
FTF-IND-F0010

John H Floros | MCD Toolbox Manager

A P R . 2 0 1 4
Agenda

• Overview: 30 minutes
  - **Introduction and Objectives**
  - Motor Control Development Toolbox: Library blocks, FreeMASTER, and Bootloader
  - Model Based Design Steps: Simulation, SIL, PIL and ISO26262
• Hands-on Demo: 20 minutes
  - Convert simple model to run on Motor Kit with MCD Toolbox and use FreeMASTER
• Motor Control: 30 minutes
  - Motor Kit ( Describe Freescale 3-Phase Motor Kit )
  - Trapezoidal control and how to use it to turn a motor
• Motor Control Hands-on Demo: 80 minutes
  - Implement Trapezoidal Motor Control on Motor Kit
  - Run software from the model and use FreeMASTER to monitor and tune parameters
• SIL/PIL Demo: 10 minutes
• Summary and Q&A: 10 minutes
Introduction: WHAT DO WE DO?

• The CodeWarrior and Engineering Tools group’s objective is to develop software enablement tools to assist our customers with rapid prototyping and accelerate algorithm development on their target Freescale MCU

• This includes software tools that automatically generate peripheral initialization code through GUI configuration, to generating peripheral driver code from a Model Based Design environment like Simulink™
Introduction: Model Based Design (MBD)

• Model Based Design is becoming more common during the normal course of software development to explain and implement the desired behavior of a system. The challenge is to take advantage of this approach and get an executable that can be simulated and implemented directly from the model to help you get the product to market in less time and with higher quality. This is especially true for electric motor controls development in this age of hybrid/electric vehicles and the industrial motor control application space.

• Many companies model their controller algorithm and the target motor or plant so they can use a simulation environment to accelerate their algorithm development.

• The final stage of this type of development is the integration of the control algorithm software with target MCU hardware. This is often done using hand code or a mix of hand code and model-generated code. Motor Control Development Toolbox allows this stage of the development to generate 100% of the code from the model.
Introduction: Motor Control Development Toolbox

- The Motor Control Development Toolbox includes an embedded target supporting Freescale MCUs and Simulink™ plug-in libraries which provide engineers with an integrated environment and tool chain for configuring and generating the necessary software, including initialization routines, device drivers, and a real-time scheduler to execute algorithms specifically for controlling motors.

- The toolbox also includes an extensive Math and Motor Control Function Library developed by Freescale’s renowned Motor Control Center of Excellence. The library provides dozens of blocks optimized for fast execution on Freescale MCUs with bit-accurate results compared to Simulink™ simulation using single-precision math.

- The toolbox provides built-in support for Software and Processor-in-the-Loop (SIL and PIL), which enables direct comparison and plotting of numerical results.

MathWorks products required for MC Toolbox:
- MATLAB (32-Bit or 64-Bit)*
- Simulink
- MATLAB Coder
- Simulink Coder
- Embedded Coder

*Earlier released products only support 32-bit
Introduction: Reduce Development Time With MBD and MC Toolbox

- **System Requirements**: Use software-based model vs. paper-based method, and start testing at very earliest stage.
- **Modeling/Simulation**: Convert model to SIL and now can test ANSI-generated software. Can also use MC library with SIL testing.
- **Rapid Prototype**: With MC library and MC Toolbox, test Model using target MCU and compiler through PIL testing.
- **Target MCU Implementation**: With MC Toolbox, auto-generate code for direct interface of peripherals for target hardware without any manual hand code.
- **Functional Testing**: Fewer defects found in this phase of testing, where finding defects is expensive.
- **HIL Testing**: Now that more testing on target has occurred earlier in the process, HIL testing time is reduced.

Reduce Time from This...
Introduction: Reduce Development Time With MBD and MC Toolbox

- System Requirements
- Modeling/Simulation
- Rapid Prototype
- Target MCU Implementation
- HIL Testing
- Functional Testing

To This!
Objectives

After this Hands-on Workshop, you will be able to:

- Use the Motor Control Development Toolbox to auto-generate and build software for the MCU directly from the MATLAB™/Simulink™ environment.
- Configure the MCU peripherals required to implement three phase motor control using the MCU and the low-voltage Three Phase Motor Control Kit.
- Implement Trapezoidal Motor Control from a model based design environment and auto generate the code to run the brushless DC Motor provided with the Motor Kit.
- Know how Motor Control Development Toolbox can help with your motor control development projects and Freescale MCUs.
Objectives

- Freescale’s hardware/software enablement

**TWR-ELEV**
TOWER System
Modular, expandable and cost-effective development platform

**TWR-KV10Z32**

**TWR-MC-LV3PH**
3-Phase Motor Control Kit

**Motor Control Development Toolbox**
with Simulink™

Model-based design driver configuration, Assignment to pins, & initialization setup

Real-time Debugging Tool
Data acquisition and calibration
Agenda

• Overview: 30 minutes
  - Introduction and Objectives
  - **Motor Control Development Toolbox: Library blocks, FreeMASTER, and Bootloader**
    - Model Based Design Steps: Simulation, SIL, PIL and ISO26262
• Hands-on Demo: 20 minutes
  - Convert simple model to run on Motor Kit with MCD Toolbox and use FreeMASTER
• Motor Control: 30 minutes
  - Motor Kit (Describe Freescale 3-Phase Motor Kit)
  - Trapezoidal control and how to use it to turn a motor
• Motor Control Hands-on Demo: 80 minutes
  - Implement Trapezoidal Motor Control on Motor Kit
  - Run software from the model and use FreeMASTER to monitor and tune parameters
• SIL/PIL Demo: 10 minutes
• Summary and Q&A: 10 minutes
# MCD Toolbox: Toolbox Library Contents

<table>
<thead>
<tr>
<th>Peripherals</th>
<th>Configuration/Modes</th>
<th>Utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>• General</td>
<td>• Compiler Options</td>
<td>• FreeMASTER Interface</td>
</tr>
<tr>
<td>- ADC conversion</td>
<td>- CodeWarrior</td>
<td>• Data acquisition</td>
</tr>
<tr>
<td>- Digital I/O</td>
<td>- Wind River DIAB</td>
<td>• Calibration</td>
</tr>
<tr>
<td>- PIT timer</td>
<td>- Green Hills</td>
<td>• Customize GUI</td>
</tr>
<tr>
<td>- ISR</td>
<td>- Cosmic</td>
<td></td>
</tr>
<tr>
<td>• Communication Interface</td>
<td>- IAR</td>
<td></td>
</tr>
<tr>
<td>- CAN driver</td>
<td>- GCC</td>
<td></td>
</tr>
<tr>
<td>- SPI driver</td>
<td>- RAM/FLASH targets</td>
<td></td>
</tr>
<tr>
<td>- I2C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Motor Control Interface</td>
<td>• Simulation Modes</td>
<td>• Profiler Function</td>
</tr>
<tr>
<td>- Cross triggering unit</td>
<td>- Normal</td>
<td>• Exec. time measurement</td>
</tr>
<tr>
<td>- PWM</td>
<td>- Accelerator</td>
<td>• Available in PIL</td>
</tr>
<tr>
<td>- eTimer block(s)</td>
<td>- Software in the Loop (SIL)</td>
<td>• Available in standalone</td>
</tr>
<tr>
<td>- Sine wave generation</td>
<td>- Processor in the Loop (PIL)</td>
<td></td>
</tr>
<tr>
<td>- ADC Command List</td>
<td>• MCU Option</td>
<td></td>
</tr>
<tr>
<td>- GDU (Gate Drive Unit)</td>
<td>- Multiple packages</td>
<td>• MPC5643L</td>
</tr>
<tr>
<td>- PTU (Prog Trigger Unit)</td>
<td>- Multiple Crystal frequencies</td>
<td>• MPC567xK</td>
</tr>
<tr>
<td>- TIM Hall Sensor Port</td>
<td></td>
<td>• MPC574xP</td>
</tr>
<tr>
<td>- FTM (Flex Timer Module)</td>
<td></td>
<td>• S12ZVM</td>
</tr>
<tr>
<td>- PDB (Programmable Delay Block)</td>
<td></td>
<td>• KV10Z</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 56F82xx (Coming Soon)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• KV31/30 (Coming Soon)</td>
</tr>
</tbody>
</table>

**MCUs Supported**

- MPC5643L
- MPC567xK
- MPC574xP
- S12ZVM
- KV10Z
- 56F82xx (Coming Soon)
- KV31/30 (Coming Soon)

---

**NOTE:** Peripheral Block and compiler availability is dependant on which MCU is use.
# MCD Toolbox: Math and Motor Control Library Contents

## GFLIB
- **Trigonometric Functions**
  - GFLIB_Sin
  - GFLIB_Cos
  - GFLIB_Tan
  - GFLIB_Asin
  - GFLIB_Acos
  - GFLIB_Atan
  - GFLIB_AtanXY
- **Limitation Functions**
  - GFLIB_Limit
  - GFLIB_LowerLimit
  - GFLIB_UpperLimit
  - GFLIB_VectorLimit
- **PI Controller Functions**
  - GFLIB_ControllerPIr
  - GFLIB_ControllerPIrAW
  - GFLIB_ControllerPIp
  - GFLIB_ControllerPIpAW
- **Linear Interpolation**
  - GFLIB_Lut1D
- **Hysteresis Function**
  - GFLIB_Hyst
- **Signal Integration Function**
  - GFLIB_IntegratorTR
- **Sign Function**
  - GFLIB_Sign
- **Signal Ramp Function**
  - GFLIB_Ramp

## GDFLIB
- **Finite Impulse Filter**
  - GDFLIB_FilterFIR
- **Moving Average Filter**
  - GDFLIB_FilterMA
- **1st Order Infinite Impulse Filter**
  - GDFLIB_FilterIIR1init
  - GDFLIB_FilterIIR1
- **2nd Order Infinite Impulse Filter**
  - GDFLIB_FilterIIR2init
  - GDFLIB_FilterIIR2

## GMCLIB
- **Clark Transformation**
  - GMCLIB_Clark
  - GMCLIB_ClarkInv
- **Park Transformation**
  - GMCLIB_Park
  - GMCLIB_ParkInv
- **Duty Cycle Calculation**
  - GMCLIB_SvmStd
- **Elimination of DC Ripples**
  - GMCLIB_ElimDcBusRip
- **Decoupling of PMSM Motors**
  - GMCLIB_DecouplingPMSM
MCD Toolbox: RAppID Bootloader Utility

The RAppID Bootloader works with the built-in Boot Assist Module (BAM) included in the Freescale Qorivva and also supports MagniV, Kinetis, and DSCs family of parts. The Bootloader provides a streamlined method for programming code into FLASH or RAM on either target EVBs or custom boards. Once programming is complete, the application code automatically starts.

**Modes of Operation**

The Bootloader has two modes of operation: for use as a stand-alone PC desktop GUI utility, or for integration with different user required tools chains through a command line interface (i.e. Eclipse Plug-in, MATLAB/Simulink, …)

**MCUs Supported**

MPC5534, MPC5601/2D, MPC5602/3/4BC, MPC5605/6/7B, MPC564xB/C, MPC567xF, MPC567xK, MPC564xL, MPC5604/3P, MPC574xP, S12ZVM, KV10 and 56F82xx.

![Graphical User Interface](image1)

![Command](image2)

Status given in two stages: Bootloader download, then application programming
FreeMASTER – Run Time Debugging Tool

- User-friendly tool for real-time debug monitor and data visualization
  - Completely non-intrusive monitoring of variables on a running system
  - Display multiple variables changing over time on an oscilloscope-like display, or view the data in text form
  - Communicates with an on-target driver via USB, BDM, CAN, UART
- Establish a Data Trace on Target
  - Set up buffer (up to 64KB), sampling rate and trigger
  - Near 10-µs resolution

http://www.freescale.com/freemaster
MCD Toolbox: Summary of Application Support

**System Infrastructure**
- MC library set
  - GFLIB (General functions)
  - GDFLIB (Digital filtering)
  - GMCLIB (Motor Control)
- Drivers
  - Efficient Reflecting the chip features
- FreeMaster Support
- Boot Loader Support

**Application SW**
- API
- Algorithm Libraries
- Drivers
- On-Chip Peripherals

**External Hardware**
- External Connections
- PINS

**Target Platform**

**User Application Software**

**Documentation**
Motor Control Development Toolbox

Any Questions?
Agenda

• Overview: 30 minutes
  - Introduction and Objectives
  - Motor Control Development Toolbox: Library blocks, FreeMASTER, and Bootloader
  - **Model Based Design Steps: Simulation, SIL, PIL and ISO26262**

• Hands-on Demo: 20 minutes
  - Convert simple model to run on Motor Kit with MCD Toolbox and use FreeMASTER

• Motor Control: 30 minutes
  - Motor Kit (Describe Freescale 3-Phase Motor Kit)
  - Trapezoidal control and how to use it to turn a motor

• Motor Control Hands-on Demo: 80 minutes
  - Implement Trapezoidal Motor Control on Motor Kit
  - Run software from the model and use FreeMASTER to monitor and tune parameters

• SIL/PIL Demo: 10 minutes

• Summary and Q&A: 10 minutes
Idealized simulation of the controller and the motor to refine the control technique. Done on host PC without regard for embedded controller. Can optionally add analog device models for fault detection and signal control.
Model Based Design Steps: Step 2 (SIL)

(SIL) Generated code executes as atomic unit on PC

Still done on host PC without regard for embedded controller. Instead using generated C code that is compiled using a PC-based compiler. Run same test vectors as in simulation for C Code Coverage analysis and verify functionality.
Execute the model on the target MCU and perform numeric equivalence testing. Co-execution with MCU and Model Based Design working together while collecting execution metrics on the embedded controller of the control algorithm. Validate performance on the MCU.
Model Based Design Steps: Step 3 (PIL)

Verification and Validation at Code Level

• This step allows:
  – Translation validation through systematic testing
  – To demonstrate that the execution semantics of the model are being preserved during code generation, compilation, and linking with the target MCU and compiler

• Numerical Equivalence Testing:
  – Equivalence Test Vector Generation
  – Equivalence Test Execution
  – Signal Comparison
Example IEC 61508 and ISO 26262 Workflow for Model-Based Design with MathWorks Products*

Model advisor, modeling standards checking

Simulation (model testing), model coverage, RMI

Module and integration testing at the model level

Review and static analysis at the model level

PIL testing using embedded IDE links

Real-Time Workshop Embedded Coder traceability report or model vs. code coverage comparison

Equivalence testing

Prevention of unintended functionality

Model used for production code generation

Generated C code

Object code

Textual requirements

Executable specification

Modeling

Simulink / Stateflow / Simulink Fixed Point

Real-Time Workshop Embedded Coder

Simulink / Stateflow / Simulink Fixed Point

Modeling

Real-Time Workshop Embedded Coder

*Workflow from The Mathworks™ Presentation Material Model-Based Design for IEC 61508 and ISO 26262
Model Based Design Steps: Step 4 (Target MCU)*

Generate production code to run on embedded MCU with real motor while collecting execution metrics on the embedded controller of control algorithm. Validate performance on MCU and use FreeMASTER to tune control parameters and perform data logging.

* I/O peripheral driver blocks can be included in the model, providing the analog driver interfaces needed to directly interface to devices external from the MCU.
Model Based Design Steps: Summary

**Step 1 – System Requirements:**
MBD Simulation Only
Software requirements
Control system requirements
Overall application control strategy
Modeling style guidelines applied
Algorithm functional partitioning
Interfaces are defined here

**Step 2 – Modeling/Simulation:**
MBD Simulation with ANSI C Code using SIL
Control code generation
Determine execution time on MCU
Test harness to validate all requirements
Start testing implementation approach
Testing of functional components of algorithm
Equivalence testing
Creates functional baseline of model

**Step 3 – Rapid Prototype:**
MBD Simulation with ANSI C Code using PIL
Controller code generation
Determine execution time on MCU
See memory/stack usage on MCU
Start testing implementation approach
Target testing controls algorithm on MCU
Refine model for code generation
Function/File partitioning
Data typing to target environment done here
Scaling for fixed point simulation and code generation
Testing of functional components of algorithm
Test harness to validate all requirements
Test coverage of model here
Creates functional baseline of model

**Step 4 – Target MCU Implementation**
ANSI C Code Running on Target HW & MCU
Validation/verification phase
Controller code generation
Determine execution time on MCU
Start testing implementation on target ECM
Code generate control algorithm + I/O drivers
Execute code on target MCU
Functional testing in target environment
Utilize calibration tools for data logging and parameter tuning

Ensure execution on target is correct as well as code generation on target is performing as desired.

---

External Use | 23
Agenda

• Overview: 30 minutes
  - Introduction and Objectives
  - Motor Control Development Toolbox: Library blocks, FreeMASTER, and Bootloader
  - Model Based Design Steps: Simulation, SIL, PIL and ISO26262

• Hands-on Demo: 20 minutes
  - Convert simple model to run on Motor Kit with MCD Toolbox and use FreeMASTER

• Motor Control: 30 minutes
  - Motor Kit (Describe Freescale 3-Phase Motor Kit)
  - Trapezoidal control and how to use it to turn a motor

• Motor Control Hands-on Demo: 80 minutes
  - Implement Trapezoidal Motor Control on Motor Kit
  - Run software from the model and use FreeMASTER to monitor and tune parameters

• SIL/PIL Demo: 10 minutes

• Summary and Q&A: 10 minutes
Hands-on Demo: Tower Hardware

**TWR-MKV10Z32**

**Features:**
- MKV10Z32VLF7 MCU (48LQFP)
- OpenSDA debug circuit with Micro USB connector and virtual serial port
- MMA8451Q 3-axis digital accelerometer (i2c interface)
- LEDs with connected buffers to PWM channels for dimming
- 2 push buttons for user input or interrupts
- 4 Thermistors for single ended or differential analogue inputs, 2 motor control auxiliary connectors
- Compatible with TWR-MC-LV3PH motor driver peripheral module (NOTE: TWR-MC-LV3PH module and TWR-ELEV module must be ordered separately if required.

**TWR-MC-LV3PH**

**Features:**
- Input voltage 12-24V DC
- Output current 5-10 Amps
- 3-phase MOSFET inverter
- 3-phase pre-driver MC33937
- Analog sensing
- Motor speed/position sensors interface
- 2 pole-pair BLDC motor with Hall sensors (4000 RPM rated speed)
- On-board power regulation for Tower System (single power supply via TWR-MC-LV3PH power jack)
Hands-on Demo: Simple Model

Run Simple Model Simulation

1. Open Model “Blinky.slx”
2. You will see a model that toggles outputs and will change the way it toggles the outputs based on an input.
3. Run simulation and open the scope. You should see the following on the scope:
Hands-on Demo: Simple Model

Convert Simple Model and Run

1. Save model as “KV10_Blinky.slx”
2. Select KV10 TLC file to configure model for the MCU
3. Open Simulink library
4. Go to Motor Control Toolbox for Configuration Information Block
5. Drag the block into the model
6. Open block and go to PIL/BAM setup tab
7. Check download code after build
8. Enter the COM port number that you are using from PC
9. Delete Step block and Scope
10. Go back to library under General Purpose Blocks, drag in a Digital Output block and connect to each mode output a Digital Output Block. Select the output pins to use which are connected to an LED (See next Slide).
11. Go back to library under Motor Control Blocks and drag in a Digital input block to read SW1 (PTA4).
12. Go back to the library and under utilities get a Profiler Function block so that the execution time can be measured.
Hands-on Demo: LED Simple Model

Convert Simple Model and Run

- This is what the model should look like after step 12
Hands-on Demo: Simple Model

Convert Simple Model and Run
13. Go to Tools pull down menu and then select generate code / Build Model.
14. Wait for model to generate code and then a prompt from the RAppID Bootloader Utility will appear. Reset the MCU and then select “OK”.
15. Once the download is complete you will observe LEDs blinking.
16. Press SW1 and you will see not all the LEDs blinking.
Hands-on Demo: Simple Model

Using FreeMASTER with Hands-on Demo

17. Start FreeMASTER and open project KV10_Blinky.pmp. Just press OK if a message comes up that the map file has been updated.

18. Go to Project Options Pull Down and select “Options”. Verify that COM settings are the same as what were set in your model.

19. Once the COM settings are correct, press the STOP button and press SW 1. You should see the following (next slide):
Hands-on Demo: Read A/D and Toggle LED Simple Model

Using FreeMASTER with Hands-on Demo

• This is what you should see after step 19
Hands-on Demo: Simple Model

Any Questions?
Agenda

• Overview: 30 minutes
  - Introduction and Objectives
  - Motor Control Development Toolbox: Library blocks, FreeMASTER, and Bootloader
  - Model Based Design Steps: Simulation, SIL, PIL and ISO26262
• Hands-on Demo: 20 minutes
  - Convert simple model to run on Motor Kit with MCD Toolbox and use FreeMASTER
• Motor Control: 30 minutes
  - Motor Kit (Describe Freescale 3-Phase Motor Kit)
  - Trapezoidal control and how to use it to turn a motor
• Motor Control Hands-on Demo: 80 minutes
  - Implement Trapezoidal Motor Control on Motor Kit
  - Run software from the model and use FreeMASTER to monitor and tune parameters
• SIL/PIL Demo: 10 minutes
• Summary and Q&A: 10 minutes
Tower Motor Kit Overview

Features
• Electrical specifications:
  - Input voltage 12-24V DC
  - Output current 5-10 Amps
• 3-phase MOSFET inverter
• 3-phase pre-driver MC33937, configurable thru SPI analog sensing (dcb voltage, dcb current, phase currents, back-EMF voltage)
• Motor speed/position sensors interface (Encoder, Hall)
• Hardware over-current fault protection
• On-board power regulation for Tower System (single power supply via TWR-MC-LV3PH power jack)
• Brushless DC (BLDC) Motor Linix 45ZWN24-40

Purpose
• The purpose of the Low Voltage Motor Control Tower module is to be used by customers to prototype DC, BLDC and PMSM motor control solutions and to evaluate/demonstrate various algorithms.
Tower Motor Kit: TWR-MC-LV3PH Module

Motor Connector

MOSFET H-Bridge

Power Supply Connector

Motor Hall/Encoder Connector

Freescale 3-Phase Pre-Driver Chip (MC33937)
Tower Motor Kit: 3PP Driver Chip (MC33937)

Features
- Fully specified from 8.0V to 40V; covers 12V and 24V automotive systems
- Extended operating range from 6.0V to 58V covers 12V and 42V systems
- 1.0A gate drive capability with protection
- Protection against reverse charge injection
- Includes a charge pump to support full FET drive at low battery voltages
- Dead time is programmable via the SPI
- Simultaneous output capability enabled via safe SPI command

Purpose
- The IC contains three high-side FET pre-drivers and three low-side FET pre-drivers. Three external bootstrap capacitors provide gate charge to the high side FETs.

The IC interfaces to an MCU via six input control signals, a SPI port for device setup and asynchronous reset, enable and interrupt signals.
Tower Motor Kit: Brushless DC Motor

Features
Model: Linix 45ZWN24-40
Motor Type: Brushless DC
Windings: “Y” Connection Method
Pole Pairs: 2 pairs
Rated Voltage: 24V
Rated Current: 2.3 A
Rated Torque: 990 g.cm
Rated Power: 40 Watts
Rated Speed (Load): 4000 RPM
Speed (Un-Loaded): 4900 RPM
Position Sensing: Hall Type (A, B, C)

Note: Pole pair count for this motor means that every single mechanical revolution equals two electrical revolutions. State change in Hall sensors is every 60 degrees electrical.
Introducing Kinetis V Series
Motor Control and Power Conversion MCUs

• Builds on Freescale’s **motor and power control expertise** to address **NEW mass market** customers.

• Enables **efficient, next generation** BLDC, PMSM and ACIM designs through **optimized performance, analog and timing IP**. High speed **DSC peripherals** are ideal for advanced motor control and power conversion and include the **fastest ADC** in the Freescale MCU portfolio.

• Features **scalable, low-power families** built on **ARM® Cortex® processors** – starting with the industry’s **fastest ARM Cortex-M0+ MCU**.

• Includes sophisticated enablement tools like the new, easy-to-use **Kinetis motor suite** which helps to **reduce development time and cost** for every customer.
Kinetis V Series Target Applications

**Motor Control**
- Sensored BLDC / PMSM
  - High Dynamic Control
- Sensored ACIM
- Sensorless VOC
  - PMSM/BLDC
  - High Dynamic Control
  - Low Dynamic Control
- Sensorless ACIM

**Digital Power Conversion**
- Solar Inverters
  - Grid-Tied
  - Non Grid Tied
- Power factor correction
- Switch Mode Power Supplies
  - AC/DC
  - DC/DC
- UPS
  - On-Line
  - Offline
- Inductive cooking
  - Multi cook plate
Solutions for Motor Control and Digital Power Conversion

**KV1x Family**
- BLDC, entry level PMSM
- + Motor Control Software

**KV3x Family**
- Mid range PMSM, UPS power control
- + High Speed ADC
- + High Resolution PWM

**KV4x Family**
- High performance motors, UPS, solar and mid range AC/DC control
- + Advanced Memory, Connectivity and Communications
- + Multi Channel Timers
- + Floating Point Unit

**KVxx Families**
- Increasing performance and feature integration
- Optimized memory configurations

**Feature Integration**
- Freescale IDE, RTOS, Software Libraries and Motor Control Development Tools
Kinetis V Series Enablement

**New Kinetis Motor Suite**

A simple-to-use, motor control development suite that allows customers of all experience levels to design applications quickly, efficiently and without the need for in-depth motor control expertise.

- **Turnkey solutions initially targeting fans, compressors and pumps.** MCU interaction via GUI. Additional application areas will be developed over time.
- **Advanced solutions for user development of app. code,** with Kinetis motor suite configuring and controlling the motor subsystem.

**Motor Control Tools**

**New Freescale Enablement Software**
- Kinetis SDK, Kinetis Design Studio, Kinetis Bootloader

**FreeMASTER:** GUI-based run-time debug monitor and data visualization tool

**MCAT:** FreeMASTER plug-in for real-time monitoring, tuning and updating of control parameters

**Motor Control Toolbox:** MATLAB™ modeling environment plug-in for automatic code generation

**Software Libraries for CM0+/M4 & IEC60730:** Math, General, Filter and Motor Control libraries. Sensorless algorithms for advanced control

**Tower Development System, IDEs & Auto-code Generators:** KDS, IAR, Keil & Processor Expert

**Extensive Ref. Design and App. Note Library:** BLDC, PMSM, High Voltage Power Stage and more

Supported by Freescale’s Motor Control Center of Excellence with 20+ years of expertise in MCU development, enablement and customer support
# Kinetis V Series Availability

<table>
<thead>
<tr>
<th>MCU Family</th>
<th>Core / Frequency</th>
<th>Flash Range</th>
<th>ADC</th>
<th>FlexTimers / eFlexPWM</th>
<th>Starting From Price (10K# SRP, USD)</th>
<th>Expected availability (Samples / Production)</th>
</tr>
</thead>
<tbody>
<tr>
<td>KV4x</td>
<td>Cortex-M4 150 MHz</td>
<td>64-256 KB</td>
<td>2x 12bit 4.1 MSps / 1.9 MSps</td>
<td>Up to 8-ch. FlexTimer / 12-ch. eFlexPWM + Nano-Edge</td>
<td>*$2.99</td>
<td>Sept</td>
</tr>
<tr>
<td>KV3x</td>
<td>Cortex-M4 100/120 MHz</td>
<td>64-512 KB</td>
<td>2x 16-bit 1.2 MSps</td>
<td>Up to 20-ch. FlexTimers</td>
<td>*$1.79</td>
<td>Apr / Aug</td>
</tr>
<tr>
<td>KV1x</td>
<td>Cortex-M0+ 75 MHz</td>
<td>16-32 KB</td>
<td>2x 16-bit 1.2 MSps</td>
<td>Up to 10-ch. FlexTimers</td>
<td>$0.99</td>
<td>Now</td>
</tr>
</tbody>
</table>

*Subject to change*
**Tower Motor Kit: Kinetis V Series KV1x Overview**

**Key Features:**

**Core/System**
- 75MHz ARM CM0+ with 4ch DMA
- H/W DIV & SQRT block

**Memory**
- 32KB Flash, 8KB SRAM

**Communications**
- Multiple serial ports

**Analog**
- 2 x 8ch 16-bit ADC
  - 835nS conversion time
- 1 x12-bit DAC,
- 2 x ACMP w/ 6b DAC

**Timers**
- 1x6ch FlexTimer (PWM)
- 2x2ch FlexTimer (PWM/QDEC)
- Programmable Delay Block

**Others**
- 32-bit CRC
- Inter-module Crossbar Switch
- Up to 35 I/Os
- 1.71V-3.6V; -40 to 105°C

**Packages**
- 32QFN, 32LQFP, 48LQFP
- Pin-to-pin compatible with K series

---

A high performance, cost-optimized and best-in-class enabled 32-bit ARM Cortex-M0+ MCU for low/mid range Brushless DC and FOC PMSM Motor Control

---

**Core**
- ARM Cortex-M0+ 75MHz
- Debug Interfaces
- HW Divide & SqrRoot
- Interrupt Controller

**System**
- Internal and External Watchdogs
- Inter-Module Crossbar

**Memories**
- Program Flash 32KB
- SRAM 8KB

**Clocks**
- Frequency-Locked Loop
- Low/High Frequency Oscillators
- Internal Reference Clocks

**Security and Integrity**
- Cyclic Redundancy Check (CRC)

**Analog**
- 2 x16-bit ADC
- 2 x ACMP
- 1 x12-bit DAC

**Timers**
- 6ch FlexTimer
- 2x2ch FlexTimer
- Programmable Delay Block
- Low-Power Timer

**Communication Interfaces**
- 1xI2C
- 2xUARTs
- 1xSPI

**HMI**
- GPIO
Tower Motor Kit: System Diagram

3PP Driver Chip (MC33937)

Phase Voltages

Phase A
Phase B
Phase C

BLDC Motor

3PP

Vb+
Vb-

FlexTimer
UART

KV10

SPI

GPIO

FTM/GPIO

UART

RAppID BL Utility

FREEMASTER

Freescale

External Use | 44
Tower Motor Kit

Any Questions?
Agenda

• Overview: 30 minutes
  - Introduction and Objectives
  - Motor Control Development Toolbox: Library blocks, FreeMASTER, and Bootloader
  - Model Based Design Steps: Simulation, SIL, PIL and ISO26262

• Hands-on Demo: 20 minutes
  - Convert simple model to run on Motor Kit with MCD Toolbox and use FreeMASTER

• Motor Control: 30 minutes
  - Motor Kit (Describe Freescale 3-Phase Motor Kit)
  - Trapezoidal control and how to use it to turn a motor

• Motor Control Hands-on Demo: 80 minutes
  - Implement Trapezoidal Motor Control on Motor Kit
  - Run software from the model and use FreeMASTER to monitor and tune parameters

• SIL/PIL Demo: 10 minutes

• Summary and Q&A: 10 minutes
Trapezoidal Control: Brushless DC Motor

- A BLDC motor consists of a rotor with permanent magnets and a stator with phase windings. A BLDC motor needs electronic commutation for the control of current through its three phase windings.
Trapezoidal Control: Commutation Method

- Trapezoidal control is one type of commutation method used to turn a motor where only two phase windings will conduct current at any one time. With direction also to consider, that leaves six possible patterns.
Trapezoidal Control: Commutation Control

- By adding switches, the current flow can be controlled by a MCU to perform trapezoidal control.
Trapezoidal Control: Turning the Motor CW

With the switches, the stator can be used to turn the motor to the desired direction and location by creating a magnetic field that affects the magnets on the rotor.

<table>
<thead>
<tr>
<th>CW</th>
<th>Phase A</th>
<th>Phase B</th>
<th>Phase C</th>
</tr>
</thead>
<tbody>
<tr>
<td>0/180°</td>
<td>Vb-</td>
<td>NC</td>
<td>Vb+</td>
</tr>
<tr>
<td>30°</td>
<td>Vb-</td>
<td>Vb+</td>
<td>NC</td>
</tr>
<tr>
<td>60°</td>
<td>NC</td>
<td>Vb+</td>
<td>Vb-</td>
</tr>
<tr>
<td>90°</td>
<td>Vb+</td>
<td>NC</td>
<td>Vb-</td>
</tr>
<tr>
<td>120°</td>
<td>Vb+</td>
<td>Vb-</td>
<td>NC</td>
</tr>
<tr>
<td>150°</td>
<td>NC</td>
<td>Vb-</td>
<td>Vb+</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Top Switch</th>
<th>Vb+</th>
<th>Vb-</th>
<th>NC</th>
</tr>
</thead>
<tbody>
<tr>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bottom Switch</th>
<th>Vb+</th>
<th>Vb-</th>
<th>NC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td></td>
</tr>
</tbody>
</table>
Trapezoidal Control: Turning the Motor CCW

• With the switches, the stator can be used to turn the motor to the desired direction and location by creating a magnetic field that affects the magnets on the rotor.

<table>
<thead>
<tr>
<th>CCW</th>
<th>Phase A</th>
<th>Phase B</th>
<th>Phase C</th>
</tr>
</thead>
<tbody>
<tr>
<td>0/180°</td>
<td>Vb+</td>
<td>NC</td>
<td>Vb-</td>
</tr>
<tr>
<td>30°</td>
<td>Vb+</td>
<td>Vb-</td>
<td>NC</td>
</tr>
<tr>
<td>60°</td>
<td>NC</td>
<td>Vb-</td>
<td>Vb+</td>
</tr>
<tr>
<td>90°</td>
<td>Vb-</td>
<td>NC</td>
<td>Vb+</td>
</tr>
<tr>
<td>120°</td>
<td>Vb-</td>
<td>Vb+</td>
<td>NC</td>
</tr>
<tr>
<td>150°</td>
<td>NC</td>
<td>Vb+</td>
<td>Vb-</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Vb+</th>
<th>Vb-</th>
<th>NC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top Switch</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td>Bottom Switch</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
</tr>
</tbody>
</table>
Trapezoidal Control: Motor Position

- In order to commutate correctly for trapezoidal control, motor position information is required for proper motor rotation. The motor position information enables the MOSFETs or IGBTs in the inverter to properly be switched ON and OFF to ensure proper direction of current flow through the phase windings. Therefore, Hall sensors are used as position sensors for trapezoidal control. Each Hall sensor is placed 120 degrees apart and delivers a “high” state when facing a “north pole” and a “low” state when facing a “south pole”.

![Diagram of Hall sensor placement](image)
Trapezoidal Control: Motor Position CW

With three Hall sensors, it is possible to have eight states with two invalid states. That leaves six valid states that can be used to determine which two phase coils to drive the current through and in which direction. The six states are generated due to rotation of the motor.

<table>
<thead>
<tr>
<th>Hall A</th>
<th>Hall B</th>
<th>Hall C</th>
<th>State</th>
<th>CW</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>0/180°</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6</td>
<td>30°</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>60°</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>90°</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>120°</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>150°</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Invalid</td>
<td>n/a</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Invalid</td>
<td>n/a</td>
</tr>
</tbody>
</table>

Hall A

Hall B

Hall C
Trapezoidal Control: Motor Position CCW

• With three Hall sensors, it is possible to have eight states with two invalid states. That leaves six valid states that can be used to determine which two phase coils to drive the current through and in which direction. The six states are generated due to rotation of the motor.

<table>
<thead>
<tr>
<th>Hall A</th>
<th>Hall B</th>
<th>Hall C</th>
<th>State</th>
<th>CCW</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>0/180°</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>30°</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>60°</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>90°</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>120°</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6</td>
<td>150°</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Invalid</td>
<td>n/a</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Invalid</td>
<td>n/a</td>
</tr>
</tbody>
</table>

Hall A

Hall B

Hall C
Trapezoidal Control: Bringing It All Together

• With the commutation table and the motor position table, a full trapezoidal control algorithm can be developed.

<table>
<thead>
<tr>
<th>Hall A</th>
<th>Hall B</th>
<th>Hall C</th>
<th>State</th>
<th>CW</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>0/180°</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6</td>
<td>30°</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>60°</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>90°</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>120°</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>150°</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Invalid</td>
<td>n/a</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Invalid</td>
<td>n/a</td>
</tr>
</tbody>
</table>

Motor Position Table Input

<table>
<thead>
<tr>
<th>CW</th>
<th>Phase A</th>
<th>Phase B</th>
<th>Phase C</th>
</tr>
</thead>
<tbody>
<tr>
<td>0/180°</td>
<td>Vb+</td>
<td>NC</td>
<td>Vb-</td>
</tr>
<tr>
<td>30°</td>
<td>Vb+</td>
<td>Vb-</td>
<td>NC</td>
</tr>
<tr>
<td>60°</td>
<td>NC</td>
<td>Vb-</td>
<td>Vb+</td>
</tr>
<tr>
<td>90°</td>
<td>Vb-</td>
<td>NC</td>
<td>Vb+</td>
</tr>
<tr>
<td>120°</td>
<td>Vb-</td>
<td>Vb+</td>
<td>NC</td>
</tr>
<tr>
<td>150°</td>
<td>NC</td>
<td>Vb+</td>
<td>Vb-</td>
</tr>
</tbody>
</table>

Commutation Table Output

<table>
<thead>
<tr>
<th>Top Switch</th>
<th>Vb+</th>
<th>Vb-</th>
<th>NC</th>
</tr>
</thead>
<tbody>
<tr>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bottom Switch</th>
<th>Vb+</th>
<th>Vb-</th>
<th>NC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td></td>
</tr>
</tbody>
</table>
Trapezoidal Control: Bringing It All Together

- With the commutation table and the motor position table, a full trapezoidal control algorithm can be developed.

### Trapezoidal Control Algorithm Clockwise Rotation

<table>
<thead>
<tr>
<th>Hall A</th>
<th>Hall B</th>
<th>Hall C</th>
<th>State</th>
<th>CW</th>
<th>Phase A</th>
<th>Phase B</th>
<th>Phase C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>0/180°</td>
<td>Vb+</td>
<td>NC</td>
<td>Vb-</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6</td>
<td>30°</td>
<td>Vb+</td>
<td>Vb-</td>
<td>NC</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>60°</td>
<td>NC</td>
<td>Vb-</td>
<td>Vb+</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>90°</td>
<td>Vb-</td>
<td>NC</td>
<td>Vb+</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>120°</td>
<td>Vb-</td>
<td>Vb+</td>
<td>NC</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>150°</td>
<td>NC</td>
<td>Vb+</td>
<td>Vb-</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Invalid</td>
<td>n/a</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Invalid</td>
<td>n/a</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

### Switch Control

<table>
<thead>
<tr>
<th>Top Switch</th>
<th>Vb+</th>
<th>Vb-</th>
<th>NC</th>
</tr>
</thead>
<tbody>
<tr>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bottom Switch</th>
<th>Vb+</th>
<th>Vb-</th>
<th>NC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
</tr>
</tbody>
</table>
Trapezoidal Control: Bringing It All Together

• With the commutation table and the motor position table, a full trapezoidal control algorithm can be developed.

Trapezoidal Control Algorithm Counter Clockwise Rotation

<table>
<thead>
<tr>
<th>Hall A</th>
<th>Hall B</th>
<th>Hall C</th>
<th>State</th>
<th>CW</th>
<th>Phase A</th>
<th>Phase B</th>
<th>Phase C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>Vb-</td>
<td>NC</td>
<td>Vb+</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>30°</td>
<td>Vb-</td>
<td>Vb+</td>
<td>NC</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>60°</td>
<td>NC</td>
<td>Vb+</td>
<td>Vb-</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>90°</td>
<td>Vb+</td>
<td>NC</td>
<td>Vb-</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>120°</td>
<td>Vb+</td>
<td>Vb-</td>
<td>NC</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6</td>
<td>150°</td>
<td>NC</td>
<td>Vb-</td>
<td>Vb+</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Invalid</td>
<td>n/a</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Invalid</td>
<td>n/a</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Vb+  Vb-  NC

Top Switch
On  Off  Off

Bottom Switch
Off  On  Off
Trapezoidal Control

Any Questions?
Agenda

• Overview: 30 minutes
  - Introduction and Objectives
  - Motor Control Development Toolbox: Library blocks, FreeMASTER, and Bootloader
  - Model Based Design Steps: Simulation, SIL, PIL and ISO26262

• Hands-on Demo: 20 minutes
  - Convert simple model to run on Motor Kit with MCD Toolbox and use FreeMASTER

• Motor Control: 30 minutes
  - Motor Kit (Describe Freescale 3-Phase Motor Kit)
  - Trapezoidal control and how to use it to turn a motor

• Motor Control Hands-on Demo: 80 minutes
  - Implement Trapezoidal Motor Control on Motor Kit
  - Run software from the model and use FreeMASTER to monitor and tune parameters

• SIL/PIL Demo: 10 minutes

• Summary and Q&A: 10 minutes
Hands-on Demo: Implement Trapezoidal Motor Control on Motor Kit

Summary Trapezoidal Motor Control on KV10 steps:

1. Open Trap_Ctrl.slx
2. Save model as KV10_Trap_Ctrl.slx
3. Configure KV10 configuration block
4. Configure Input port blocks to read motor hall position state
5. Configure Input Edge Capture Blocks to detect change in motor position sensors
6. Configure an Input Edge Capture Block to measure one of the Hall sensor pulse width for RPM calculation
7. Configure Digital Input for use in controlling RPM Request
8. Configure DSPI blocks to interface to Freescale 3PP driver
9. Connect and configure PWM blocks for output to switches
Hands-on Demo: Implement Trapezoidal Motor Control on Motor Kit

Target: MKV10Z32VLF7
System clock: 75 MHz
XTAL clock: 10 MHz
Compiler: GCC
Download Code after build: (COM59, 115200)
Freemaster: UART0 (115200)
System Tick Interrupt Priority: 1
Hands-on Demo: Implement Trapezoidal Motor Control on Motor Kit

Configure Hall Sensor Input Block using Digital I/O steps:
Set input blocks to correct pins.
Hands-on Demo: Implement Trapezoidal Motor Control on Motor Kit

The Trapezoidal Control algorithm needs to run on any change of state or event so that we can change our Commutation State as quickly as possible. Therefore we want to trigger off of any rising or falling edge of the hall sensors.
Measure the pulse width of a Hall sensor so that motor speed can be calculated.
Hands-on Demo: Implement Trapezoidal Motor Control on Motor Kit

- Using the Input Edge Capture blocks we can monitor the transitions of the hall sensors to trigger a function call to set a new commutation state. We can measure the hall sensor pulse time to determine RPM.

- We also can capture the number of changes in the hall sensors states and save that for determining the motor RPM.
Hands-on Demo: Implement Trapezoidal Motor Control on Motor Kit

Make sure to check Hands-on Demo: Implement Trapezoidal Motor Control on Motor Kit.
Hands-on Demo: Implement Trapezoidal Motor Control on Motor Kit

- Input Edge Capture block to get time stamp on the channel input event.
  - Input signals:
    - Ch[0] input pin PTD6: [47] : [31] : [31]
Hands-on Demo: Implement Trapezoidal Motor Control on Motor Kit

MotorSpeedReqInput Block steps:
Hands-on Demo: Implement Trapezoidal Motor Control on Motor Kit

3PhaseDutyCycleOut Block with Flex PWM Blocks steps:
Pull Simple PWM phase block from library, connect to phase A and configure.
Hands-on Demo: Implement Trapezoidal Motor Control on Motor Kit

PWM Block steps:
Hands-on Demo: Implement Trapezoidal Motor Control on Motor Kit

PWM Block steps:
Hands-on Demo: Implement Trapezoidal Motor Control on Motor Kit

3PhaseDutyCycleOut Block with Flex PWM Blocks steps:
Hands-on Demo: FreeMASTER to Monitor and Tune Parameters

Using FreeMASTER with Hands-on Demo

1. Start FreeMASTER and open projectKV10_Trap_Ctrl.pmp. Press OK if a message comes up that the map file has been updated.

2. Go to Project Options pull-down and select “Options”. Verify that COM settings are the same as what they were set to in your model.

3. Once the COM settings are correct, press the STOP button.

4. Change MotorSpeedReqFreemaster Variable to 1000 RPM.
Hands-on Demo: How can we make the response better

Using FreeMASTER and PWM resolution

1. Adjust the Proportional and Integral gain constants for faster response.
2. See if changing the PWM resolution can help (see screen shot)

Change scaling so that instead of 0-100 make it 0-1000 effectively making the PWM resolution 0.1% vs. 1.0%.
Hands-on Demo: Trapezoidal Motor Control

Any Questions?
Agenda

• Overview: 30 minutes
  - Introduction and Objectives
  - Motor Control Development Toolbox: Library blocks, FreeMASTER, and Bootloader
  - Model Based Design Steps: Simulation, SIL, PIL and ISO26262
• Hands-on Demo: 20 minutes
  - Convert simple model to run on Motor Kit with MCD Toolbox and use FreeMASTER
• Motor Control: 30 minutes
  - Motor Kit (Describe Freescale 3-Phase Motor Kit)
  - Trapezoidal control and how to use it to turn a motor
• Motor Control Hands-on Demo: 80 minutes
  - Implement Trapezoidal Motor Control on Motor Kit
  - Run software from the model and use FreeMASTER to monitor and tune parameters
• SIL/PIL Demo: 10 minutes
• Summary and Q&A: 10 minutes
SIL/PIL Demo

1. Open Model “KV10_FOC_PIL_REF.slm
2. You will see a motor simulation of an FOC control algorithm
3. Will run model and review results
1. You can switch between SIL and PIL by using the tools menu.
SIL/PIL Demo

Any Questions?
Agenda

• Overview: 30 minutes
  - Introduction and Objectives
  - Motor Control Development Toolbox: Library blocks, FreeMASTER, and Bootloader
  - Model Based Design Steps: Simulation, SIL, PIL and ISO26262
• Hands-on Demo: 20 minutes
  - Convert simple model to run on Motor Kit with MCD Toolbox and use FreeMASTER
• Motor Control: 30 minutes
  - Motor Kit (Describe Freescale 3-Phase Motor Kit)
  - Trapezoidal control and how to use it to turn a motor
• Motor Control Hands-on Demo: 80 minutes
  - Implement Trapezoidal Motor Control on Motor Kit
  - Run software from the model and use FreeMASTER to monitor and tune parameters
• SIL/PIL Demo: 10 minutes
• Summary and Q&A: 10 minutes
Summary

You now know how to do Trapezoidal Control and auto-generate software with the Motor Control Development Toolbox directly from the MATLAB™/Simulink™ for Freescale MCUs.

You now understand how to run SIL and PIL with MCD Toolbox and how it can accelerate your development, including systems being developed under IEC 61508 using PIL.

You have gained a good working knowledge of how FreeMASTER works and how it can be used with MCD Toolbox to accelerate development when working with the target hardware.

You have a working knowledge of how to use the Freescale Pre-Driver chip with a motor control application.

You know how the Freescale MCU covered can fit into your motor control application.
Summary: Publications

MathWorks Announces Simulink Code Generation Targets in New Freescale Motor Control Development Toolbox
www.mathworks.com

- Simulink and Embedded Coder enable engineers to generate production code for Freescale MCUs in IEC 61508 (SIL3) and ISO 26262 (ASIL-D) compliant systems.

Freescale likes model-based design, says MathWorks
www.ElectronicsWeekly.com

- MathWorks says Freescale has made a major commitment to model-based design methodologies by adopting Simulink code generation targets in its motor control development toolbox. The toolbox, consisting of Simulink motor control blocks and target-ready …

A model-based tool to support rapid application development for Freescale MCUs
www.Freescale.com - Beyond Bits—Issue VIII

- Model-based design (MBD) is becoming the standard methodology for developing embedded systems that implement the desired behavior of a control system. MBD is a graphical method …
Summary

Any Questions?

Please Fill Out Your Surveys.

Thank you for your time.