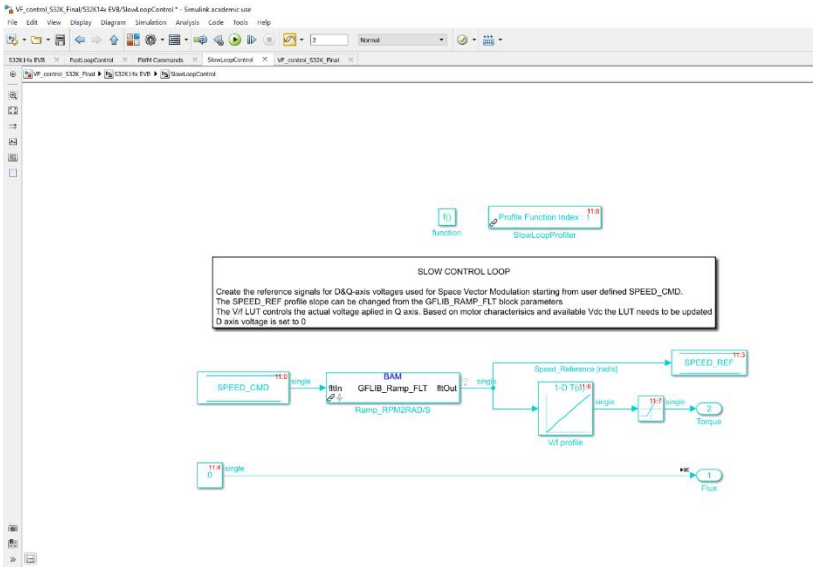
- Upon successful completion of compilation: reset the board, and verify it is spinning

- Available are modules for configuration of UART behavior, as well the behavior of the motor in response to commands



- The following represent the mathematical models used to control the motors operation. These modules can be freely configured per implementation.


FAST CONTROL LOOP
BAM
GMCLIB_ParkInv_FLT
GMCLIB_SvmStd_FLT
PWM Commands
GFLIB_SlwGne_FLT
SpeedRef_rad/s


Profile Function Index : 1
SlowLoopProfiler
SLOW CONTROL LOOP
Create the reference signals for D&Q-axis voltages used for Space Vector Modulation starting from user defined SPEED_CMD.
The SPEED_REF profile slope can be changed from the GFLIB_RAMP_FLT block parameters
The V/f LUT controls the actual voltage aplied in Q axis. Based on motor characterisics and available Vdc the LUT needs to be updated
D axis voltage is set to 0
SPEED_CMD
BAM
GFLIB_Ramp_FLT
Ramp_RPM2RAD/S
Speed_Reference [rad/s]
SPEED_REF
1-D T(u)
V/f profile
Torque
Flux

Optional Part : SVM, PWM and S/F Scalar Control

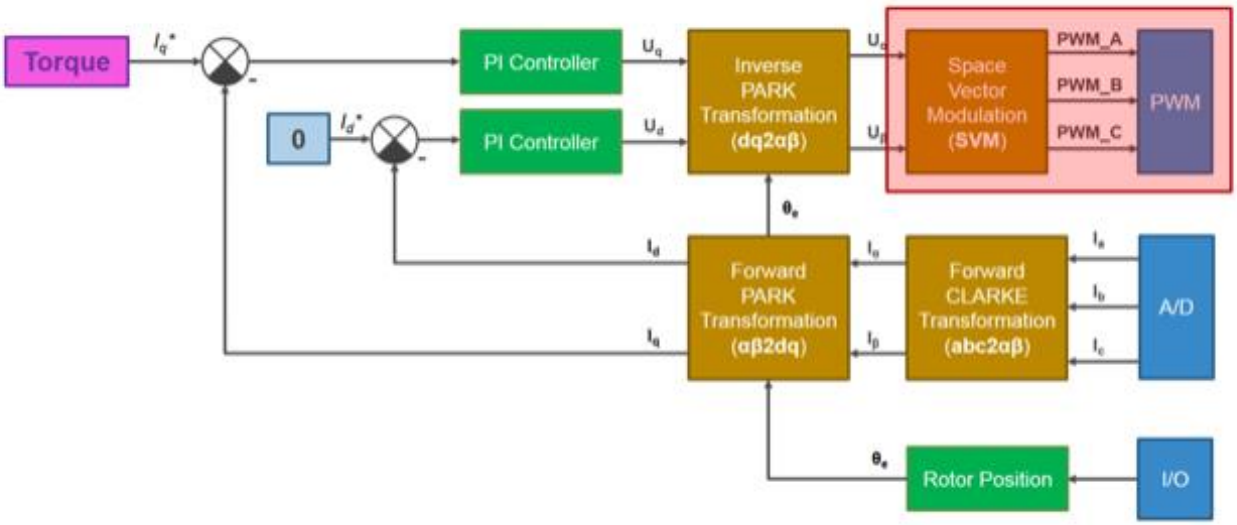
Part III: SVM and PWM

The focus of the first part of this lab is on the Space Vector Modulation (SVM) technique needed for generating the appropriate PWM commands for the 3-phase inverter used for controlling the speed and position of PMSM's rotor.

The goal is to explain step-by-step the aspects related with power conversion from DC bus voltage to AC phase voltages that are fed to motor windings and cause the motor to spin.

The Field Oriented Control (FOC) is one of the main control structure for generating the control quantities that drives the rotor in any position we want.

In this part of the lab, we will focus on methods to convert the digital information provided by FOC into actual voltages that can be applied to the PMSM windings.

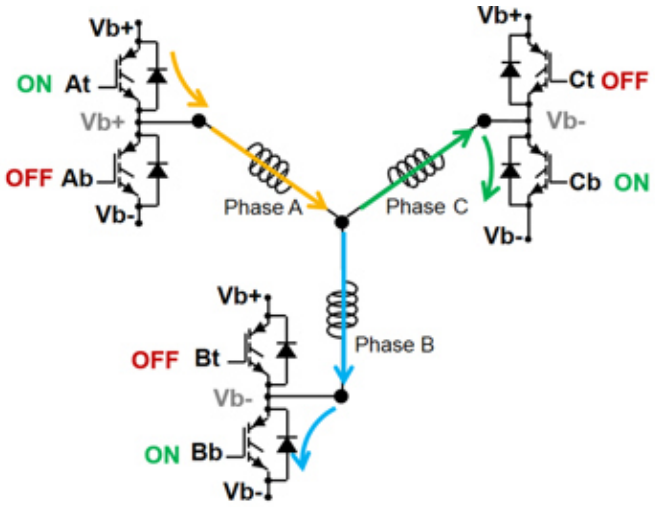


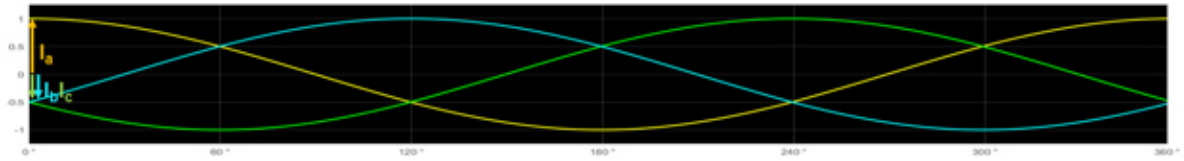
i. Commutation used for PMSM

In case of PMSM that are controlled with Field Oriented Control, all three phase windings can always conduct current.

Using three pairs of switches connected to a voltage source, we can control the current's flow and the magnitude by switching ON or OFF the transistors.

The selection of switches ON/OFF state, will determine the current sense, and the time while the which is connected to actual power source will determine the magnitude of the currents that flow thru the windings.





To control a PMSM we will need 6 switches that needs to be connected in pairs of two: one connected to a positive voltage source (we call that the TOP switch) and the other one connected to a negative voltage (we call this one the BOTTOM switch).

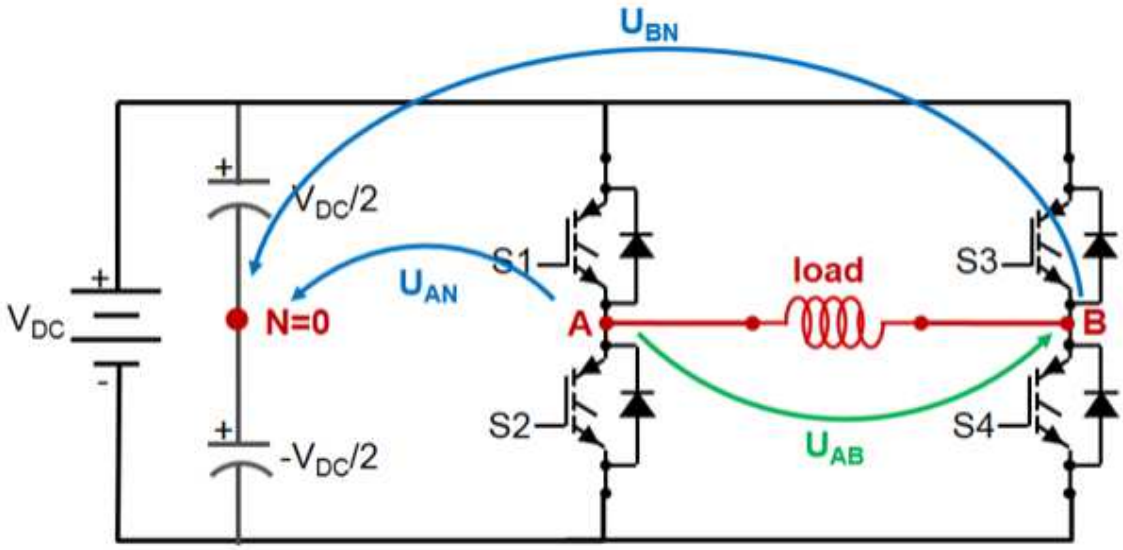
Each pair of switches will be controlled in complementary mode – meaning the switches can't be ON in the same time. In general, as a precaution, to avoid any accident like short-circuit, there is a small period referred as dead-time when both transistors are OFF.

ii. Bipolar vs. Unipolar Modulation

Let's start with a simple use-case: the single-phase inverter as the one shown next.

In general, to control the voltage applied on an inductive load, a single-phase inverter made up of 2 pairs of switches (also referred as legs) is used.

The voltage on the load (the green arrow) is defined based on the Kirchhoff circuit law as the difference of terminal voltages.



Since we have four switches, we will have four possible combinations of those and three possible values for the voltage applied on the load as shown in the following table.

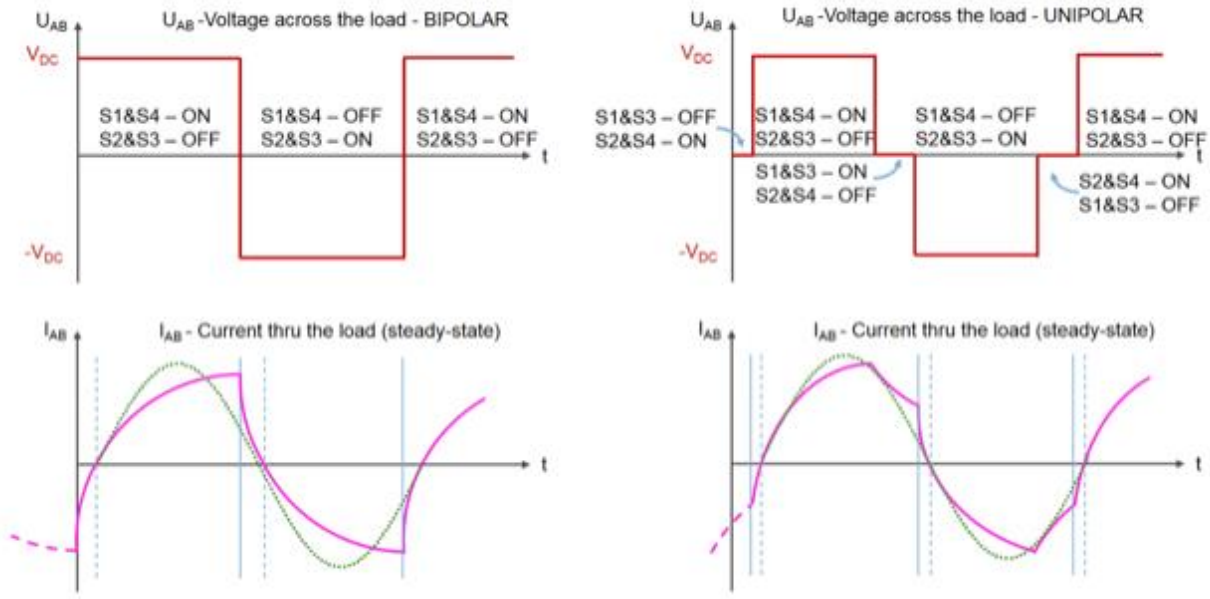
S1	S3	U_{AN}	U_{BN}	U_{AB}
OFF	OFF	$-V_{DC}/2$	$-V_{DC}/2$	0
OFF	ON	$-V_{DC}/2$	$+V_{DC}/2$	$-V_{DC}$
ON	OFF	$+V_{DC}/2$	$-V_{DC}/2$	$+V_{DC}$
ON	ON	$+V_{DC}/2$	$+V_{DC}/2$	0

Depending on how we choose to control the switches, we can have two types of PWM modulations:

- **BIPOLAR** – we use only rows 2 and 3 in table to control the switches. In this case the voltage applied on the load can have only 2 distinct values: +Vdc or –Vdc

- **UNIPOLAR** - we use all four combinations of switches leading us to a load voltage that can have 3 distinct levels: zero, plus or negative DC bus voltage.

The main differences between BIPOLAR and UNIPOLAR commutations are highlighted in the following figure.

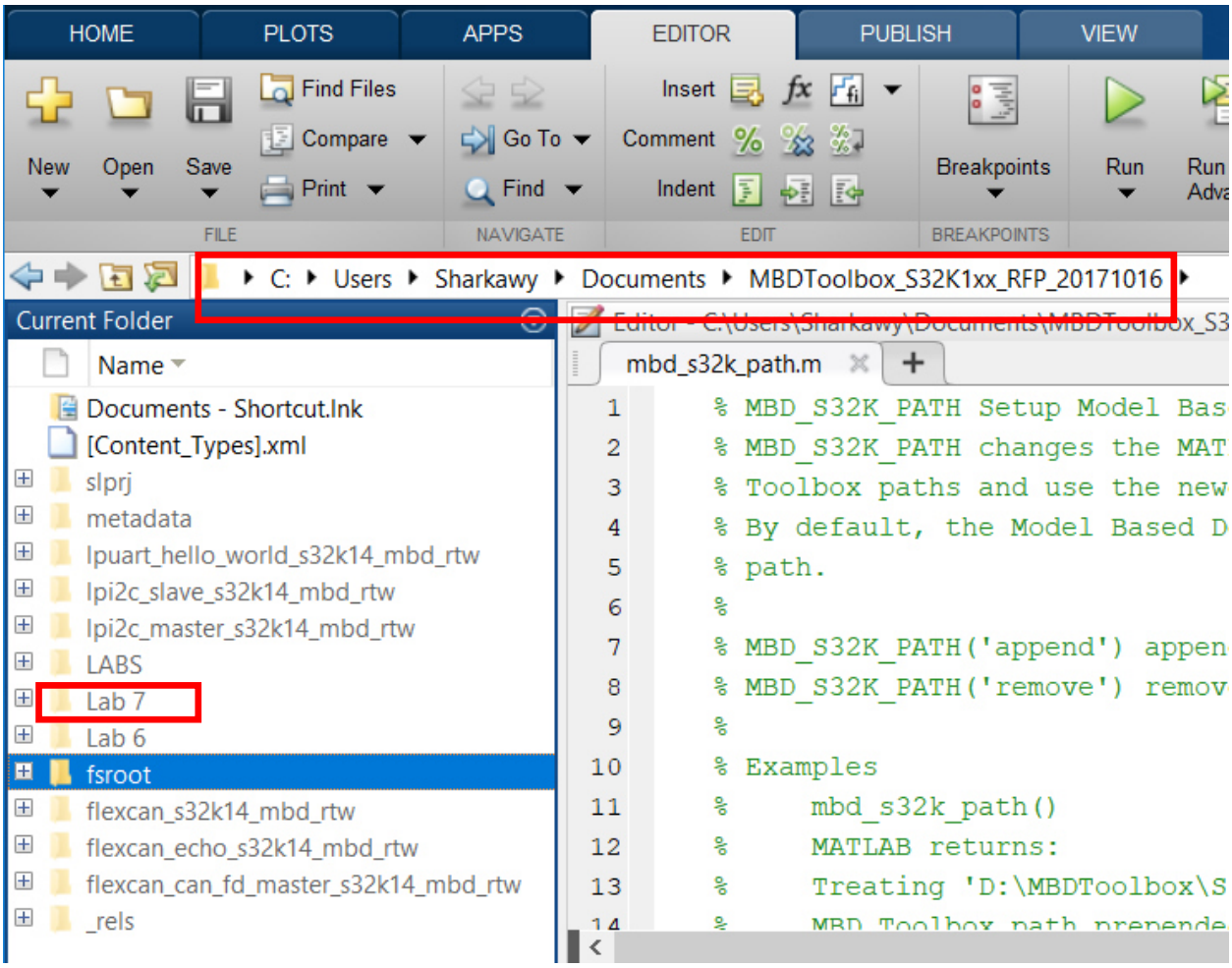


On the left, we have the load voltage and current variation for bipolar case while on the right-hand side we have the same quantities but this time for unipolar commutation method.

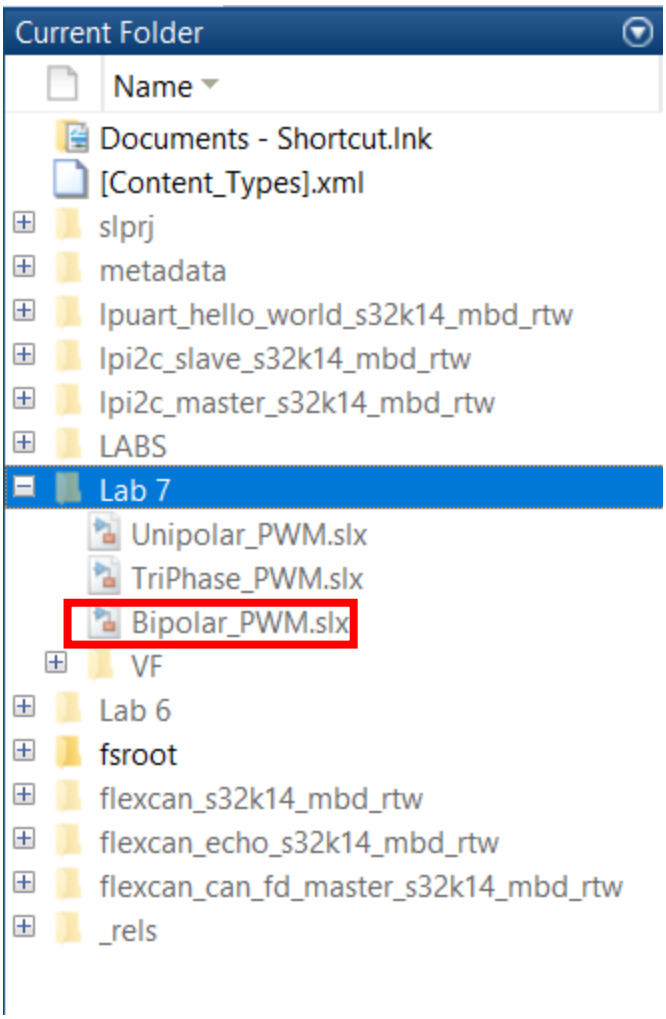
iii. Bipolar PWM Example:

- For this example, you need MATLAB version R2017a (or R2017b) and MBDT version 3.

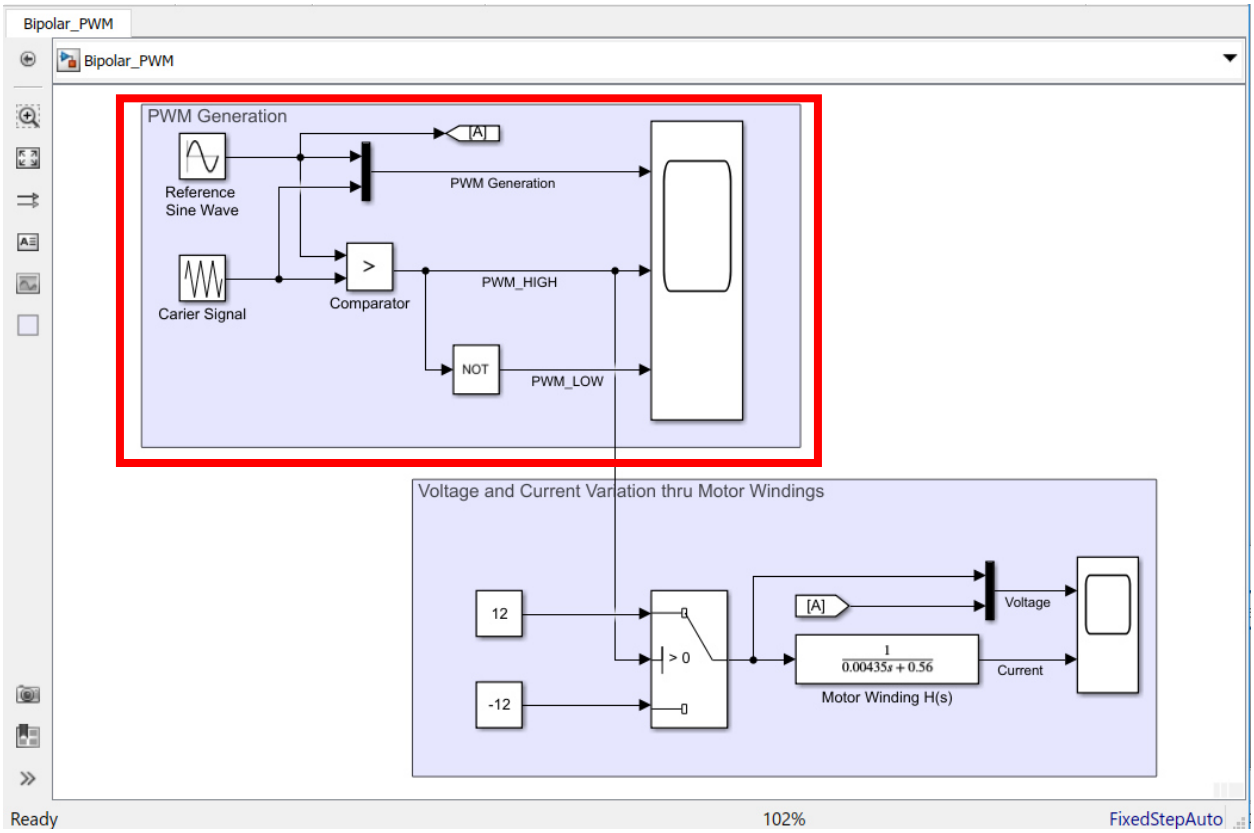
- Locate Lab 7 folder at the course website. Copy the lab 7 folder into the "MBDToolbox_S32K1xx_RFP_20171016" folder.



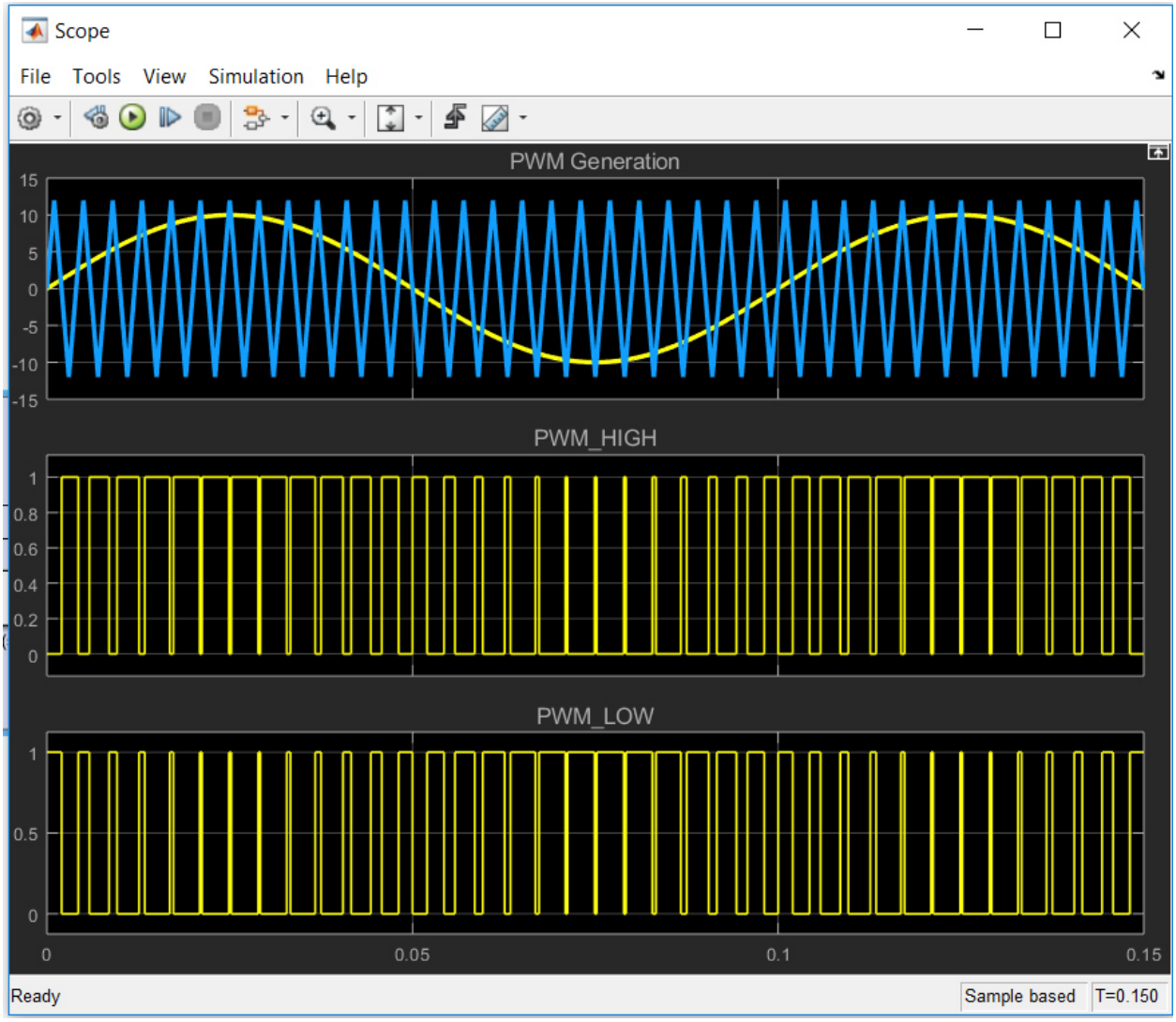
- Double click on Bipolar_PWM Simulink model.



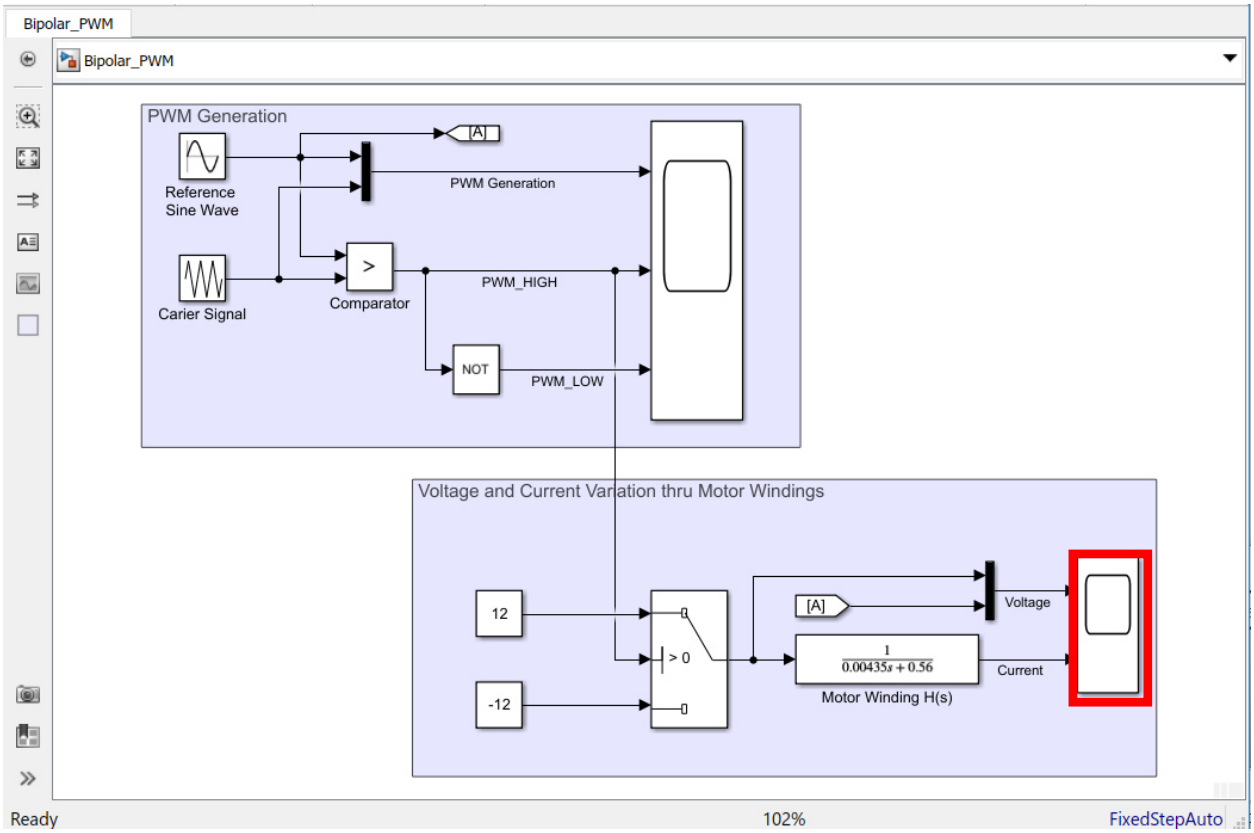
- The bipolar modulation is quite simple and the logic to implement such method is elementary. All we need is a carrier signal, usually a sawtooth signal that is compared against the signal waveform you wish to obtain. The values obtained after comparison are used to control the top and bottom switches that forms one inverter leg. Since the control pulses varies in width – hence the name of Pulse Width Modulation (PWM).

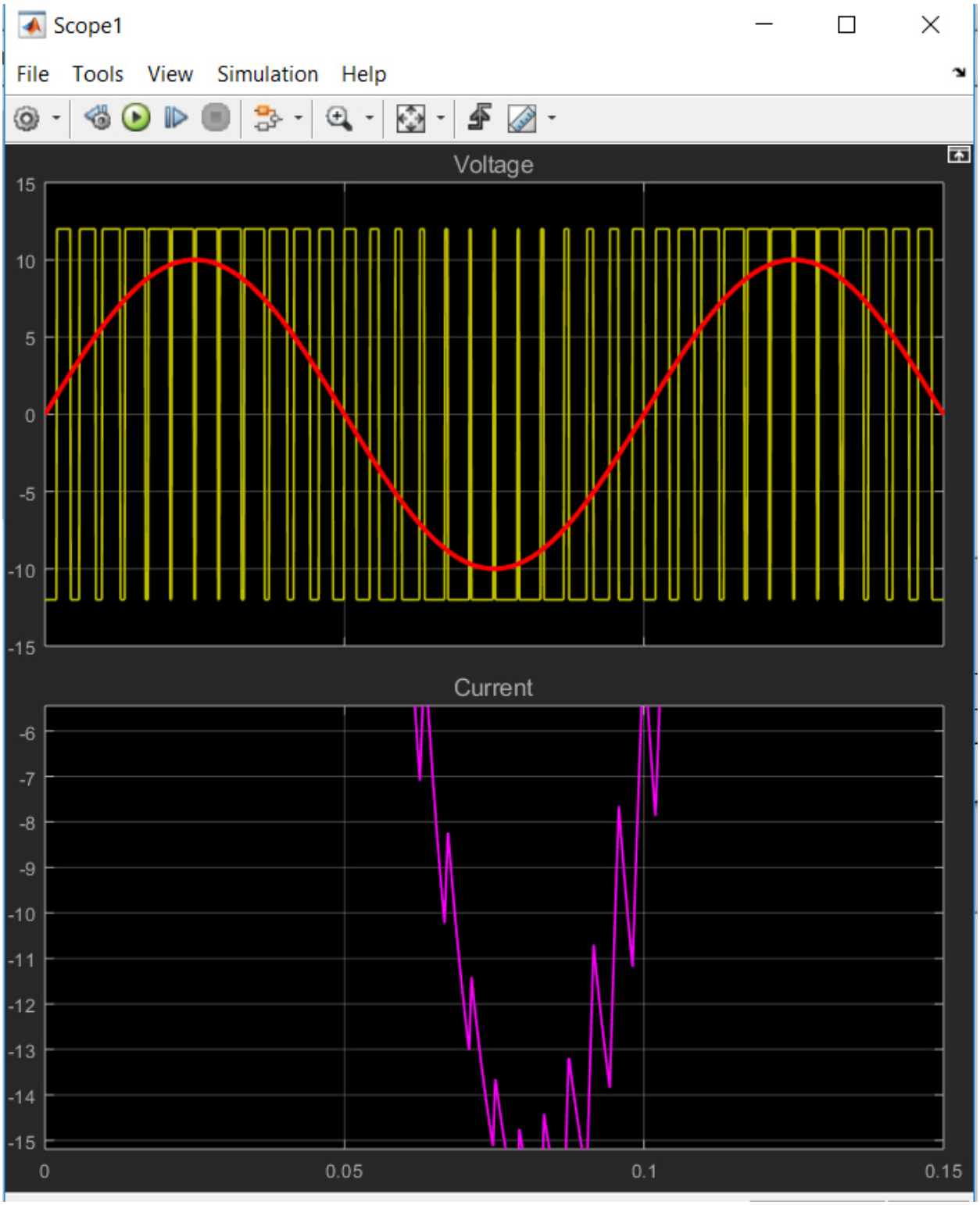


- Click on  to run your simulation.
- Double click on the upper scope.



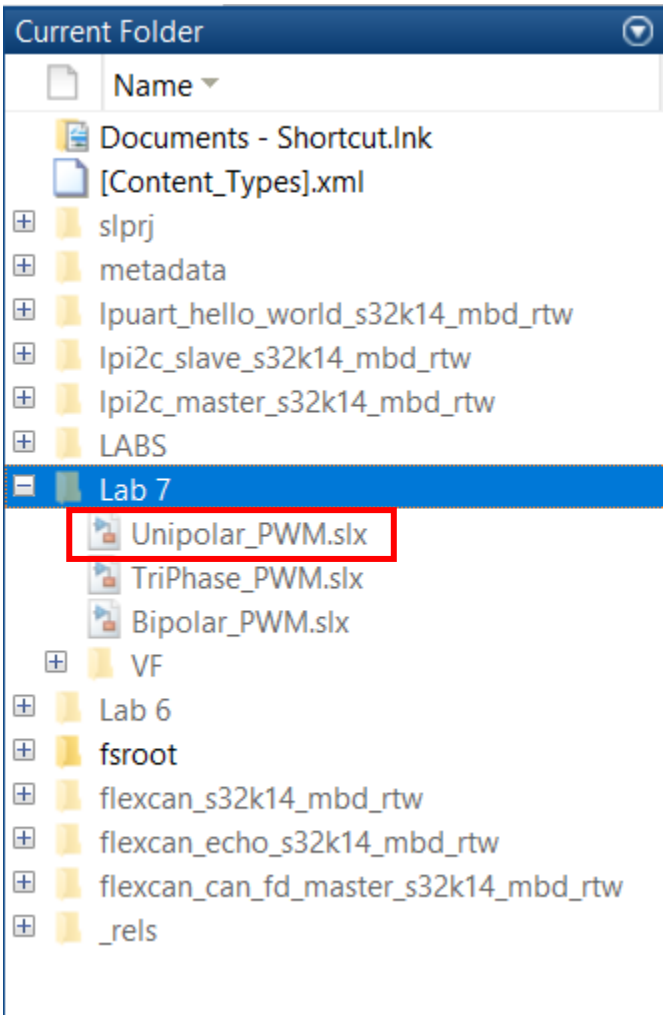
- Double click on the lower scape to display voltage and current variation through motor windings.

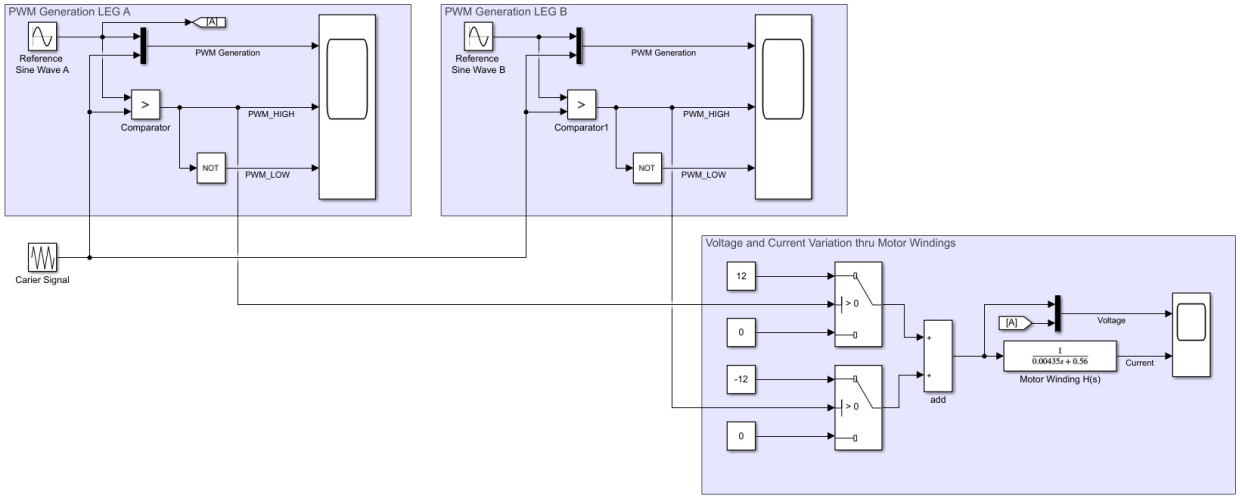

Bipolar_PWM
Bipolar_PWM
PWM Generation
[A]
Reference
Sine Wave
PWM Generation
Carier Signal
>
Comparator
PWM_HIGH
NOT
PWM_LOW
Voltage and Current Variation thru Motor Windings
12
-12
> 0
[A]
Voltage
1
0.00435s + 0.56
Current
Motor Winding H(s)
Ready
102%
FixedStepAuto


Scope1
File Tools View Simulation Help
Voltage
15
10
5
0
-5
-10
-15
Current
-6
-7
-8
-9
-10
-11
-12
-13
-14
-15
0
0.05
0.1
0.15

iv. Unipolar PWM Example:

- Double click on Unipolar_PWM Simulink model.

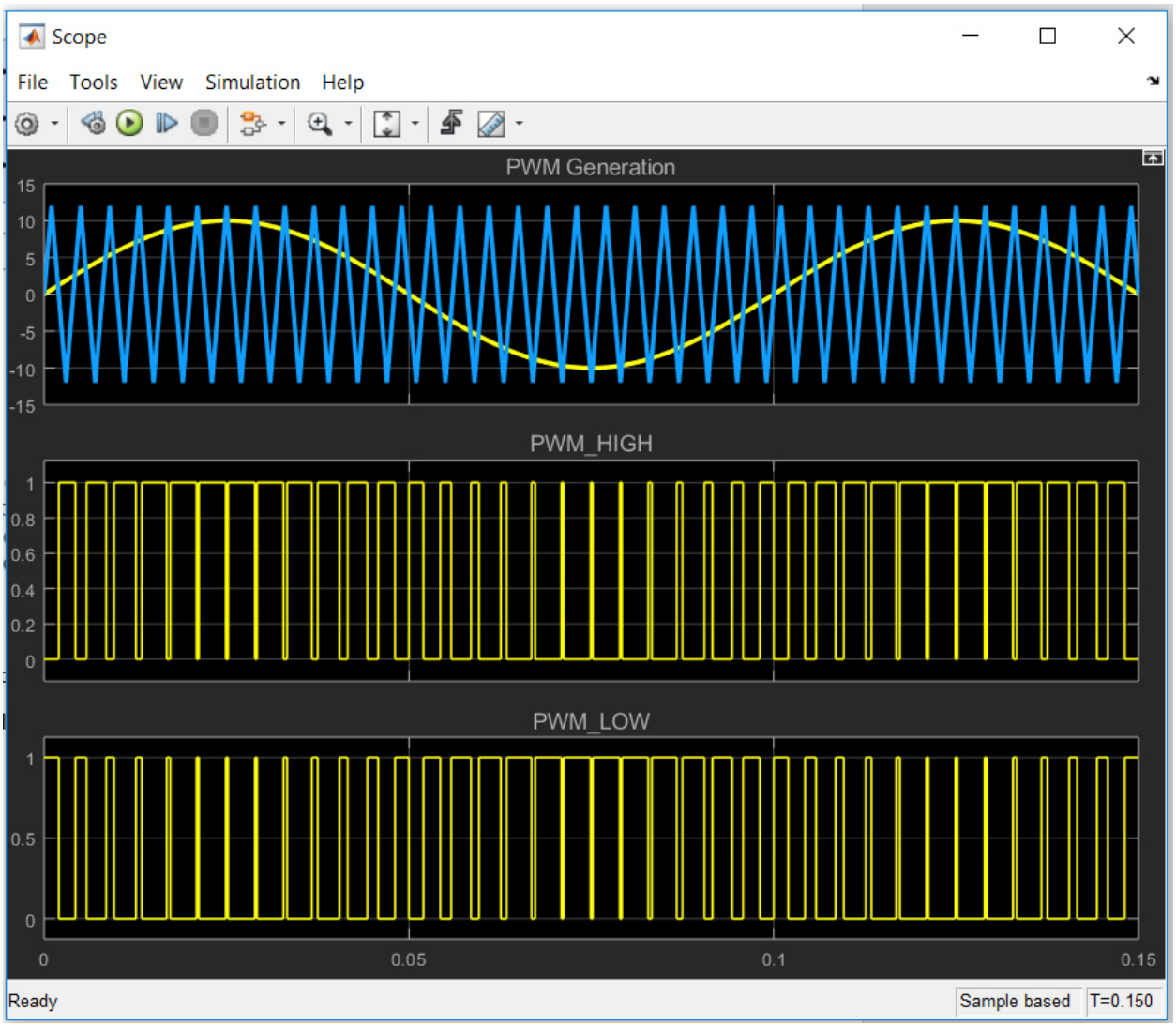




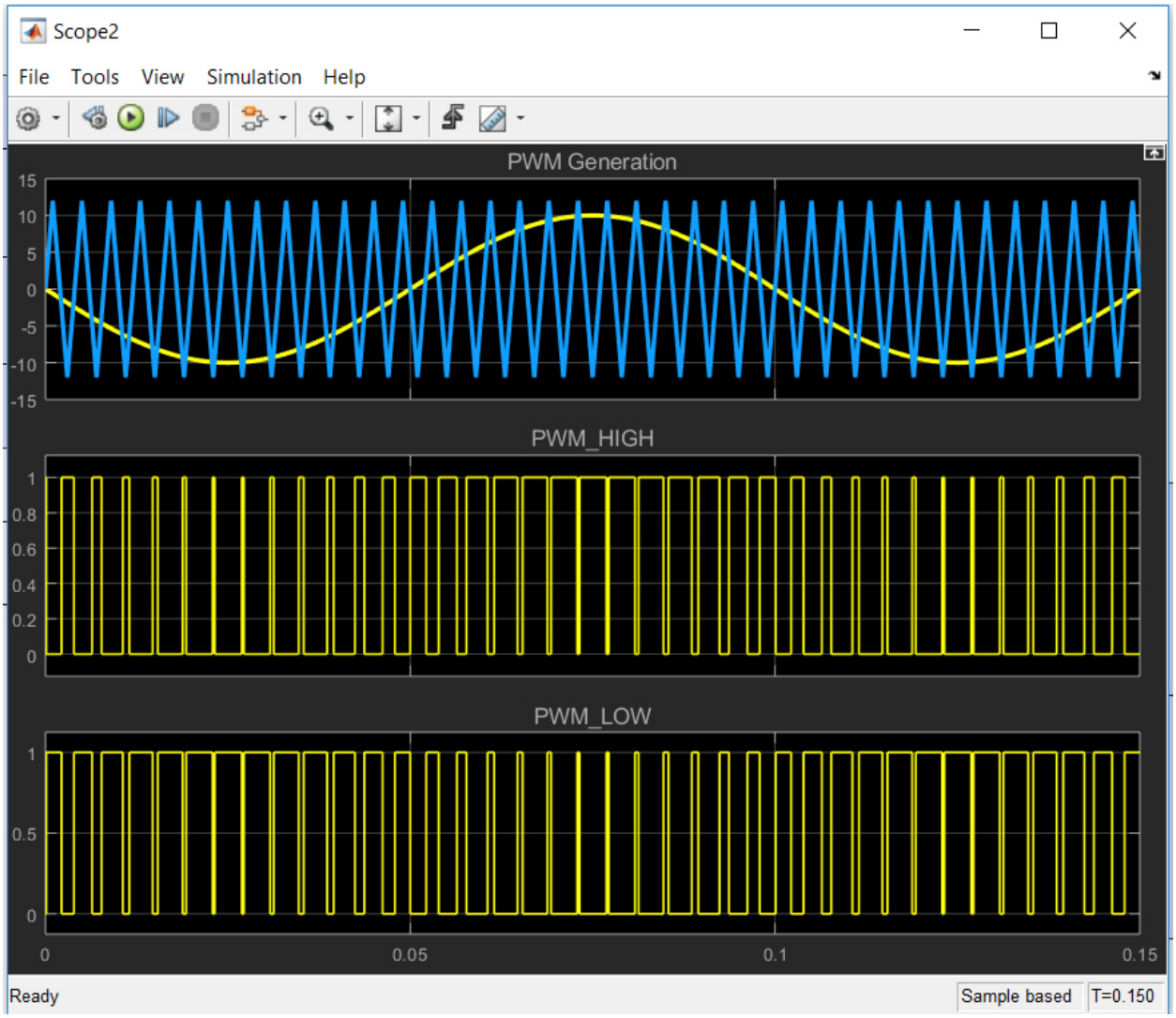
- In case of unipolar modulation, the logic to generate such signals looks similar, the only difference being the fact that we need to add a second reference source to control the switches on the second inverter leg. Usually this second reference signal is phased out with 180 degrees compared with the source for the first inverter leg.

- Click on  to run your simulation.

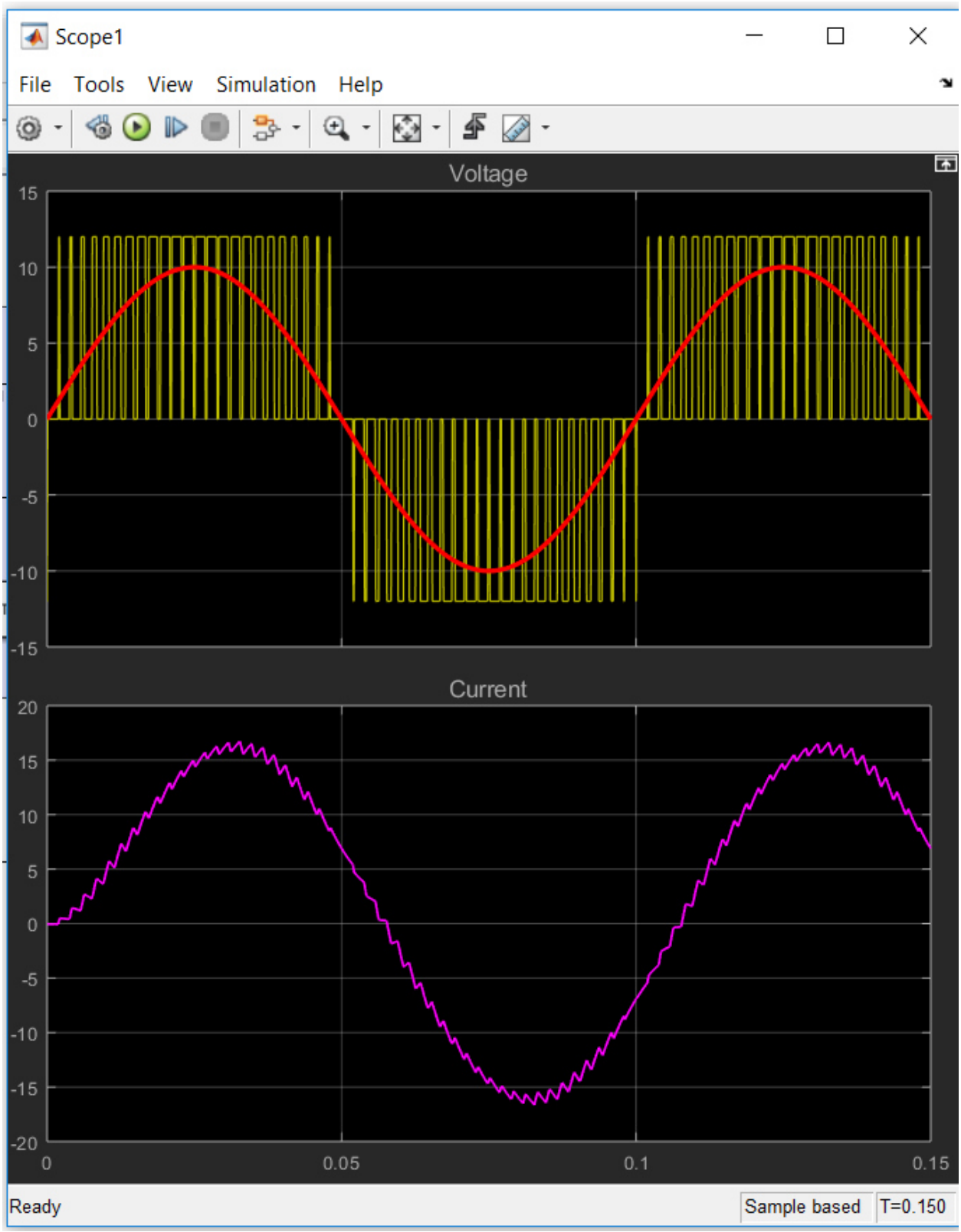
- Double click on the upper left scope.



- Double click on the upper right scope.

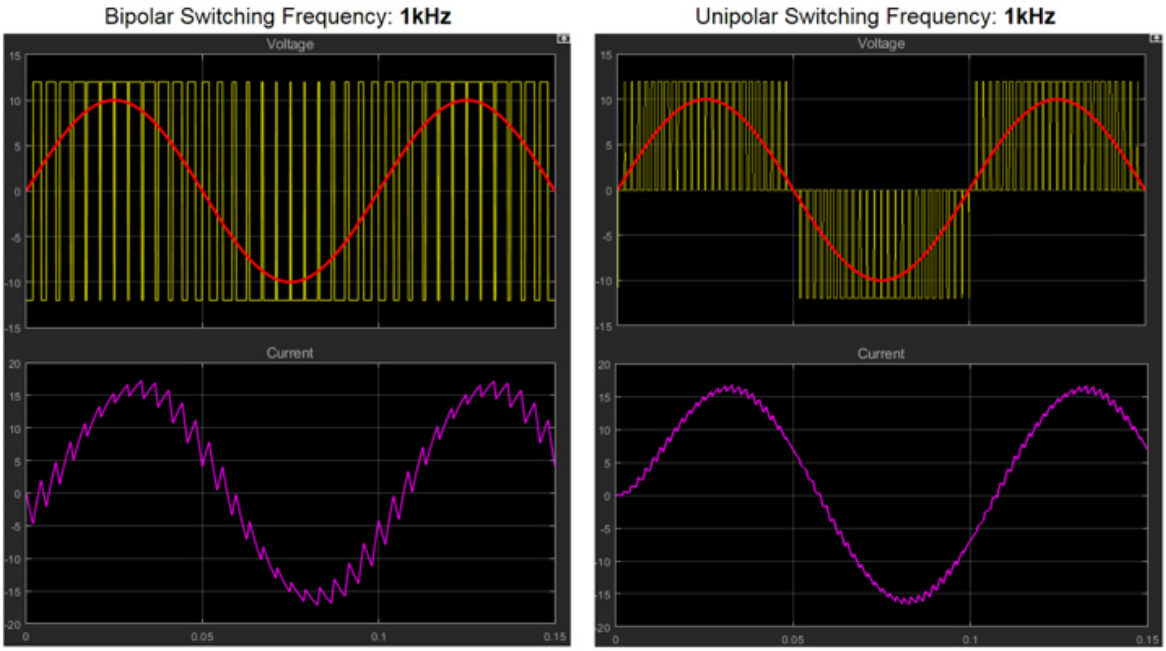


- Double click on the lower scope to display voltage and current.


Scope1
File Tools View Simulation Help
Voltage
15
10
5
0
-5
-10
-15
Current
20
15
10
5
0
-5
-10
-15
-20
0
0.05
0.1
0.15
Ready
Sample based
T=0.150

- So far, we have an example of switching commands for bipolar versus unipolar commutation methods in case of 10 Hz sinusoidal waveform and a 1 kHz carrier signal.

*



As we can see there are a few fundamental differences between bipolar and unipolar modulation techniques:

- The first notable difference is the **current waveform** that is much smoother in case of unipolar commutation than the corresponding one obtained with bipolar modulation. The fact that we apply a smaller effective voltage helps keeping the current ripple smaller and reduces the harmonics produced. As you can see the current waveform resemble better the sinusoidal waveform in case of the unipolar modulation.

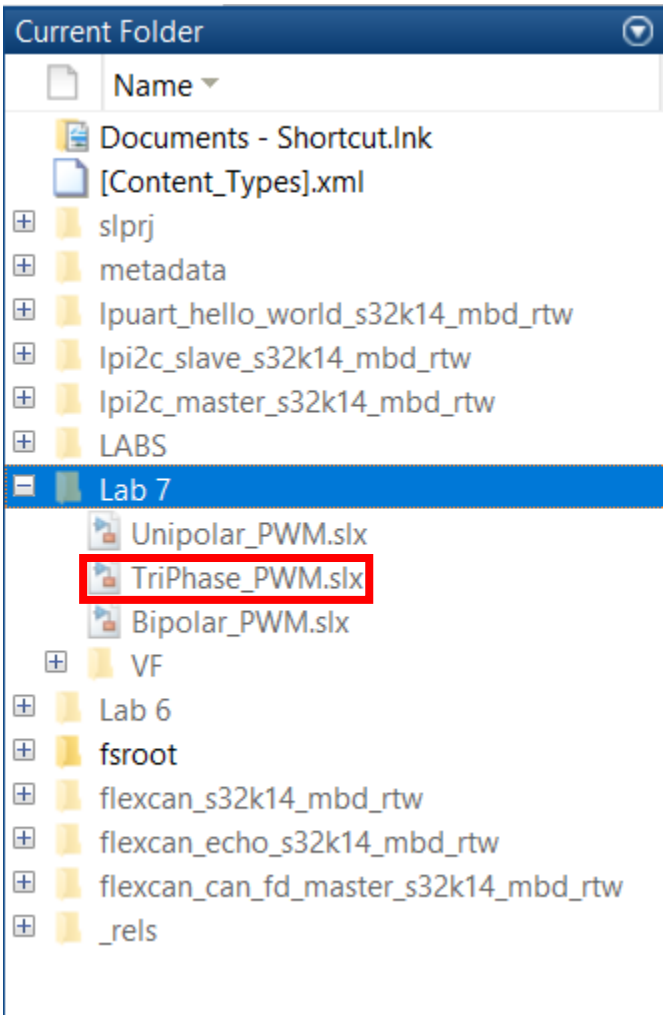
- **Less voltage stress** seen by the transistors in case of unipolar compared to bipolar method where the transistors needs to switch off 2 times the amount of DC voltage, in unipolar technique they see only half of that stress. This makes the unipolar method more suitable of high-voltage applications.

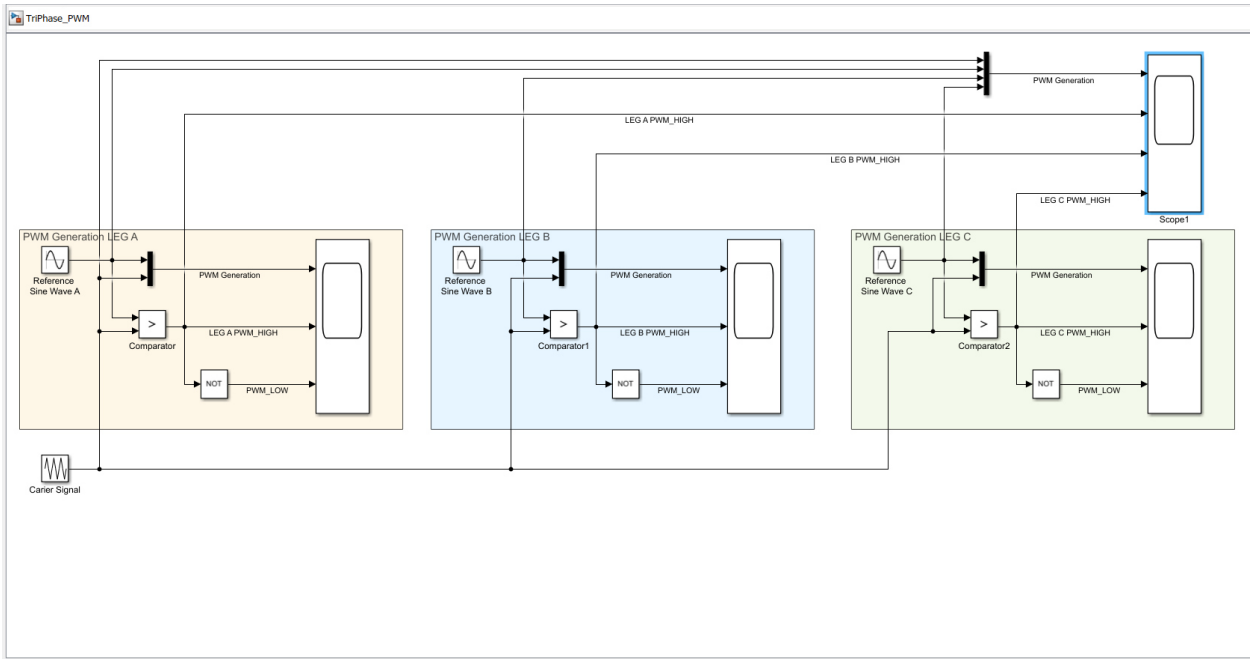
- **The switching logic** is more complicated in case of unipolar method since we need to pass thru an intermediate state.

- **Effective switching frequency** which is higher in case of unipolar commutation even if we used the same 1khz carrier signal.

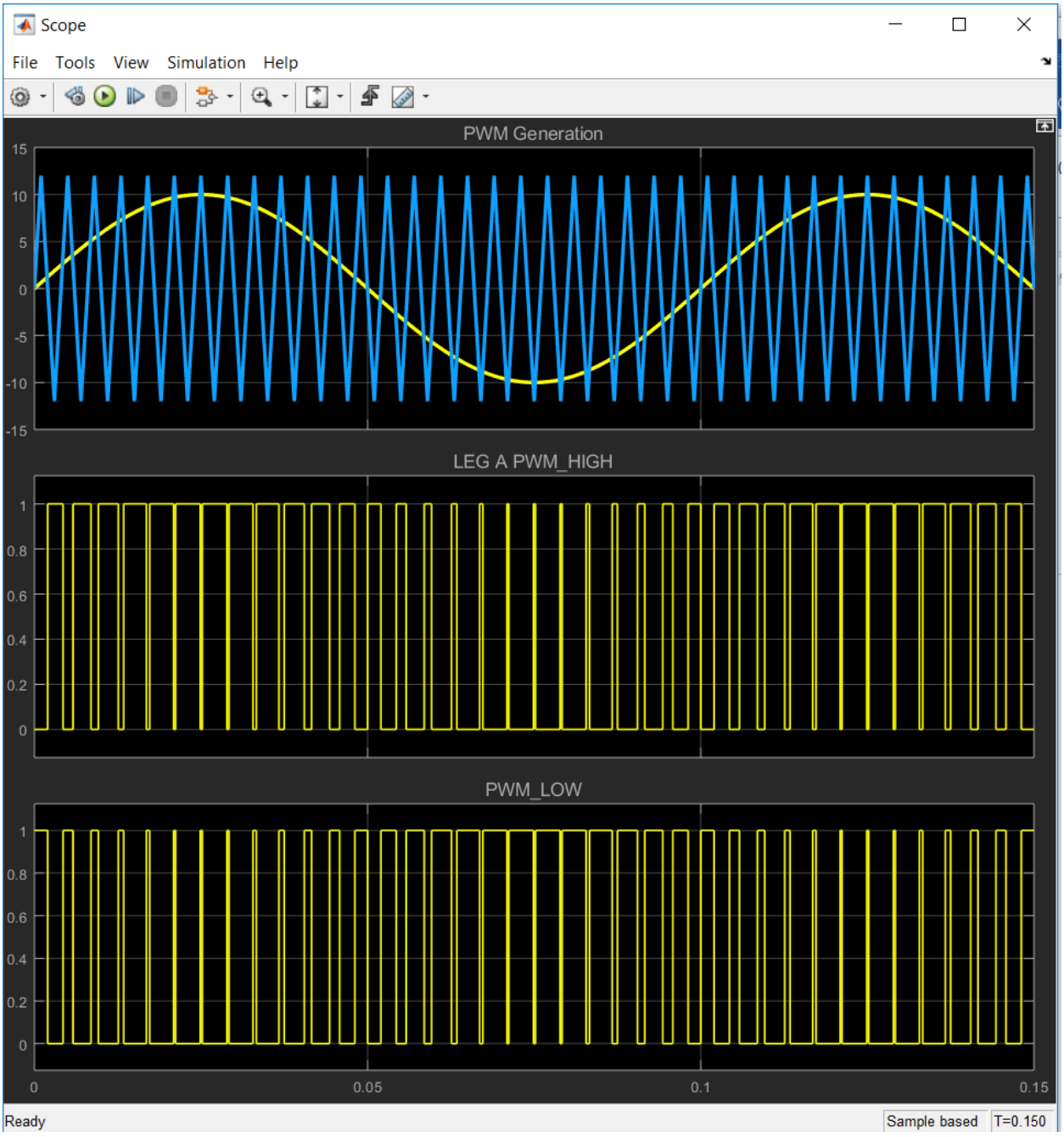
v. TriPhase PWM Example:

- Double click on TriPhase_PWM Simulink model.

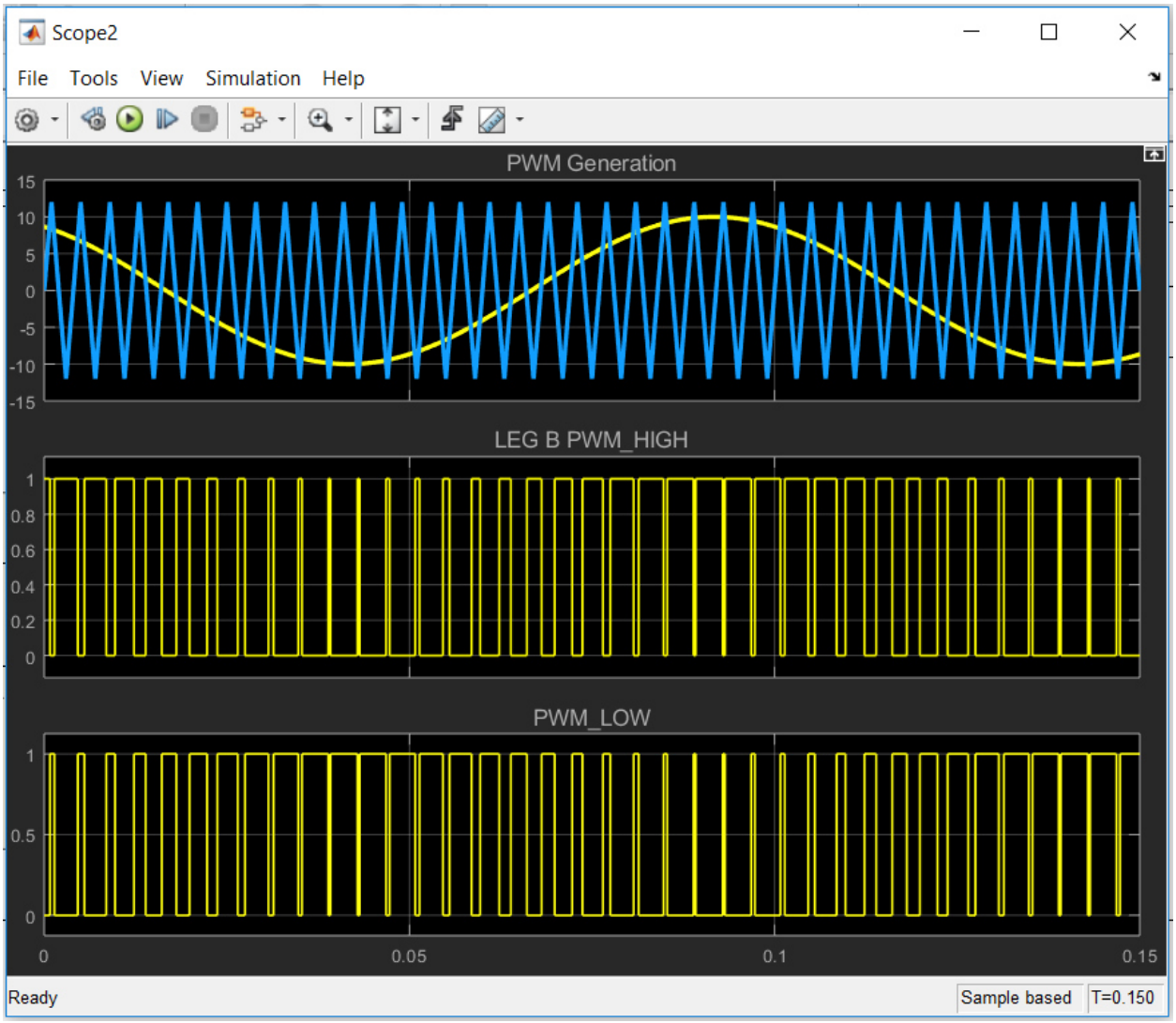




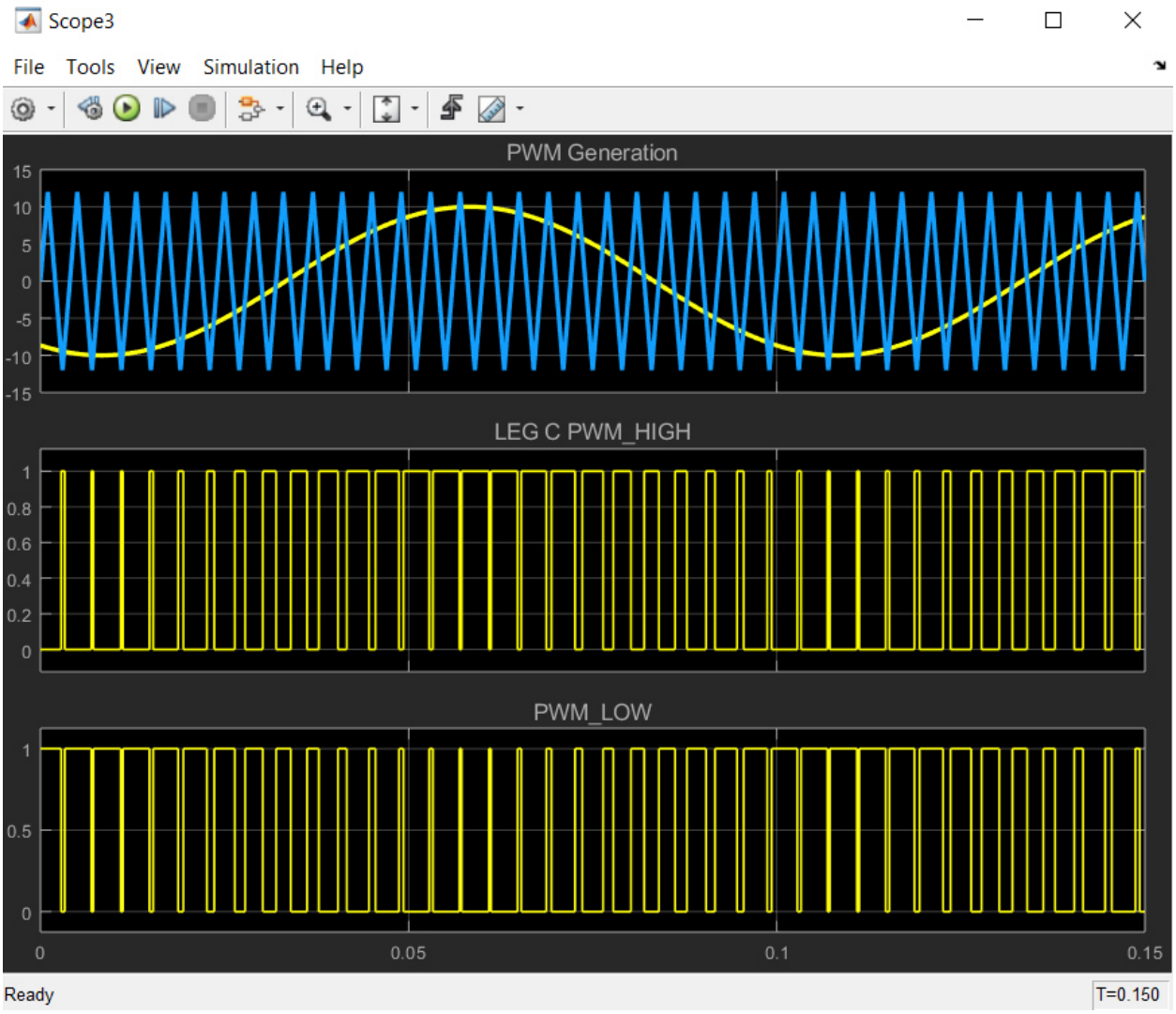
- Click on  to run your simulation.
- Double click on the lower left scope.


Scope
File Tools View Simulation Help
PWM Generation
15
10
5
0
-5
-10
-15
LEG A PWM_HIGH
1
0.8
0.6
0.4
0.2
0
PWM_LOW
1
0.8
0.6
0.4
0.2
0
0
0.05
0.1
0.15
Ready
Sample based
T=0.150

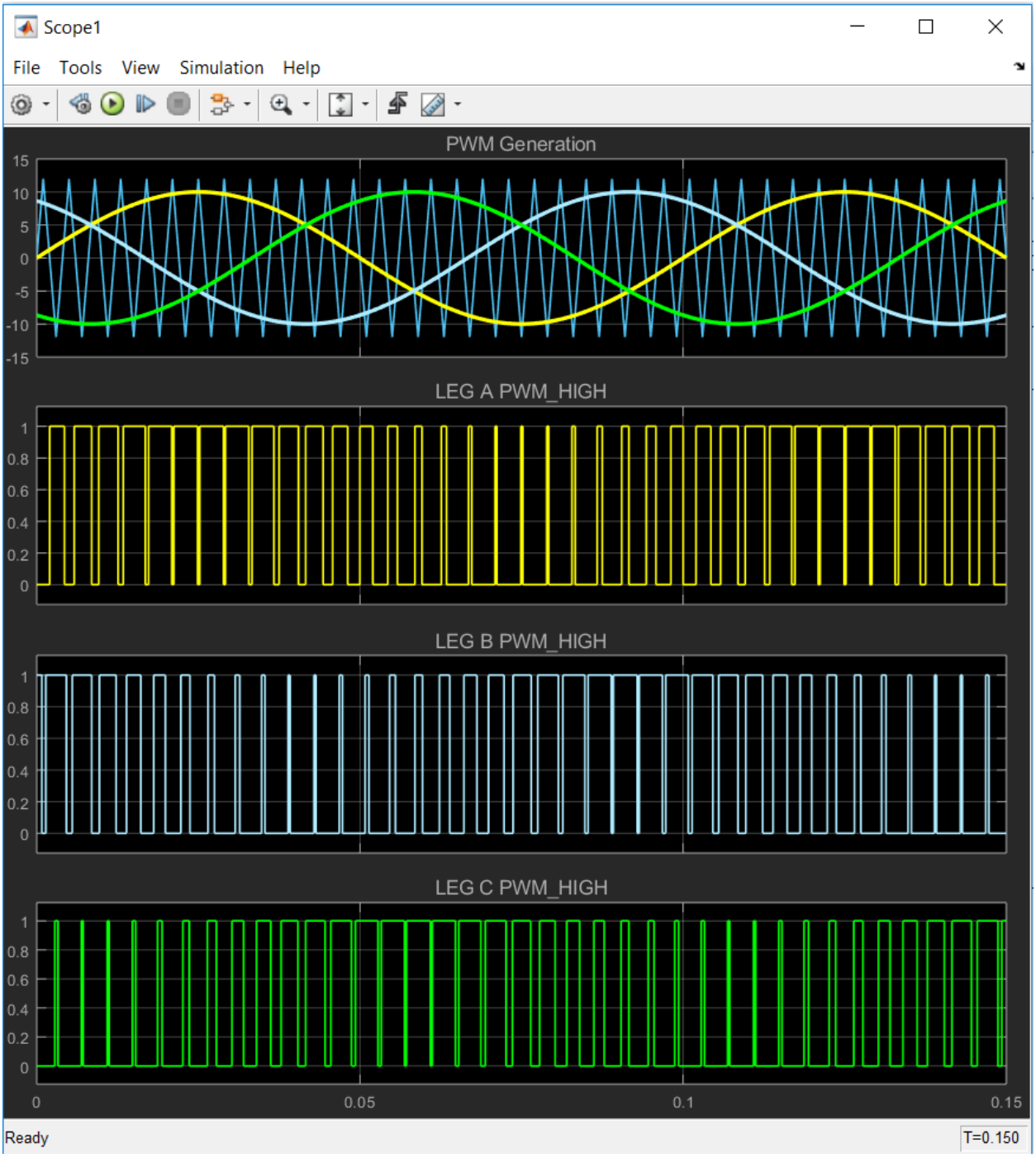
- Double click on the middle left scope.



- Double click on the right left scope.



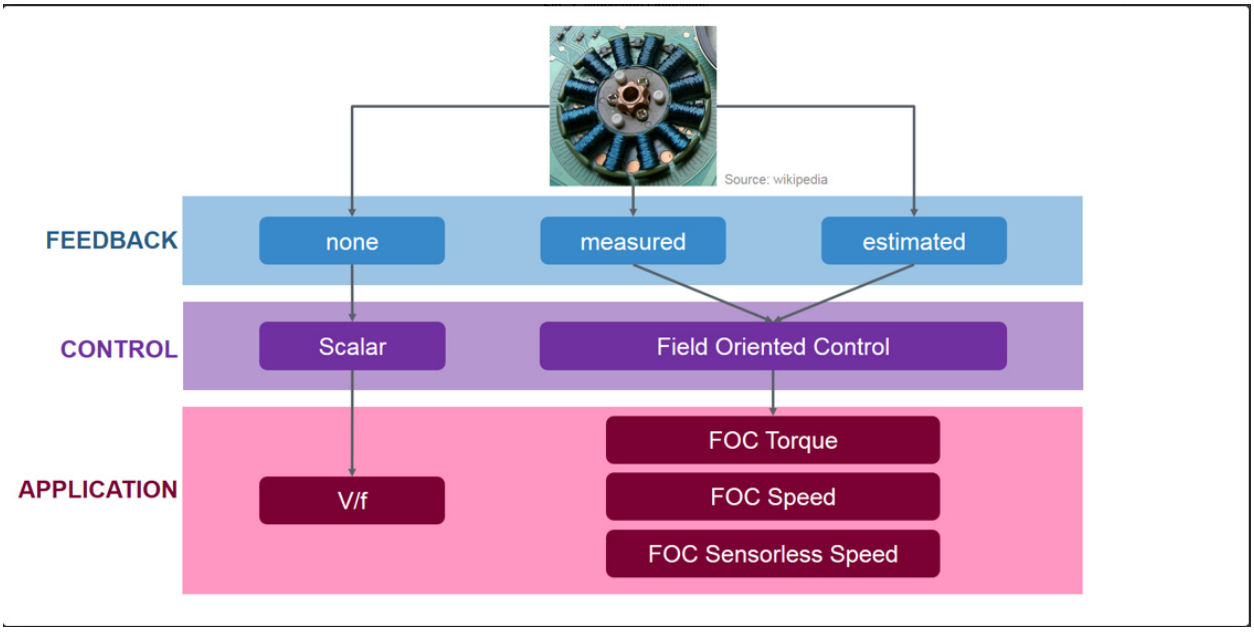
- Double click on the upper scope.


Scope1
File Tools View Simulation Help
PWM Generation
15
10
5
0
-5
-10
-15
LEG A PWM_HIGH
1
0.8
0.6
0.4
0.2
0
LEG B PWM_HIGH
1
0.8
0.6
0.4
0.2
0
LEG C PWM_HIGH
1
0.8
0.6
0.4
0.2
0
0
0.05
0.1
0.15
Ready
T=0.150

Part VI: V/F Scalar Control

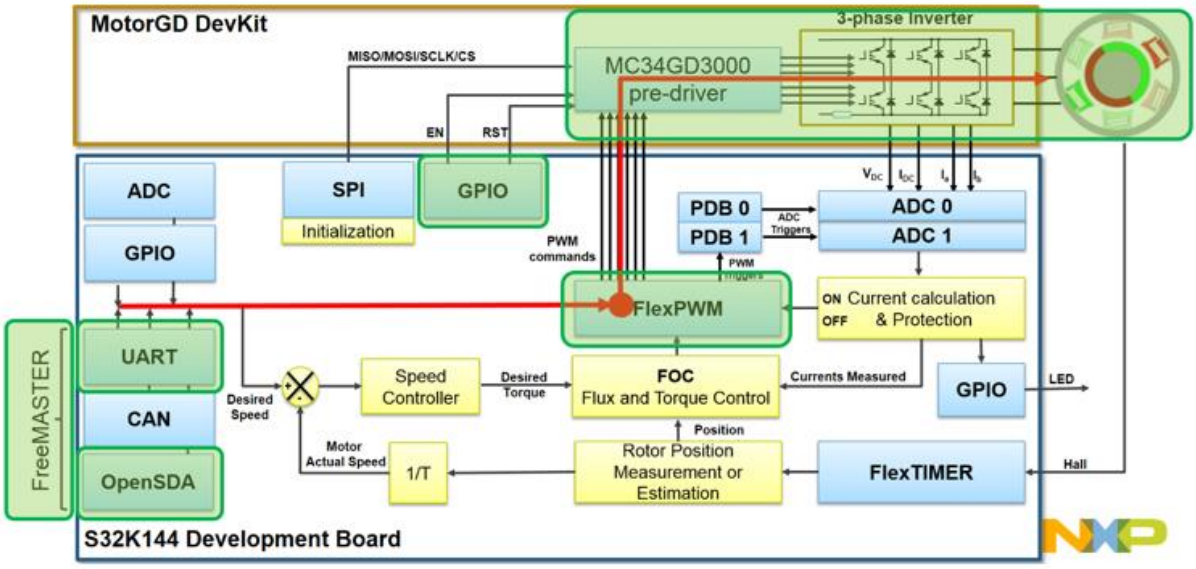
A PMSM can be led to steady state regime by one of the following control techniques:

- simple voltage fed controller that we are going to implement here.
- current type controller (DTC).
- speed controlled scheme (FOC).



In this part of the lab, we are going to simulate the spinning of the PMSM with S32K and MotorGD Development Kit. We are building a PMSM mathematical model in Simulink to facilitate the development and validation of the control algorithm. The simulation of the algorithm allows us to learn about implementing it on the real hardware.

We are implementing the V/F Scalar Control method. Another name for this method is Volts per Hertz (V/Hz). The main principle is the ratio between voltage and frequency is kept constant throughout the motor speed range.



In general, HW & SW modules used for V/F Scalar Control are highlighted in green.

The figure shows the main hardware blocks that need to be configured and the control path need to be followed (with red line).

To achieve V/F scalar control, we need to configure the S32K FlexTimer peripheral to generate six PWM signals that are used to control the MotorGD Development Kit DC to AC power inverter via a dedicated MOSFET pre-driver chip: MC34GD3000.

To enable the MC34GD3000 chip operation, two additional signals need to be controlled: Enable and Reset via dedicated GPIO.

The interactions between host PC and hardware need to be done via OpenSDA serial communication that will be used to download the code generated from MATLAB and to visualize various control signals with FreeMASTER.

In this part of the lab, we are going to simulate this system.

i. V/F Scalar Control:

The main equations of the PMSM motor are:

$$\begin{bmatrix} u_d \\ u_q \end{bmatrix} = R_s \begin{bmatrix} i_d \\ i_q \end{bmatrix} + \begin{bmatrix} s & \omega_e \\ -\omega_e & s \end{bmatrix} \begin{bmatrix} \psi_d \\ \psi_q \end{bmatrix} \quad \text{(eq. 1)}$$

$$\begin{bmatrix} \psi_d \\ \psi_q \end{bmatrix} = \begin{bmatrix} L_d & 0 \\ 0 & L_q \end{bmatrix} \begin{bmatrix} i_d \\ i_q \end{bmatrix} + \psi_{PM} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{(eq. 2)}$$

where:

u_d, u_q, i_d, i_q voltages and current in d-axis and q-axis respectively

L_d, L_q, R_s d and q-axis inductances and stator winding resistance

$\psi_d, \psi_q, \psi_{PM}$ d and q-axis flux linkage and permanent magnet flux linkage

ω_e electrical rotor speed

Rewriting (eq.1) in scalar from and time domain we will obtain:

$$u_d = R_s \cdot i_d + \frac{d}{dt}\psi_d - \omega_e \psi_q$$

$$u_q = R_s \cdot i_q + \frac{d}{dt}\psi_q + \omega_e \psi_d \quad \text{(eq. 3)}$$

In steady state regime, the flux linkage variation is zero, and for further simplification we are going to assume the stator winding resistance is neglectable. Taking into consideration these simplifications and the flux linkage equation (eq. 2) then the equations (eq. 3) becomes:

$$u_d = -\omega_e L_q i_q$$
$$u_q = \omega_e L_d i_d + \omega_e \psi_{PM}$$

(eq. 4)

At this point we can transform the electric speed in frequency and rewrite the (eq.4) as a ratio of V/F:

$$\frac{u_d}{f} = -(2\pi) L_q i_q$$
$$\frac{u_q}{f} = (2\pi) \cdot (L_d i_d + \psi_{PM})$$

(eq. 5)

In V/F scalar control method the frequency of the stator magnetic flux is set according with the desired synchronous rotor speed while the magnitude of the stator voltage is adjusted to keep the ratio between them constant. No control over voltage or current vectors angles is utilized, hence the name scalar control.

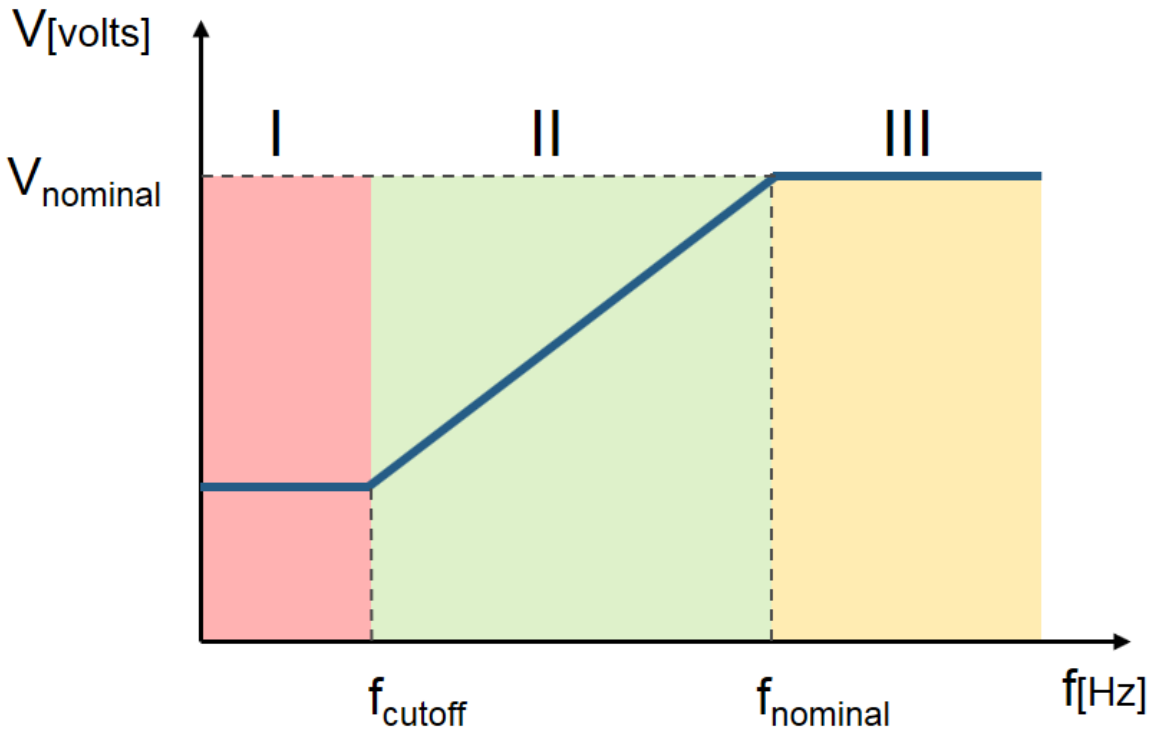
The V/F ratio is calculated from the nominal values of the PMSM voltage and frequency parameters. By maintaining a constant V/F ratio between the amplitude and frequency of 3-phase voltage waveforms, then the stator flux of the PMSM can be maintained relatively constant in steady state. However, in practice a typical V/F profile is not constant over the entire range of motor speed.

As can be seen next, the V/F profile may be divided in three main regions:

I. **Compensation** – a higher than normal voltage is required to compensate the voltage drop across the stator resistance that was neglected for simplified mathematical model

II. **Linear** - follows the constant V/f relationship as derived from (eq. 5)

III. **Field weakening** - constant V/f ratio cannot be satisfied due to the stator voltages limitation at the rated value in order to avoid insulation breakdown



V/F profile practical aspects

In case of PMSM, the V/F scalar control is a good alternative in applications where good dynamic performance is not required (e.g.: HVAC, fans, pumps or blowers). In such cases the V/F scalar control is performed without the need of a position/speed sensor.

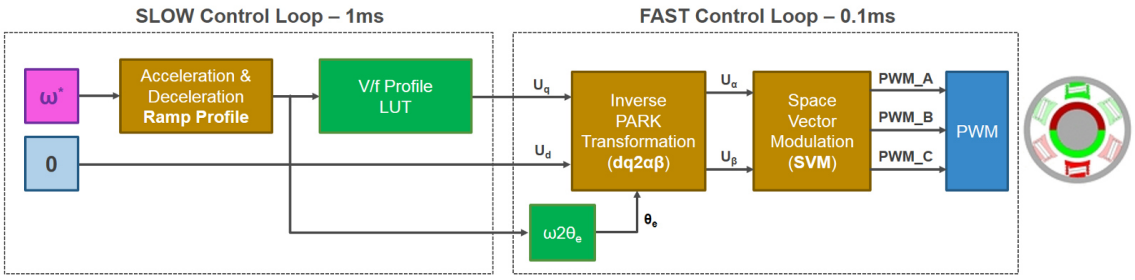
By using V/F scalar control there is no need for high capability CPU as in the case of FOC, but keep in mind that this kind of simplicity also comes with some disadvantages:

- instability of the system after exceeding a certain applied frequency
- systems low dynamic performance, which limits the use of this control method
- poor fault protection against stall detection and over-currents

In case of PMSM, both open and closed-loop control of the speed can be implemented based on the V/F scalar control.

Open-loop control is used in applications where system dynamic response is not a concern. For such use cases, the frequency is determined based on the desired speed and the assumption that the rotor will ultimately follow the synchronous speed.

Throughout this part of the lab, we are going to implement an open-loop control system topology for a 3-phase PMSM using V/F scalar control as the one shown next.



V/F scalar control block diagram

Such control structure will allow us to control the PMSM speed without any feedback of motor parameters or rotor position. The motor is driven by the conventional 3 phase voltage-source inverter via MotorGD DevKit. The NXP's S32K MCU is being used to generate the six PWM signals using a modified SVM PWM technique with 3-rd harmonic injection.

The V/F scalar control application based on:

- FAST control loop is executed at each 0.1ms and computes the PWM duty cycles based on Space Vector Modulation. The reference voltages for (dq)-axes and the motor electrical speed represent the inputs. The electrical angle needed for Inverse Park Transformation is computed based on electrical speed reference.

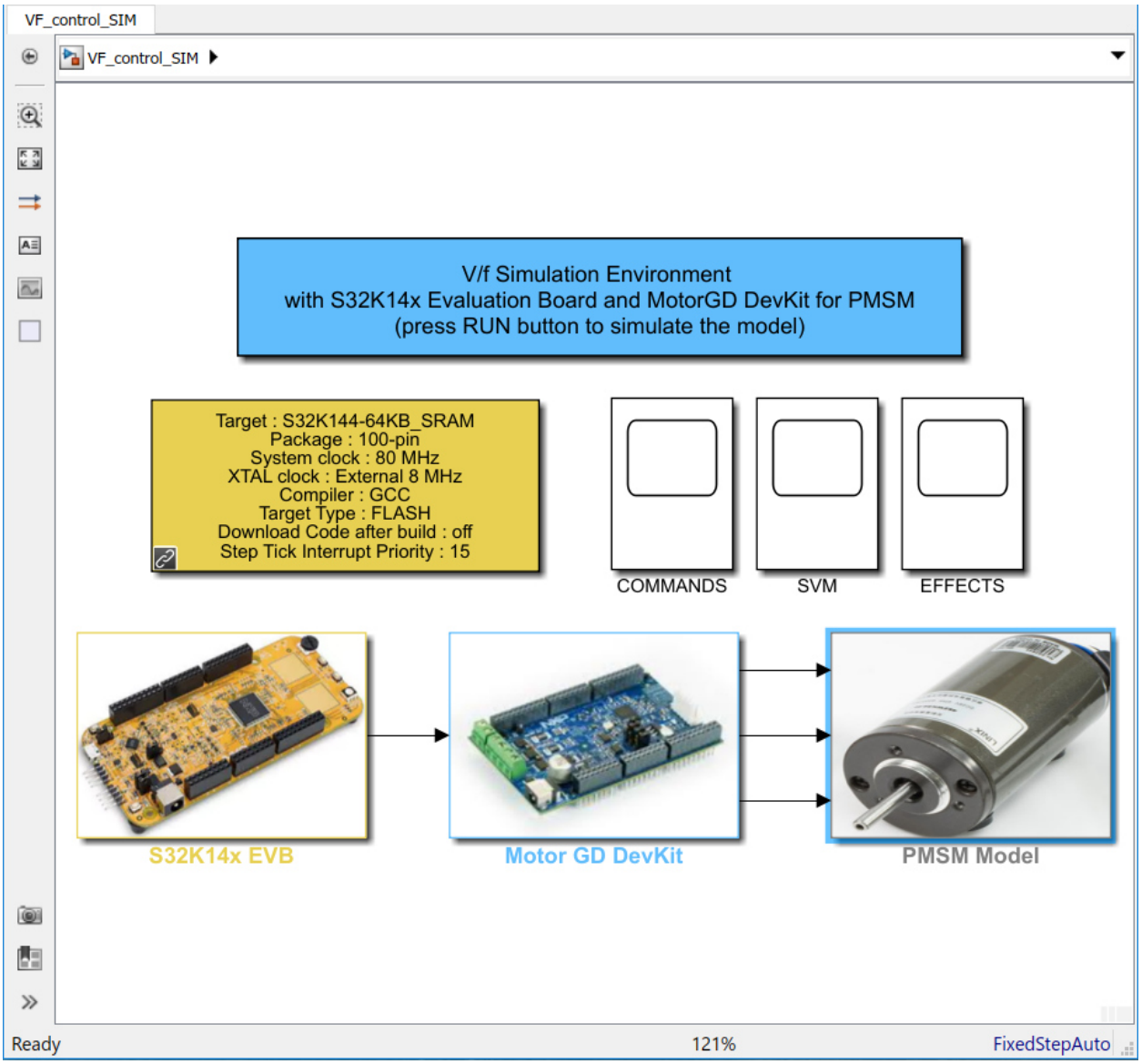
- SLOW control loop is executed at each 1ms and provides the voltage references for Inverse Park transformation and the electrical speed profile. Inside this loop, the user commands are handled and based on a parameterisable Ramp Profile the appropriate control parameters are generated. The V/F profile is implemented via a tunable Look-Up-Table (LUT)

ii. MATLAB Modelling of Control Algorithm

Using Model-Based Design strategy we can develop, implement, simulate and test the control method entirely in simulation environment and once we are satisfied with the results we can proceed further by deploying the generated code on the real target and perform final validation and testing.

- Double click on VF_control_SIM Simulink model.

  The display shows the Simulink model we used for V/F scalar control implementation, testing and validation.



Simulink model for the entire plant: MCU-DC2AC INVERTER-PMSM

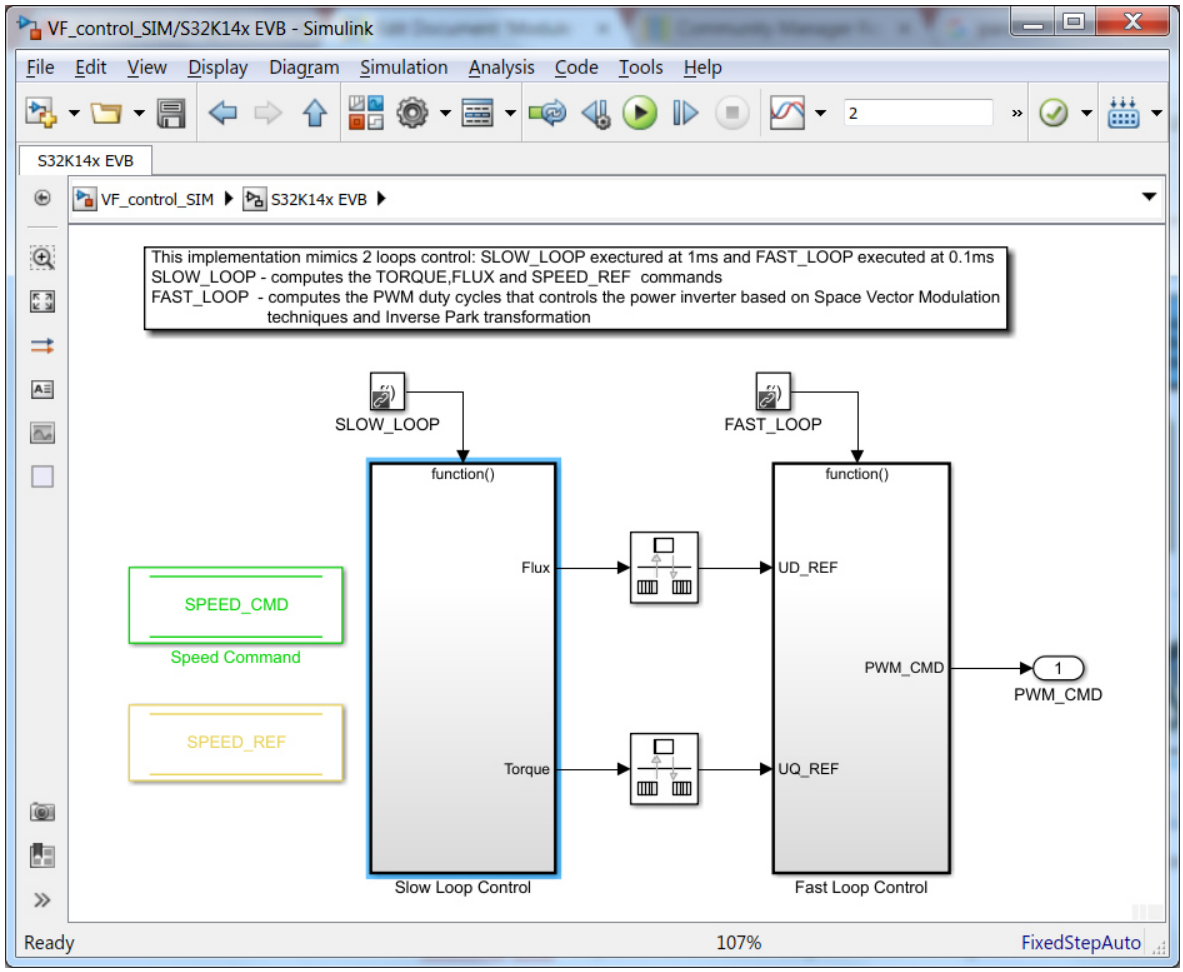
The model consists in three major subsystems that mimics the actual hardware arrangement, each one of them implementing a specific functionality:

- **S32K14x EVB** - implements the S32K MCU logic that allows the control algorithm to compute the PWM commands for the power inverter;
- **Motor GD DevKit** - implements a simplified model of power inverter that converts the PWM commands into high voltage continuous signals;

- **PMSM Model** - implements a simplified PMSM motor model that replicates the behavior for the real LINIX motor.

- Double click on the S32K14x EVB Subsystem

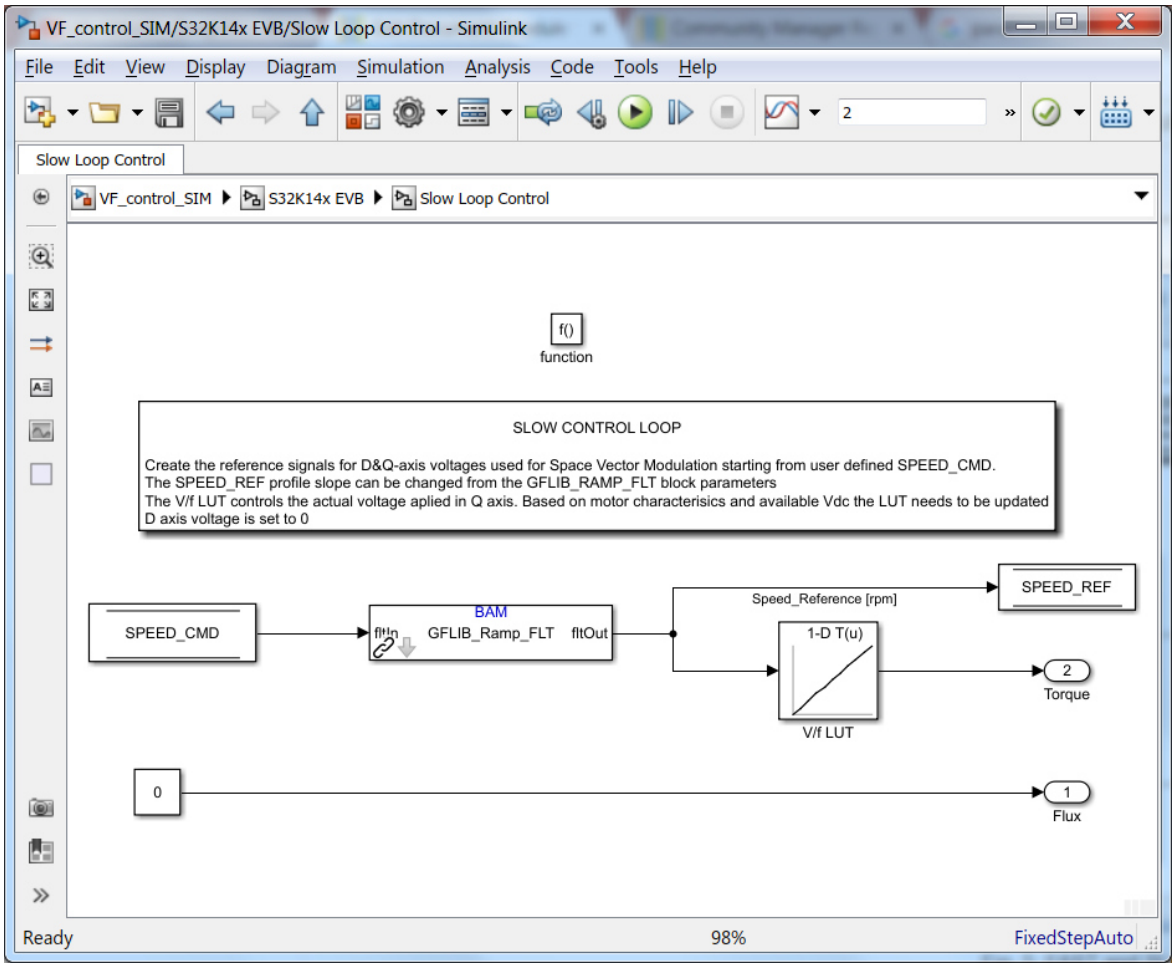
  This is a discrete model that implements the V/F scalar control block diagram. There are 2 subsystems that are triggered based on specific timing intervals. These 2 subsystem mimics the exitance of the separated digital control loops: SLOW & FAST



FAST and SLOW control loop model

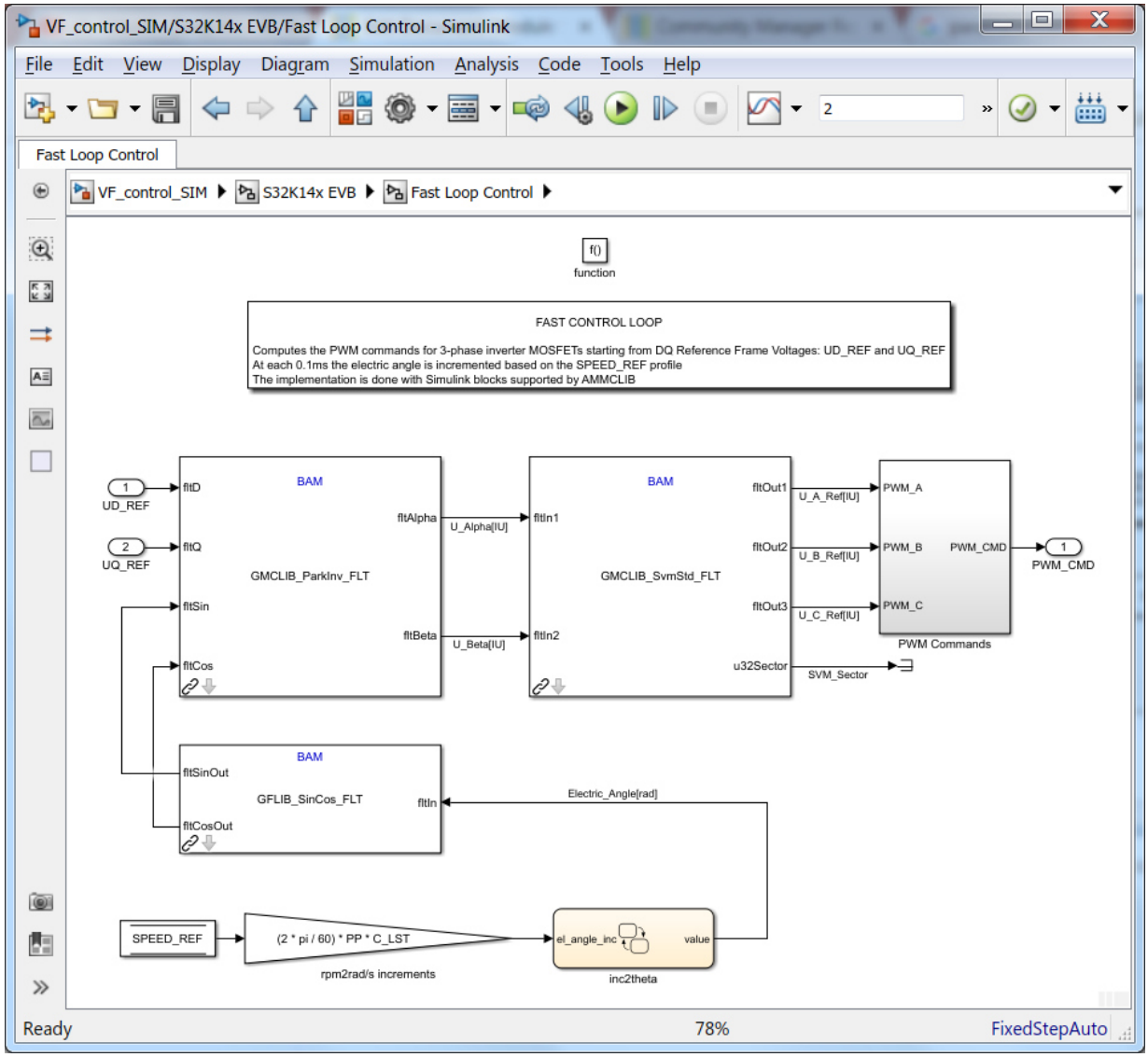
- The SLOW Loop Subsystem updates the commands for the FAST Loop Subsystem.

- Based on the SPEED_CMD set by the user as motor target speed, a trapezoidal speed profile is generated using a dedicated Simulink block exposed by the AMMCLIB Add-on. Based on this SPEED_REF the electrical angle used for Inverse Park transformation is going to be derived.



SLOW control loop model

- The quadrature voltages alpha-beta are computed based on the electrical angle and the (dq) reference voltages obtained in SLOW Loop. In case of UD_REF the value is set to 0 just to emulate the theory behind FOC and in case of UQ_REF the value is obtained from a Look-Up-Table (LUT) that implement the concept of V/F scalar control.



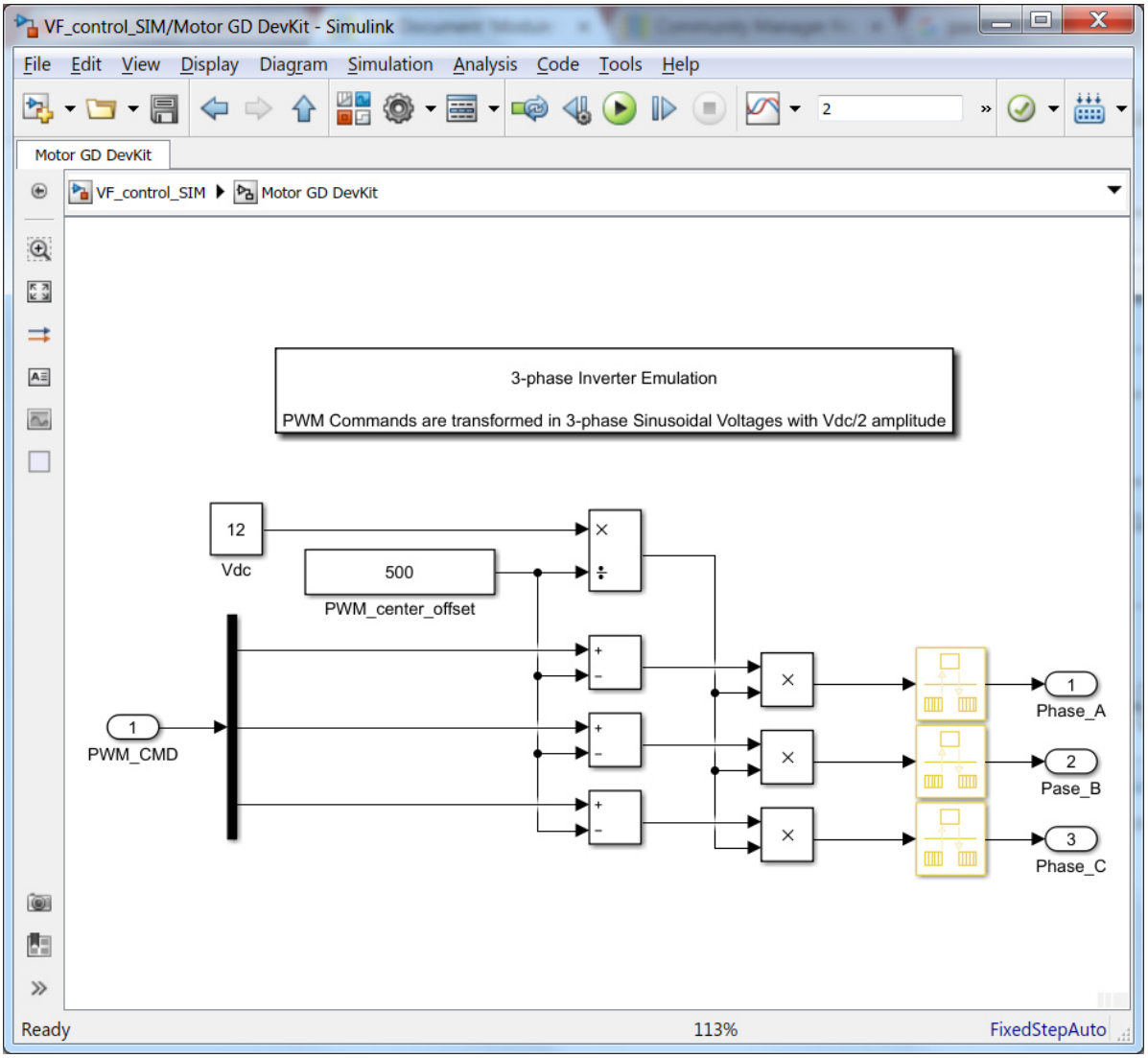
FAST control loop model

At each 0.1ms the PWM commands are computed and updated based on AMMCLIB specialized Simulink block that implements and optimized Space Vector Modulation with 3rd harmonic injection. These PWM commands are then passed to the next subsystem that emulates the operations for power inverter.

- Double click on MotorGD DevKit Subsystem

This subsystem implements a basic model to emulate the DC2AC power inverter. The PWM commands are converted into high voltage signals

(depending on the Vdc value selected) that are then applied to the motor windings.

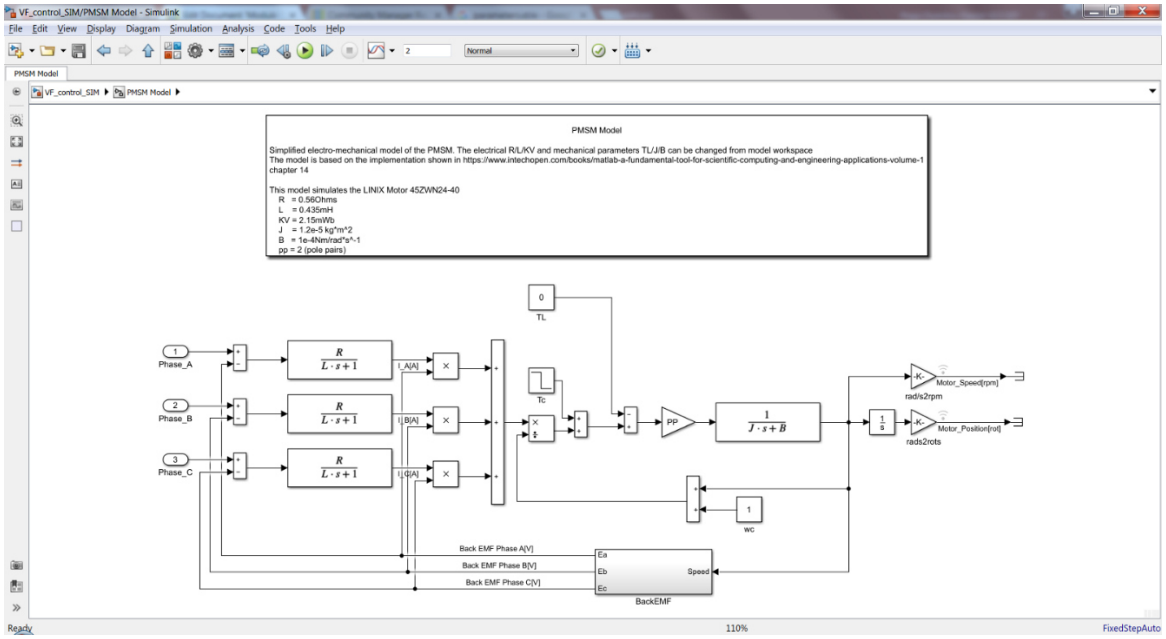


Simplified Power Inverter model

- Double click on PMSM Model Subsystem

  The Simulink model consists of:

  - stator windings electrical model
  - electro-mechanical model
  - sinusoidal back-EMF model

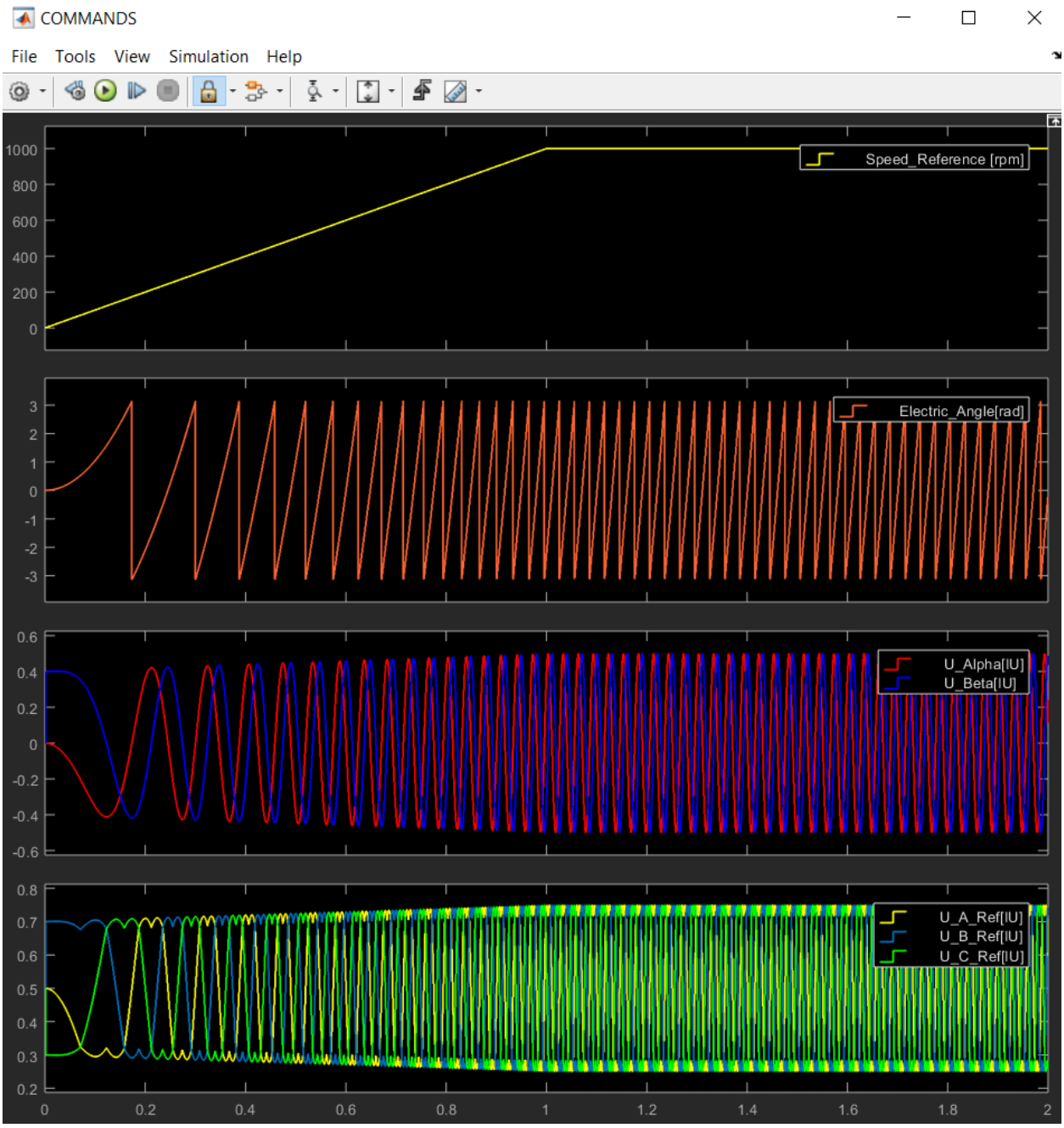


3-phase PMSM model

iii. MATLAB Simulation of the Plant

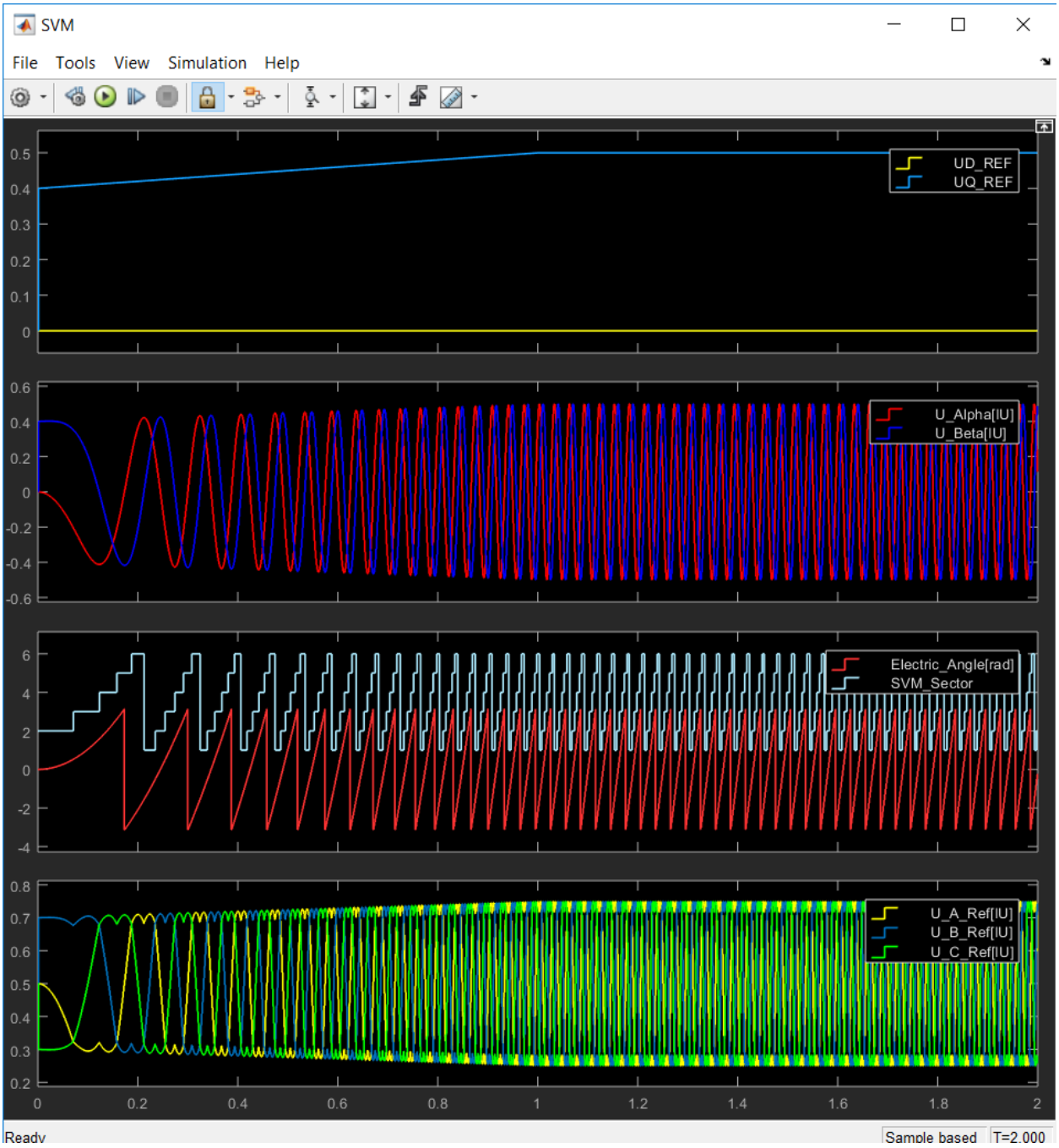
The entire V/F scalar control algorithm can be validated using this simulation environment provided in MATLAB/Simulink. This way we can validate the Space Vector Modulation, V/f scalar control and PMSM motor dynamic responses even without having the actual hardware setup.

- Click on  to run your simulation.
- Double click on the Commands scope.



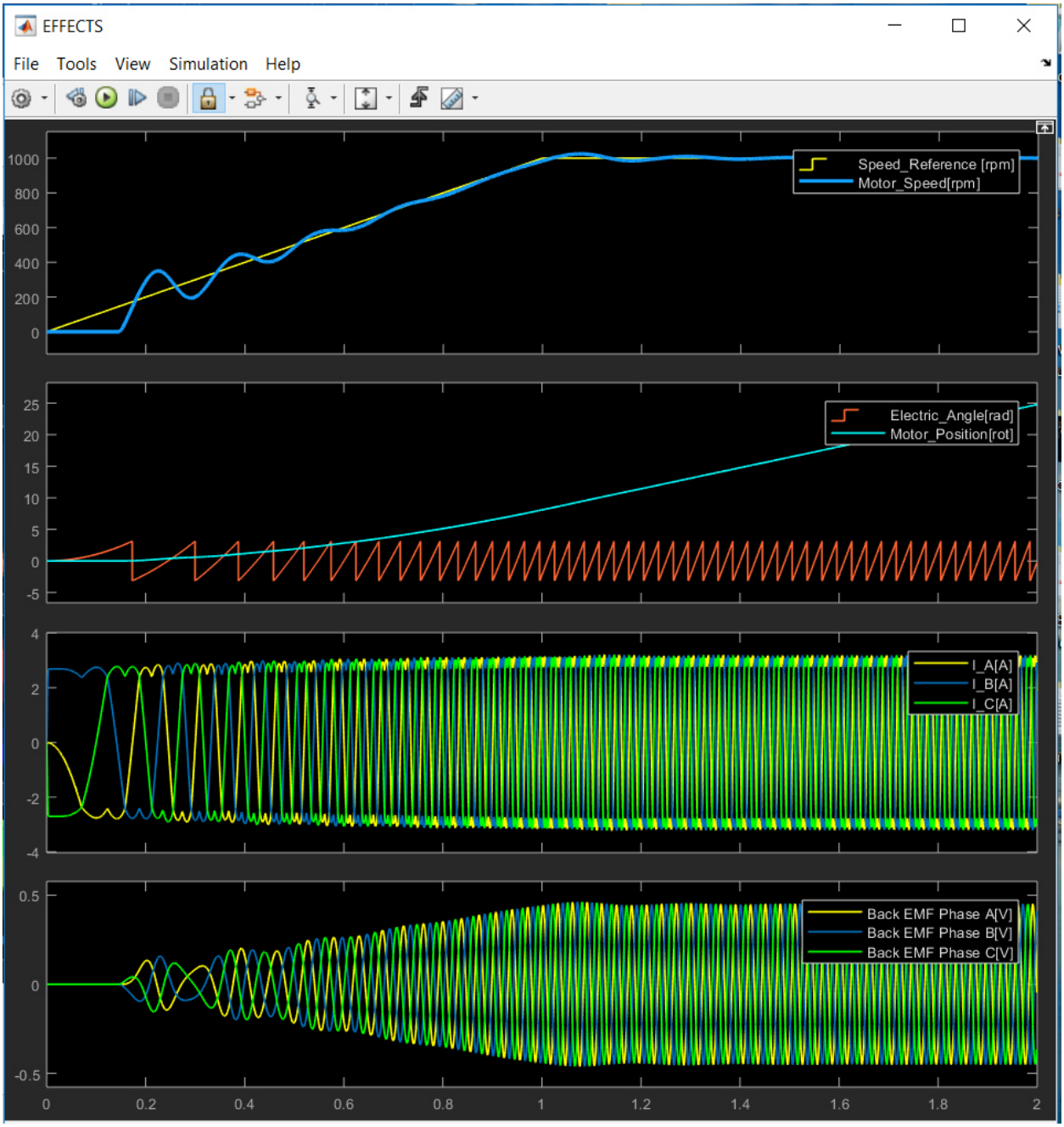
The above figure shows the V/F scalar control commands in case of start-up ramp: (1) Speed Profile, (2) Electric Angle, (3) quadrature voltage references and (4) 3-phase stator voltages references.

- Double click on the SVM scope.



The above figure shows the Space Vector Modulation verification: Inputs & Outputs

- Double click on the Effects scope.



The above shows the PMSM Model Responses in case of start-up sequence: (1) Command vs. Actual Motor Speed, (2) Rotor angle vs. Rotor position, (3) Stator Currents and (4) Back EMF voltages

The above three figures show the main control signals and outputs for each subsystems discussed earlier.

As can be seen there is a good match between the results obtained in simulation and the PMSM motor theory.