

C28x Digital Motor Control

Introduction

In this module, we will look into an application area that is not usually the domain of Digital Signal Processors: real-time control of electrical motors. In the old days, control of the speed and torque of electrical motors was done purely using analog technology. Since the appearance of microprocessors, more and more control units have been designed digitally, using the advantages of digital systems. This improves the degree of efficiency and allows the implementation of more advanced control schemes, thanks to increase real-time computing power. It is a natural progression to use the internal hardware computing units of a DSP to transfer the calculation from a standard microprocessor to a DSP. This way, we can implement more advanced algorithms in a given period of time.

However, to use a digital controller for motor control, the system needs a little more than computing power. The output signals of the digital controller to the power electronic are usually generated as pulse width modulated signals (PWM). It would be most cost-effective if the controller could be equipped with an internal PWM-unit. To control the operation of the motor we need to do some measurement for currents and voltages – analogue to digital converters (ADC) will be helpful as well. A typical unit to perform a position/speed measurement is an optical encoder; quite often, we build in a Quadrature Encoder (QEP). Recalling all parts of the C28x we discussed in chapters 1-9, you can imagine that the C28x is an ideal device for Digital Motor Control (DMC).

The chapter will not impart a deep knowledge of electrical motors and drives. Instead, it will give you a sense what needs to be done to use the C28x to control the motor of a vacuum cleaner or the motor of an electrical vehicle. To fully understand the principles, it needs a lot more classes at university. If you are in a course of electrical engineering that focuses on drives and power engineering, you might be familiar with most of the technical terms. If not, see this chapter as a challenge for you to open up another application field for a Digital Signal Processor.

Module 15 is based on a Texas Instruments Application Note (SPRC129, “C28x & F28x PMSM3_1: 3-Phase sensored Field Oriented Control“, Version 3.0, May-17-2003). Depending on the laboratory equipment at your university, you might be offered the chance to attend a laboratory session to build a working solution for such a motor control.

After a general discussion of the basics of motor control, we will focus on the computing steps necessary to build a control scheme for Field Oriented Control (FOC). This will be done in an incremental way, to help you to understand the computing steps. We will not discuss any power electronic hardware requirements or design constraints.

A complete working reference design is available in the appendix of this CD-ROM, based on a small 24V 3-phase PMSM, the eZdspTMS320F2812 and the DMC550, a power electronic adapter board available from Spectrum Digital.

Module Topics

C28x Digital Motor Control	15-1
<i>Introduction</i>	<i>15-1</i>
<i>Module Topics.....</i>	<i>15-2</i>
<i>Basics of Electrical Motors.....</i>	<i>15-3</i>
Motor Categories	15-3
3 – phase Motor	15-4
Permanent Magnet Synchronous Motor	15-6
Brushless Direct Current Motor (BLDC)	15-7
3 – Phase Power Switches	15-8
<i>Motor Control Principles.....</i>	<i>15-9</i>
Scalar Control (“V/Hz”)	15-9
Field Oriented Control (FOC)	15-10
FOC Control Scheme.....	15-11
<i>FOC Core Math Operations</i>	<i>15-12</i>
PARK Transform.....	15-12
CLARKE Transform	15-14
PARK Transform Summary	15-15
<i>Texas Instruments Digital Motor Control Library.....</i>	<i>15-16</i>
Library Modules	15-18
FOC for PMSM	15-19
Hardware Laboratory Setup.....	15-20
PMSM Library Modules.....	15-22
PMSM Software Flowchart.....	15-23
<i>Lab 15: PMSM control project</i>	<i>15-24</i>
Build Level 1	15-24
Build Level 2	15-28
Build Level 3	15-30
Build Level 4	15-32
Build Level 5	15-35

Basics of Electrical Motors

Motor Categories

In order to classify the different electrical motors families, we can distinguish motors driven by direct current (DC) and motors driven by an alternating current (AC). DC motors are the most popular ones: both stators and rotors carry an excitation created by coils or windings in which DC current circulates. In order to ensure the motor rotation by commutating the windings, brushes are permanently in contact with the rotor.

Electrical Motor families

Motor Classification:

◆ Direct Current Motors (DC)

◆ Alternating Current Motors (AC)

- ◆ Asynchronous Induction Motor (ACI)
- ◆ Permanent Magnet Synchronous Motor (PMSM)
- ◆ Synchronous Brushless DC Motor (BLDC)

15 - 2

Under the classification of AC motors, we have synchronous motors and asynchronous motors; both motor types are induction machines.

Asynchronous machines require a sinusoidal voltage distribution on the stator phases in order to induce current on the rotor, which is not fed by any currents nor carries any magnetic excitation.

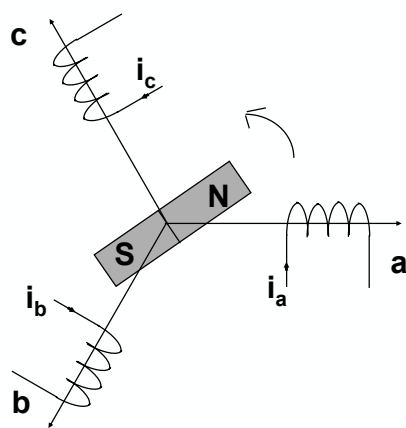
Synchronous motors are usually called “Brushless DC Motors” (BLDC) but can also be called “Permanent Magnet Synchronous Motors” (PMSM) depending on their construction and the way of being controlled. In this type of motor, we have one sinusoidal or trapezoidal excitation on the stator and one constant flux source on the rotor (usually a magnet).

Because of the selected application note as the base of this module, we will focus on the PMSM – type for the rest of this module.

3 – phase Motor

Let us start with a brief discussion of a 3-phase motor. We can start with a magnet rotating in front of an electrical winding. This can be expanded into a three phase winding approach. Windings a, b and c are physically spaced 120° apart from each other. While the magnet is rotating with a mechanical speed Ω , each winding sees a varying flux that induces voltage and current at their ends. Applying the Faraday law to each of the phases, we can obtain a three-phase equation system that will be the base of our rotating machine study.

3 – Phase - Motor (PMSM)



$$\begin{cases} i_a = I_s \cdot e^{j\omega t} \\ i_b = I_s \cdot e^{j\omega t - \frac{2\pi}{3}} \\ i_c = I_s \cdot e^{j\omega t - \frac{4\pi}{3}} \end{cases}$$

$$\begin{cases} e_a = E \cdot e^{jp\Omega t} \\ e_b = E \cdot e^{jp\Omega t - \frac{2\pi}{3}} \\ e_c = E \cdot e^{jp\Omega t - \frac{4\pi}{3}} \end{cases}$$

- ◆ For most three phase machines, the winding is stationary, and magnetic field is rotating
- ◆ Three phase machines have three stator windings, separated 120° apart physically
- ◆ Three phase stator windings produce three magnetic fields, which are spaced 120° in time

15 - 3

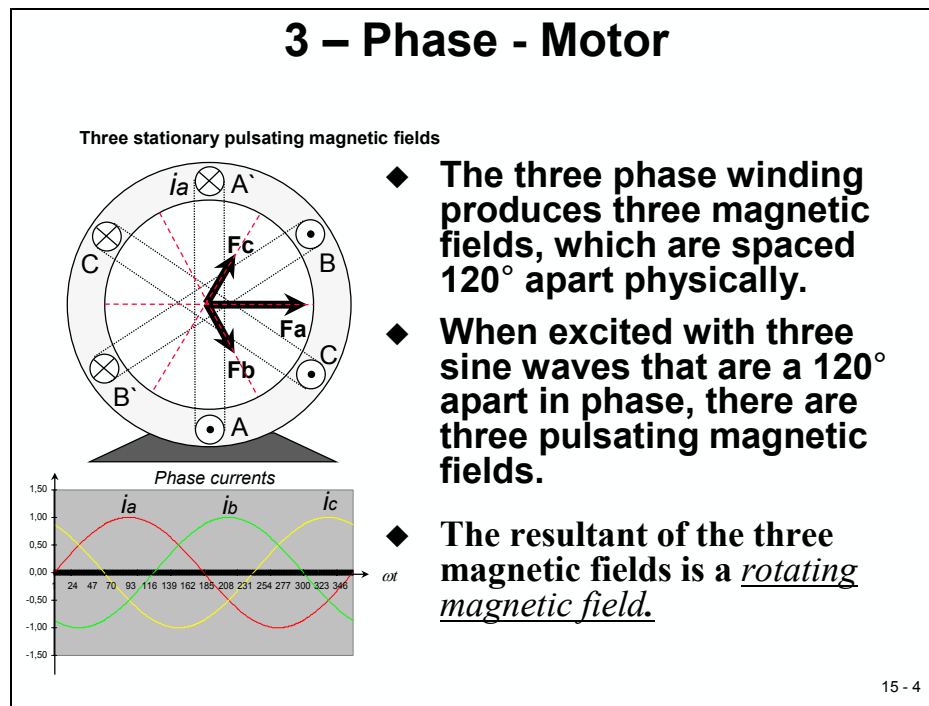
Depending on the number of pole pairs (p) of the rotating magnet, the electrical pulsation of the induced voltage will change. For a two pole pairs rotating magnetical excitation, the electrical pulsation is twice the mechanical pulsation. For a complete 360° turn of the magnet, the windings see two North and South poles. This observation for two pole pair magnet can be generalized to $\omega = p \cdot \Omega$ with electrical (ω) and mechanical (Ω) pulsation. Parameter e is the inductive voltage, called „Back Electromotive Force (Bemf)“.

We can represent the system using the following equations (in real notation as an example):

$$\begin{aligned} e_a &= \Phi \omega \sqrt{2} \sin(\omega t) = E \sqrt{2} \sin(p\Omega t) \\ e_b &= \Phi \omega \sqrt{2} \sin\left(\omega t - \frac{2\pi}{3}\right) = E \sqrt{2} \sin\left(p\Omega t - \frac{2\pi}{3}\right) \\ e_c &= \Phi \omega \sqrt{2} \sin\left(\omega t - \frac{4\pi}{3}\right) = E \sqrt{2} \sin\left(p\Omega t - \frac{4\pi}{3}\right) \end{aligned}$$

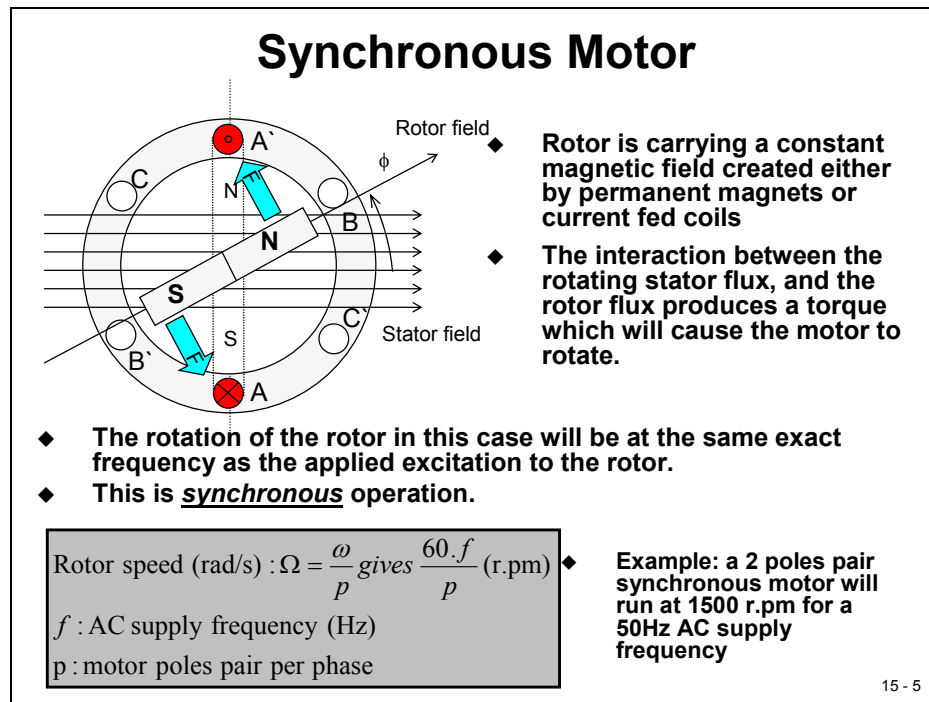
Instead of rotating the magnet in front of windings and inducing a flux variation and a B_{emf} inside the three phases we can turn the principle around:

We can place a sinusoidal source current distribution on the stator that creates a rotating magnetic field. In this case, we apply a rotating force to a magnetic field placed in the center of the stator. This leads us to the principle of the synchronous motor.



Permanent Magnet Synchronous Motor

Synchronous motor construction: Permanent magnets glued or screwed tightly to the rotating axis create the rotor flux (constant). When excited, the stator windings create electromagnetic poles. By controlling the stator currents, we control the stator magnetic field and the revolution of the rotor.



The permanent magnet approach is suitable for synchronous machine ranging up to a few kilo Watts (kW). For higher power ranges, the rotor is composed of windings in which a DC current circulates with the appropriate sequence of North and South poles, in order to get the desired pole pair number.

This action of the rotor chasing after the electromagnet poles on the stator is the fundamental action used in synchronous permanent magnet motors.

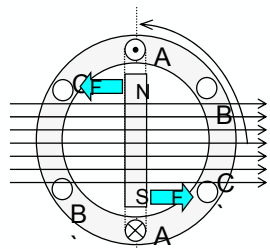
The phase difference between the rotor and the rotating stator field must be controlled to produce torque and to achieve a high degree of efficiency. This implies:

- The rotating stator field must revolve at the same frequency as the rotor permanent magnetic field. If not, the rotor will stop chasing after the stator field.
- The angle between the rotor field and the stator field must be equal to 90° to obtain the highest mutual torque production. This synchronization requires knowing the rotor position in order to generate the right stator field.

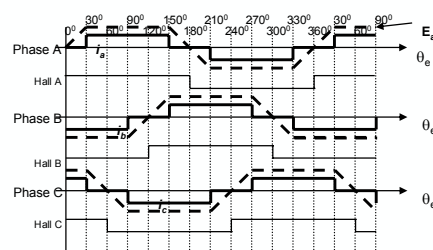
Brushless Direct Current Motor (BLDC)

We can distinguish two types of synchronous motor: Brushless Direct Current motor (BLDC) and Permanent Magnet Synchronous Motor (PMSM). This terminology defines the shape of the Back EMF of the synchronous motor. Both a BLDC and a PMSM motor can have permanent magnets on the rotor, but different Back EMF shapes. It is mainly linked to the way of mounting and disposing of the magnets on the rotor. To get the best performances out of the synchronous motor, it is important to identify the type of motor in order to apply the most appropriate type of control scheme.

Synchronous Motors: BLDC and PMSM

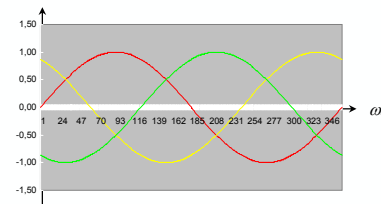


Back EMF of BLDC Motor



- ◆ Both (typically) have permanent-magnet rotor and a wound stator
- ◆ BLDC (Brushless DC) motor is a permanent-magnet brushless motor with trapezoidal back EMF
- ◆ PMSM (Permanent-magnet synchronous motor) is a permanent-magnet brushless motor with sinusoidal back EMF

Back EMF of PMSM



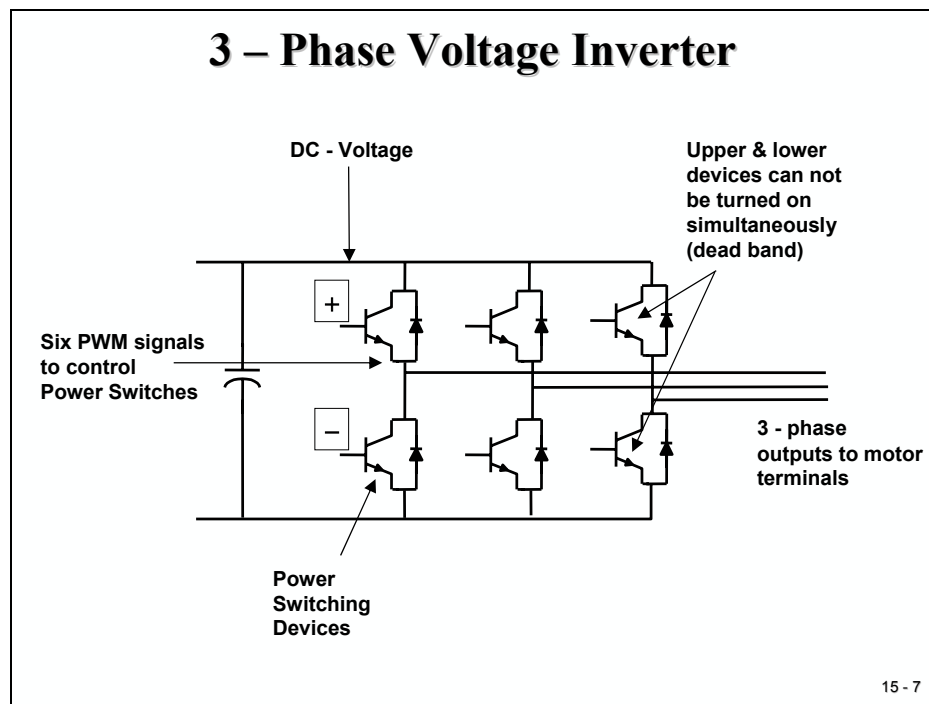
15 - 6

3 – Phase Power Switches

As we saw in the previous basic diagrams, we need to apply three 120° phase shifted excitation signals into the power circuitry of the motor. As you have seen in Module 5 („Event Manager“) a PWM signal can be used to modulate sine wave shaped signals. With three independent switching pattern streams and six power switches, we can deliver the necessary phase voltages to generate the required torque imposed by the load. The goal is to build the correct IGBT's conduction sequences to deliver sine wave shaped currents to the motor to transform it in a mechanical rotation.

This is traditionally achieved by comparing a three-phase sinusoidal waveform with a triangular carrier. In the digital world, on the DSP processor, we compute a sinusoidal command and apply it to the PWM units that generate the appropriate PWM outputs usually connected to gate drivers of the IGBT's from the inverter.

Basically we are “chopping” a DC voltage, carried by the DC bus capacitor, in order to build the appropriate voltage shapes to the stator phases, with the goal of having a good efficiency during this energy conversion process. This is a power electronics concern: we need to minimize the noise introduced by these conducting sequences source of harmonics.



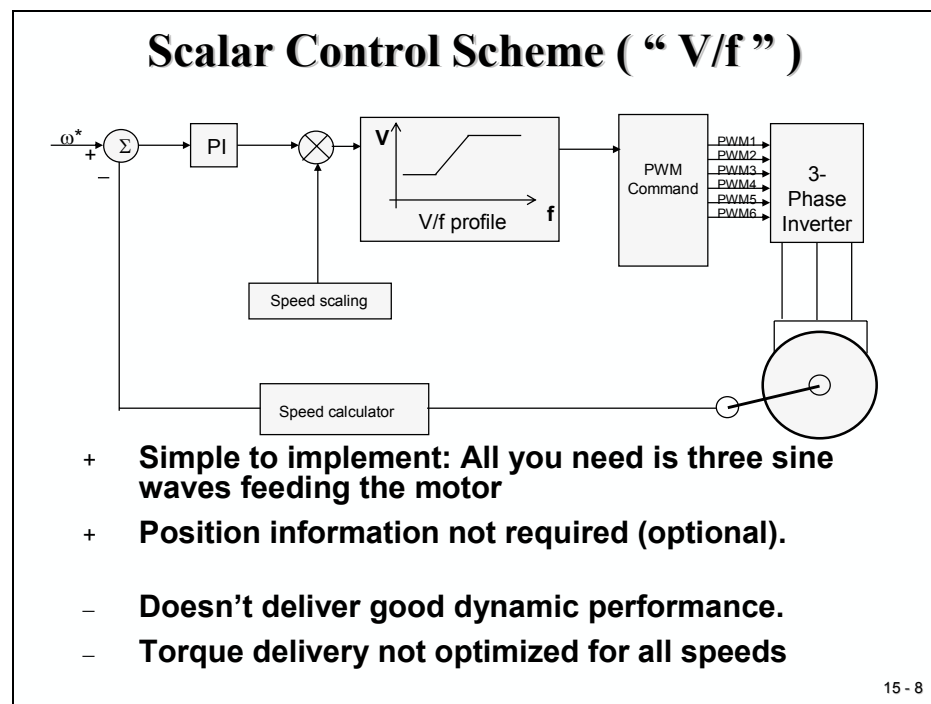
Motor Control Principles

Scalar Control (“V/Hz”)

The V/Hz regulation scheme is the simplest one that can be applied for an **asynchronous motor**. The goal is to work in an area where the rotor flux is constant (Volts proportional to speed).

In practical solutions, the speed sensor is optional as the control is tuned to follow a predefined “speed-profile versus load“- table, assuming the load characteristics are known in advance.

Obviously, this type of control bases itself on the steady electrical characteristic of the machine and assumes that we are able to work with a constant flux on the complete speed range the application targets. This is why this type of control does not deliver a good dynamic performance and a good transient response time; the V/Hz profile is fixed and does not take into account conditions other than those seen in a steady state. The second point is the problem at startup AC induction motors, which cannot deliver high torques at zero speed; in this case, the system cannot maintain a fixed position. In practice for low speed, we need to increase the delivered voltage to the stator compared to the theoretical V/Hz law.



Field Oriented Control (FOC)

Instead of using a pure sine wave shaped modulation of the PWM stage, in recent years the space vector theory has demonstrated some improvements for both the output crest voltage and the harmonic copper loss. The maximum output voltage based on the space vector theory is 1.155 times larger than the conventional sinusoidal modulation. It makes it possible to feed the motor with a higher voltage than the simpler sub-oscillation modulation method. This modulator enables higher torque at high speeds, and a higher efficiency. Torque distortion is also reduced.

The space vector PWM technique implemented into the existing TI DMC library reduces the number of transistor commutations. It therefore improves EMI behavior.

Field Oriented Control (FOC)

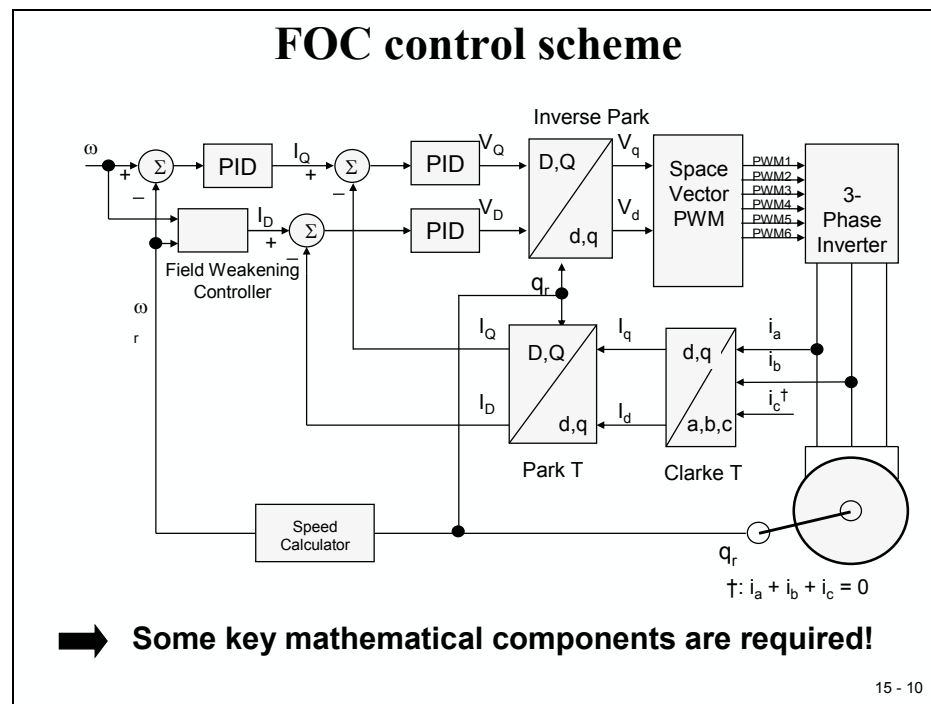
- ◆ **Field Oriented Control (FOC) or Vector Control, is a control strategy for 3-phases induction motors where the torque producing and magnetizing components of the stator flux are separately controlled.**
- ◆ **The approach consists in imitating the DC motors' operation**
- ◆ **FOC will be possible with system information: currents, voltages, flux and speed.**

15 - 9

A typical characteristic of FOC - PWM command strategy is that the envelope of the generated signal is carrying the first and the third harmonics. We can interpret this as a consequence of the special PWM sequence applied to the power inverters. Literature also mentions the third harmonic injection to boost out the performance we get out of the DC bus capacitor. This third-harmonic exists in the phase to neutral voltage but disappears in the phase-to-phase voltage.

FOC Control Scheme

The overall system for implementation of the 3-phase PMSM control is depicted in the next slide. The PMSM is driven by the conventional voltage-source inverter. The C28x is generating six pulse width modulation (PWM) signals by means of space vector PWM technique for six power-switching devices in the inverter. Two input currents of the PMSM (i_a and i_b) are measured from the inverter and they are sent to the C28x via two analog-to-digital converters (ADCs).



Theoretically, the field oriented control for the PMSM drive allows the motor torque be controlled independently with the flux-like DC motor operation. In other words, the torque and flux are decoupled from each other. The rotor position is required for variable transformation from stationary reference frame to synchronously rotating reference frame. As a result of this transformation, the so called Park transformation, the q-axis current will be controlling torque while the d-axis current is forced to zero for a PMSM. Here “d” means direct and “q” means Quadrature.

Therefore, an important key module of this system is the information of rotor position from QEP encoder.

Note that the same control scheme with an additional field weakening controller can be used to control 3-phase asynchronous AC induction motors.

FOC Core Math Operations

PARK Transform

The PARK transform is not something new. This theory has been around for 75 years. As we will see, this technique requires a large amount of mathematical calculations involving in particular matrix multiplications. Thanks to new control processor technologies, it becomes now possible to use this real-time control technique.

Let us consider the voltage vector (V_s) applied to the stator of the three phase machine we are working on (ACI or PMSM).

$$(V_s) = \begin{bmatrix} v_{s1} \\ v_{s2} \\ v_{s3} \end{bmatrix}$$

This theory is not limited to sinusoidal distribution and can be applied to any kind of vector. For now, we will consider the general case.

The PARK transform is only a referential change defined as below:

PARK transform (1929):

- ◆ **(V_s): voltage vector applied to motor stator (index s)**
- ◆ **Park transform is a referential change**

$$(V_s) = \begin{bmatrix} v_{s1} \\ v_{s2} \\ v_{s3} \end{bmatrix}$$

$$\begin{bmatrix} v_{s1} \\ v_{s2} \\ v_{s3} \end{bmatrix} = \begin{bmatrix} \cos \theta_s & -\sin \theta_s & 1 \\ \cos(\theta_s - \frac{2\pi}{3}) & -\sin(\theta_s - \frac{2\pi}{3}) & 1 \\ \cos(\theta_s - \frac{4\pi}{3}) & -\sin(\theta_s - \frac{4\pi}{3}) & 1 \end{bmatrix} \begin{bmatrix} v_{sd} \\ v_{sq} \\ v_{so} \end{bmatrix}$$

$$\begin{bmatrix} v_{s1} \\ v_{s2} \\ v_{s3} \end{bmatrix} = [P(\theta_s)] \begin{bmatrix} v_{sd} \\ v_{sq} \\ v_{so} \end{bmatrix} \text{ and } \begin{bmatrix} v_{sd} \\ v_{sq} \\ v_{so} \end{bmatrix} = [P(\theta_s)]^{-1} \begin{bmatrix} v_{s1} \\ v_{s2} \\ v_{s3} \end{bmatrix}$$

15 - 11

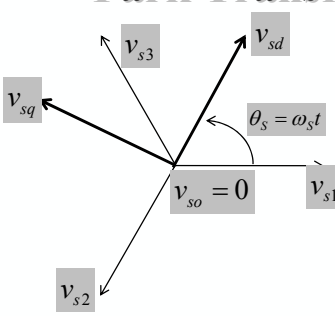
The index “S” indicates that we work with the stator parameters. This referential change transforms the V_s vector in a new vector as written below:

$$\begin{bmatrix} v_{s1} \\ v_{s2} \\ v_{s3} \end{bmatrix} = [P(\theta_s)] \begin{bmatrix} v_{sd} \\ v_{sq} \\ v_{s0} \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} v_{sd} \\ v_{sq} \\ v_{s0} \end{bmatrix} = [P(\theta_s)]^{-1} \begin{bmatrix} v_{s1} \\ v_{s2} \\ v_{s3} \end{bmatrix}$$

PARK coordinates will play an important role in the FOC control from a regulation point of view. Instead of using the general PARK transform, literature prefers the normalized PARK transform for which inverted matrix is much easier to calculate and which allows building an orthogonal referential change. The normalized PARK transform is defined as follows:

$$[P(\theta_s)] = \sqrt{\frac{2}{3}} \begin{bmatrix} \cos \theta_s & -\sin \theta_s & \frac{1}{\sqrt{2}} \\ \cos(\theta_s - \frac{2\pi}{3}) & -\sin(\theta_s - \frac{2\pi}{3}) & \frac{1}{\sqrt{2}} \\ \cos(\theta_s - \frac{4\pi}{3}) & -\sin(\theta_s - \frac{4\pi}{3}) & \frac{1}{\sqrt{2}} \end{bmatrix}$$

Park Transform key components



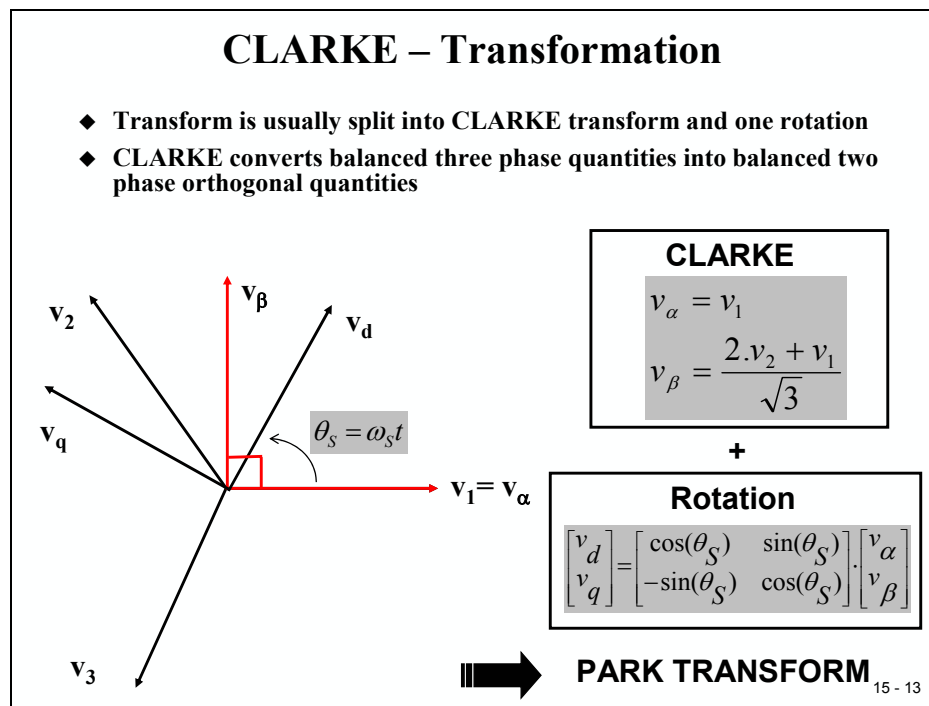
$\begin{cases} \vec{v}_{s1} + \vec{v}_{s2} + \vec{v}_{s3} = \vec{0} \quad (\text{tri-phases balanced system}) \\ \vec{v}_{sd} \cdot \vec{v}_{sq} = 0 \\ \vec{v}_{sd} \cdot \vec{v}_{so} = 0 \\ \vec{v}_{sq} \cdot \vec{v}_{so} = 0 \end{cases}$

- ◆ (v_{sd}, v_{sq}, v_{so}) are called the Park coordinates
- ◆ v_{sd} : direct Park component
- ◆ v_{sq} : squaring Park component
- ◆ v_{so} : homo-polar Park component
- ◆ v_{so} is null for a three-phases balanced system
- ◆ Each pair of components is perpendicular to each other

15 - 12

CLARKE Transform

The normalized PARK can be seen as the result of the combination of the CLARKE transform combined with a rotation. Literature sometimes refers to PARK in this way: this is the case for the TI Digital Motor Control library. This gives an intermediate step that helps to build the regulation scheme.

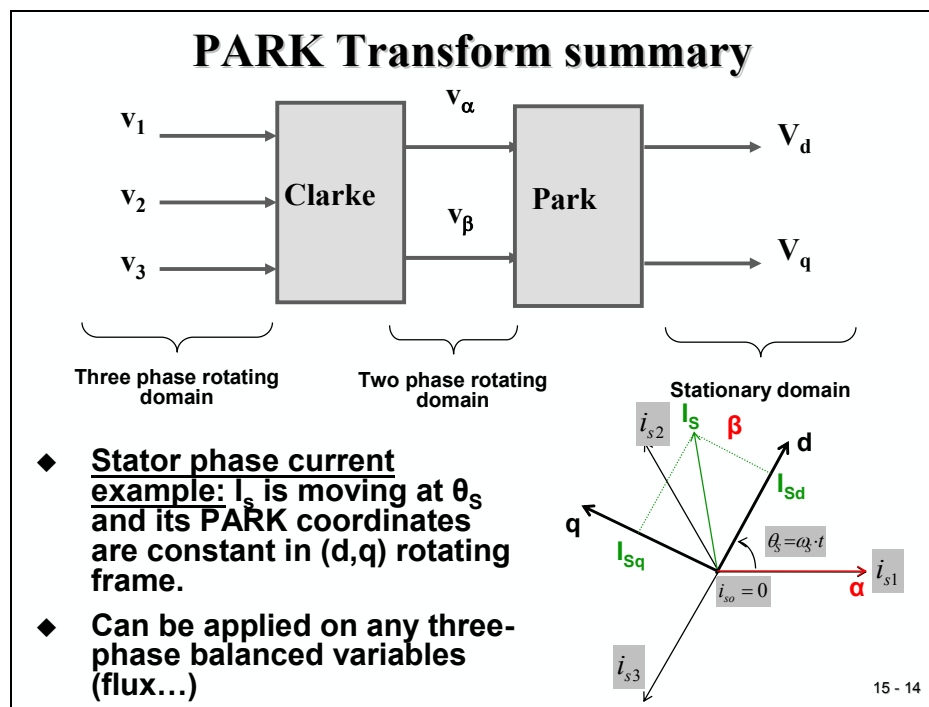


We start from a three-phase balanced system that we first transform in a two-phase balanced system: this is the role of the CLARKE transform that defines a two-phase rotating frame (α , β) that rotates at the speed of the rotating stator magnetic field (ω_s). If we “sit” on the rotating frame (α , β) then we see stationary variables. This is the role of the θ_s angle rotation that allows us to move from the rotating domain to the stationary domain.

PARK Transform Summary

In the next slide, the two modules “PARK” and “CLARKE” are shown in a “one-piece” transform. Texas Instruments DMC library uses two C-callable functions to perform the coordinate transform. The library functions are based on the fixed point math „IQ-Math“.

Combining the CLARKE and PARK transforms as defined above, we move from the three phase rotating domain to the stationary domain: we just need to control DC quantities in real-time.



To drive the power switches with new calculated values we have to re-transform these stationary control values back into the three phase rotating domain. This is done with a similar transform function called “Inverse PARK”, shown in slide 15-10.

The control itself is done with 3 instances of a C-callable function “PID” (see slide 15-10). This function implements a discrete proportional (P)-integral (I)-derivative (D) control scheme with an additional anti-windup feedback. To learn more about the theory refer to file „pid_reg3.pdf“ in the appendix.

Texas Instruments Digital Motor Control Library

Texas Instruments Digital Motor Control (DMC) Library is available free of charge and can be downloaded from the website. It consists of a number of useful functions for motor control applications. Among those functions, there are pure motor control modules (Park and Clark transforms, Space Vector PWM, ...) as well as traditional control modules (PID controller, ramp generator, ...) and peripherals drivers (for PWM, ADC, ...)

Based on this DMC library, Texas Instruments has developed a number of application notes for different types of electrical motors. All applications examples are specially designed for the C2000 platform and come with a working example of the corresponding software, background information and documentation.

One branch of this library is dedicated to the C28x and takes advantage of the 32 Bit IQ-Math data format.

The following slide shows the examples available for the C28x:

Texas Instruments Motor Control Solution

“C2000 - Digital Motor Control Library (DMC)” :

- ◆ **Single Phase ACI Motor Control Using Constant V/Hz**
- ◆ **3-Phase ACI Motor Constant V/Hz Control**
- ◆ **3-Phase ACI Motor Field Oriented Control**
- ◆ **3-Phase Sensored Field Oriented Control (PMSM)**
- ◆ **3-Phase Sensorless Field Oriented Control (PMSM)**
- ◆ **3-Phase Sensored Trapezoidal Control (BLDC)**
- ◆ **3-Phase Sensorless Trapezoidal Control (BLDC)**

15 - 15

In the remaining part of this chapter we will focus on the “C28x & F28x PMSM3_1: 3-phase Sensored Field Oriented Control” – Literature Number SPRC129. The laboratory setup will be based on the following hardware modules:

- TMS320F2812 eZdsp (Spectrum Digital Inc.) with 5V DC power supply
- DMC550 power drive adapter board (Spectrum Digital Inc.),
- A 24V 3-Phase PMSM (Applied Motion Inc.)
- External 24V – DC power supply unit

Digital Motor Control Library (DMC-Lib)

- ◆ **The DMC-Library is a collection of most commonly used algorithms and function blocks for motor control systems**
- ◆ **For every algorithm and function:**
 - **Essential theoretical background information**
 - **Data types for input/output parameters with numerical range and precision**
 - **Function prototypes and calling conventions**
 - **code size (program and data memory)**
 - **Build Level based code examples**

15 - 16

NOTE:

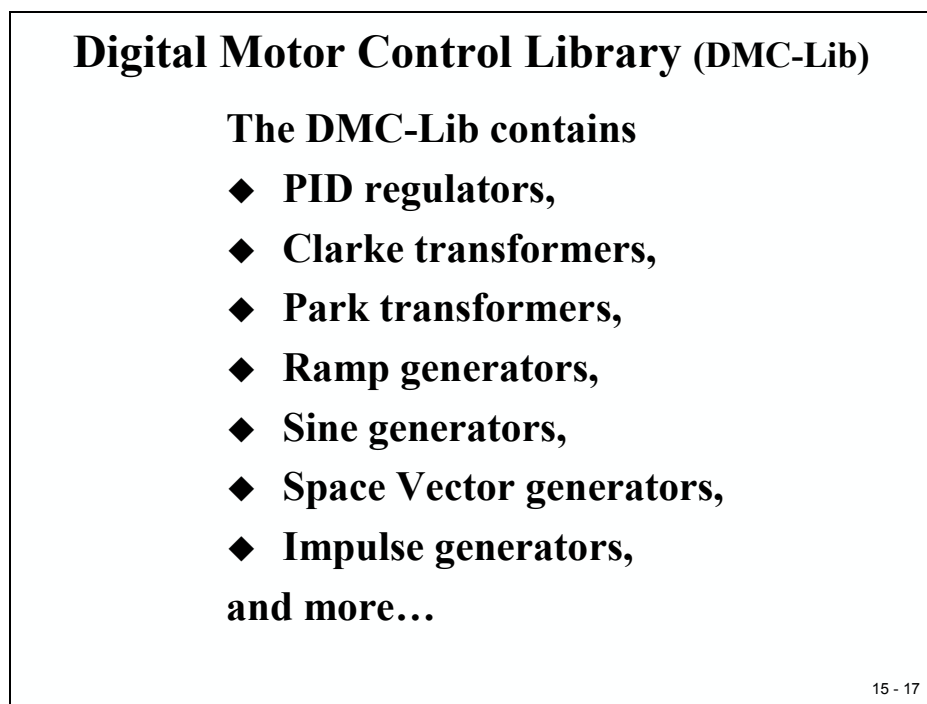
Depending on existing equipment at your university laboratory, you might be able to attend sessions based on other motor types. The procedure for all library solutions will be similar and is based on TI’s modular approach for each of the motor solutions.

To accomplish the lab exercises you will need an additional student user guide, which is tailored to your university lab.

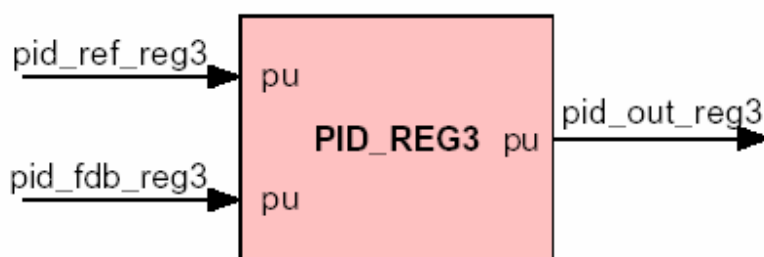
Ask your professor if this optional laboratory exercise is available to you.

Library Modules

The following slide lists all groups of C-callable modules which are part of the library. Each block is explained in detail with its accompanying documentation. After the installation of the library, a separate subfolder (“C:\tides\dmc\c28\lib\dmclib\doc”) includes the documentation.



For example, the file “pid_reg3.pdf” explains the interface and the background of the PID-controller:



All functions are coded for 32-Bit variables in IQ-Math-format, which was explained in module 11. Successful completion of module 11 is necessary to be up to speed with the following software modules. All functions are used as instances of a predefined object class, based on a structure definition in a header file.

The following slide is the data flow chart of the complete application for a 3-Phase PMSM. This software project will be implemented step-by-step during the laboratory.



It consists of three PID-controller loops that are executed periodically. This period is defined to 20 (in kHz) by parameter “ISR_FREQUENCY” in file “parameter.h” and can be adapted to the needs of the motor, which is used in your laboratory.

The coordinate transform modules, which we discussed previously, form part of the control scheme as well as the QEP – support module to estimate speed and position of the rotor.

Hardware Laboratory Setup

Before we busy ourselves with the software project, let's summarize the hardware equipment that we need to continue:

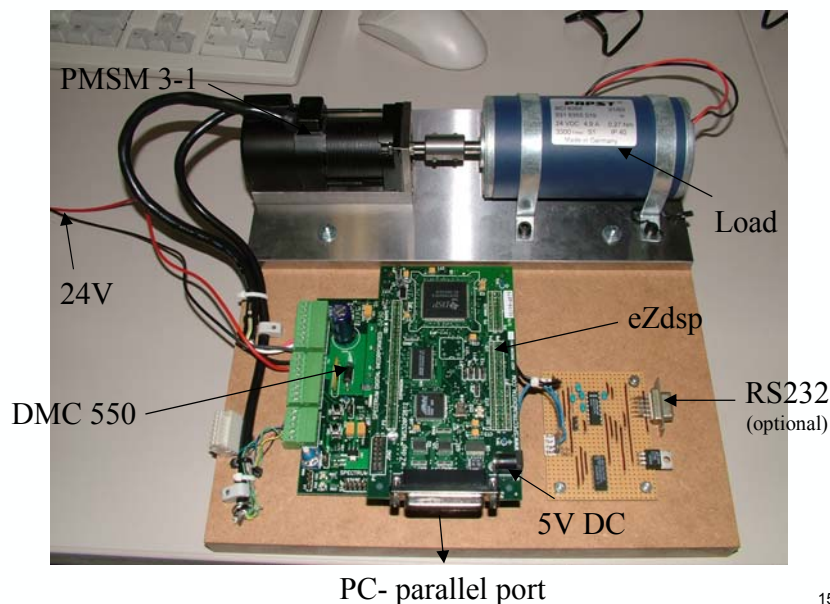
TI Library Solution “PMSM 3-1” (sprc129)

Hardware for Laboratory setup:

- Spectrum Digital eZdsp TMS320F2812
- Spectrum Digital DMC550 drive platform
- 3-phase PMSM with a QEP encoder
 - Applied Motion 40mm Alpha Motor
 - Type: A0100-104-3-100
- 24V DC power supply (DC bus voltage)
- load , e.g. DC Motor as Generator
- PC parallel port to JTAG
- 5V DC (eZdsp)
- RS232 (optional)
- Oscilloscope

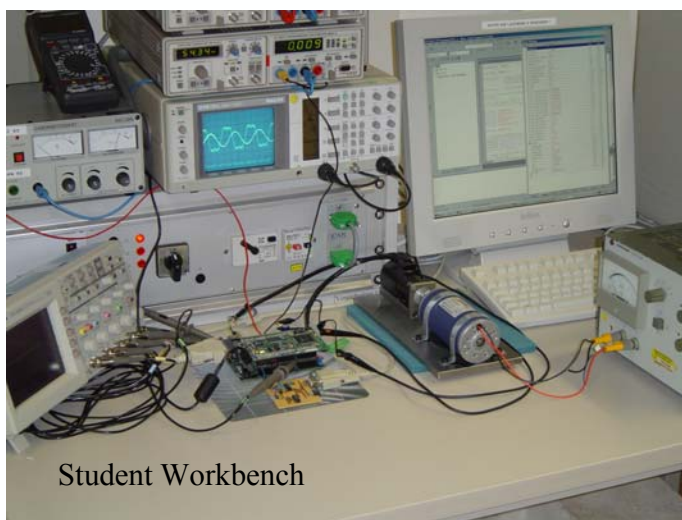
15 - 19

TI Library Solution “PMSM 3-1” (sprc129)



15 - 20

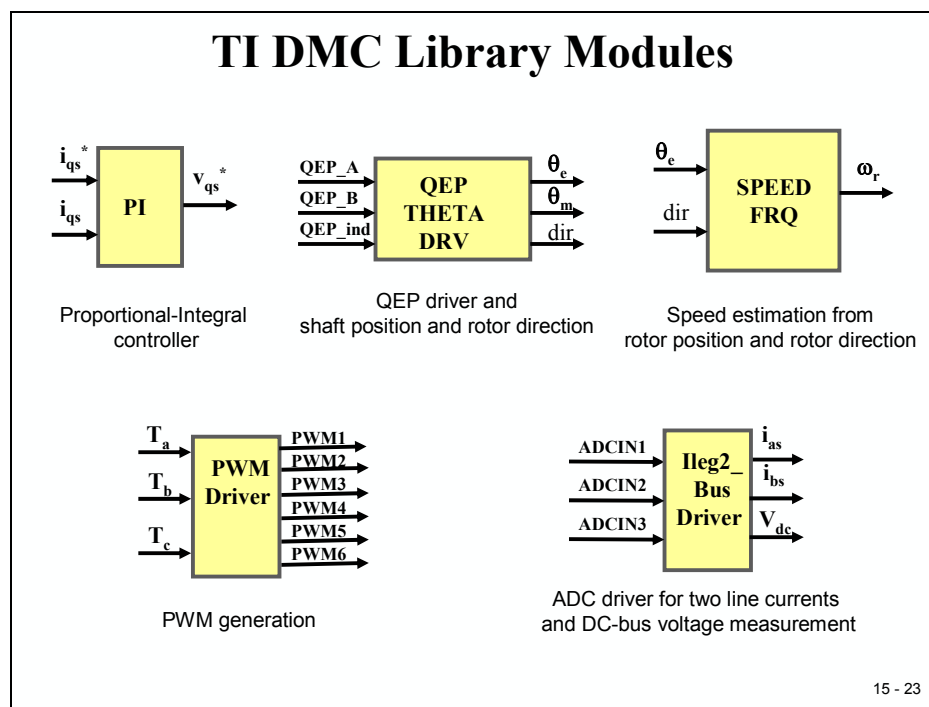
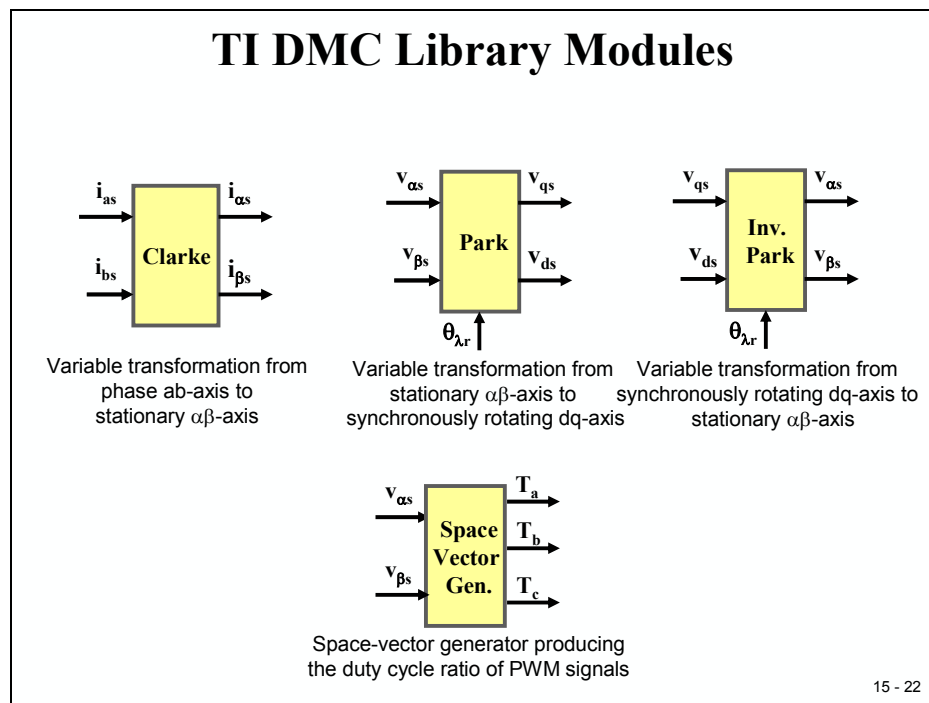
PMSM 3-1 Laboratory



15 - 21

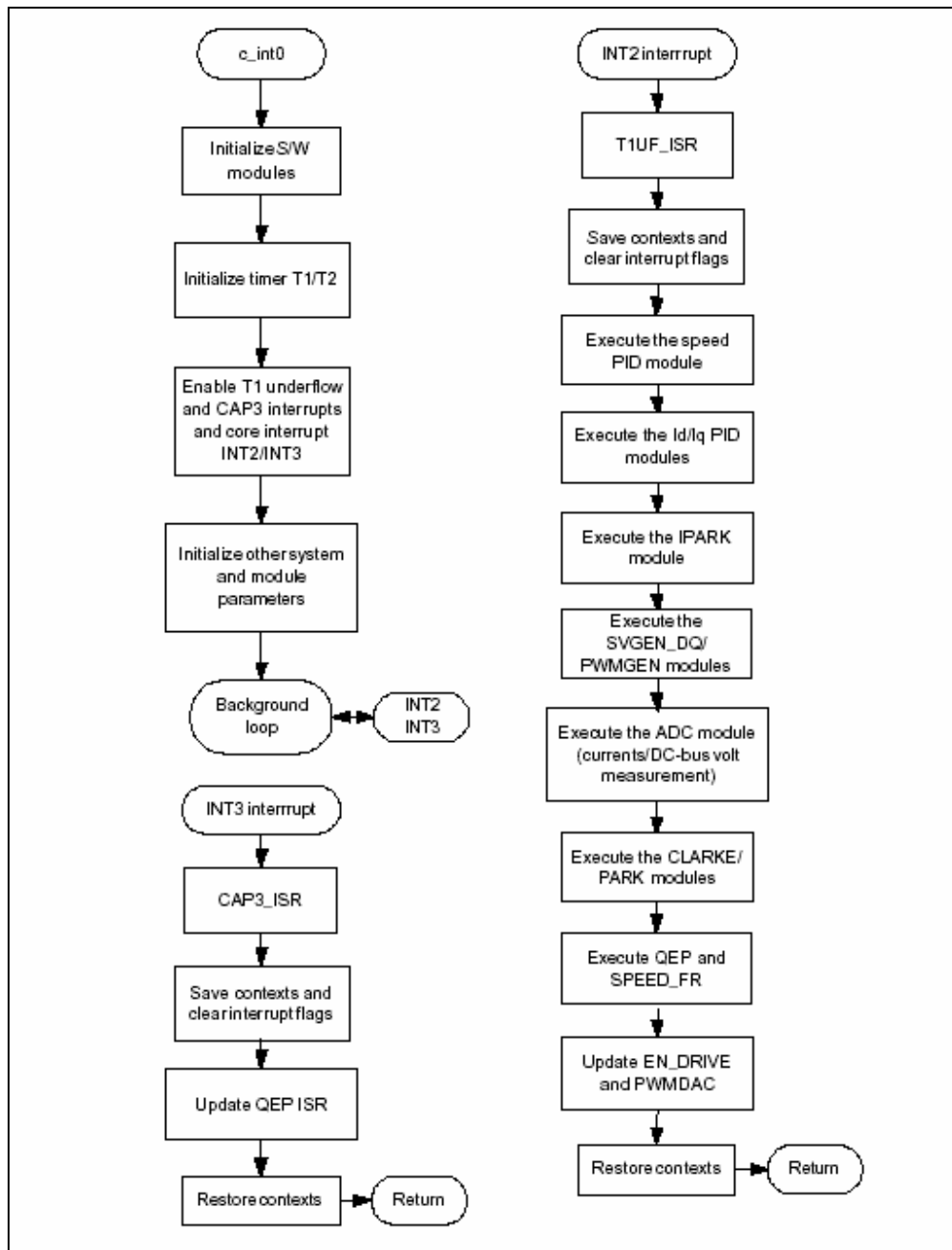
PMSM Library Modules

The next two slides are a summary of all the library modules used in the software project.



PMSM Software Flowchart

The flow of the control software consists of an initialize part (c_int00 plus main) and two interrupt service routines (INT2, INT3). After main() has done the primary initialization, it stays in an endless loop. All further activities are done by the two ISR's. INT3 belongs to the QEP-unit and is called when the rotor passes through zero degrees. INT2 belongs to Timer 1 underflow and executes the control loop periodically.



Lab 15: PMSM control project

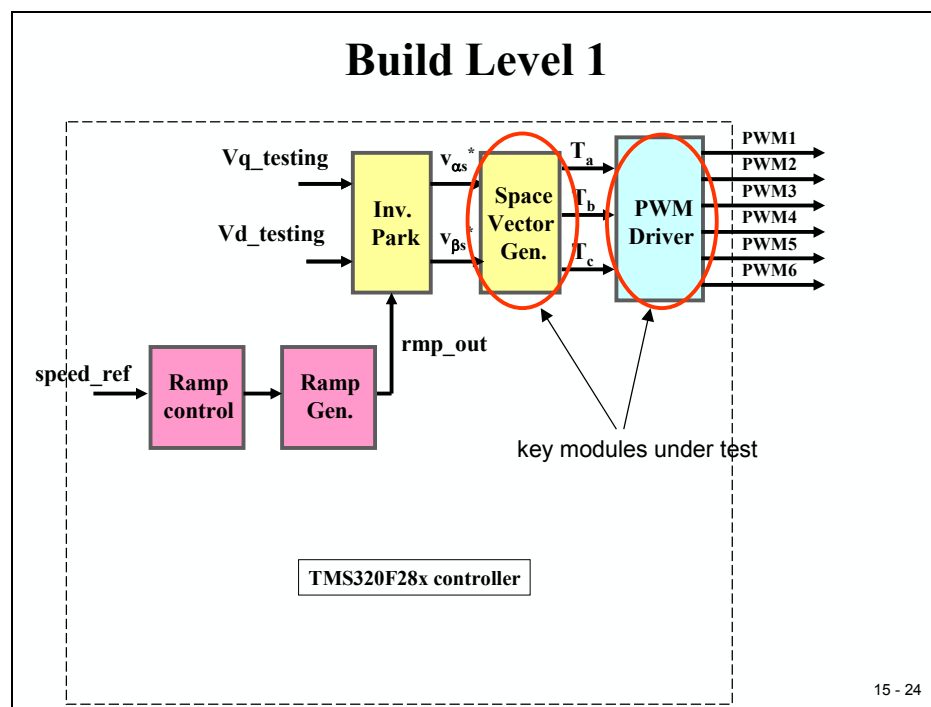
The laboratory experiment is gradually built-up in order that the final system can be confidently operated. Five phases of the incremental system build are designed to verify each of the major software modules used in the system.

Build Level 1

This first level describes the steps for a “minimum” system check-out, which confirms correct operation of system interrupts, the peripheral and target independent I_PARK and SVGEN_DQ modules and the peripheral dependent PWM_DRV module.

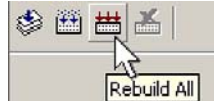
Notice that only the x2812 eZdsp is used in this phase. The PMSM and DMC550 are not to be connected yet.

In the **build.h** header file located under *../pmsm3_1/cIQmath/include* directory, select phase 1 incremental build option by setting the parameter “build level” to “level 1”. Use the ‘Rebuild All’ feature of CCS to save the program, compile it and load it to the target.

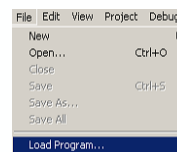


Code Composer Studio with Real Time Mode

1. Load a workspace file '*pmsm3_1.wks*'
2. In *build.h*, `#define BUILDLEVEL LEVEL1`
3. Rebuild all



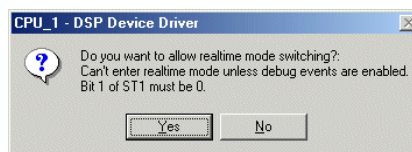
4. Load program to target
(..\pmsm3_1.out)



15 - 25

Code Composer Studio with Real Time Mode

5. In Debug menu, "Reset CPU" and then set "Real-time Mode". Then, click "Yes" when the message box pops up.



6. Click "Run" icon 

15 - 26

After running and setting real time mode, set variable "enable_flg" to 1 in the Watch Window in order to enable interrupt T1UF. The variable named "isr_ticker" will be increased incrementally, as can be seen in Watch Window to confirm the interrupt working properly.

Code Composer Studio with Real Time Mode

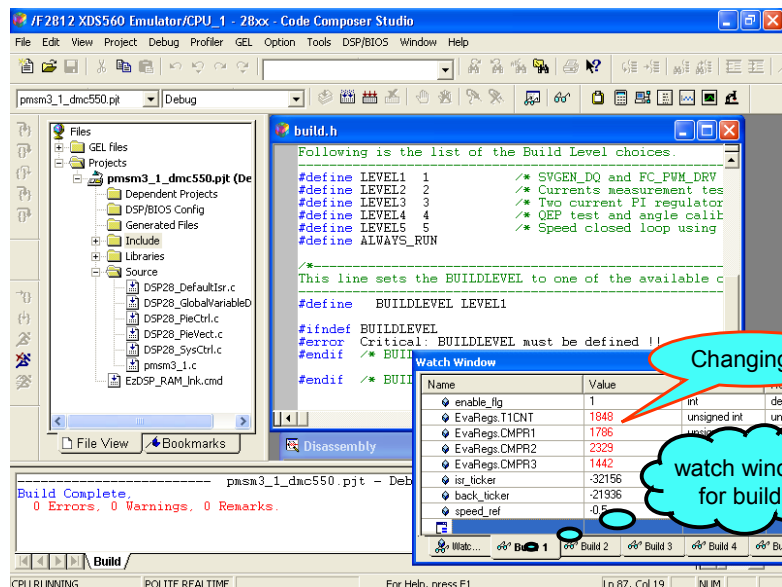
7. Right click on watch window. Then, check “Continuous Refresh”.



8. Set “enable_flg” to 1 in watch window.
(to enable T1UF interrupt and PWM drive on DMC550)

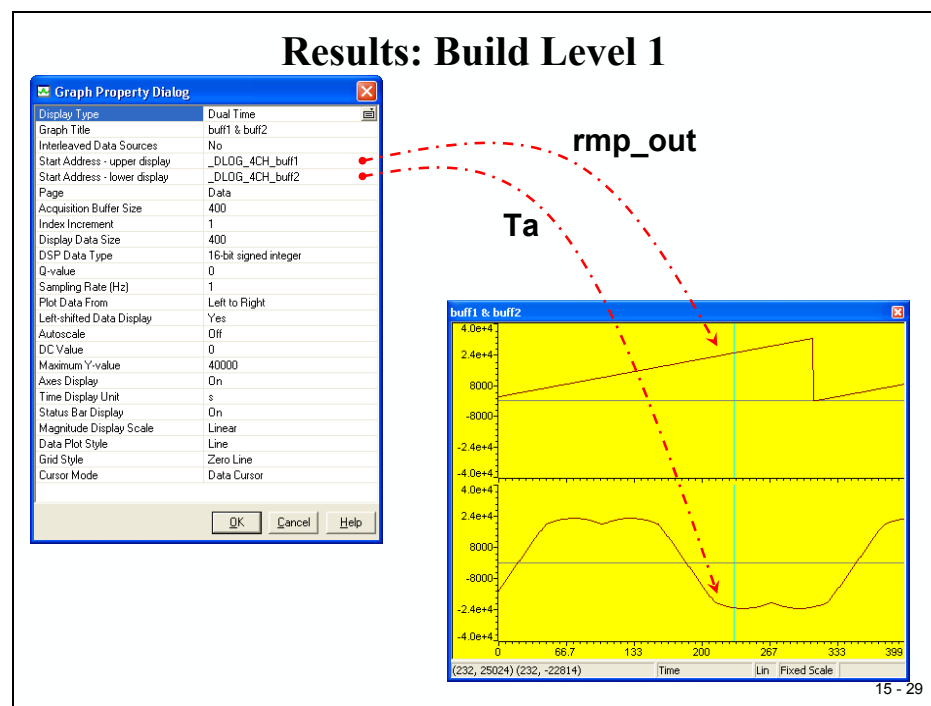
15 - 27

Code Composer Studio Level 1



15 - 28

The speed_ref value is specified to the RAMP_GEN module via RAMP_CNTL module. The I_PARK module is generating the outputs to the SVGEN_DQ module. Three outputs from SVGEN_DQ module are monitored via the PWMDAC module with external low-pass filter and an oscilloscope. The expected output waveform can be seen in the next slide. Waveforms Ta, Tb, and Tc are 120° apart from each other. Specifically, Tb lags Ta by 120° and Tc leads Ta by 120°.



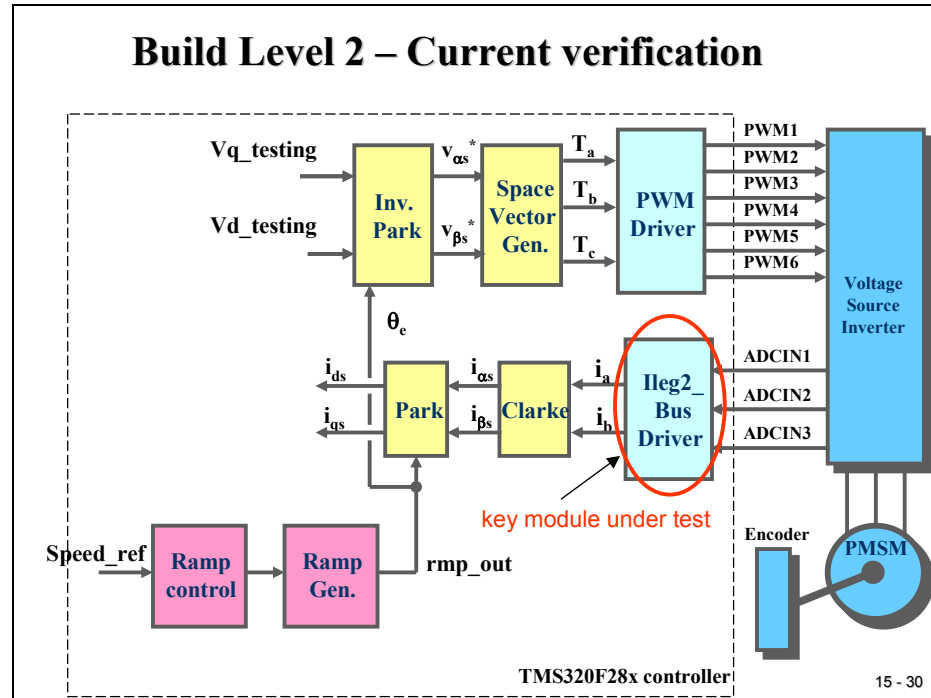
Next, the PWM_DRV module is tested by looking at the six PWM output pins.

Note:

A simple 1st – order low-pass filter RC circuit must be created to visualize the integral of the PWM-signals with an oscilloscope. Ask your laboratory technician about provisions or additional recommendations.

Once the low-pass filter is connected to the PWM pins of the x2812 eZdsp, the filtered version of the PWM signals are monitored by oscilloscope. The waveform shown on the oscilloscope should appear as same as one shown above. It is emphasized that the Ta waveform may be out of phase comparing with the filtered PWM1 signal.

Build Level 2



Assuming Level 1 is completed successfully, this section verifies the analog-to-digital conversion (ILEG2_DCBUS_DRV) and the Clarke/Park transformations (CLARKE/PARK). Now the PMSM motor and DMC550 must be connected since the PWM signals are successfully generated from phase 1 incremental build.

Note:

The DMC550 drive platform must be adjusted BEFORE we can continue. Ask the laboratory technician if this was done prior to the class. This step includes the correct setting of jumper 13, 3, 4, 10, 11, and 14. The ADC input voltages must be limited by R5, R6, R14 and R15 of the DMC550. Do NOT modify this setup! Otherwise, you might damage the hardware!

In the **build.h** header file located under `../pmsm3_1/cIQmath/include` directory, select phase 2 incremental build option by setting the build level to level 2. Use the 'Rebuild All' feature of CCS to save the program, compile it and load it to the target.

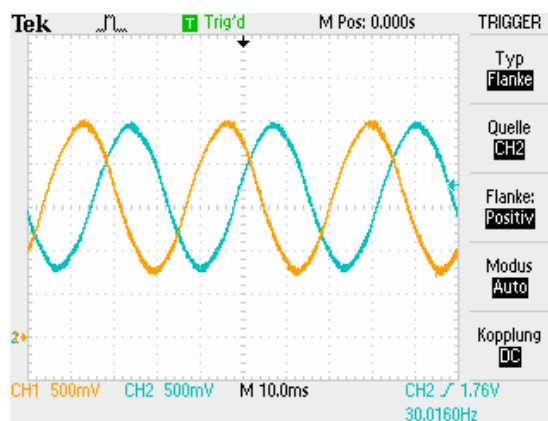
Instructions: Build Level 2

1. Tune the 24V Power Supply to 10 Volts with 1 Amp limit
2. Load a workspace file 'pmsm3_1.wks'
3. In build.h, #define BUILDLEVEL LEVEL2
4. Reset CPU, Compile, Load, start RTM and Run
5. Switch on 24V Power Supply
6. Set variable "enable_flg" to 1 in watch window.
7. Try to change motor speed by setting "speed_ref" (p.u.) in watch window. Then, motor should change its speed accordingly.

15 - 31

Results: Build Level 2

1. PMSM should run open-loop smoothly
2. The currents in the motor phases should be sinusoidal.



15 - 32

The slide above shows the measured signals „ia“ (yellow) and „ib“ (blue) from module „ILEG2_DCBUS_DRV“.

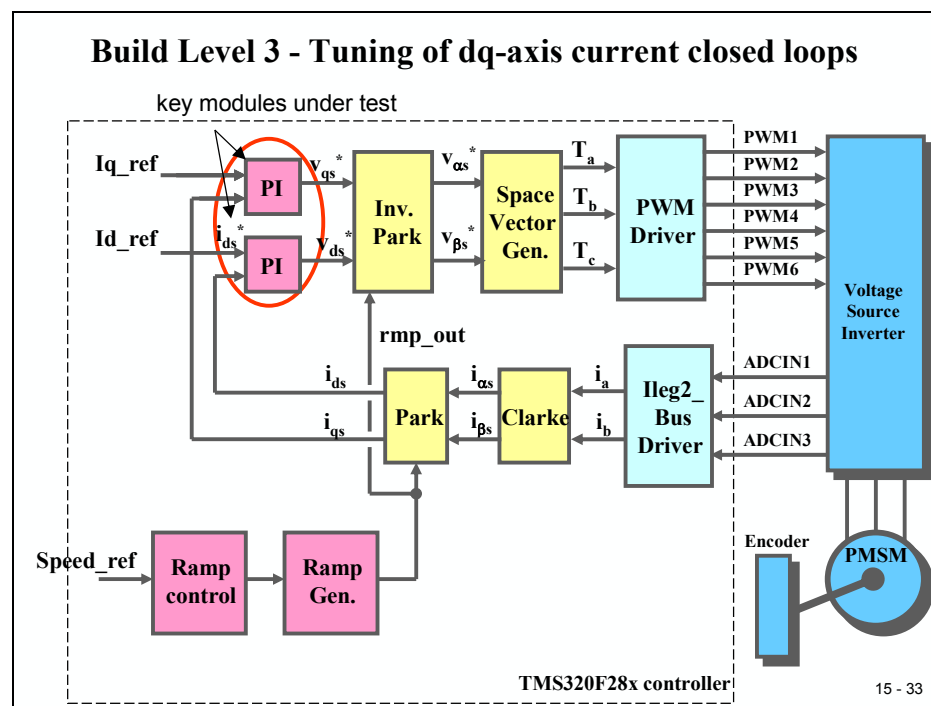
At this stage, the motor should rotate already, but without any control. Therefore we can't connect the load yet. We can modify the speed of the motor by changing variable "speed_ref" between -0.15 (anti-clockwise), 0 (stop) and +0.15 (clockwise) rotation.

With variable "rc1.rmp_dly_max", we can modify the acceleration of the motor with values between 0 (fast) and 100 (slow).

Variable "Vq_testing" will be replaced in the following levels by a control value to control the torque of the motor. Now we can use this variable to experiment with different torque control values.

Build Level 3

At this level, we close the two inner loops of the control scheme by enabling the dq-axis current regulation performed by PID_REG3 modules. To confirm the operation of current regulation, the gains of these two PID controllers are necessarily tuned for proper operation.



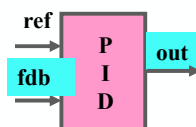
In the **build.h** header file located under `../pmsm3_1/cIQmath/include` directory, select phase 3 incremental build option by setting the build level to level 3. Use the 'Rebuild All' feature of CCS to save the program, compile it and load it to the target.

After running and setting real time mode, set “enable_flg” to 1 in the Watch Window, in order to enable interrupt T1UF. The variable named “isr_ticker” will be incrementally increased as seen in watch windows to confirm the interrupt working properly.

In this build level, the motor is supplied by AC input voltage and the motor current is dynamically regulated by using PID_REG3 module through the park transformation on the motor currents.

Instructions: Build Level 3

1. In build.h, **#define** BUILDLEVEL LEVEL3
2. Compile, Load, start RTM and Run
3. Set “enable_flg” to 1 in watch window.
4. Tune-up the PI
 - Observe dq-axis current regulations at PI inputs (i.e., reference and feedback). For example, “pid1_iq.pid_ref_reg3” and “pid1_iq.pid_fdb_reg3”.
 - Try to change motor speed by setting “speed_ref” (p.u.) in watch window . Then, observe dq-axis current regulations.

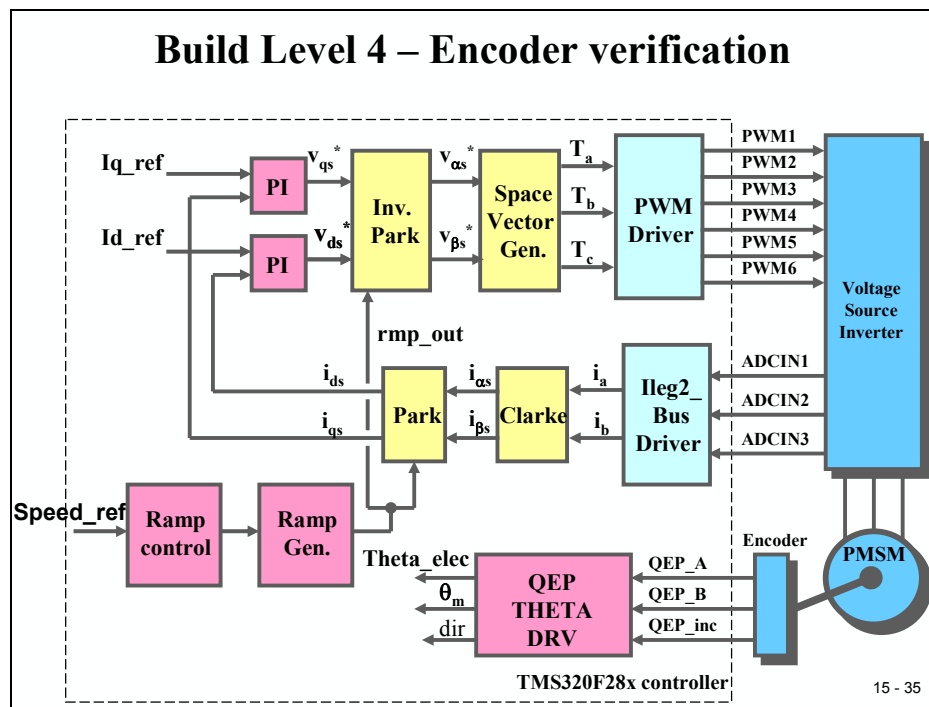


15 - 34

Build Level 4

The objective of this level is to verify the QEP driver and its speed calculation. The number of poles (p) must be set in file “parameter.h” according to the motor used in your laboratory. Next, parameter “mech_scaler” must be set according to the number of encoder-pulses per 360°-rotation:

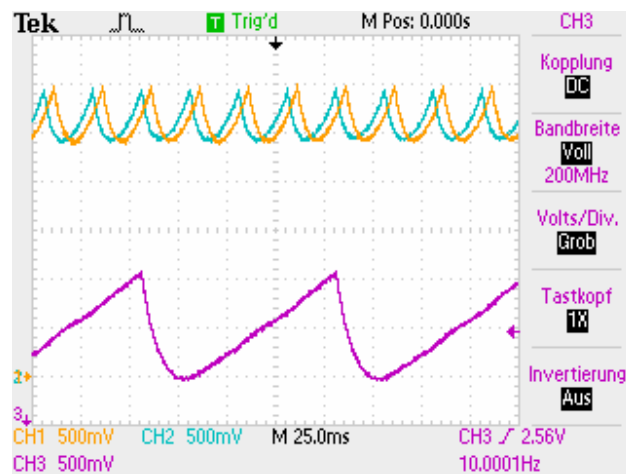
$$\text{mech-scaler} = 1 / \text{encoder-pulses} \quad (\text{in Q30 - Format})$$



Next, we have to adjust the offset angle between the index pulse of the encoder and the physical zero degrees angle of the rotor. At the end of this step, the offset will be stored as parameter “qep1.cal_angle”. With control variable “locktr_flg=0” we can activate the ramp generator in real time mode. If the motor is running, register T2CNT counts the number of pulses since the last index pulse. When we set “Locktr_flg=1”, the ramp is disconnected and the motor stops. T2CNT holds the offset between rotor position zero degrees and the QEP index pulse.

After tuning the QEP, the signals “qep1.theta_elec” and “rg1.rmp_out” should be similar in frequency and amplitude (see slide 15-37).

With the number of pole-pairs $p=4$ the frequency of “qep1.theta_mech” must be exactly $\frac{1}{4}$ of signal “qep1.theta_elec”:



Note: the signals are saw-tooth signals. Due to the low pass filter at the signal output lines the shape is smoothed.

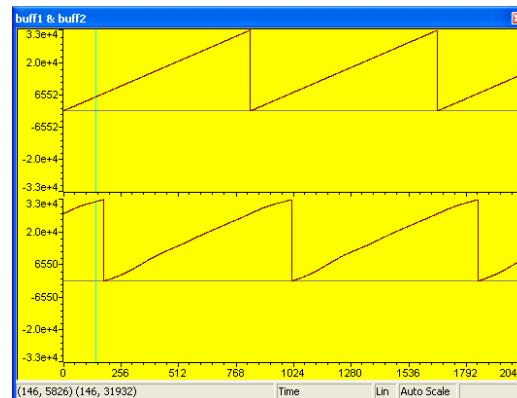
Instructions: Build Level 4

1. In `build.h`, `#define BUILDLEVEL LEVEL4`
2. Compile, Load, start RTM and Run
3. Set the DC-bus to 24 Volts 1 Amp
4. Set “enable_flg” to 1 in watch window.

15 - 36

Results: Build Level 4

◆ Emulated angle VS sensed angle

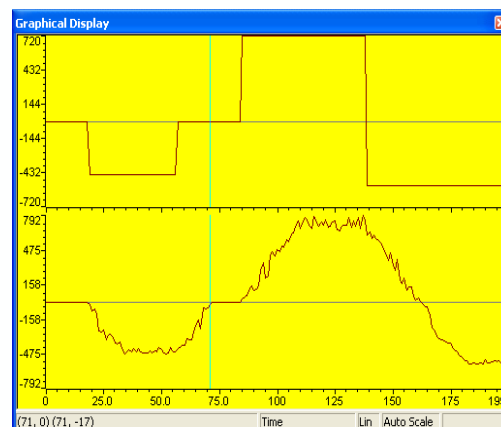


15 - 37

If you compare the changes in reference speed against real motor speed (slide 15-38), you will see that the motor follows any changes sluggishly. The reason is that we do not have closed the speed control loop yet. This will be done in the last build level 5.

Results: Build Level 4

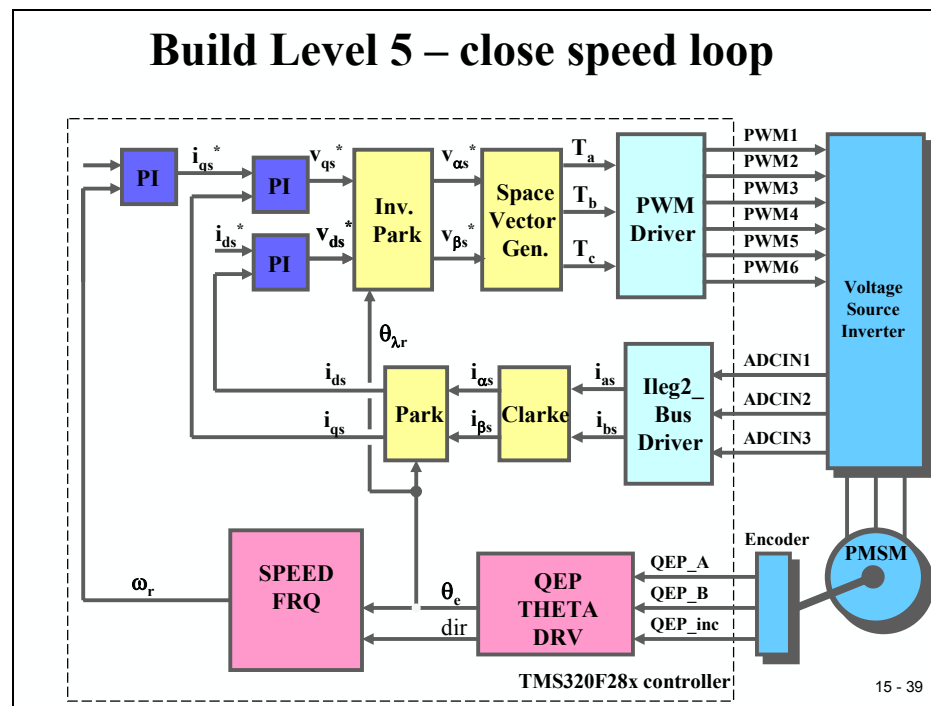
◆ Speed reference VS real speed



15 - 38

Build Level 5

After tuning the Encoder unit, we will use this feedback information as position information in our control loops. The simulated ramp modules from level 1 to 4 are no longer needed. We will also close the speed control loop.

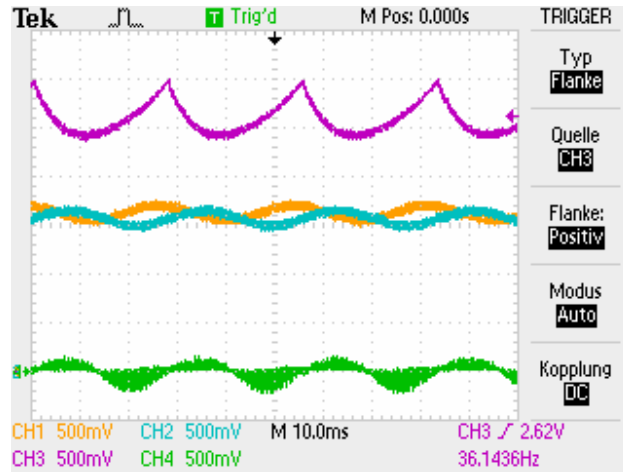


To adjust the speed reference we can use potentiometer R66 of the DMC550. Its analogue value can be connected to ADCINA7 and the converted result can be used for a calculation of signal “speed_ref”. To do this we need to modify the Interrupt Service Routine of EVA-Timer1.

Note:

This last step depends on the hardware setup in your laboratory. Ask your technician if this option is available in your case.

To verify the correct operation of module “speed_ref” we can do some measurements:



The first signal (top) is “speed1.theta_elec”. For this example and knowing that $p=4$ we can calculate the mechanical speed to:

$$\text{Speed1.theta_mech} = (36.1436 \text{ s}^{-1} / 4) * 60 = 542 \text{ rpm}$$

This value should be verified with a stroboscope, if available.

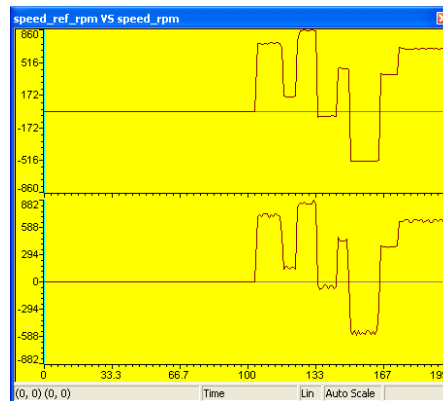
Instructions: Build Level 5

1. In build.h, **#define** BUILDLEVEL LEVEL5
2. Compile, Load, start RTM and Run
3. Set the DC-bus to 24 Volts
4. Set “enable_flg” to 1 in watch window.

If we measure the change in reference speed against real motor speed we should see a much better response of the control system (see slide 15-41).

Results: Build Level 5

- ◆ Fastest response time with the closed loop!



15 - 41

A thesis project at Zwickau University was the implementation of the C28x control scheme for a PMSM in a tram generator.

Application of PMSM 3-1:



img GmbH Nordhausen

15 - 42

This page has intentionally been left blank.