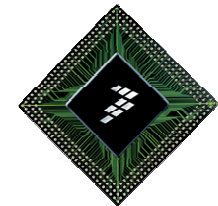




October, 2010

Motor Control with Freescale Microcontrollers



Eduardo Viramontes – Applications Engineer

Freescale, the Freescale logo, Altivec, C-5, CodeTEST, CodeWarrior, ColdFire, C-Ware, mobileGT, PowerQUICC, StarCore, and Symphony are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. BeeKit, BeeStack, CoreNet, the Energy Efficient Solutions logo, Flexis, MXC, Platform in a Package, Processor Expert, QorIQ, QUICC Engine, SMARTMOS, TurboLink and VortiQa are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. © 2010 Freescale Semiconductor, Inc.



Agenda

1. Motor Control & Power Market Trends
2. Target Applications
3. Freescale Motor Control Solutions
4. Motor Control Basics
5. Motor control portfolio
6. BLDC Control Basics
7. Commutation
8. Back EMF Basics
9. Hardware Overview
10. BLDC – Recommended Application
11. Why BLDC with MP16?
12. Controlling a BLDC motor with sensors
13. BLDC with sensors control code on the S08MP16
14. Mini hands-on: Find out how the sensors behave (Lab 0)
15. Electronic motor commutation
16. Hands-on: Run the BLDC with sensors demo (Lab1)
17. Practical sensorless motor control: How is BEMF used?
18. Measure BEMF
19. Open loop startup on a sensorless application
20. Hands-on: Run the BLDC without sensors demo (Lab 2)
21. Summary

Objectives

- ▶ By the end of this session, you should be able to
 - Understand the principles of Motor Control
 - Know the newest Control solutions provided by Freescale including
 - MP16
 - DSC portfolio
 - PPC
 - Kinetis
 - Know how to start writting motor control applications either from scratch or from Freescale reference designs.

Motor Control and Power Conversion Market Trends

Motor Control	Power Conversion
Reduce power consumption Intelligent motor control improves efficiency by 30 percent or more	Increase Conversion Efficiency Cost-effective soft switching techniques
Reduce system and development cost More on-chip peripherals to reduce component count	High Power Density Compact size: high watt per cubic inch
Reuse software, hardware and tools across platforms Ease software migration across wide performance range	High Intelligence Control Digital Controlled Power conversion
Cost-effective safety, reliability and security On-chip safety and security protection	Low Cost System monitoring and protection with less components usage



Motor Control – Freescale Alignment with Trends

Market Trend	Freescale Alignment with Customer Requirements
Reduce power consumption	Vector and sensorless control technology designed into every motor control processor
Reduce system and development cost	8- and 16-bit MCUs start at <\$0.70 16-bit DSCs for ACIM and PMSM solutions 32-bit Power Architecture® MCUs for standard/premium drives
Reuse software, hardware and tools across platforms	- Rich tools, training, reference designs and libraries - Devices are ruggedized with long life and reliability - Industrial products ship 10+ years, with high quality and expert customer support
Cost-effective safety, reliability and security	- Secure SRAM, on-chip data fusing to protect against IP cloning - Hardware encryption to protect against network data hacking - Watchdog and ECC protection against soft errors - Certified IEC software modules

Motor Control Target Applications

► Pumps and fans

- pool pumps, factory systems

► HVAC

- heating fans, air-conditioners

► Industrial drives

- Manufacturing assembly, robotics, wind turbines, printing presses

► Appliances

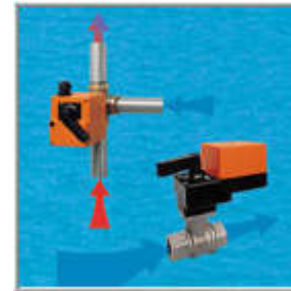
- washers, dryers, power tools

► Medical

- scanners, pumps, diagnostic and therapy

► Automotive Motor Control Trends

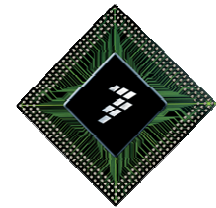
- Fuel/Water pumps, HVAC Fan Control, Window lift
- Increasing adoption of electric and hybrid vehicles continues to drive BLDC demand





Industrial Motor Control Solutions

Freescal Motor Control Processors



Freescal, the Freescal logo, Altivec, C-5, CodeTEST, CodeWarrior, ColdFire, C-Ware, mobileGT, PowerQUICC, StarCore, and Symphony are trademarks of Freescal Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. BeeKit, BeeStack, CoreNet, the Energy Efficient Solutions logo, Flexis, MXC, Platform in a Package, Processor Expert, QorIQ, QUICC Engine, SMARTMOS, TurboLink and VortiQa are trademarks of Freescal Semiconductor, Inc. All other product or service names are the property of their respective owners. © 2010 Freescal Semiconductor, Inc.

Typical Motor Control MCU Peripheral Functions

► Timer:

- PWM signals < 20Khz
- Dead time insertion
- Commutation (mask-out)
- ADC triggering
- Fault control

► ADC

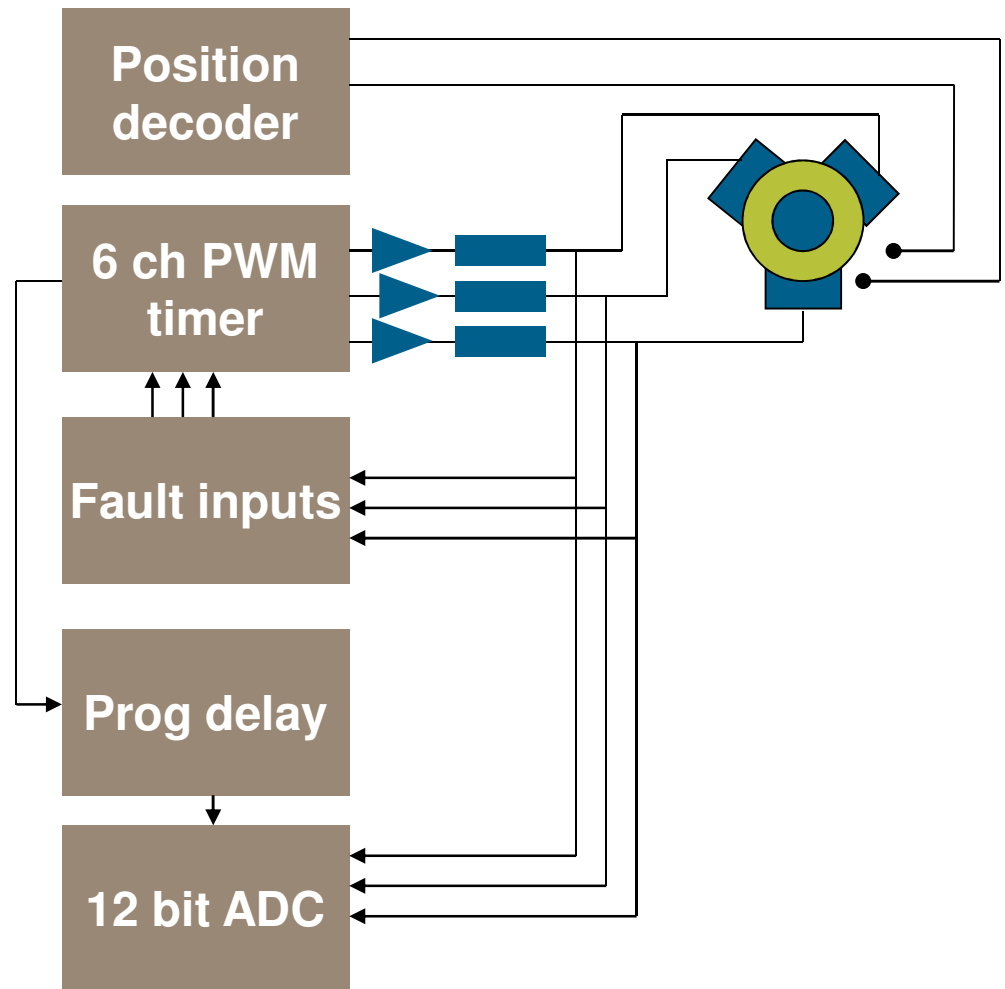
- Measure current

► Delay block

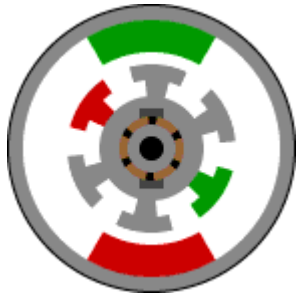
- Set ADC measurement at specific times

► Position decoder

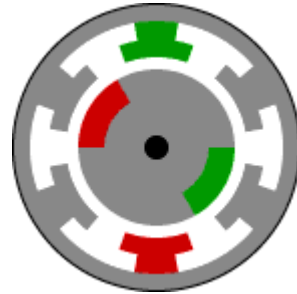
- Quadrature decoder inputs if not sensorless



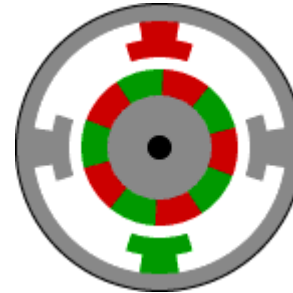
Many Different Motor Types ...



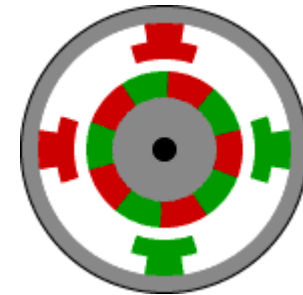
DC Motor



**Brushless DC Motor
(BLDC)**



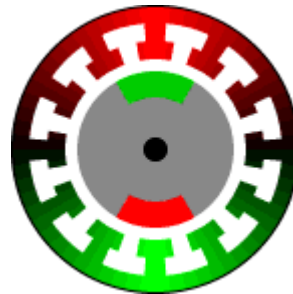
Stepper Motor (full step)



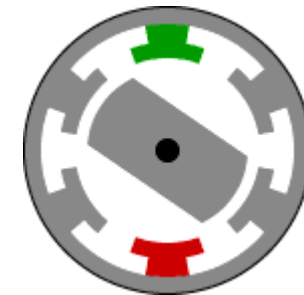
Stepper Motor (half step)



AC Induction Motor (ACIM)



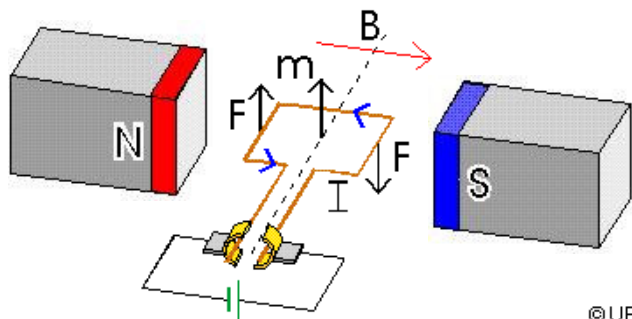
**Permanent Magnet
Synchronous Motor (PMSM)**



Switched Reluctance Motor

Understanding the Motor Basics

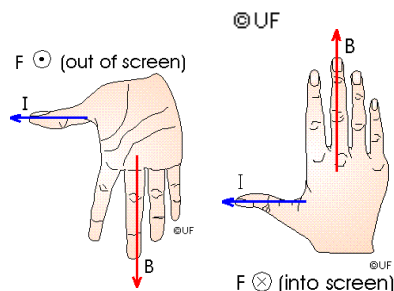
DC Motor Torque Establishment



Right Hand Rule

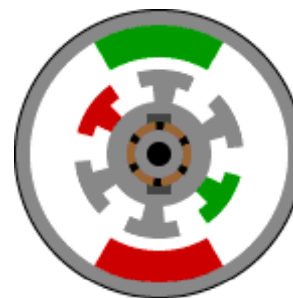
I = Current Flow

B = Magnetic Field



DC Motor Principle

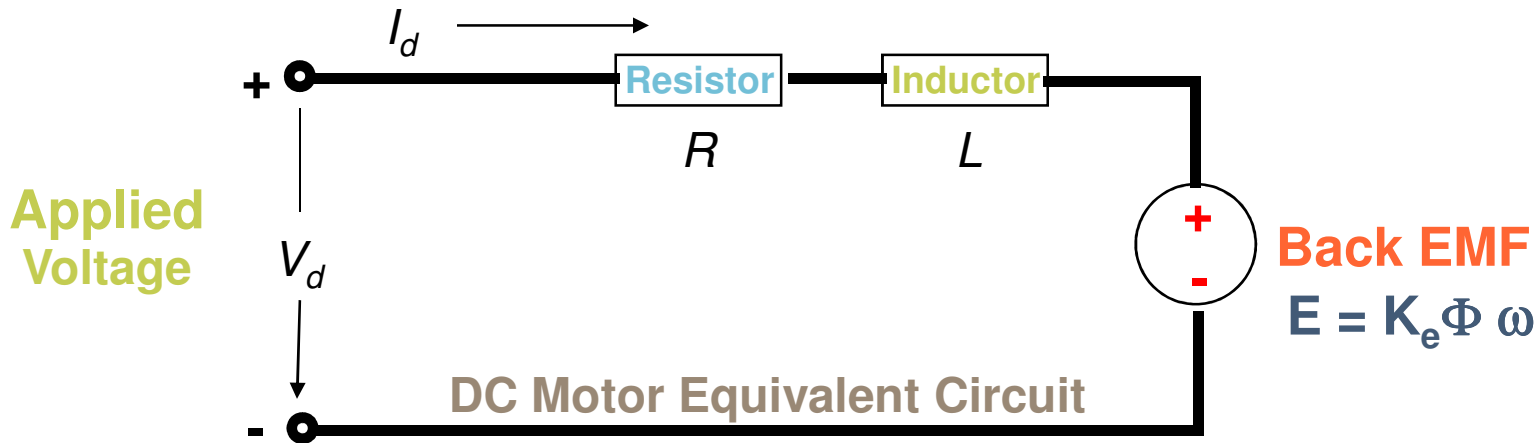
- The stator of a **Permanent Magnet DC Motor** is composed of two or more permanent magnet pole pieces



- The rotor is composed of windings which are connected to a mechanical commutator. In this case the rotor has three pole pairs

Brush DC motor control is simple:
Apply voltage → Commutation occurs mechanically

Simple Model of a DC Motor



Dynamic Motor speed:

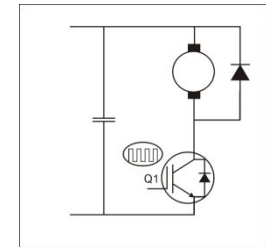
$$\omega = \frac{V_d - I_d \times R - L \frac{dI_d}{dt}}{K_e \times \Phi}$$

- ▶ Speed is increased by increasing the voltage
- ▶ Torque is controlled by controlling the current
- ▶ Direction is determined by the direction of the current

PWM Control of DC Motor

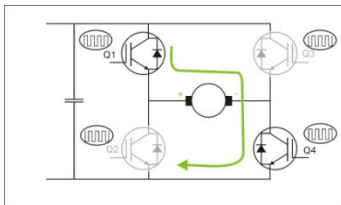
- ▶ Same motor can have different control configurations
(i.e. Simple switch vs. H-Bridge)
- ▶ Same control configuration can have different operating modes
(i.e. Bipolar PWM vs. Unipolar PWM, independent vs. complementary)
- ▶ Different modes have advantages and disadvantages

Simple Switch
(uni-directional operation)



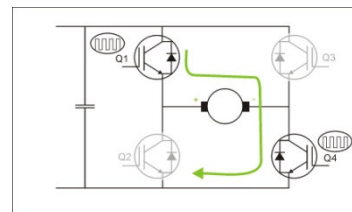
1 - PWM

H-Bridge - Complementary PWM
(4 quadrant operation)



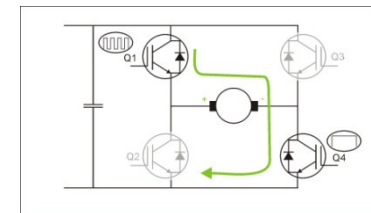
4 – PWMs with Deadtime

H-Bridge - Independent Bipolar PWM
(bi-directional operation)



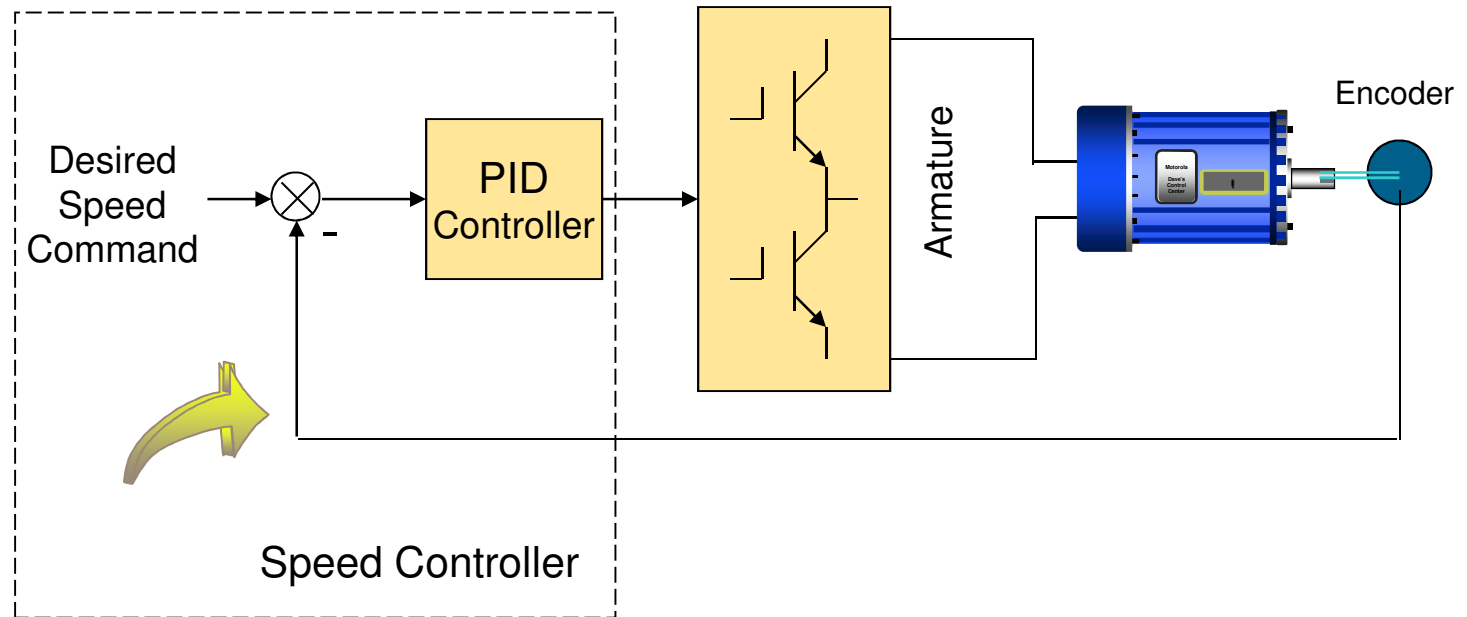
2 – PWMs

H-Bridge - Independent Unipolar PWM
(bi-directional operation)



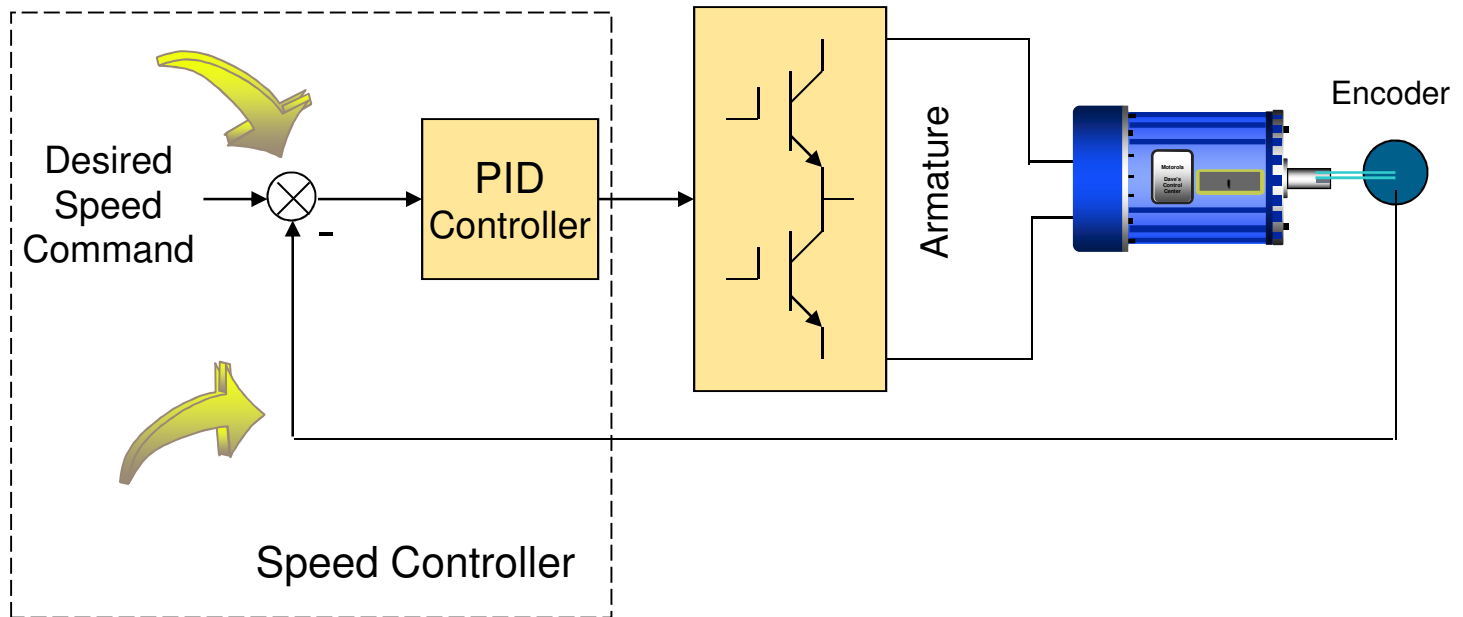
**2 – PWMs
2 – GPIO**

Simple Speed Control On A Brush DC Motor



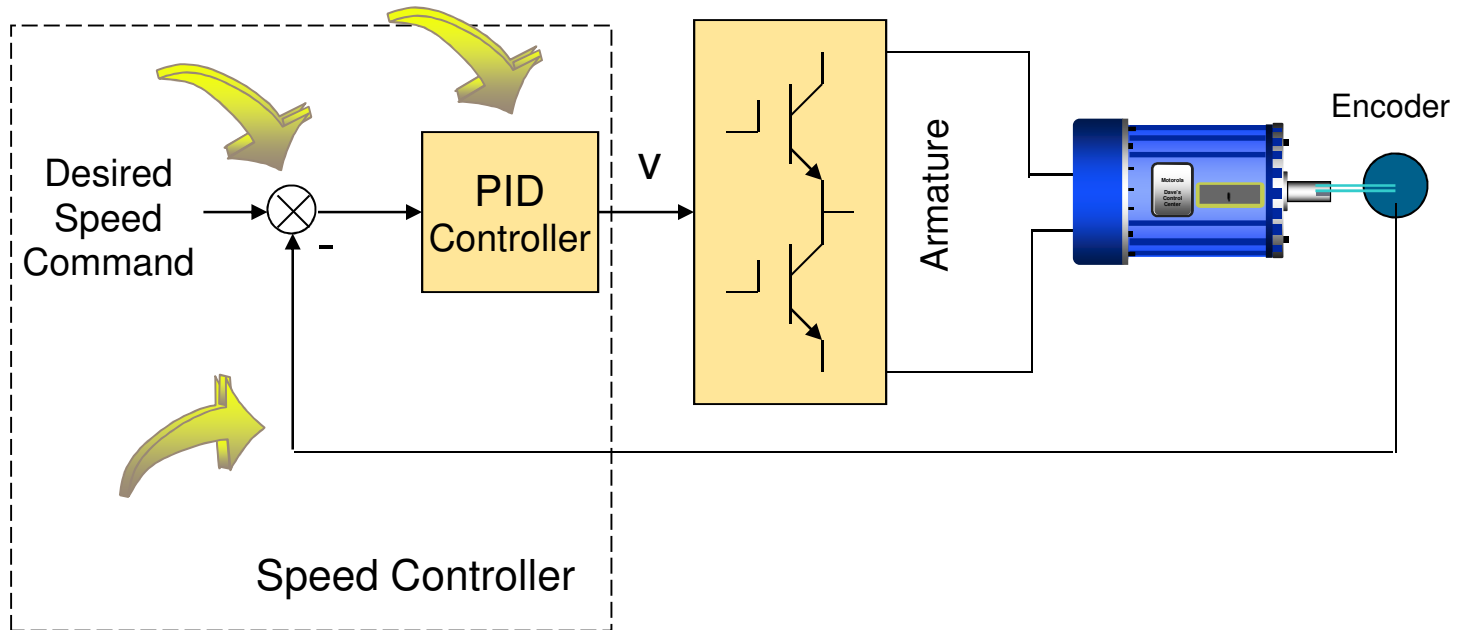
1. Measure speed of the motor

Simple Speed Control On A Brush DC Motor



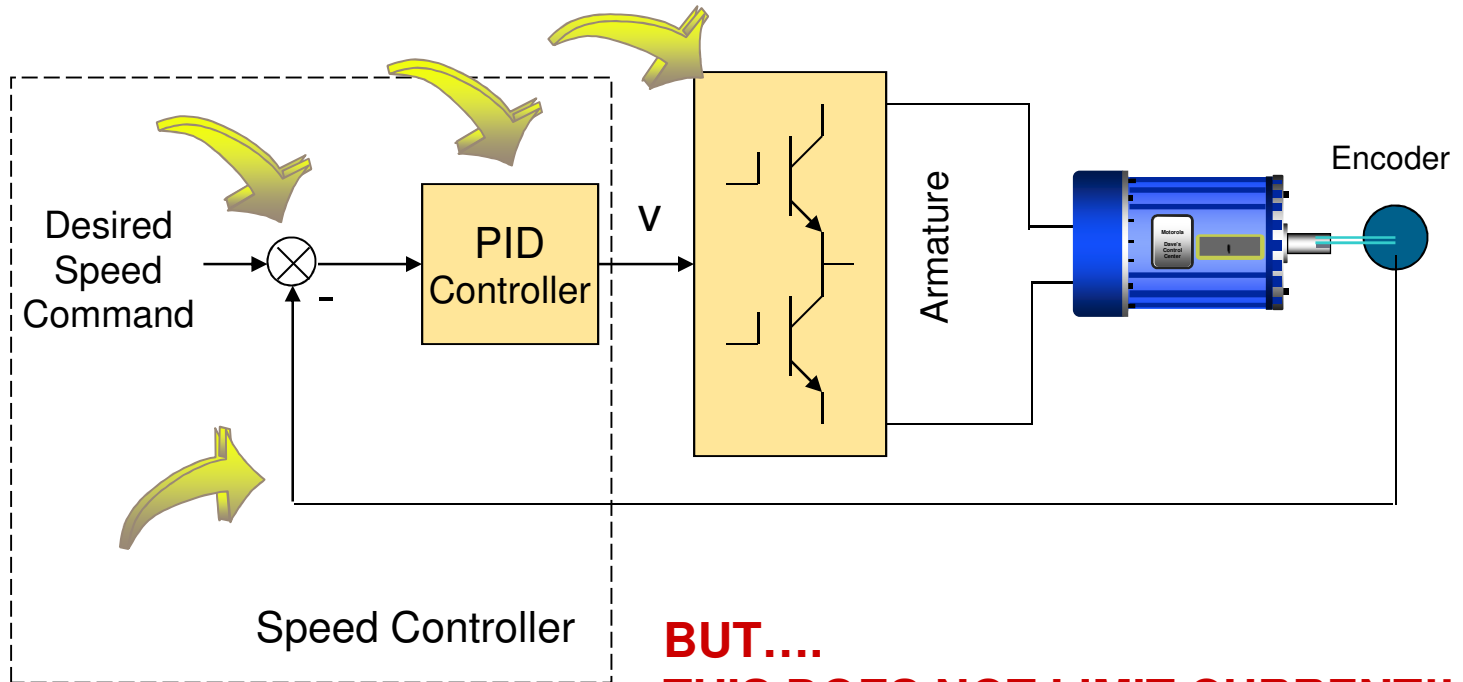
1. Measure speed of the motor
2. Compare the measured speed with the desired speed and generate an error signal

Simple Speed Control On A Brush DC Motor



1. Measure speed of the motor
2. Compare the measured speed with the desired speed and generate an error signal
3. Amplify the error signal to generate a correction voltage

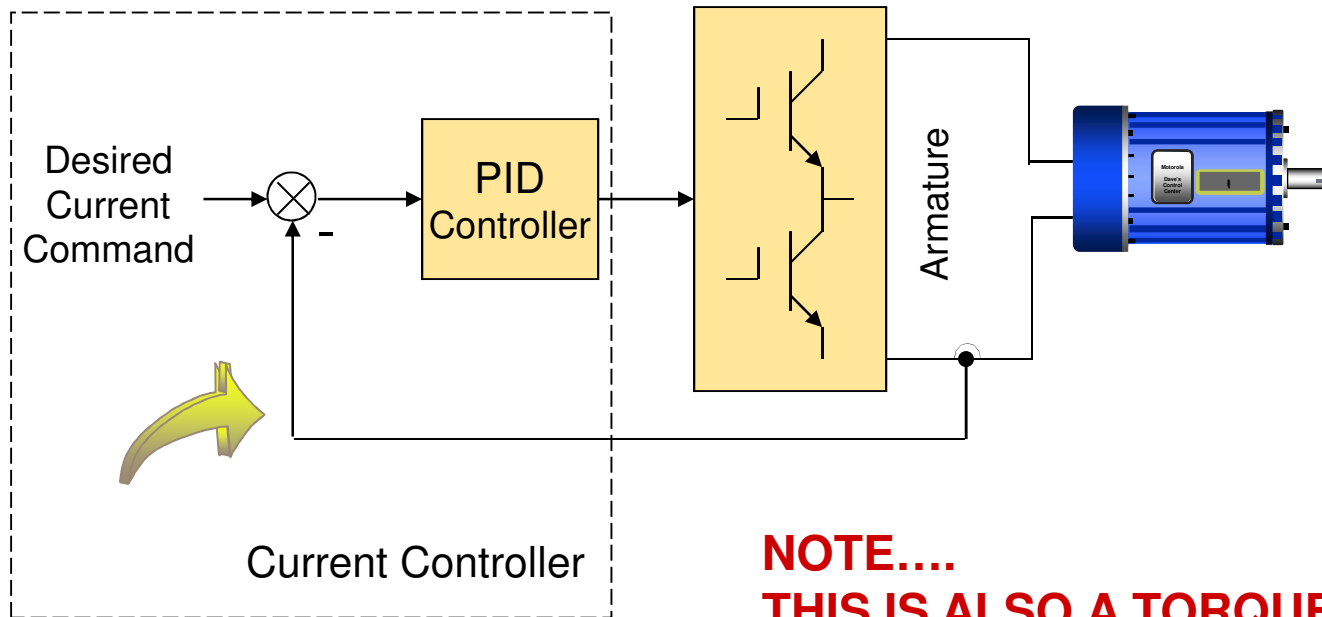
Simple Speed Control On A Brush DC Motor



**BUT....
THIS DOES NOT LIMIT CURRENT!!**

1. Measure speed of the motor
2. Compare the measured speed with the desired speed and generate an error signal
3. Amplify the error signal to generate a correction voltage
4. Modulate the correction voltage onto the motor terminals

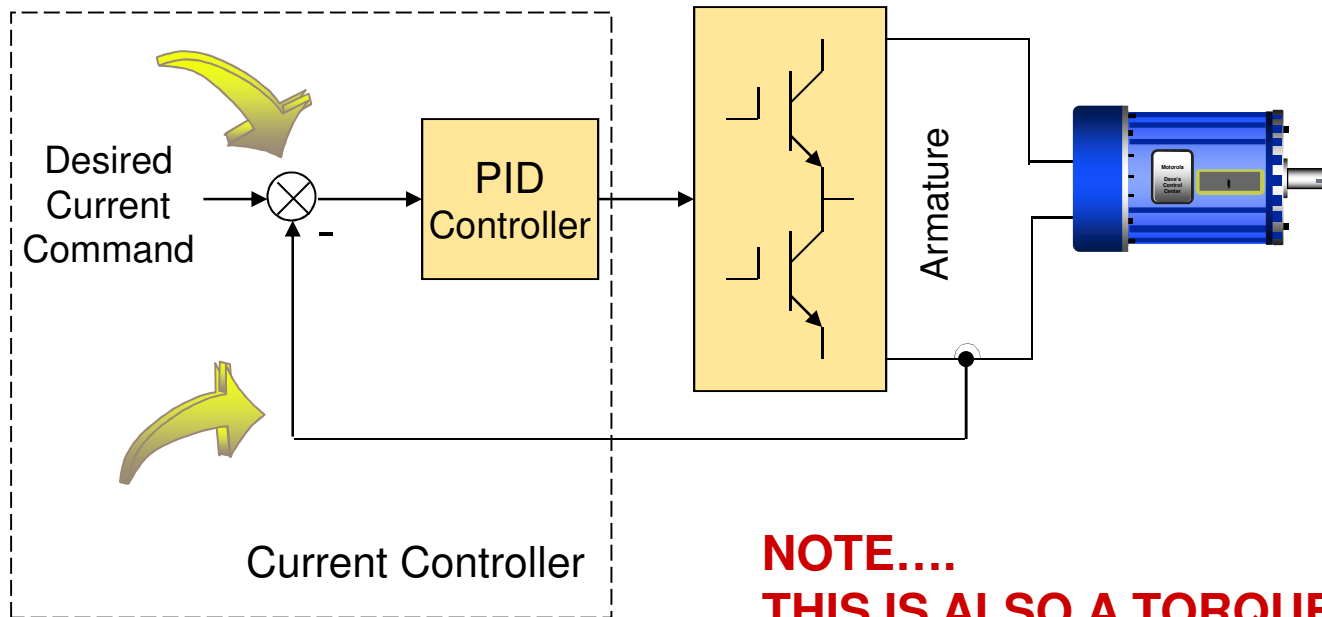
Current Control On A Brush DC Motor



**NOTE....
THIS IS ALSO A TORQUE
CONTROLLER!!**

1. Measure the current of the motor

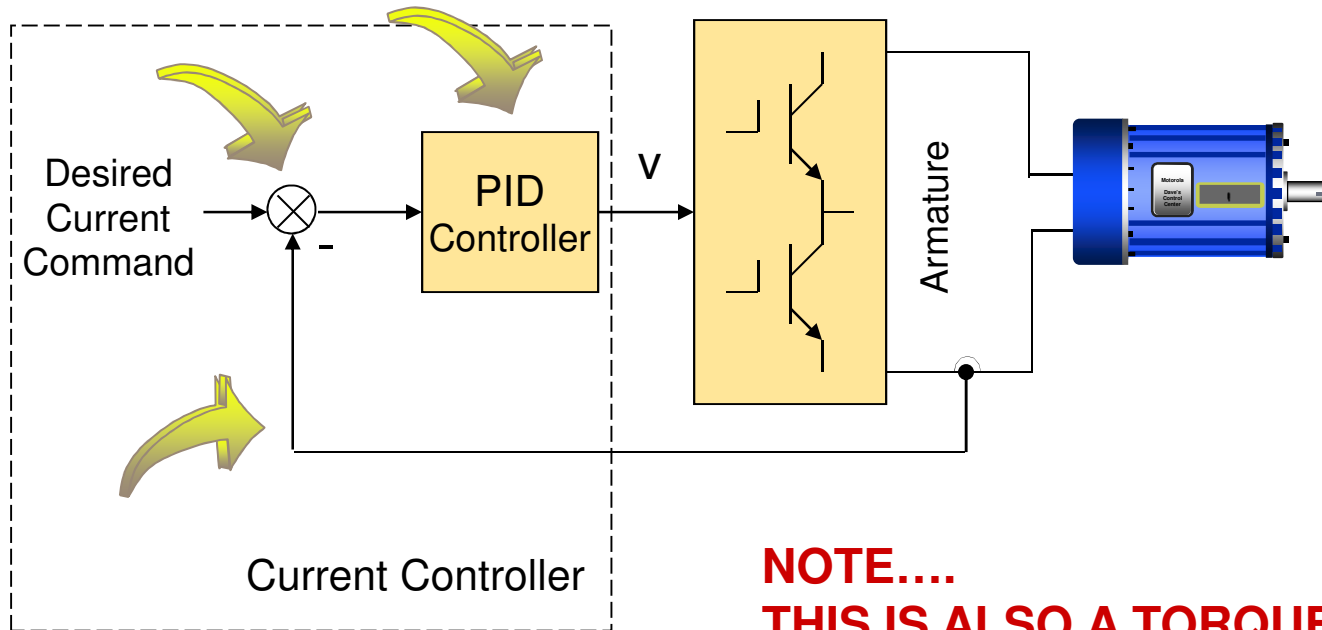
Current Control On A Brush DC Motor



**NOTE....
THIS IS ALSO A TORQUE
CONTROLLER!!**

1. Measure the current of the motor
2. Compare the measured current with the desired current and generate an error signal

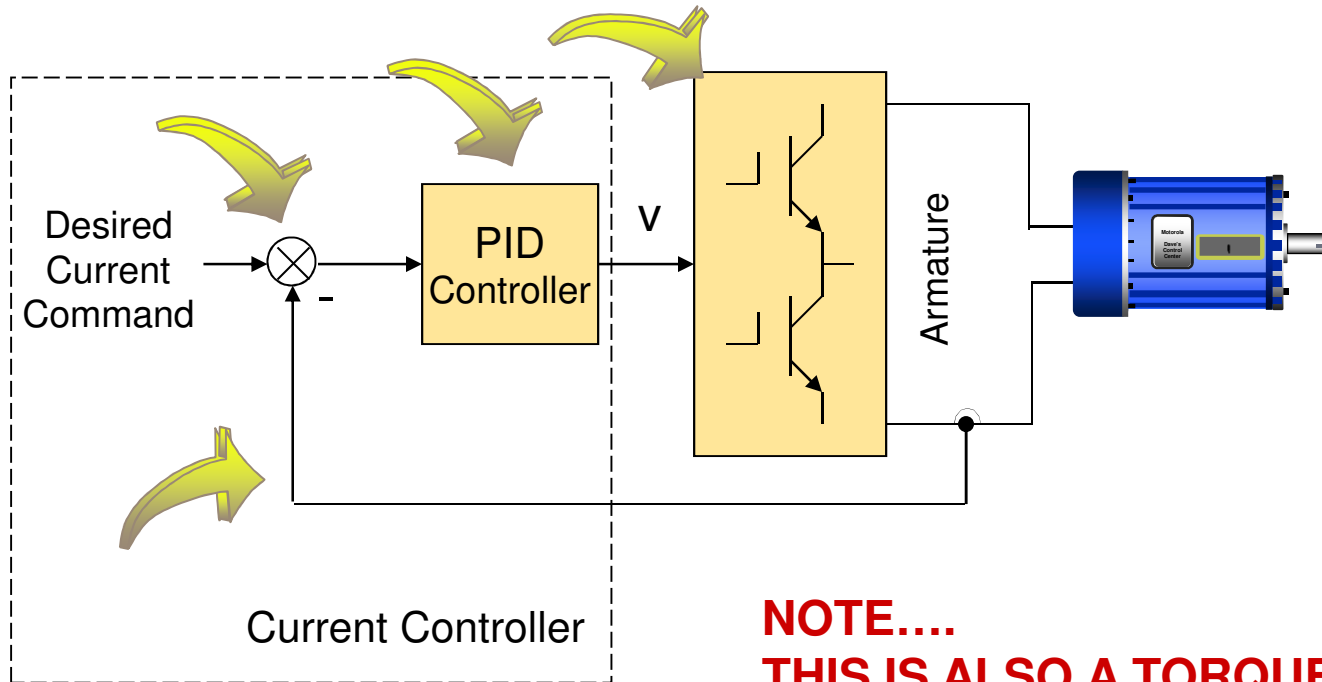
Current Control On A Brush DC Motor



**NOTE....
THIS IS ALSO A TORQUE
CONTROLLER!!**

1. Measure the current of the motor
2. Compare the measured current with the desired current and generate an error signal
3. Amplify the error signal to generate a correction voltage

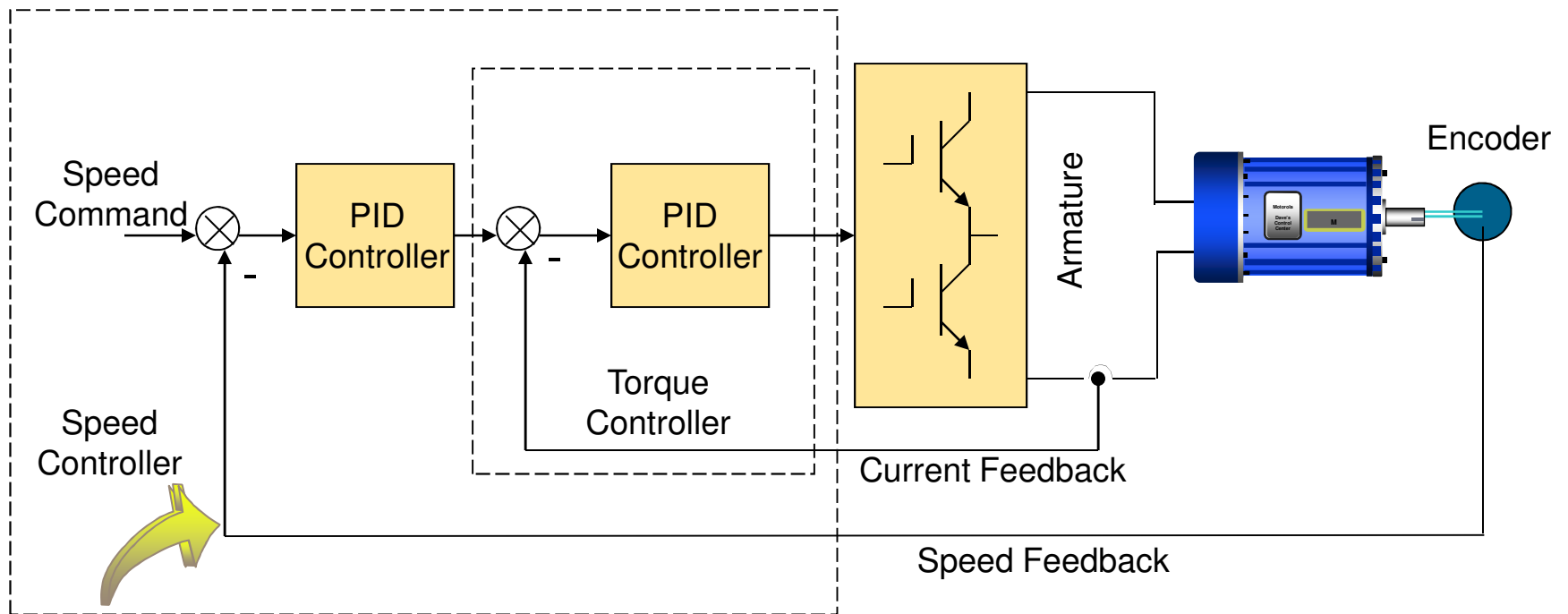
Current Control On A Brush DC Motor



**NOTE....
THIS IS ALSO A TORQUE
CONTROLLER!!**

1. Measure the current of the motor
2. Compare the measured current with the desired current and generate an error signal
3. Amplify the error signal to generate a correction voltage
4. Modulate the correction voltage onto the motor terminals

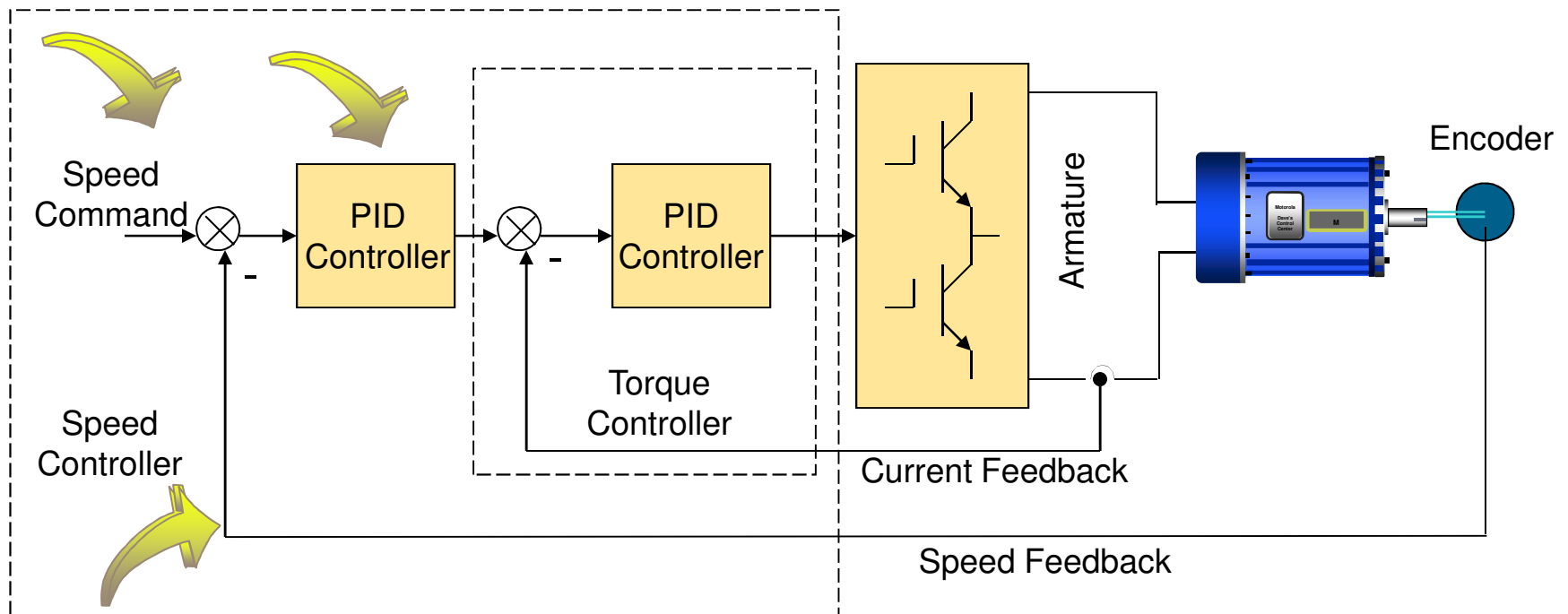
Controlling Speed and Current



1. Measure speed from the motor shaft

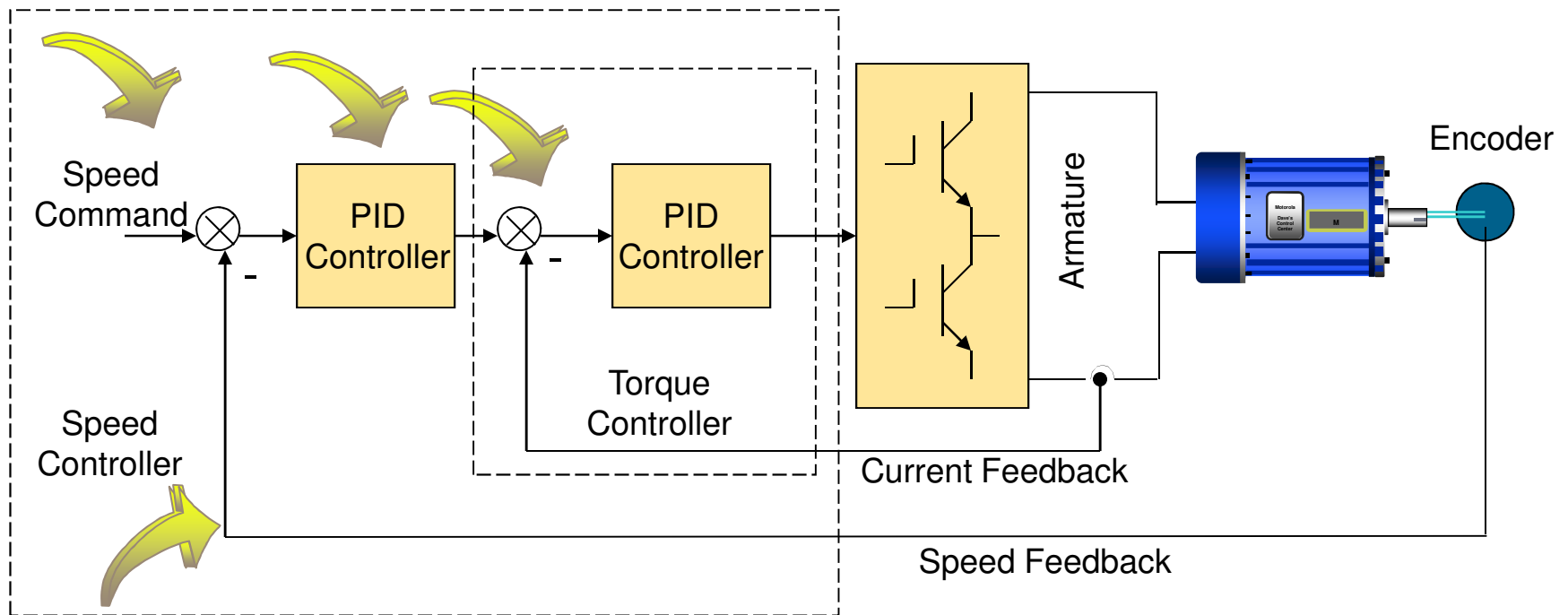


Controlling Speed and Current

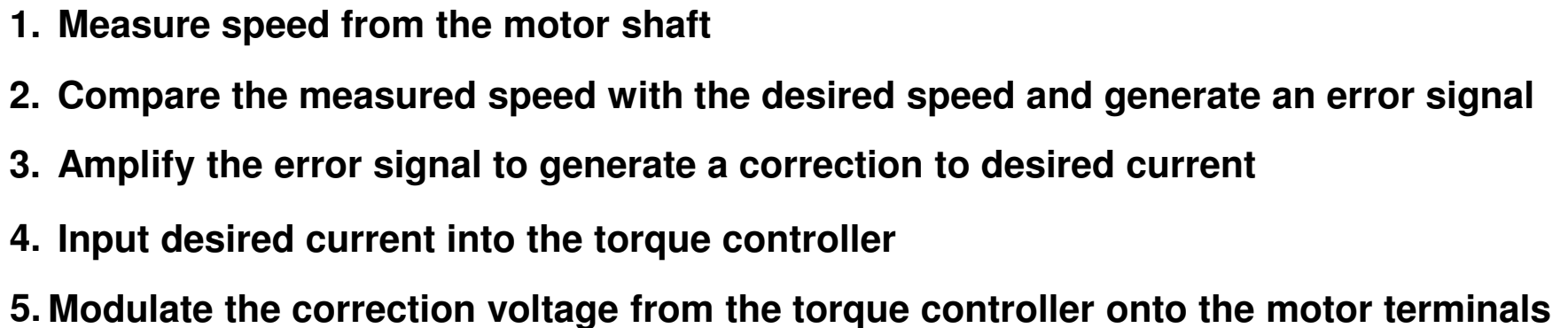


1. Measure speed from the motor shaft
2. Compare the measured speed with the desired speed and generate an error signal
3. Amplify the error signal to generate a correction to desired current

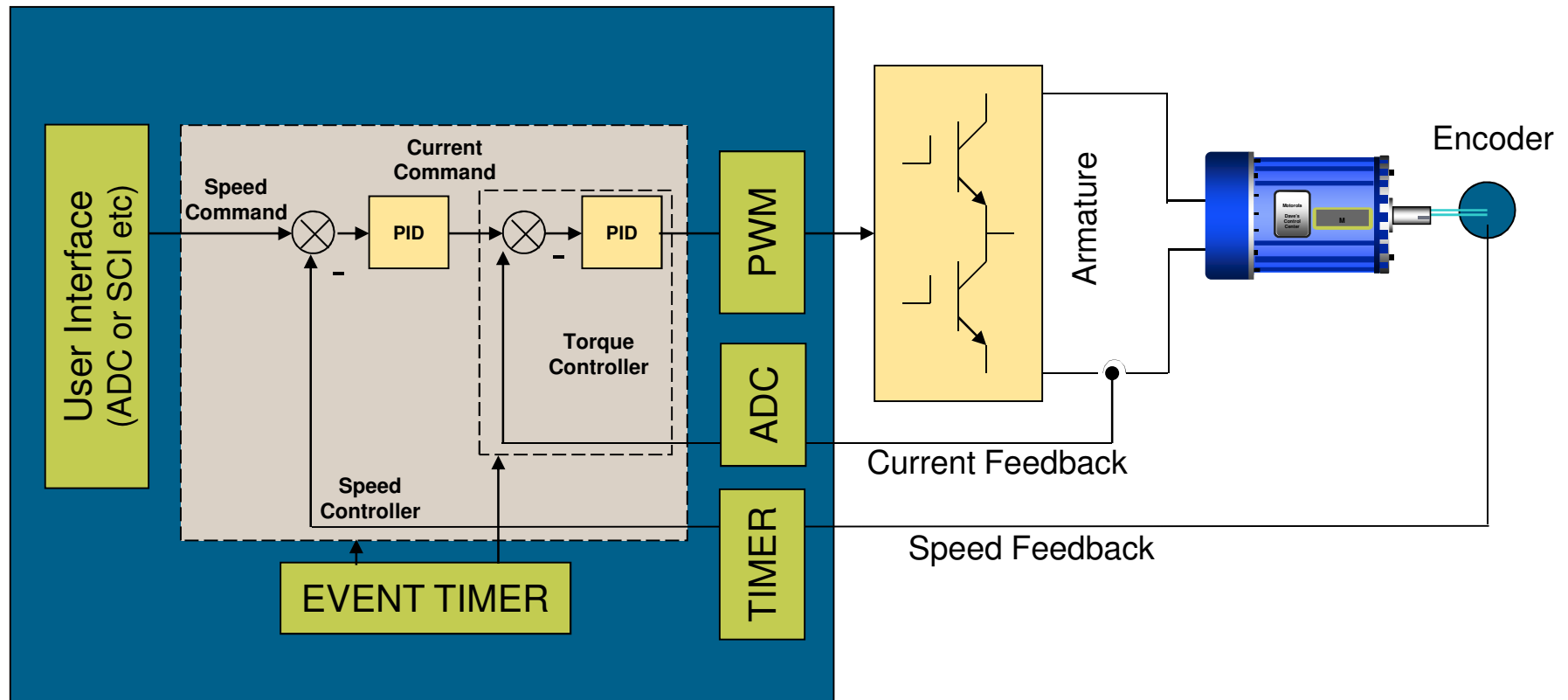
Controlling Speed and Current



1. Measure speed from the motor shaft
2. Compare the measured speed with the desired speed and generate an error signal
3. Amplify the error signal to generate a correction to desired current
4. Input desired current into the torque controller



Simple DC Motor Control from the MCU's Perspective



Motor Control Roadmap

Hi-end –DSP, 32-bit

8bit

MPC5604P

16bit

56F82xx

32bit

56F802x / 3x

56F801x

56F800x

Kinetis

51 AC

S08MP16

S08AC

S08SH

S08SF

S08QD4

Mid Range – 3-phase sensorless (vector control) 16-bit DSP

Low-end to 8-bit, Low-end 32-bit 3-phase sensed (V/Hz, Slip)

Ultra Low End – Low End 8Bit

MC33991

Dual Gauge Driver

MM908E626 Stepper Motor Driver w/ LIN

MPC17C724
0.4 Amp Dual
H-Bridge Motor

MC33879
Octal Serial Switch
with Open Load
Detect Current
Disable

MC33999

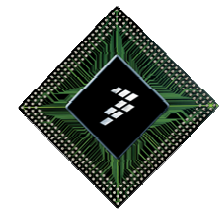
16-Output Switch with SPI and PWM Control

MC33937
Three Phase Field
Effect Transistor
Pre-driver

Analog Portfolio

MC9S08MP16

Enabling Safe, Accurate and Inexpensive BLDC Motor Control



<http://www.freescale.com/S08MP16>

MC9S08MP16/12

Core / Temp / Package

- ▶ Industrial Version: 50MHz (25MHz bus), -40to105C
- ▶ Automotive Version: 40MHz (20MHz bus), -40to125C
- ▶ 2.7V to 5.5V operating range

Memory

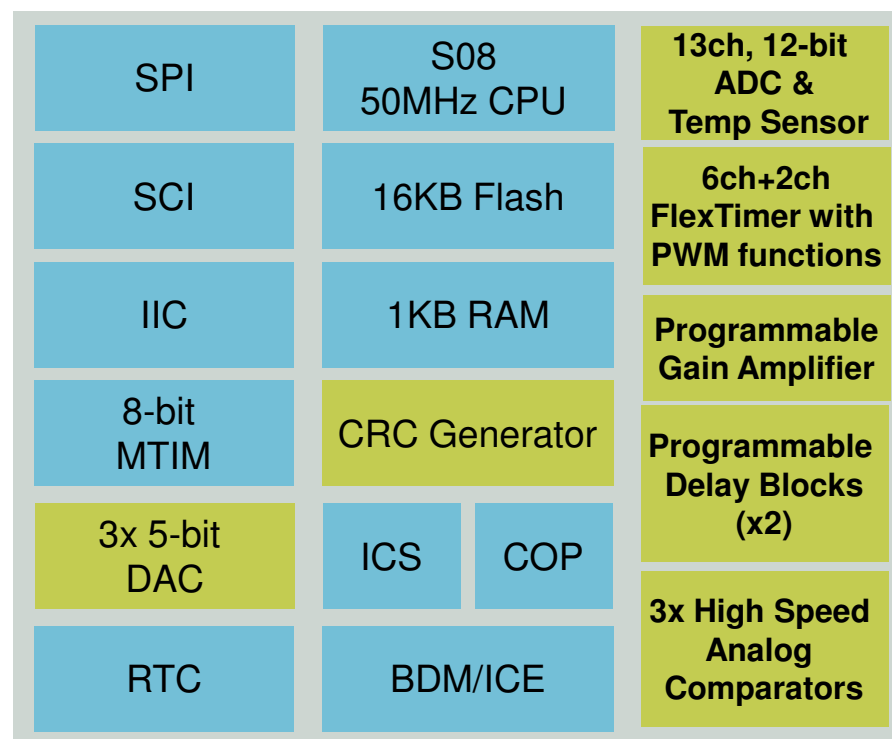
- ▶ 16KB Flash / 1KB RAM
- ▶ 12KB Flash / 512B RAM

Features

- ▶ **2x FlexTimers (6ch + 2ch) – automatic fault protection**
- ▶ **3 Analog Comparators – h/w sample trigger from PWM module allowing comparison at any point in cycle**
- ▶ 2x Programmable Delay Blocks (PDB)
- ▶ **12-ch 12-bit ADC – 3.5 uS conversion, h/w trigger from PWM module allowing conversion at any point in cycle**
- ▶ **Programmable Gain Amplifier (PGA)**
- ▶ 8-bit Modulo Timer Module (MTIM)
- ▶ LIN SCI, SPI, IIC
- ▶ 3x 5-bit DAC used as a 32 tap voltage reference
- ▶ RTC
- ▶ Software Programmable Internal Clock Source
- ▶ 3x low power modes & peripheral CLK gating
- ▶ Power Management Controller (PMC)
- ▶ KBI
- ▶ POR / LVI – supports 4 interrupt priority levels
- ▶ Background Debug Mode Interface/ICE

System Protection

- Cyclic Redundancy Check Generator (CRC)
- Watchdog Timer with Independent Clock Source



The S08MP16 8-bit MCU delivers safe, accurate, and inexpensive Brushless DC motor control for a wide range of Industrial and Automotive applications.

Safe Motor Control

Safe motor operation and increased system integrity for safety-critical applications.

- **Over-current protection:** Analog Comparators in conjunction with PWM FlexTimer fault inputs, provide fast and accurate over-current shutdown protection by driving PWMs to a safe state.

- **Enhanced system integrity for safety-critical applications implementing IEC60730:** Independently Clocked COP & Cyclic Redundancy Check Engine provide clock failure protection & memory content validation.

Rich Analog Integration

Rich on-chip peripheral integration minimizes BOM costs and delivers precise system control.

- **Reduced system cost:** integrated Programmable Gain Amplifier (PGA) and Analog Comparators minimise external component count. Low cost small footprint SOIC & LQFP packages with automotive-qualified high temperature option.

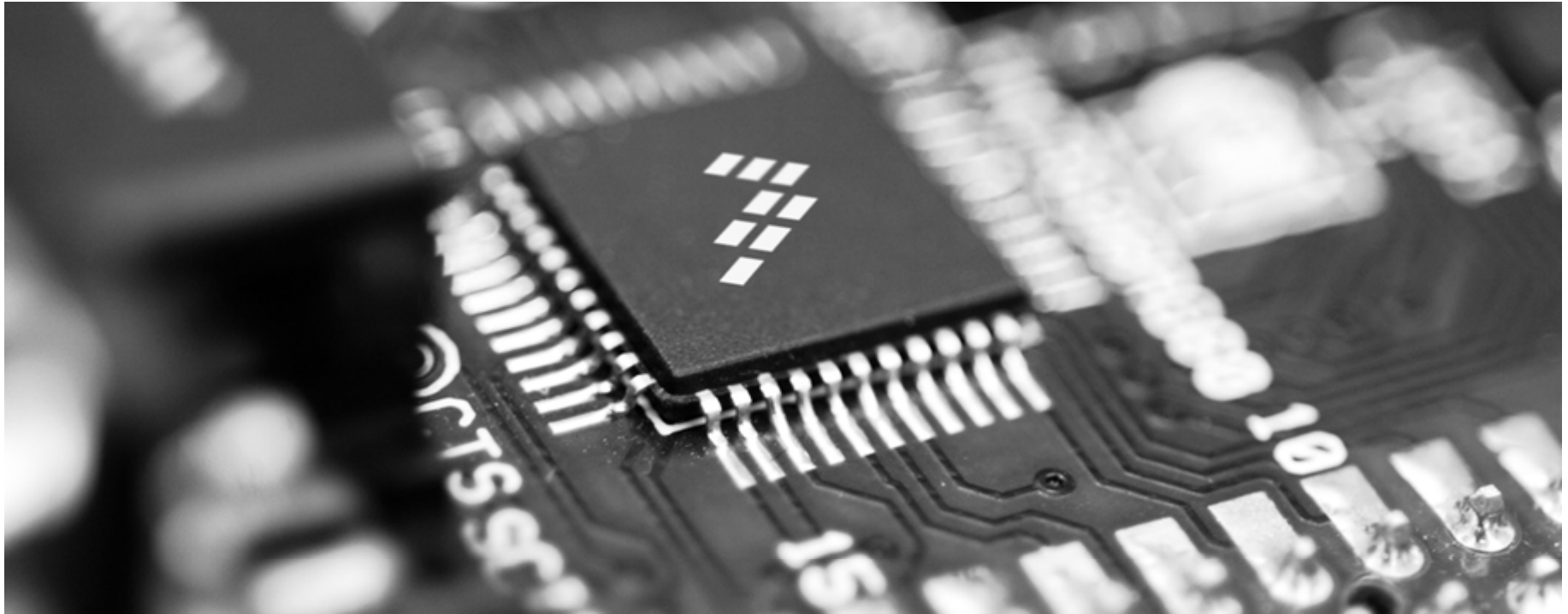
- **Precise motor control:** FlexTimer provides dead-time insertion in hardware – more accurate than software insertion and reduces CPU bandwidth usage. PGA allows high resolution ADC readings over a wide range of motor loads and speeds.

Broad Development Support

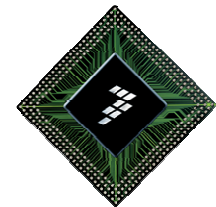
Extensive off-the-shelf development resources and complementary analog product solutions.

- **Dedicated application support:** Reference Designs, Application Notes, and Software Libraries covering all motor control topologies. Regional Motor Control Centers of Excellence provide application support from concept to delivery.

- **Reduced time-to-market:** Free CodeWarrior™ IDE up to 32KB with Processor Expert and Freemaster Real-Time Control and Monitoring Tool. One-stop-shop motor control system solutions including MC33937 3-ph FET Pre-Driver, and MC33880/87 & MC33926/32 H-bridges.



Freescale Digital Signal Controller- 56800E family



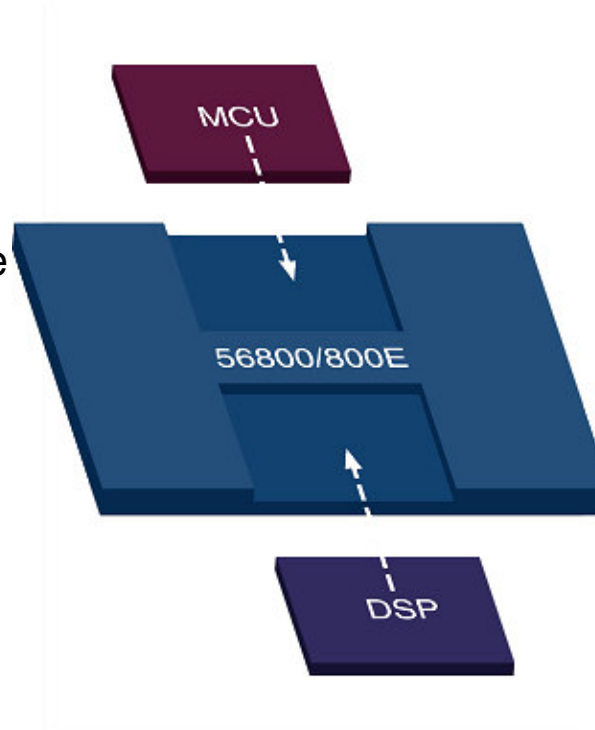
What is Digital Signal Controller

- Specialized microprocessor whose architecture contains a core engine capable of competitively performing both microcontroller and digital signal processor functionalities
- Core processing capability applicable to many types of system solutions
- Common basic features:
 - > MAC, single instruction cycle allowing several memory accesses, address generation units, algorithms for efficient looping
- Specialized Low cost, high performance on-board interfaces utilized in implementing embedded control applications :
 - > PWM; multifunction timer; high speed ADCs; DACs; Comparators; SCIs (UART); SPIs; CANs and I2Cs, etc.
- Embedded nonvolatile memory:
 - > Flash memory, ROM or EEPROM
- Easy to use development tools

56800/E Family Combining Signal Processing and Controller Functionality

Traditional Microcontroller

- Designed for Controller Code
- Compact Code Size
- Easy to Program
- **Inefficient Signal Processing**

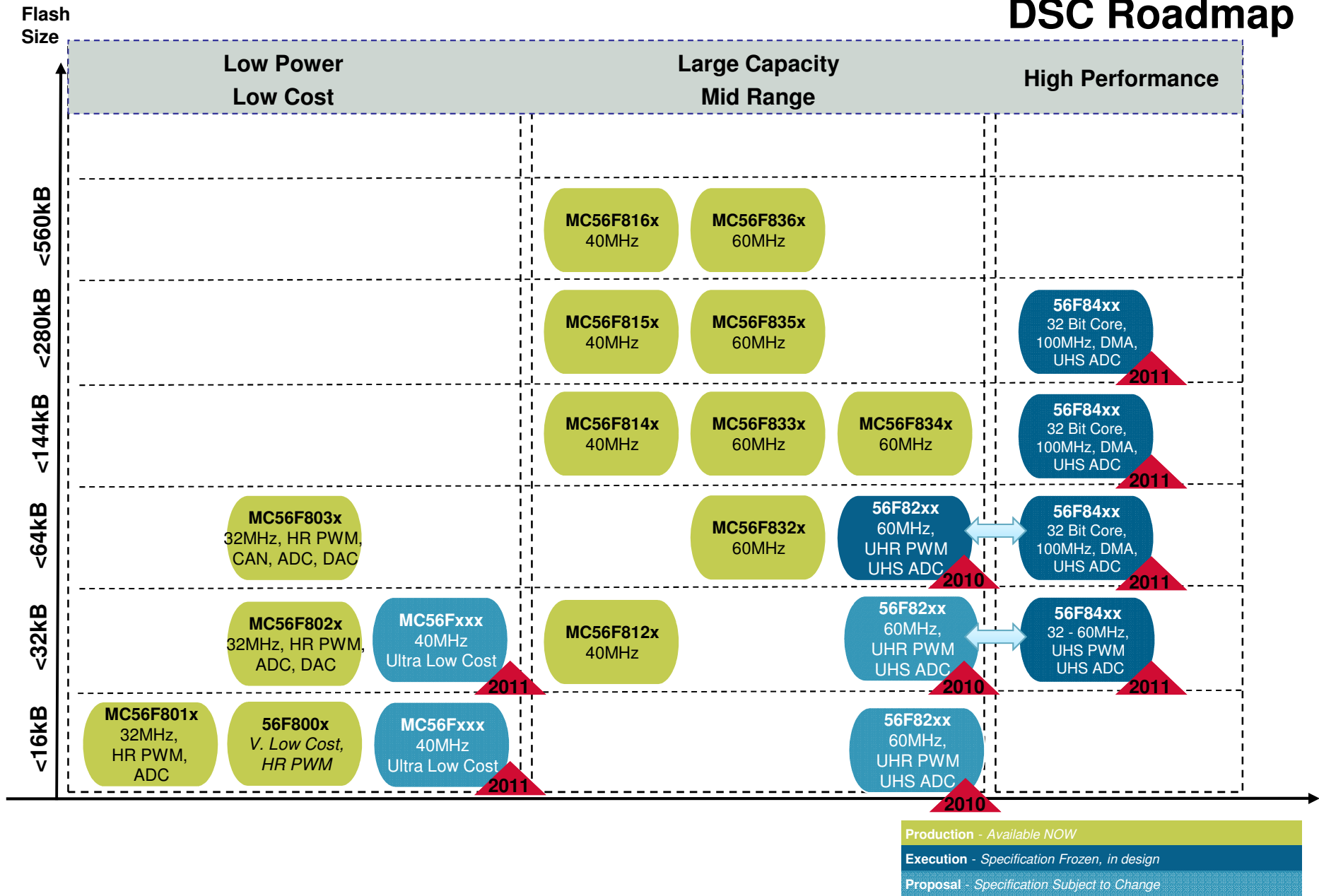


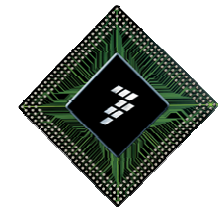
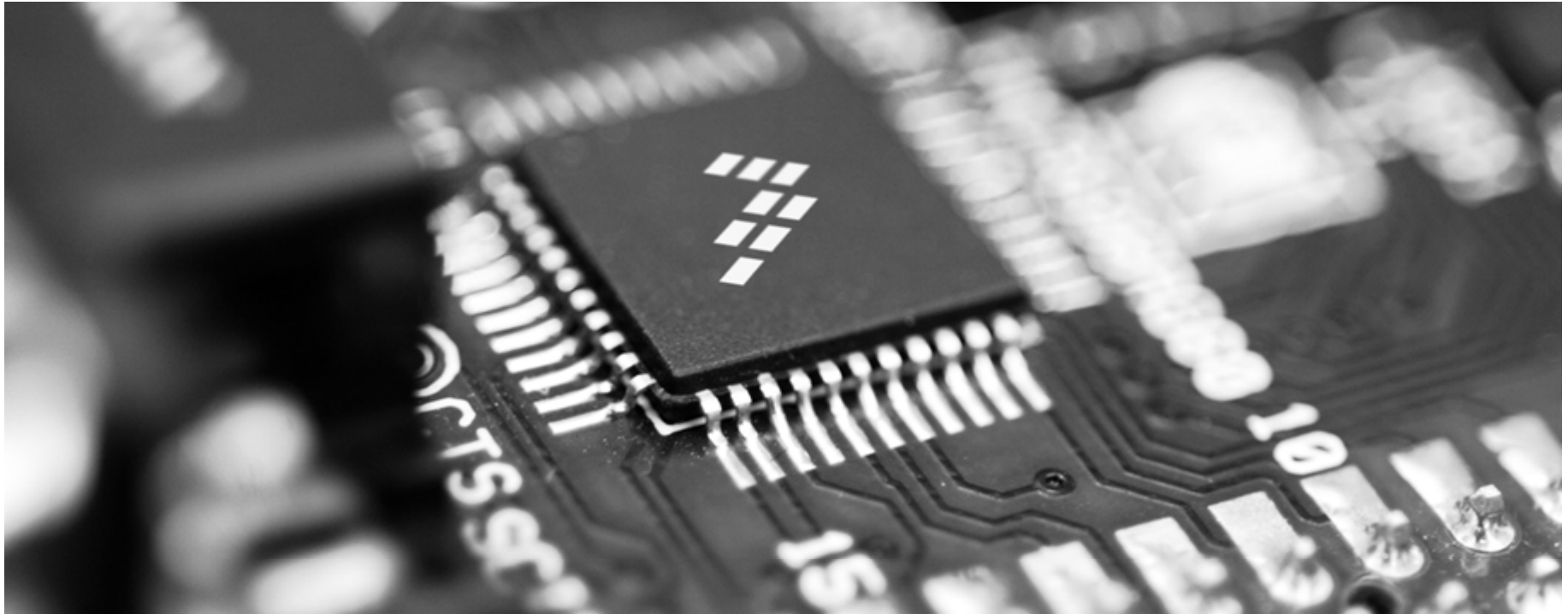
Traditional DSP Engine

- Designed for DSP Processing
- Designed for Matrix Operations
- **Complex Programming**
- **Less Suitable for Control**

- Instructions Optimized for Controller Code, DSP, Matrix Operations
- Compact Assembly and “C” Compiled Code Size
- Easy to Program
- Additional MIPS Headroom and extended addressing space

DSC Roadmap

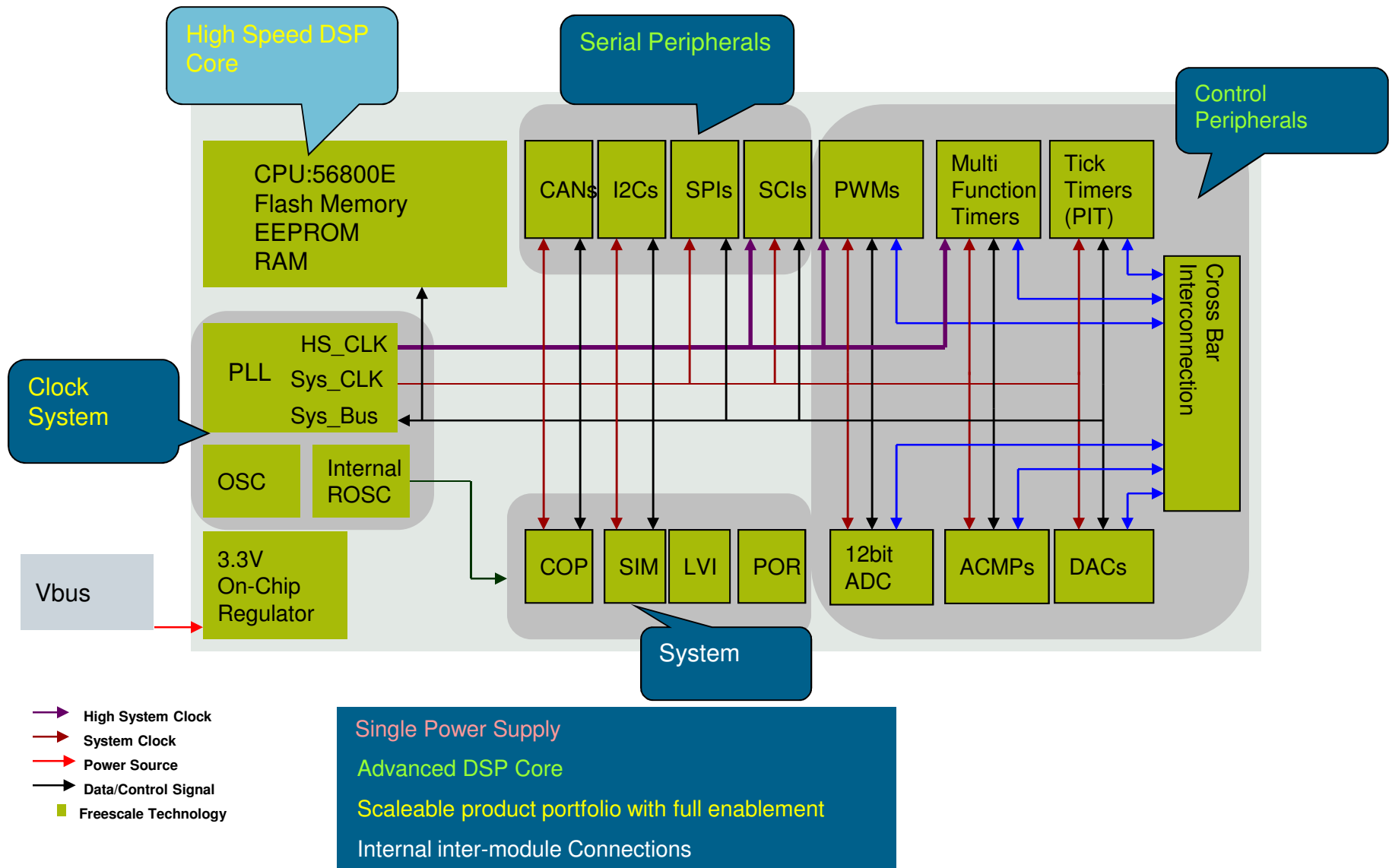




Freescal Digital Signal Controller- 56800E family

Freescal, the Freescal logo, Altivec, C-5, CodeTEST, CodeWarrior, ColdFire, C-Ware, mobileGT, PowerQUICC, StarCore, and Symphony are trademarks of Freescal Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. BeeKit, BeeStack, CoreNet, the Energy Efficient Solutions logo, Flexis, MXC, Platform in a Package, Processor Expert, QorIQ, QUICC Engine, SMARTMOS, TurboLink and VortiQa are trademarks of Freescal Semiconductor, Inc. All other product or service names are the property of their respective owners. © 2010 Freescal Semiconductor, Inc.

Freescal Single Chip Solution Strategy



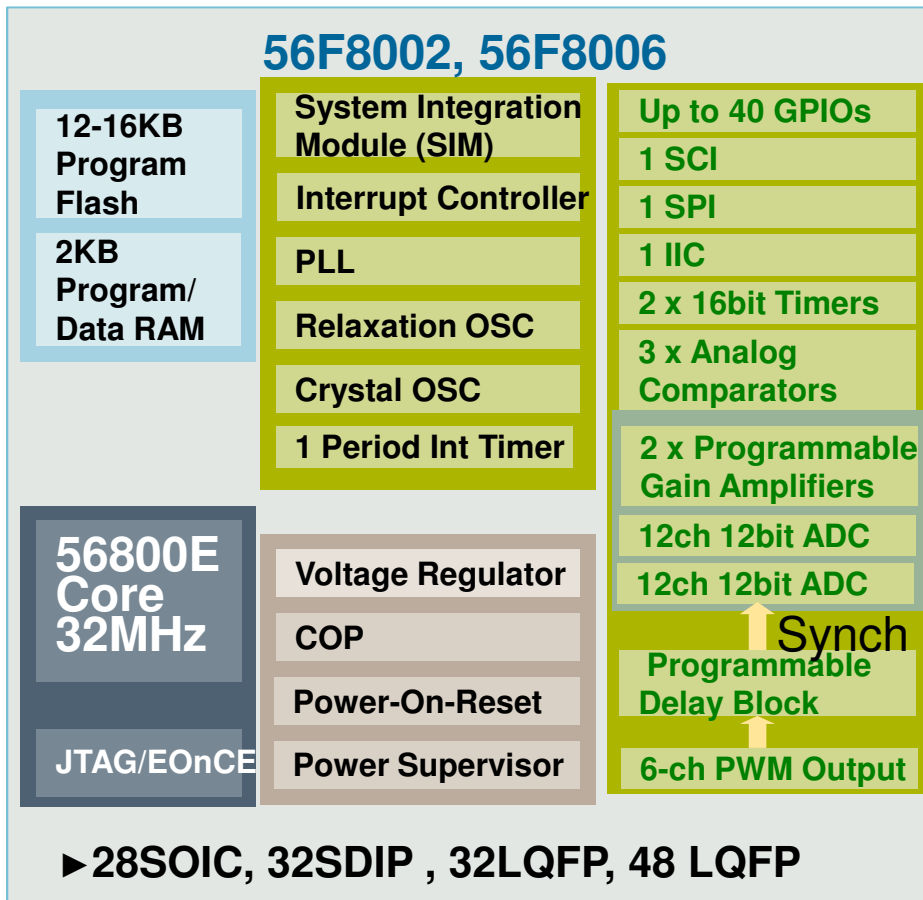
56F8000 Series Feature Summary

	56F8002	56F8006	56F8011	56F8013	56F8014	56F8023/33	56F8025/35	56F8036	56F8027/37
Performance	32MHz/MIPs	32MHz/MIPs	32MHz/MIPs	32MHz/MIPs	32MHz/MIPs	32MHz/MIPs	32MHz/MIPs	32MHz/MIPs	32MHz/MIPs
Temperature Range (V)	-40C~105C	-40C~105C	-40C~105C	-40C~125C	-40C~125C	-40C~125C	-40C~125C	-40C~125C	-40C~125C
Voltage Range	1.8V - 3.6V	1.8V - 3.6V	3.0V - 3.6V	3.0V - 3.6V	3.0V - 3.6V	3.0V - 3.6V	3.0V - 3.6V	3.0V - 3.6V	3.0V - 3.6V
Voltage Regulator	On-Chip	On-Chip	On-Chip	On-Chip	On-Chip	On-Chip	On-Chip	On-Chip	On-Chip
Program/Data Flash	12KB	16KB	12KB	16KB	16KB	32KB/64KB	32KB/64KB	64KB	32KB / 64KB
Program/Data RAM	2KB	2KB	2KB	4KB	4KB	4KB/8KB	4KB/8KB	8KB	4KB /8KB
Program Security	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
OnChip Relaxation Osc	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
PLL	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
COP (Watchdog)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
PWM (96Mhz Clock)	1 x 6ch	1 x 6ch	1 x 6ch	1 x 6ch	1 x 5ch	1 x 6ch	1 x 6ch	1 x 6ch	1 x 6ch
PWM Fault Inputs	4	4	4	4	3	4	4	4	4
12-bit ADCs	2 x 8ch	2 x 12ch	2 x 3ch	2 x 3ch	2 x 4ch	2 x 3ch	2 x 4ch	2 x 5ch	2 x 8ch
12-bit DACs	0	0	No	No	No	2	2	2	2 (Pinned out)
Analog Comparator	3	3	No	No	No	2	2	2	2
Prog Gain Amp	2	2	No	No	No	No	No	No	No
16-bit Timers	3	3	4	4	4	4	4	4	8
Prog. Interval Timers	1 (RTC)	1 (RTC)	No	No	No	1	3	3	3
GPIO (max) (+/-8mA)	23	40	26*	26*	26*	26*	35*	39*	53*
IIC	1	1	1	1	1	1 - QIIC	1 - QIIC	1 - QIIC	1 - QIIC
SCI (UART) / LIN Slave	1 - SCI	1 - SCI	1 - SCI	1 - SCI	1 - SCI	1 - QSCI	1 - QSCI	1 - QSCI	2 - QSCI
SPI (Synchronous)	1 - SPI	1 - SPI	1 - SPI	1 - SPI	1 - SPI	1 - QSPI	1 - QSPI	1 - QSPI	2 - QSPI
CAN	No	No	No	No	No	No	No	MSCAN	MSCAN
JTAG/EOnCE	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Power Consumption	IDD = 45.6mA; IDDA = 4.5mA		IDD = 42mA; IDDA = 13.5mA			IDD = 48mA; IDDA = 18.8mA		IDD = 48mA; IDDA = 18.8mA	
Package	32LQFP (.8p)	28SOIC 32LQFP 32SDIP 48LQFP	32LQFP	32LQFP	32LQFP	32LQFP	44LQFP	48LQFP	64LQFP

* 5V tolerance I/O

Freescale, the Freescale logo, Altivec, C-5, CodeTEST, CodeWarrior, ColdFire, C-Ware, mobileGT, PowerQUICC, StarCore, and Symphony are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. BeeKit, BeeStack, CoreNet, the Energy Efficient Solutions logo, Flexis, MXC, Platform in a Package, Processor Expert, QorIQ, QIICC Engine, SMARTMOS, TurboLink and VortiQa are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. © 2010 Freescale Semiconductor, Inc.

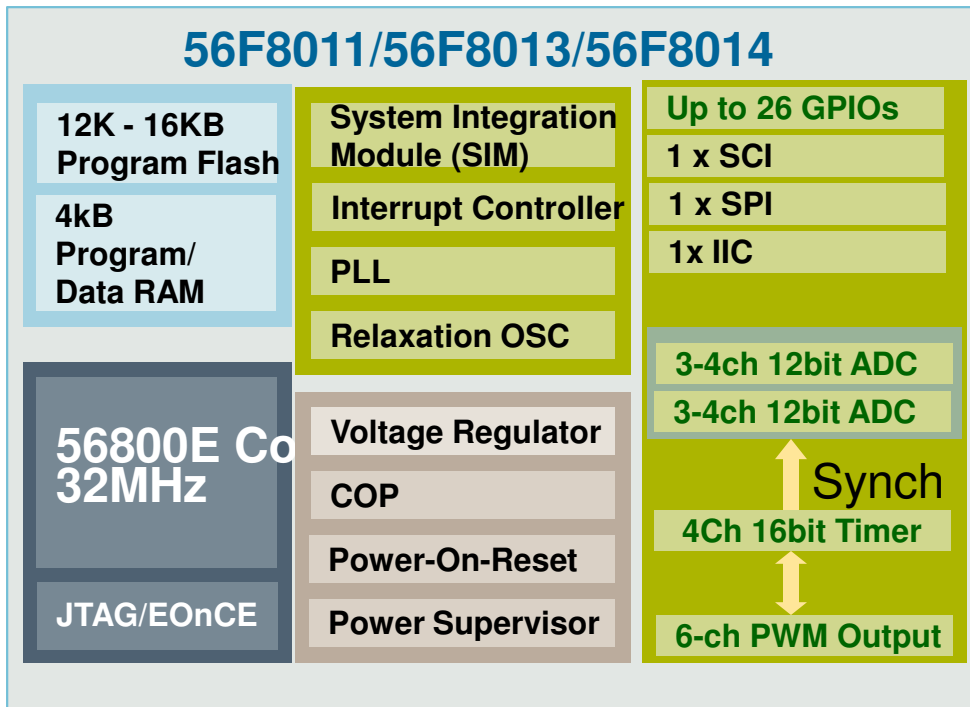
Ultra Low cost 56F8000 member: 56F8002, 56F8006



In Production
Less than \$2.00 in Volume

- 32 MHz/32 MIPS 56800E Core
- 1.8-3.6V Operation
- 12K - 16K Bytes Program FLASH with Flash security
- 2K Bytes Program/Data RAM
- Tunable Internal Relaxation Oscillator and 32KHz clock
- Phase Locked Loop (PLL)
- Up to 96 MHz Peripherals – Timers, PWM & Hi-SCI**
- 6 Output PWM Module with 4 Programmable Fault Inputs**
 - Programmable Dead timer insertion
 - Programmable PWM generation for Power supply apps
 - Multiple PWM Frequency outputs
- Two Programmable Gain Amplifiers with x2, x4, x8, x16 gains (Clocked in order to cancel input offset)**
- Two 12-bit ADCs with up to 24 Inputs, 2.5us Per conversion
- Programmable Delay Block provides precise control of ADC/PGA sample times relative to PWM reload cycles**
- Three High Speed Analog Comparators
- 2 multiple function Programmable Timers
- Computer Operating Properly Timer
- One Periodic Interval Timer (PIT)
- 1 High Speed Serial Communication Interface (Hi-SCI)
- 1 Serial Peripheral Interface (SPI)
- I²C Communications Interface
- Up to 40 GPIOs – Versatile pin usage
- JTAG/EOnCE™ Debug Port
- Lead Free “Green” Packages
- Industrial temp: -40C – 105C

Cost Effective 56F8000 Solutions 56F8011/56F8013/56F8014

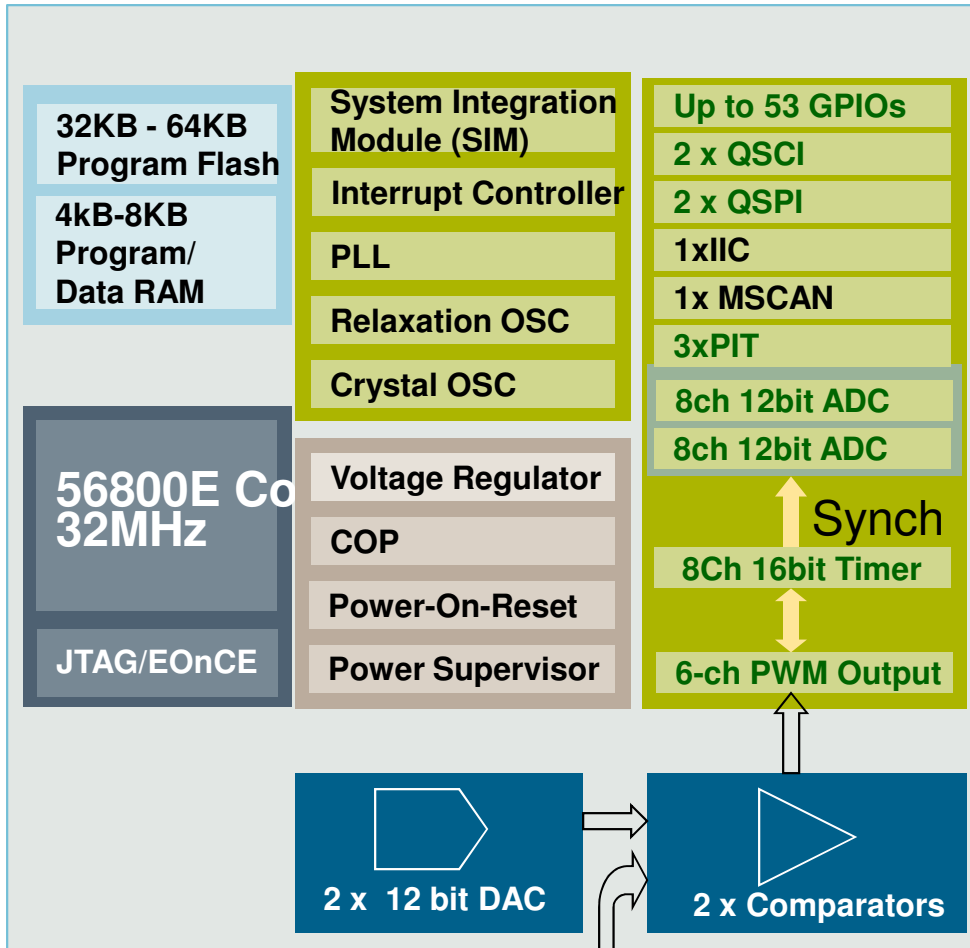


- **Package: 32LQFP**
- **In Production**

- 32 MIPS Performance
- 12K - 16 K Bytes Program FLASH
- 4 K Bytes Program/Data RAM
- Tunable Internal Relaxation Oscillator
- Software Programmable Phase Locked Loop
- **Up to 96 MHz Peripherals – Timers and PWMs**
- Up to 6-Output PWM Module with up to 4 Programmable Fault Inputs
- **Selectable PWM frequency for each complementary PWM signal pair**
- **Two 12-bit ADCs with up to 8 Inputs , 1.125us conversion rate**
- Synchronization between PWM and ADC
- Four 16-bit General Purpose Programmable Timers
- Computer Operating Properly Timer
- Serial Ports: SCI, SPI, I2C
- **Up to 26 GPIOs – Versatile pin usage**
- **Low Power Consumption – 59mA Max and .026mA Min**
- JTAG/EOnCE™ Debug Port
- Industrial & Automotive temp
- **Cost Effective**

56F8000 Family Expansion

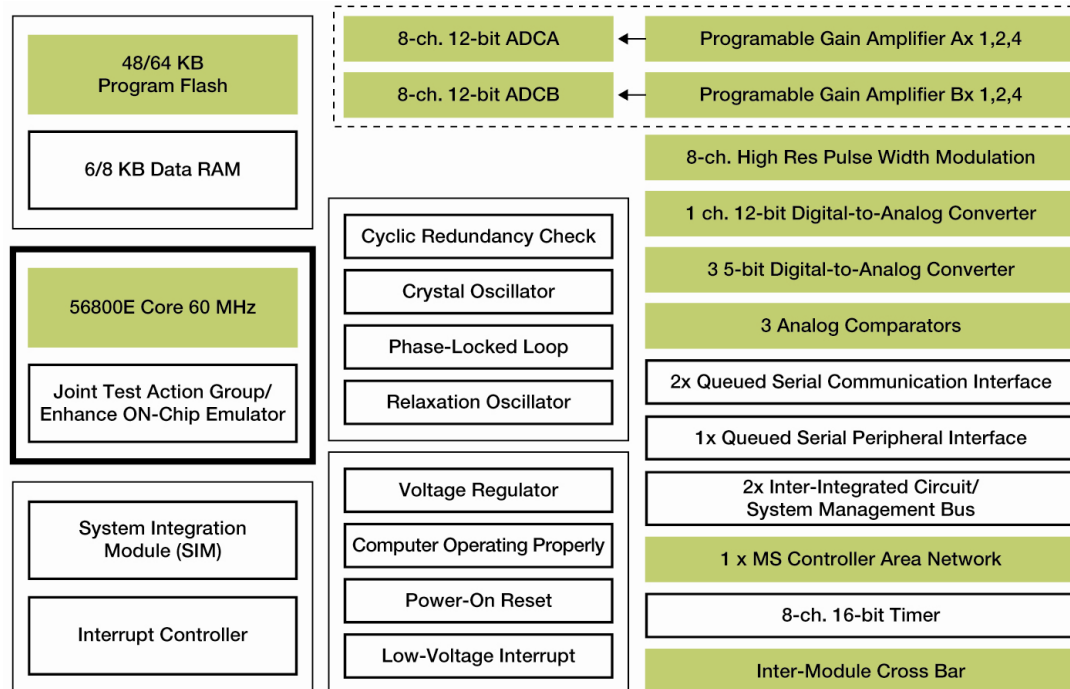
56F8023/56F8025/56F8036/56F8037 Features



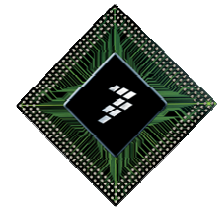
► Package 32 LQFP, 44LQFP, 48LQFP, 64LQFP

- 32 MHz/32 MIPS 56800E Core
- 3.0-3.6V Operation
- 32K-64K Bytes Program FLASH
- 4K-8K Bytes Program/Data RAM
- Flash security
- Tunable Internal Relaxation Oscillator
- Software Programmable Phase Locked Loop
- **Up to 96 MHz Peripherals – Timers and PWMs**
- 6 Output PWM Module with 4 Programmable Fault Inputs
- **Selectable PWM frequency for each complementary PWM signal pair**
- **Two 12-bit ADCs with up to 16 Inputs , 1.125us conversion rate**
- **Up to Two 12-bit Digital to Analog Converters**
- **Two Analog Comparators**
- Synchronization between PWM and ADC
- 4 or 8 16-bit General Purpose Programmable Timers
- **1 or 3 Programmable Interval Timers (PIT)**
- Computer Operating Properly Timer
- **2-Queued Serial Communications Interface**
- **2-Queued Serial Peripheral Interface**
- **Optional MSCAN**
- I²C Communications Interface
- **Up to 53 GPIOs – Versatile pin usage**
- JTAG/EOnCE™ Debug Port
- Industrial & Automotive temp

Anguilla Black 56F824x / 56F825x Features

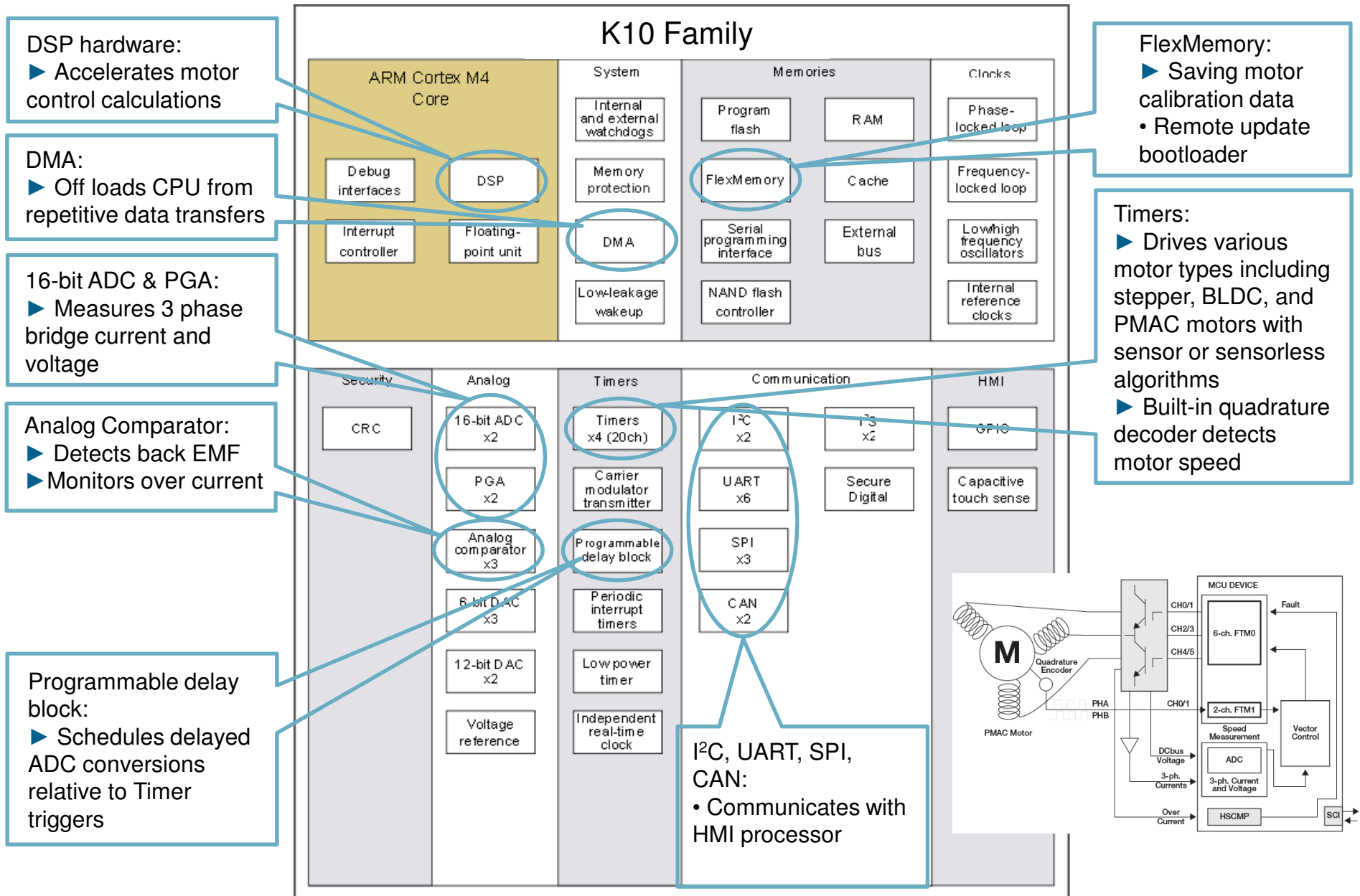


- 60 MHz/60MIPS From 56800E Core
- Up to 60 MHz Peripherals
 - 8 Output PWM Module
 - 520ps PWM duty cycle resolution
 - 2 x12-bit ADCs with total of 16 Inputs
 - 500ns conversion rate
 - Built-in PGA - 1x, 2x, 4x, gains
- 1 x 12-bit Digital to Analog Converter
- 3 x 5bit Digital to Analog Converters
- 3 Analog Comparators
- 8 x16-bit Enhanced Multifunction Programmable QTimers
- Cyclic Redundancy Check Generator (CRC)
- 5v tolerant up to 54 GPIOs
- Inter Module Cross-Bar



Kinetis

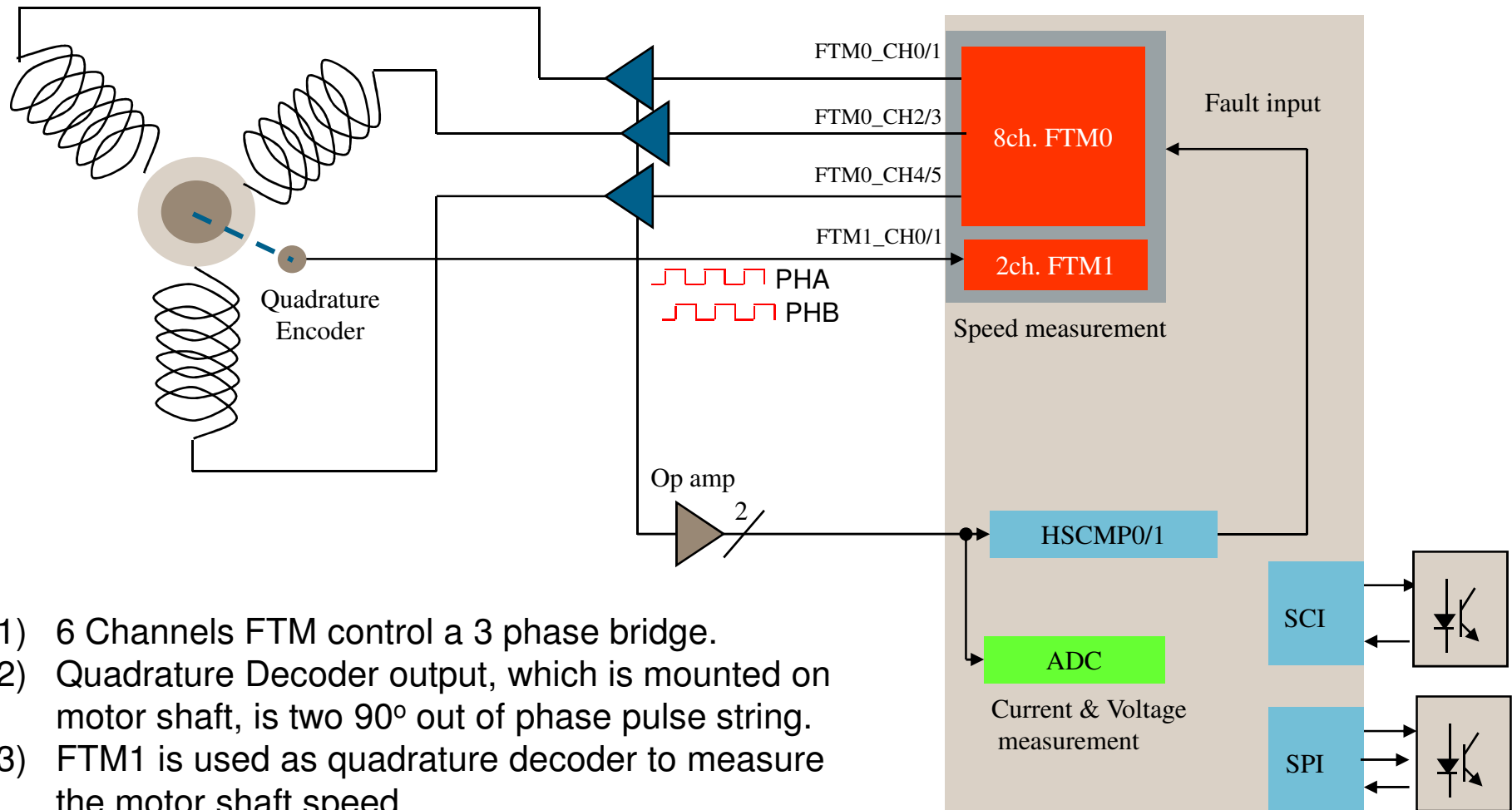
Motor Control Use Case

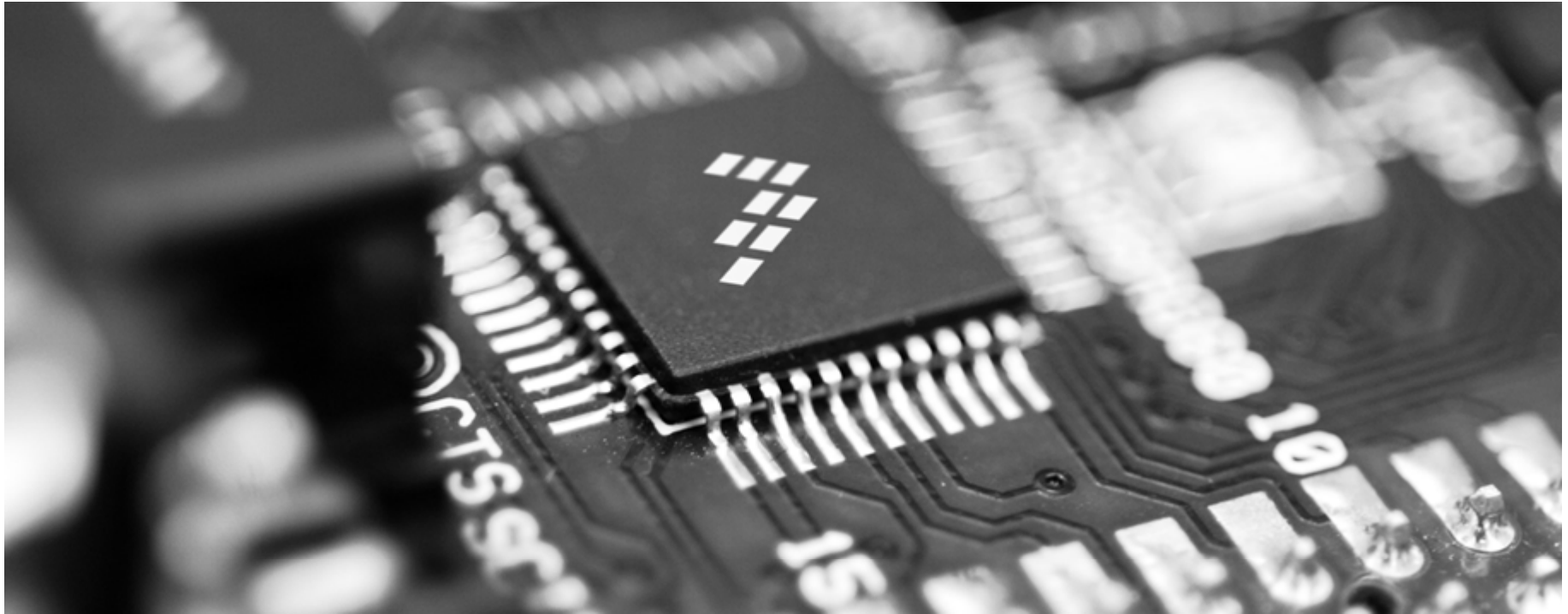


FlexTimer (FTM) Features

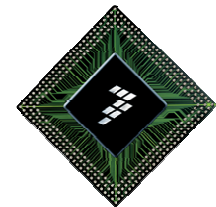
- ▶ 16-bit counter with prescaler divide-by 1,2,4,8,16,32,64, or 128
- ▶ Each channel can be configured for input capture, output compare, edge-aligned PWM mode, or center-aligned PWM mode
- ▶ Each pair of channels can be combined to generate a PWM signal with independent control of both edges of PWM signal
- ▶ Each pair can operate as complementary outputs with deadtime insertion
- ▶ Dual edge capture for pulse and period width measurement
- ▶ Quadrature decoder with input filters for relative position counting
- ▶ Global Time Base mode shares single time base across multiple FTM instances

ACIM/PMSM motor control

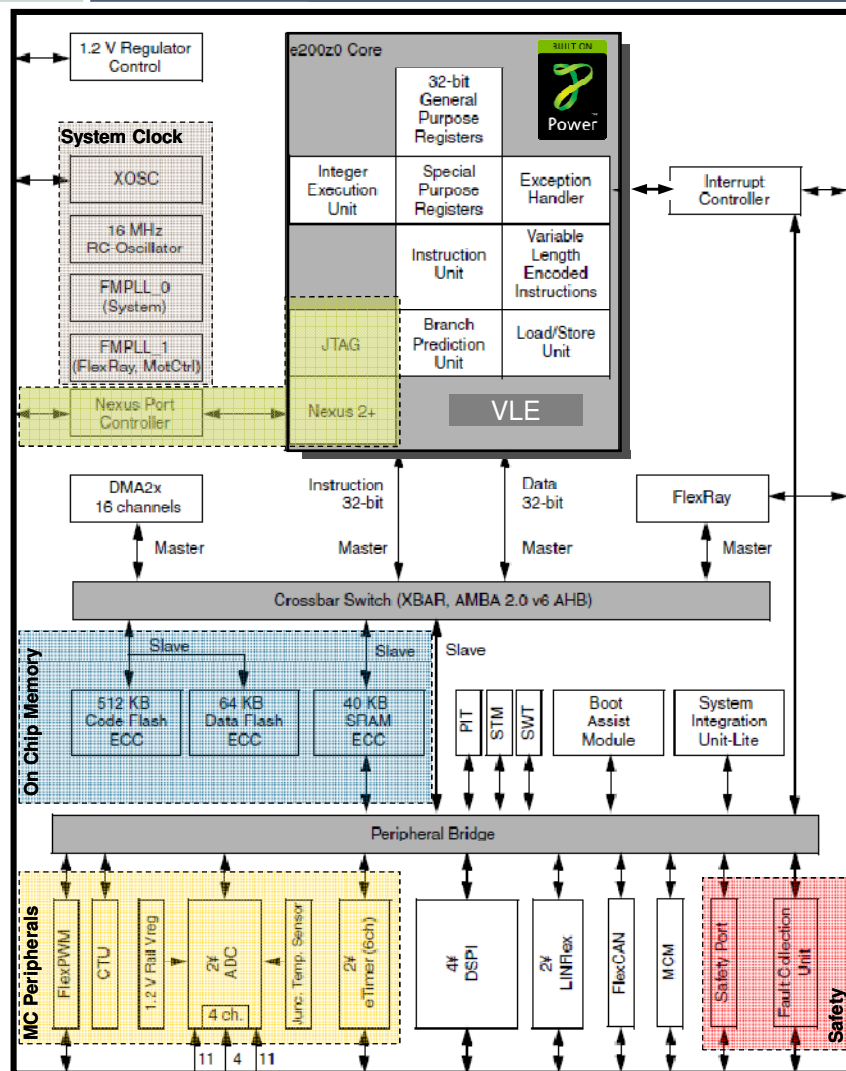




Power Architecture for Motor Control: MPC5604P



MPC5604P



CAN - Controller Area Network (FlexCAN)
 DSPI - Deserial Serial Peripheral Interface
 LINFlex - Serial Communication Interface (LIN support)
 FMPLL - Frequency-Modulated Phase-Locked Loop
 SRAM - Static Random-Access Memory

FlexPWM - Flexible Pulse Width Modulation
 eTimer - Enhanced Timer
 PIT - Periodic Interrupt Timer
 SWT - Software Watchdog Timer
 STM - System Timer Module

Power Architecture® Core

- Up to 64 MHz e200 zen0h core, 32-bit Power Architecture Book E CPU with Harvard architecture
- VLE instruction set encoding for code size footprint reduction

On-Chip Memory Options

Device	MPC5602P	MPC5603P	MPC5604P
Program Flash with ECC	256K	384K	512K
DataFlash® with ECC	64K	64K	64K
SRAM with ECC	24K	32K	40K
FlexRay	No	No	Yes
Interrupt Controller	100ch	100ch	144ch

I/O Peripherals

- 1 x High speed FlexCAN with 32 Message Buffers (MB)
- 1 x Safety port (can be used as additional FlexCAN – 32 MB)
- 1 x FlexRay Controller - Dual Channel with 32 MB
- 2 x LinFlex
- 4 x DSPI (4 independent chip selects each)
- 1 x FlexPWM (4 channels with 4 fault inputs)
- 2 x eTimer (6 channels incl. quad decode)
- 2 x ADC - 2 x 12 ch.(4 shared channels)
 - 10-bit, conversion time 700 nsec (2x 6 ch., 4shared on 100-pin package)
- 1 x CTU triggering unit: 32 input channels, 8 events, 24 ADC cmds.
- 1 x Fault collection unit

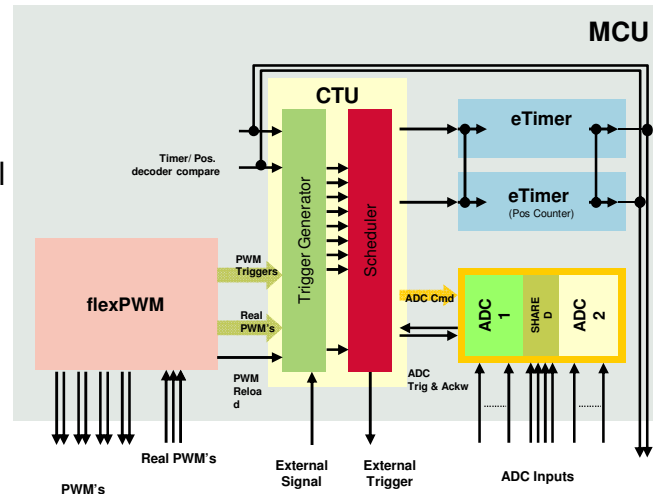
System

- 2 x PLL (one FM-PLL, one for FlexRay™)
- Crossbar switch architecture for concurrent access to peripherals
- 16-ch. eDMA
- 16 MHz internal RC OSC
- Junction temperature sensor
- Non-Maskable Interrupt
- Programmable Watchdog

Electric Motor Control Peripherals

Cross Triggering Unit

- Allows mcTIM, PWM, ATD to be synchronized
- Automatic ADC & eTimer acquisitions
- No CPU intervention during the control cycle



Timer Module:

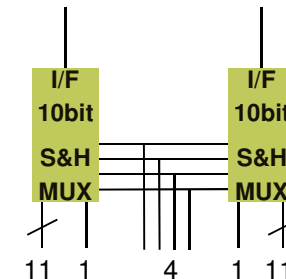
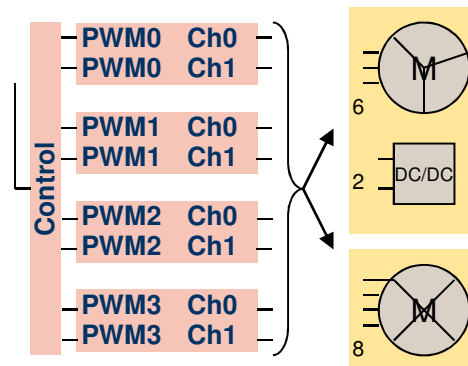
- DSC based
- Six Ch IC/OC
- Double buffered registers for detecting two edges in a row
- eDMA supported
- Integrated quad decoder support
- 2 x BUS frequency → high resolution

2x ADC

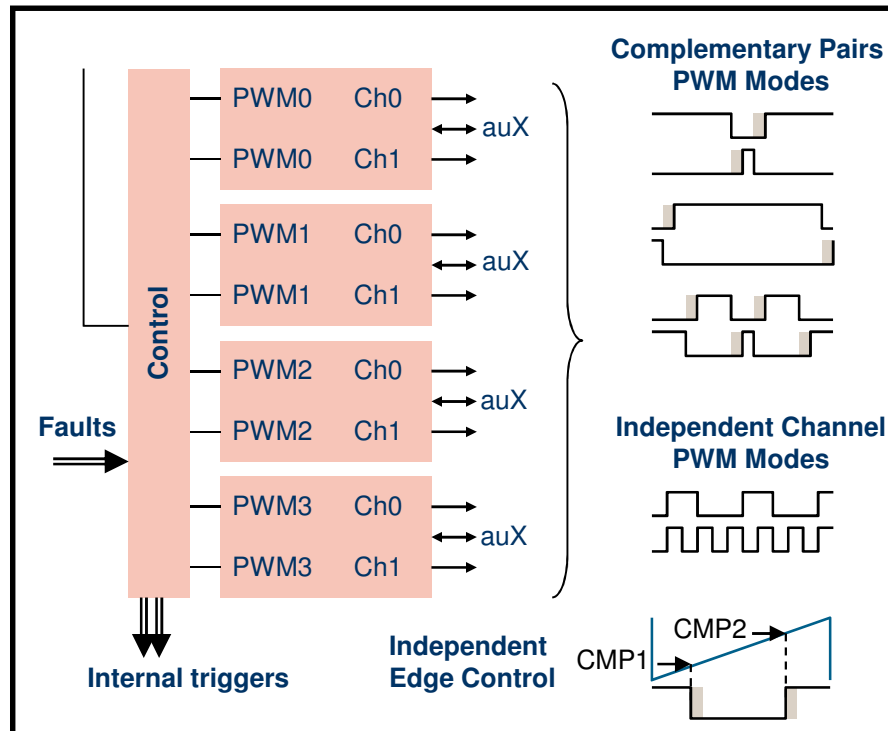
- Up to 24 independent and 4 shared channels
- 10-bit
- 700 nsec conversion time
- Limit checking & zero crossing detect

FlexPWM

- Based on DSC PWM
- Optimized for 3ph motor control
- One „extra“ pair of PWM integrated
- Includes dead time insertion, fault channels, center/edge alignment, Distortion correction, ...
- Register protections
- Double buffered registers
- eDMA supported
- 2 x BUS frequency → high resolution



Motor Control PWM Peripheral Module



- Permanent magnet synchronous motor (PMSM, PMAC)
- Brushless DC motor (BLDC)
- Brush DC motor (BDC)
- AC induction motor (ACIM)
- Switched reluctance motor (SRM)
- Variable reluctance motor (VRM)
- Stepper motors
- DC/DC converters

Main Features

- ▶ 4 Sub-modules, each with complementary PWM generation, Isense IC/OC and fault input
- ▶ 16 bits of resolution for center, edge aligned, and asymmetrical PWMs
- ▶ PWM outputs can operate as complimentary pairs or independent channels
- ▶ Independent control of both edges of each PWM output
- ▶ Independently programmable PWM output polarity
- ▶ Separate dead time for rising and falling edges
- ▶ Each complementary pair can operate with its own PWM frequency and deadtime values
- ▶ All outputs can be programmed to change simultaneously via a "Force Out" event
- ▶ Double buffered PWM registers
 - Integral reload rates from 1 to 16
 - Half cycle reload capability

Safety Features

- ▶ Write protection for critical registers
- ▶ Fault inputs can be assigned to control multiple PWM outputs
- ▶ Programmable filters for fault inputs

Summary

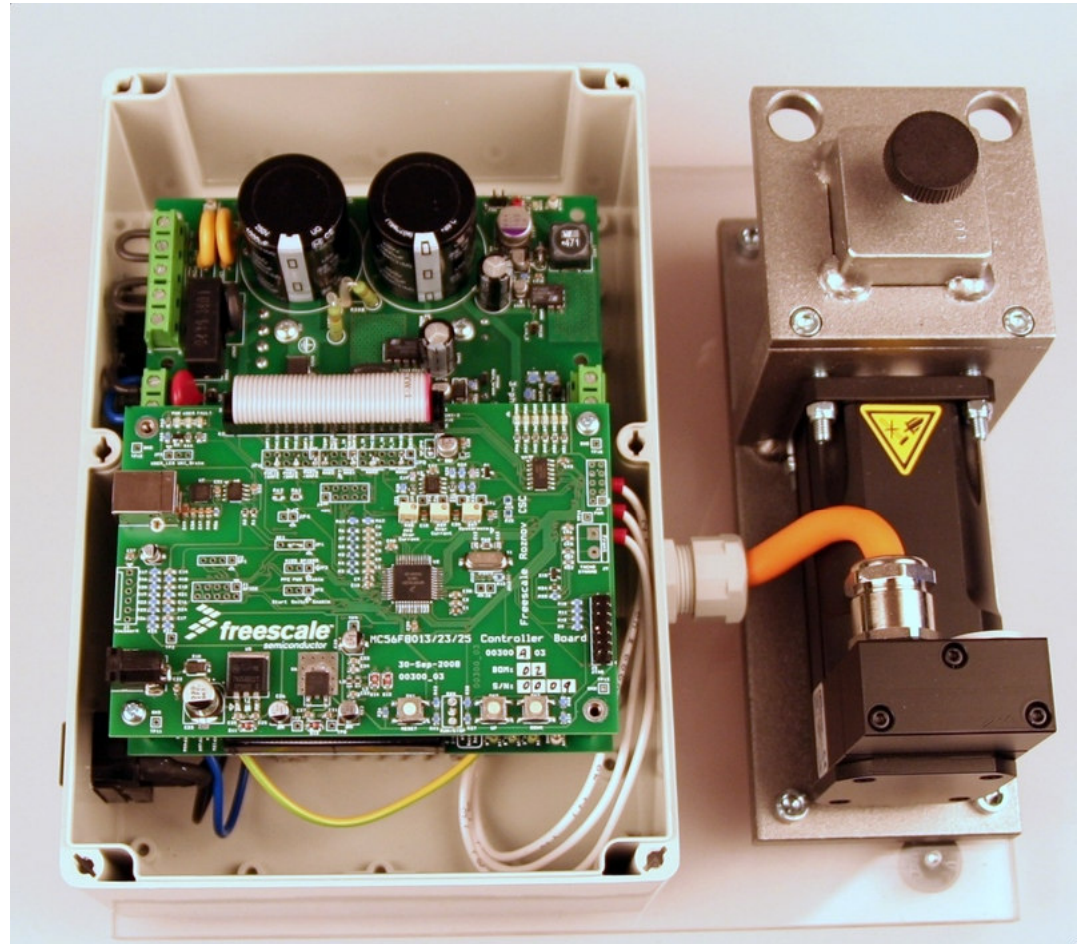
- ▶ Cutting Edge. Cost Effective. Complete.
Freescale offers technology for every motor control application
- ▶ Energy efficient motor control
Vector and sensorless control technology in motor control processors
 - 8- and 16-bit
 - 16-bit DSCs for ACIM and PMSM solutions
 - 32-bit Power Architecture® MCUs for standard and premium drives
- ▶ Strong technical support
Rich tools, training, reference designs, libraries
Devices are ruggedized with long life and reliability
Industrial products ship 10+ years with high quality and expert support
- ▶ Cost-effective safety and security on-chip
Protect against IP cloning, network data hacking and soft errors

Freescal:

- Motor Control Homepage – www.freescale.com/motorcontrol
- 8-bit Microcontrollers – www.freescale.com/8bit
- 16-bit DSC – www.freescale.com/dsc
- 32-bit Power Architecture® Processors – www.freescale.com/powerpc
- Analog Products – www.freescale.com/analog
- Industrial Segment – www.freescale.com/industrial

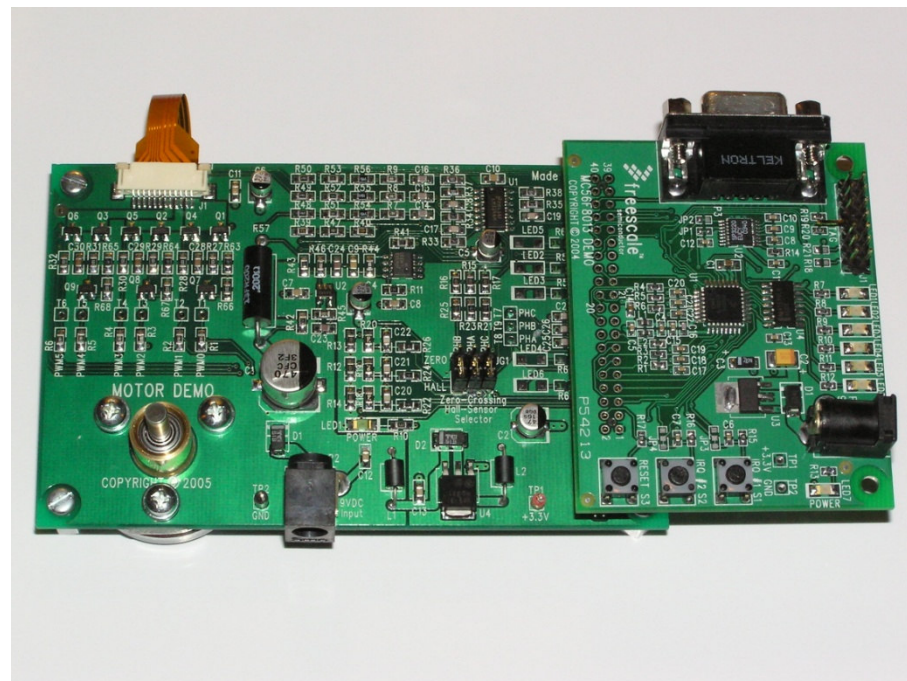
Sensorless PMSM Motor Control Using MC56F80xx

- ▶ MC56F80xx digital signal controller
- ▶ 3-phase AC/BLDC High Voltage Power Stage Board
- ▶ 1-phase line input 110/230VAC @ 50/60Hz
- ▶ Appliance PM motor
- ▶ Initial rotor position detection
- ▶ Full torque at motor start-up
- ▶ Field weakening
- ▶ Application based on C-callable library functions (GFLIB, GDFLIB, MCLIB, ACLIB)
- ▶ Current control loop execution time: 55us
- ▶ Speed control loop with Field weakening execution time : 17us
- ▶ Flash: ~ 6KB, RAM ~ 1.5KB



Low Cost BLDC Motor Control Demo Board

- ▶ Brushless motor, Maxon EC-200187, 6W 9V
- ▶ Motor interface connector
- ▶ Input power connector
- ▶ Daughter card connector for connecting the 56F8013 demonstration board
- ▶ LED power indicator
- ▶ Motor bus voltage sense logic
- ▶ Motor bus current sense logic
- ▶ Back EMF phase voltage sense logic
- ▶ Zero-crossing logic
- ▶ Hall-effect/zero-crossing selector
- ▶ 3-phase H-bridge power stage
- ▶ Power regulation logic

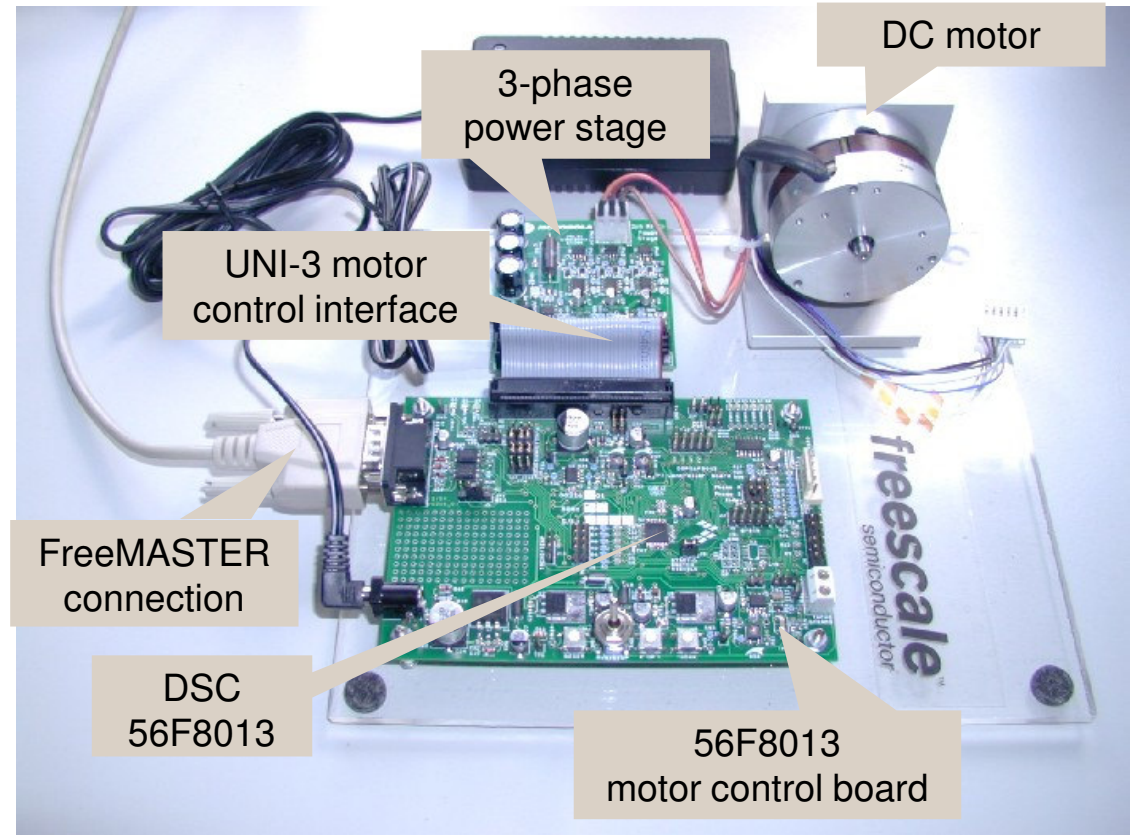


Order Number: APMOTOR56F8000

- (Optional) **Free on-board real-time user**
- FreeScale, the FreeScale logo, N/AIvec, C-S, Code-REST, CodeWarrior, ColdFire, CoreWare, mobileE-Power, QCC, S-Reg, U.S. Pat. & Tm. Off., Redlib, RedStack, RedNet, the Energy Efficient Solutions logo, Flexis, MXG, Platform in a Box and Vortex are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

BLDC Control Using MC56F8013

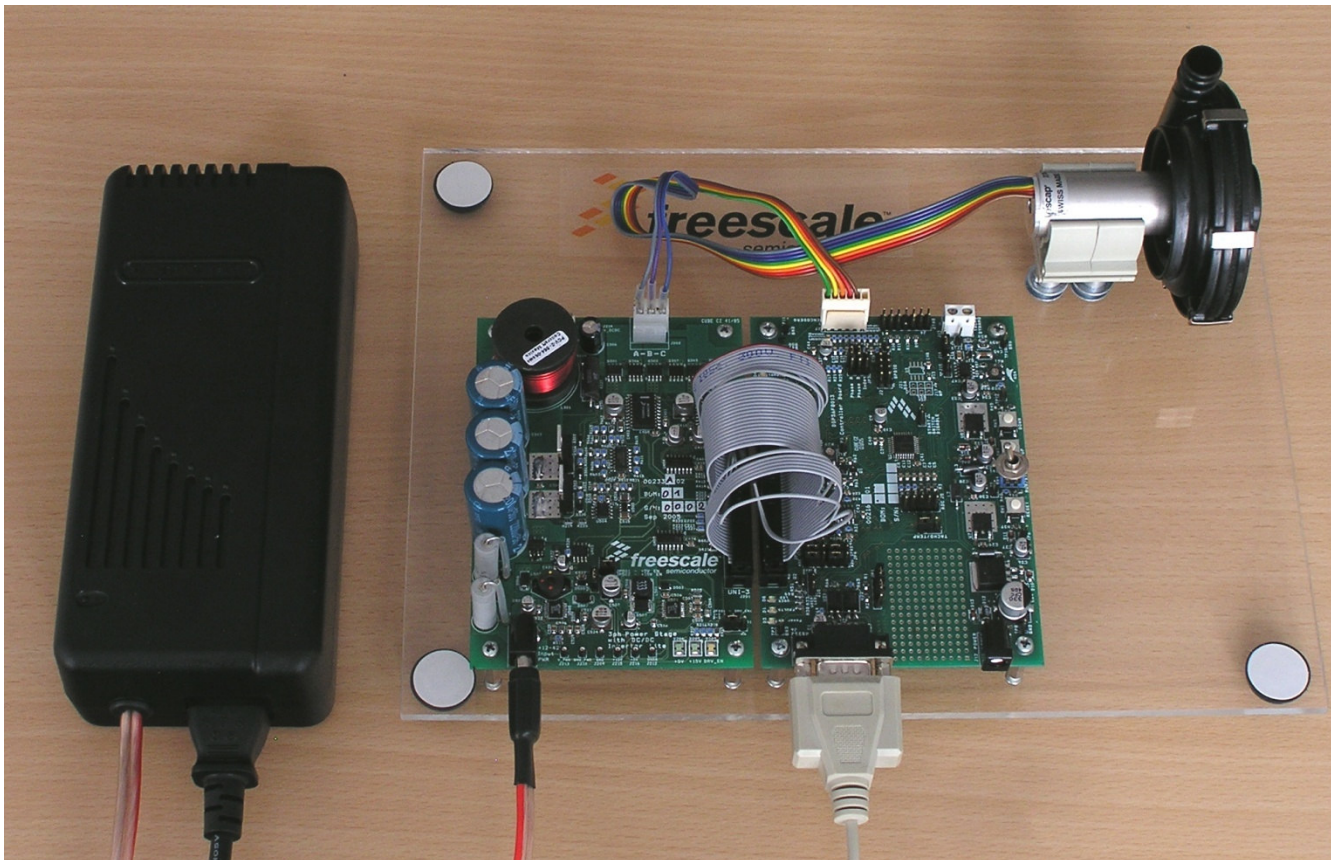
- ▶ Three-phase brushless DC motor sensorless drive
- ▶ Designed to fit into fan, pump and compressor applications
- ▶ Using MC56F8013 32 MIPS hybrid controller
- ▶ Available for two power stages and two motors
- ▶ Input power supply voltage +12 Vdc for power stages
- ▶ Control technique incorporates:
 - Sensorless, trapezoidal control of 3-phase brushless DC motor with back-EMF sensing
 - Using A / D converter zero-cross sensing for sensorless control
 - Speed and current closed loop with PI controller
- ▶ Speed range: 200 – 2000 and 500 – 5000 RPM (depending on the motor used)
- ▶ Manual interface (run/stop switch, up/down pushbuttons)



DRM070

High Speed Application

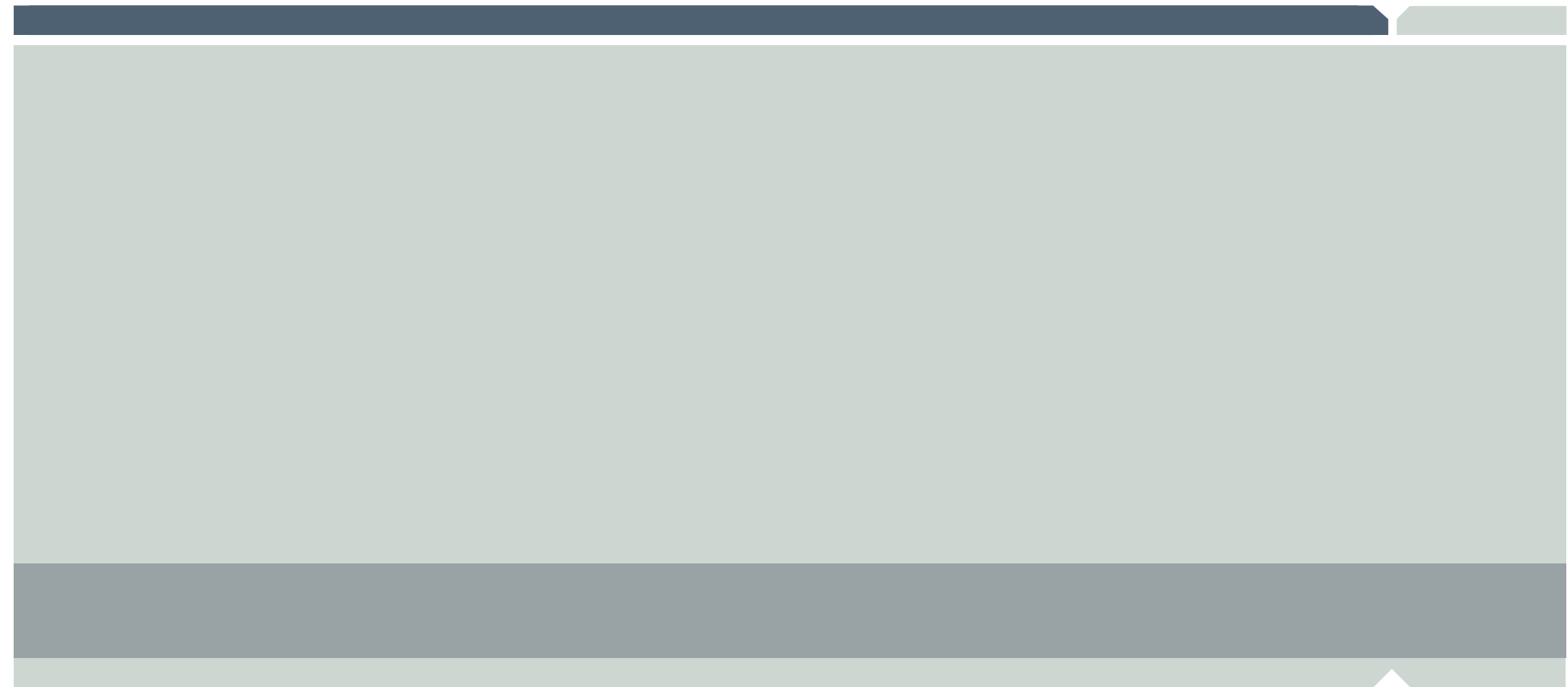
- 3-Phase BLDC Drive Using Variable DC Link Six-Step Inverter
- Application Note Number: DRM078
- Speed can exceed 10Krpm



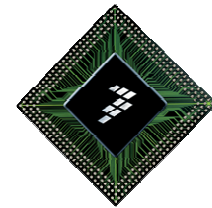
MC9S08MP16: Enablement

- DEMO9S08MP16 (generic demo board)
- Demos
 - Sensorless BLDC Motor Control using ADC approximation
- Reference Designs
 - Sensorless BLDC Motor control using Comparators
 - 3-Ph ACIM V/Hz Drive with PFC
 - Industrial HID Lamp with PFC
- Software Libraries
 - S08 Math and Embedded
- Application Notes
 - S08MP16 comparators for BLDC sensorless motor control
 - Using MP16 peripheral modules (FTM, Delay block, ADC)
 - high speed BLDC sensorless control using ADC approximation
 - BLDC Motor Control using Hall Sensors
 - Using FlexTimer in DC/BLDC Motor Control Application

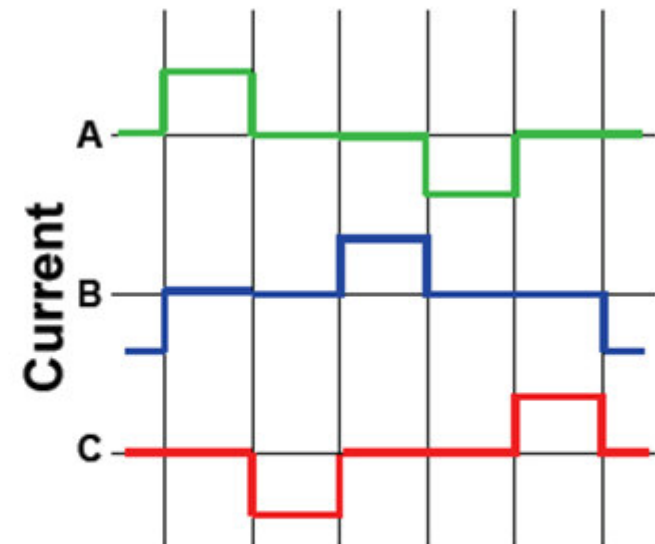
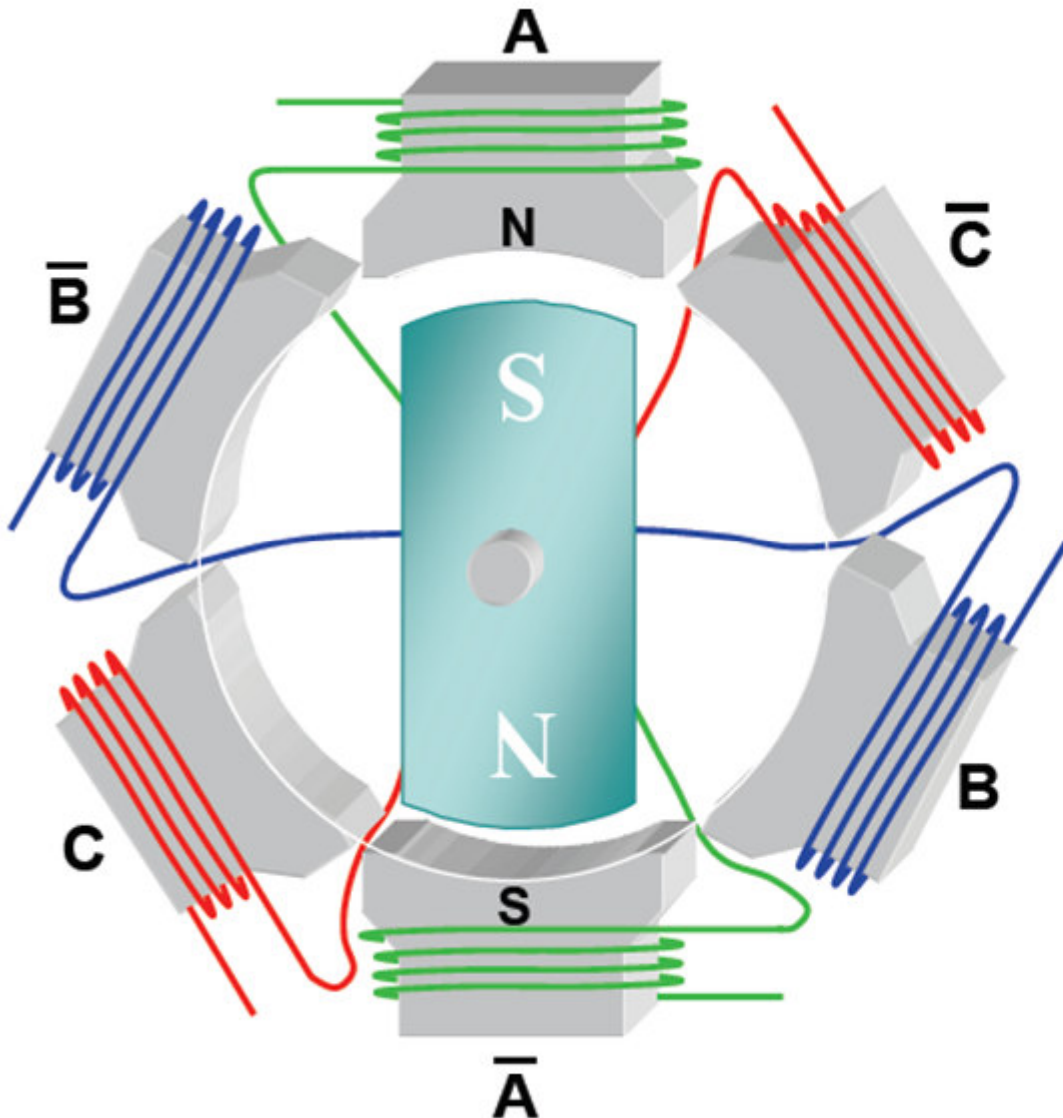




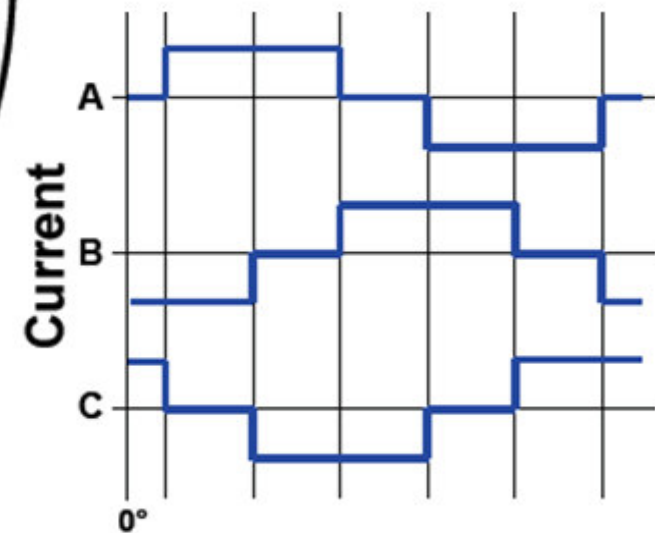
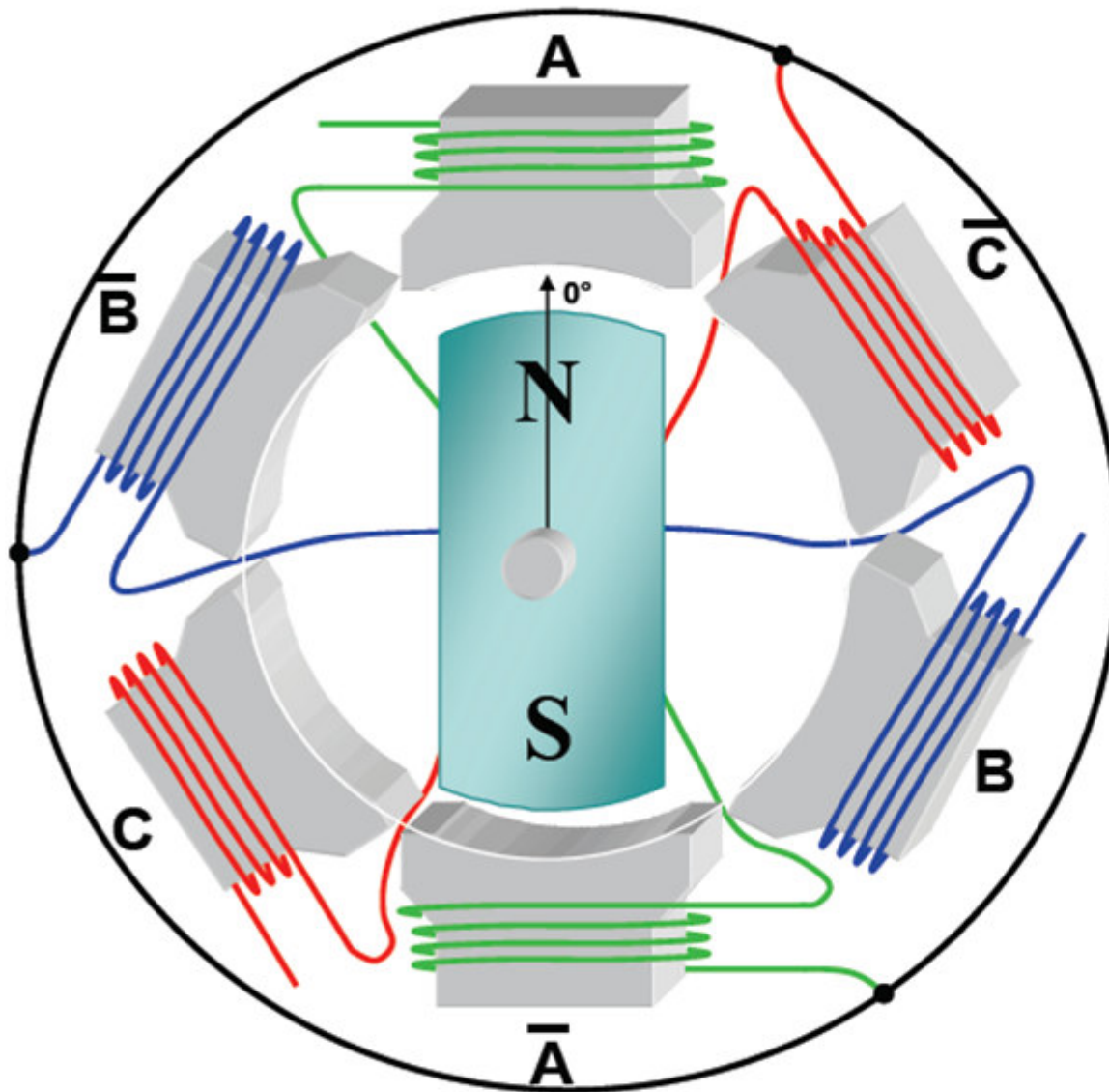
BLDC Control Basics

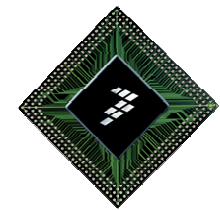


BLDC Control Basics - 1



BLDC Control Basics – 2



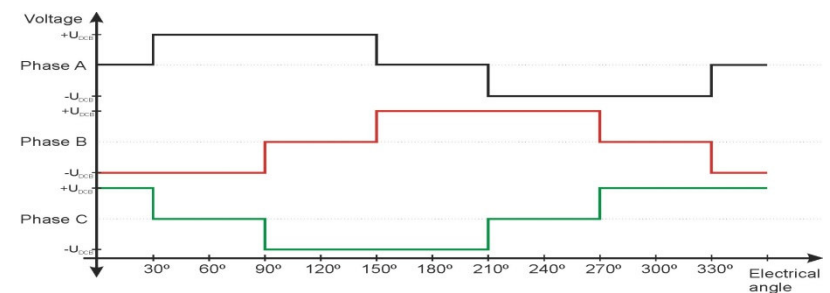
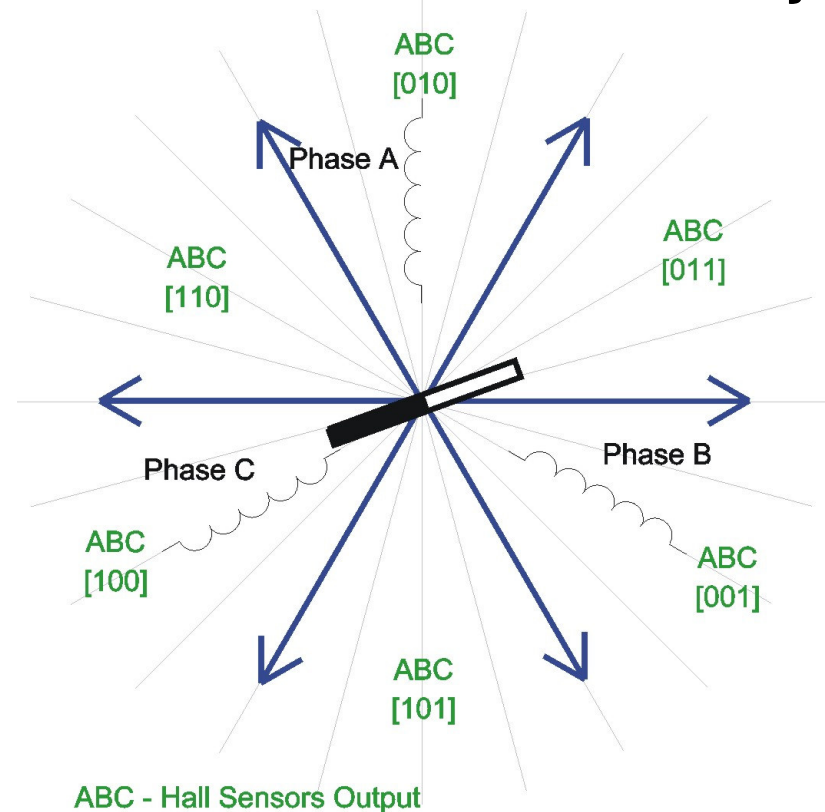
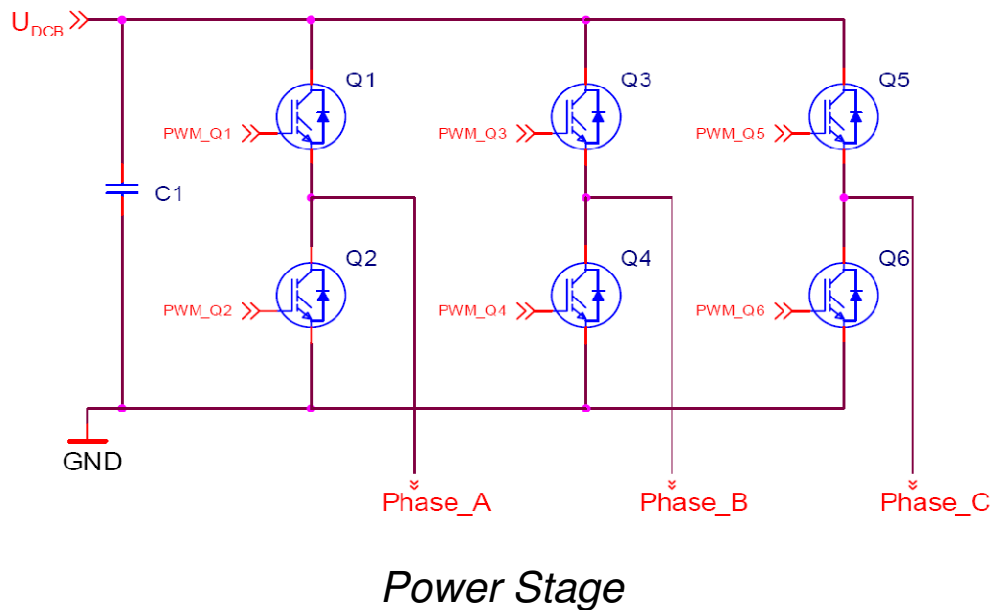


Commutation

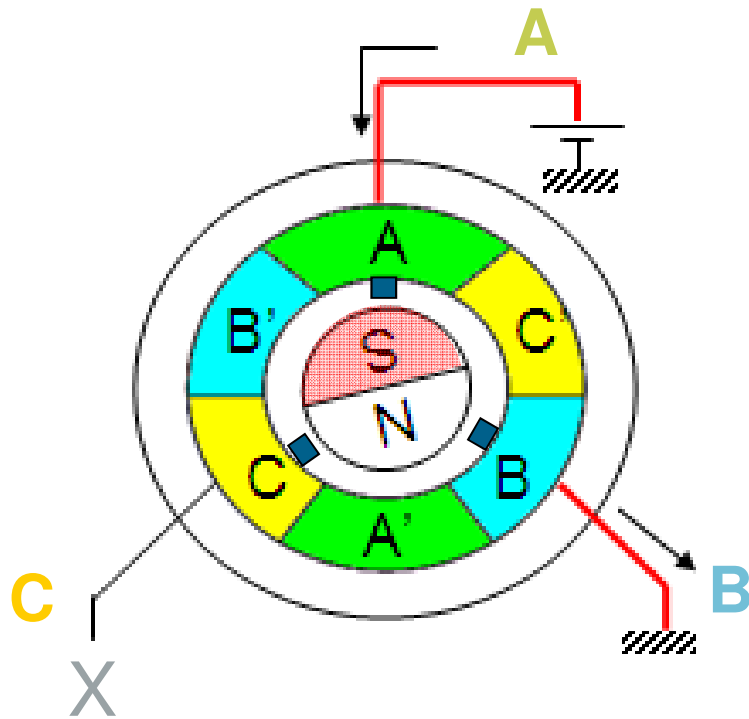
Sensorless BLDC Motor Control Theory

► Six Step BLDC Motor Control

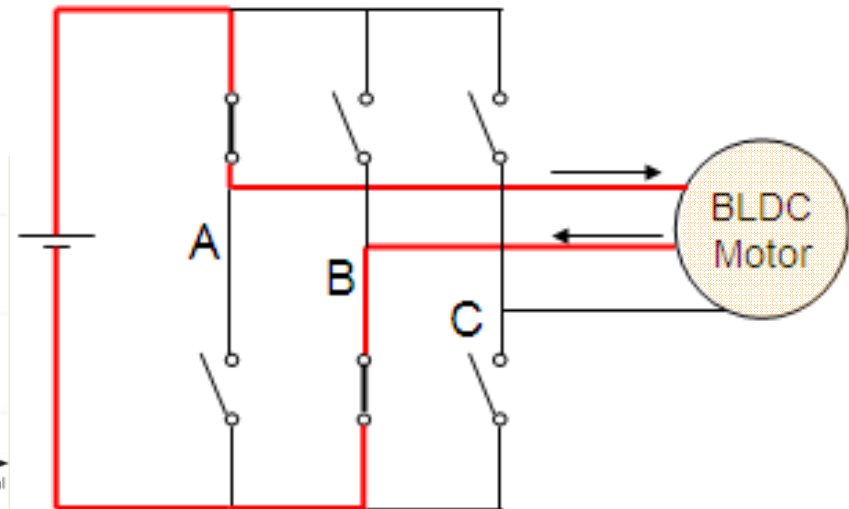
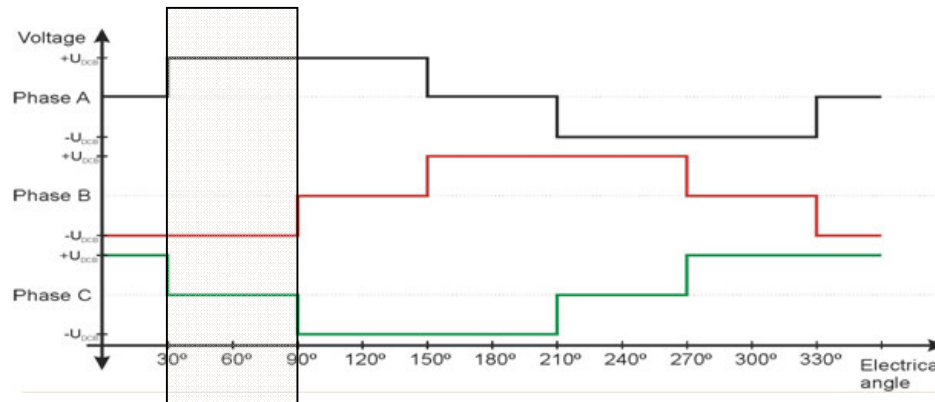
- Voltage applied on only two phases
- It creates 6 flux vectors
- Phases are powered based on rotor position
- The process is called *Commutation*



Control of 3-Phase Inverter

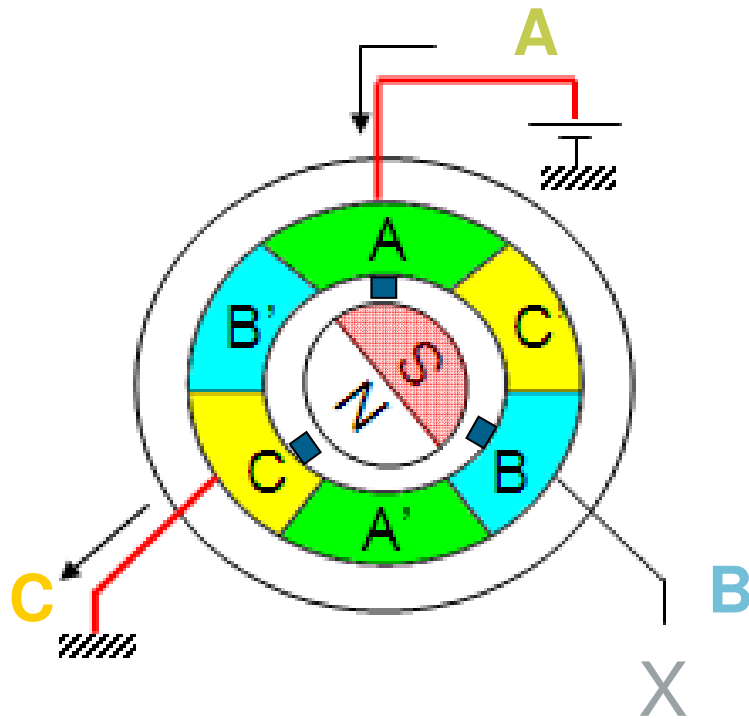


Phase A	Phase B	Phase C
$+V_{DCB}$	$-V_{DCB}$	NC
$+V_{DCB}$	NC	$-V_{DCB}$
NC	$+V_{DCB}$	$-V_{DCB}$
$-V_{DCB}$	$+V_{DCB}$	NC
$-V_{DCB}$	NC	$+V_{DCB}$
NC	$-V_{DCB}$	$+V_{DCB}$

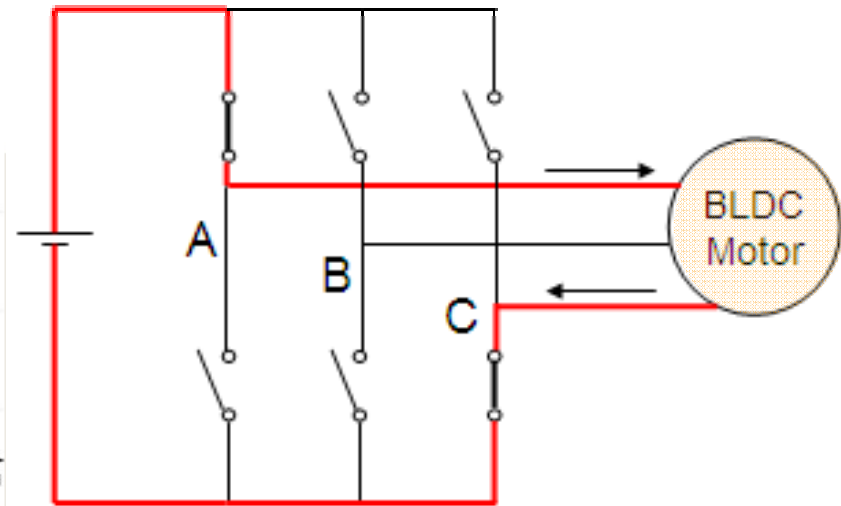
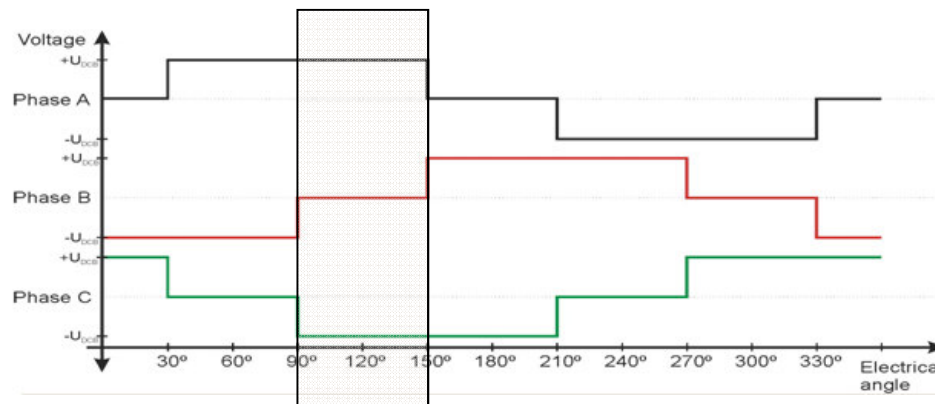


Animation part 1/6

Control of 3-Phase Inverter

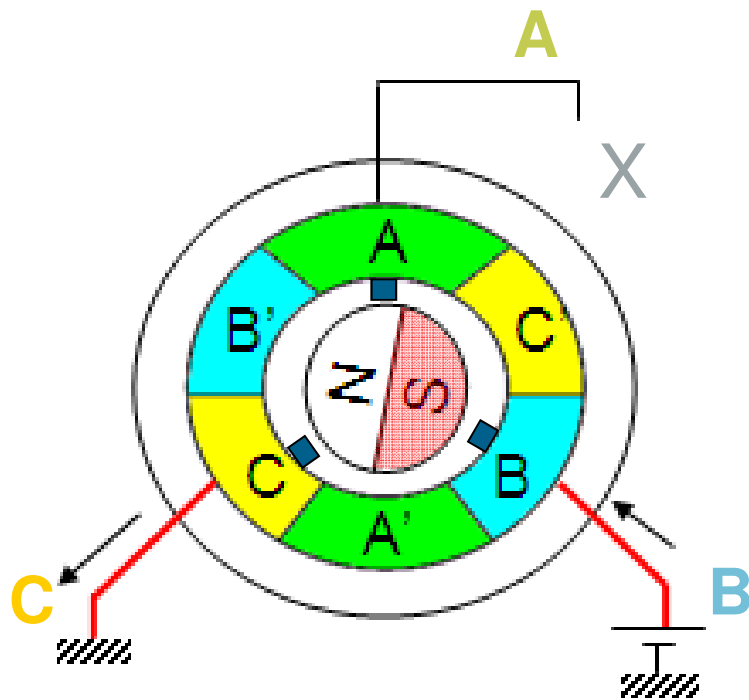


Phase A	Phase B	Phase C
$+V_{DCB}$	$-V_{DCB}$	NC
$+V_{DCB}$	NC	$-V_{DCB}$
NC	$+V_{DCB}$	$-V_{DCB}$
$-V_{DCB}$	$+V_{DCB}$	NC
$-V_{DCB}$	NC	$+V_{DCB}$
NC	$-V_{DCB}$	$+V_{DCB}$

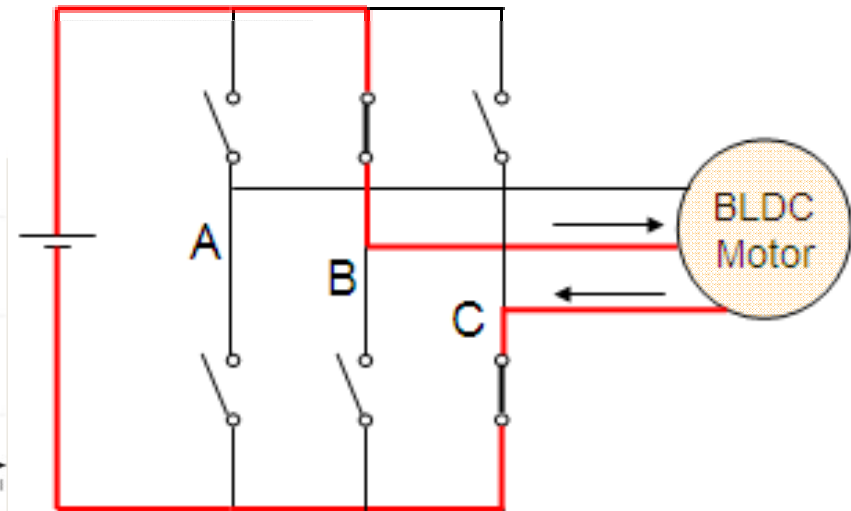
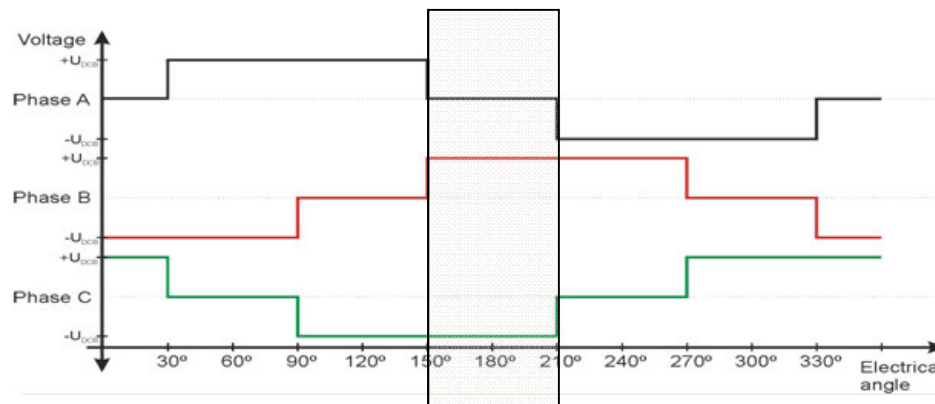


Animation part 2/6

Control of 3-Phase Inverter

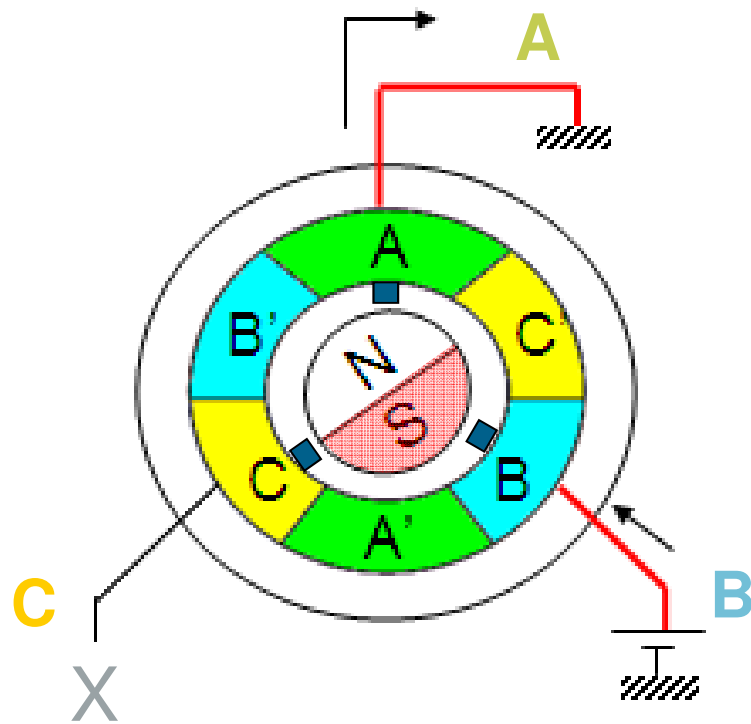


Phase A	Phase B	Phase C
$+V_{DCB}$	$-V_{DCB}$	NC
$+V_{DCB}$	NC	$-V_{DCB}$
NC	$+V_{DCB}$	$-V_{DCB}$
$-V_{DCB}$	$+V_{DCB}$	NC
$-V_{DCB}$	NC	$+V_{DCB}$
NC	$-V_{DCB}$	$+V_{DCB}$

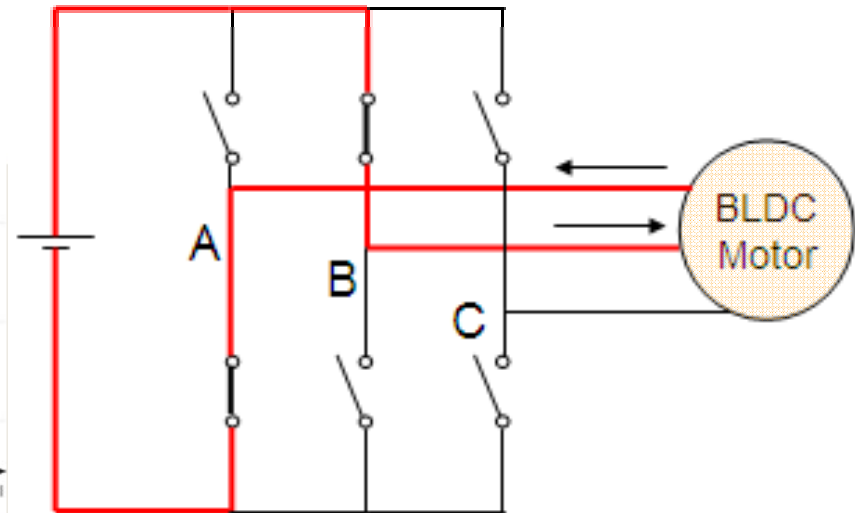
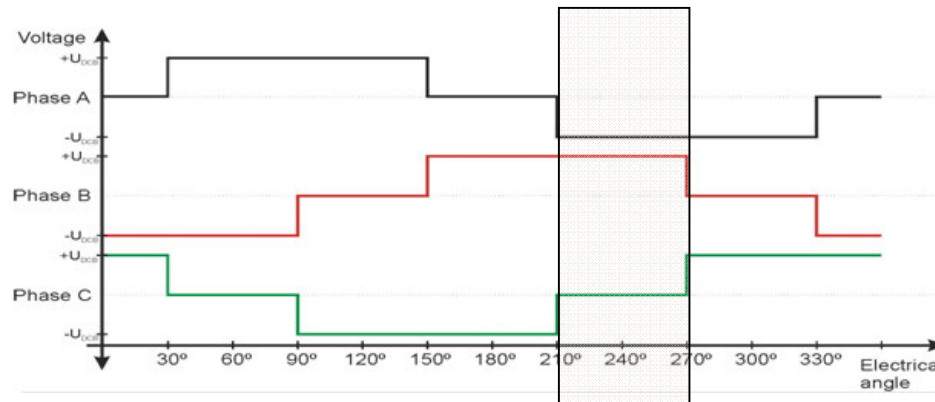


Animation part 3/6

Control of 3-Phase Inverter

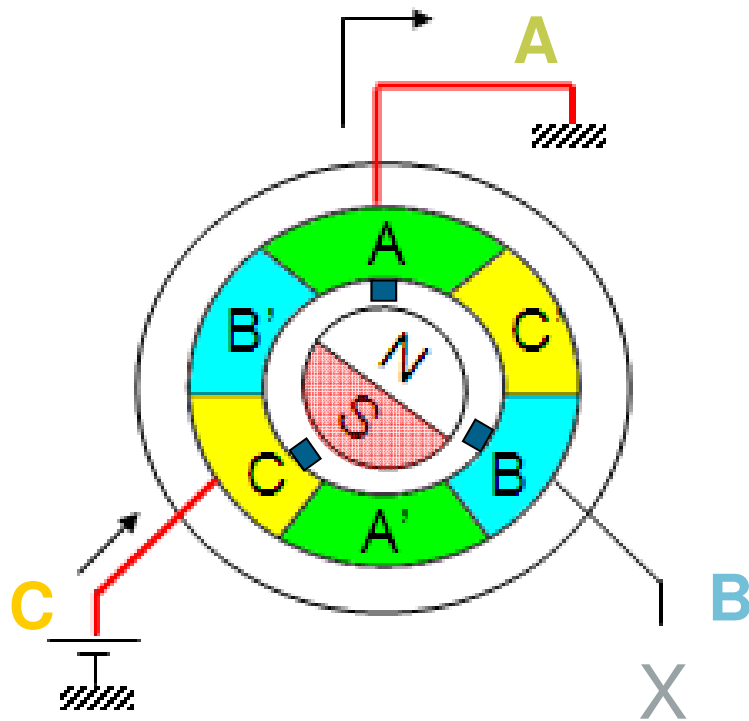


Phase A	Phase B	Phase C
$+V_{DCB}$	$-V_{DCB}$	NC
$+V_{DCB}$	NC	$-V_{DCB}$
NC	$+V_{DCB}$	$-V_{DCB}$
$-V_{DCB}$	$+V_{DCB}$	NC
$-V_{DCB}$	NC	$+V_{DCB}$
NC	$-V_{DCB}$	$+V_{DCB}$

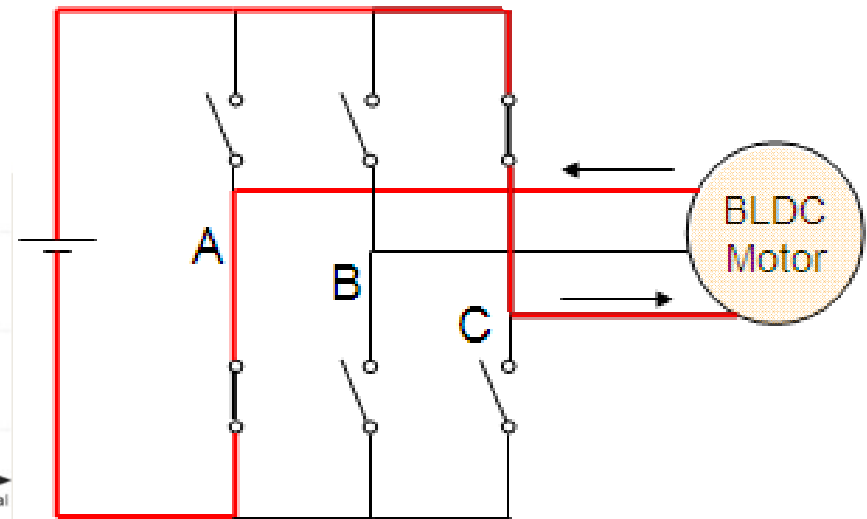
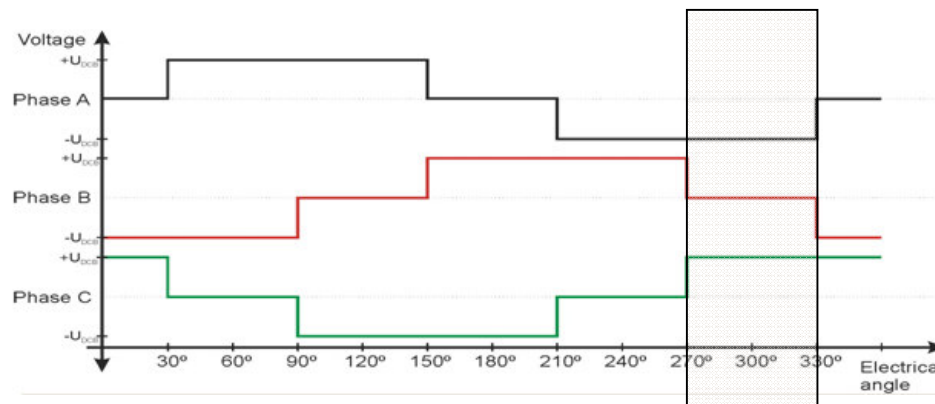


Animation part 4/6

Control of 3-Phase Inverter

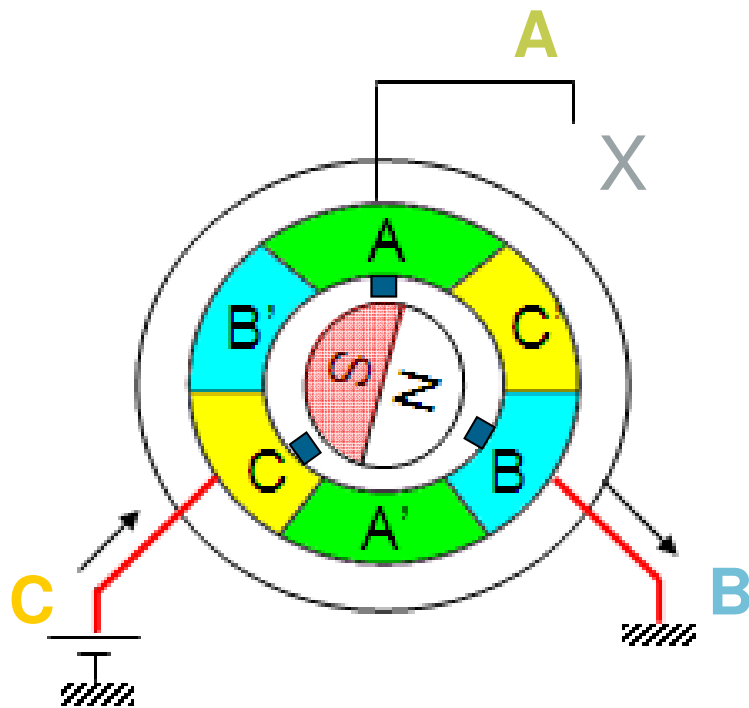


Phase A	Phase B	Phase C
$+V_{DCB}$	$-V_{DCB}$	NC
$+V_{DCB}$	NC	$-V_{DCB}$
NC	$+V_{DCB}$	$-V_{DCB}$
$-V_{DCB}$	$+V_{DCB}$	NC
$-V_{DCB}$	NC	$+V_{DCB}$
NC	$-V_{DCB}$	$+V_{DCB}$

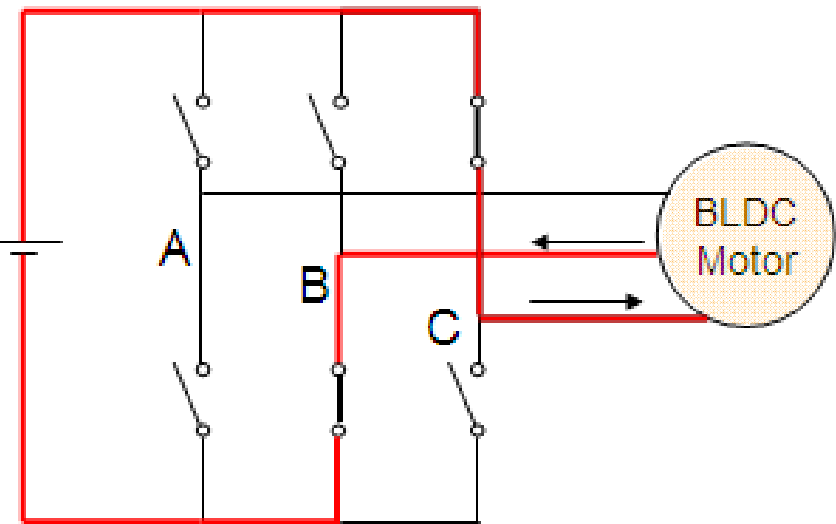
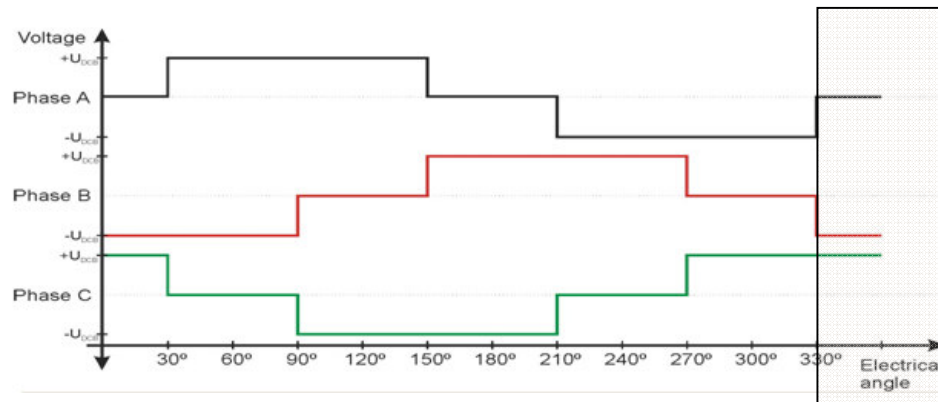


Animation part 5/6

Control of 3-Phase Inverter

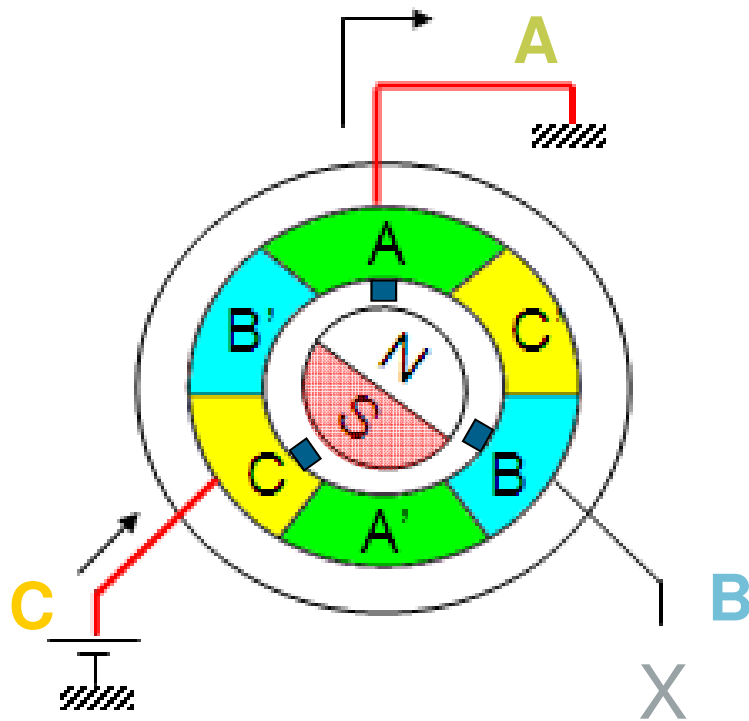


Phase A	Phase B	Phase C
$+V_{DCB}$	$-V_{DCB}$	NC
$+V_{DCB}$	NC	$-V_{DCB}$
NC	$+V_{DCB}$	$-V_{DCB}$
$-V_{DCB}$	$+V_{DCB}$	NC
$-V_{DCB}$	NC	$+V_{DCB}$
NC	$-V_{DCB}$	$+V_{DCB}$

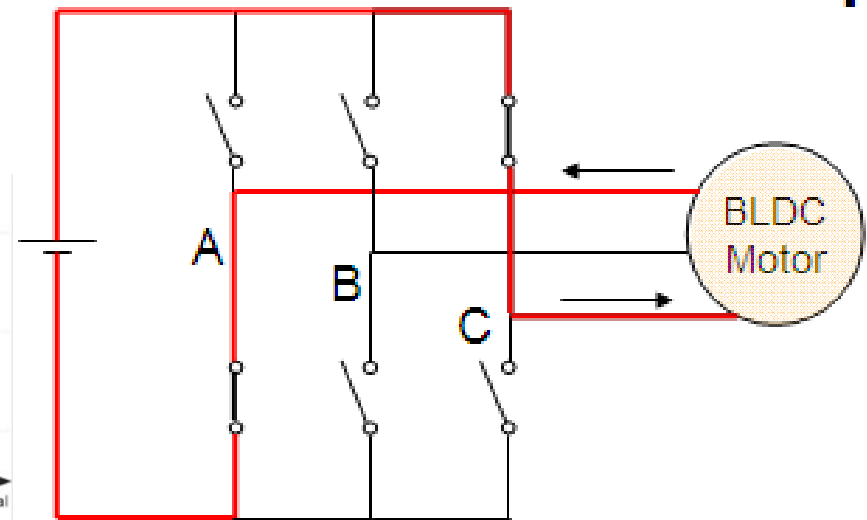
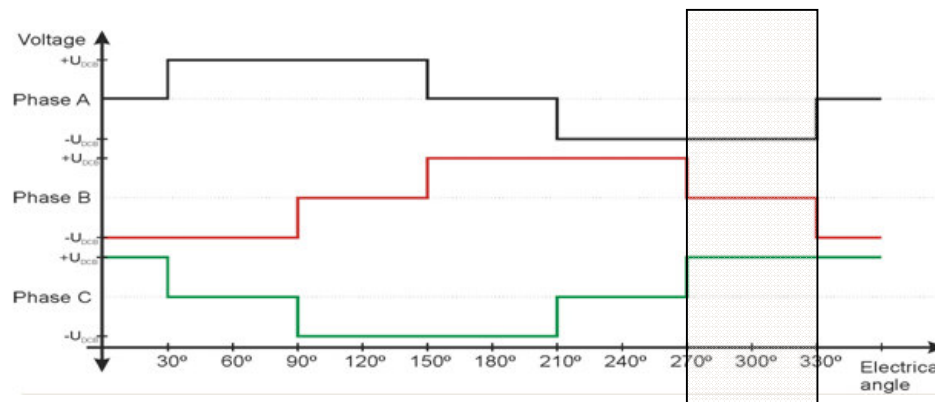


Animation part 6/6

Control of 3-Phase Inverter

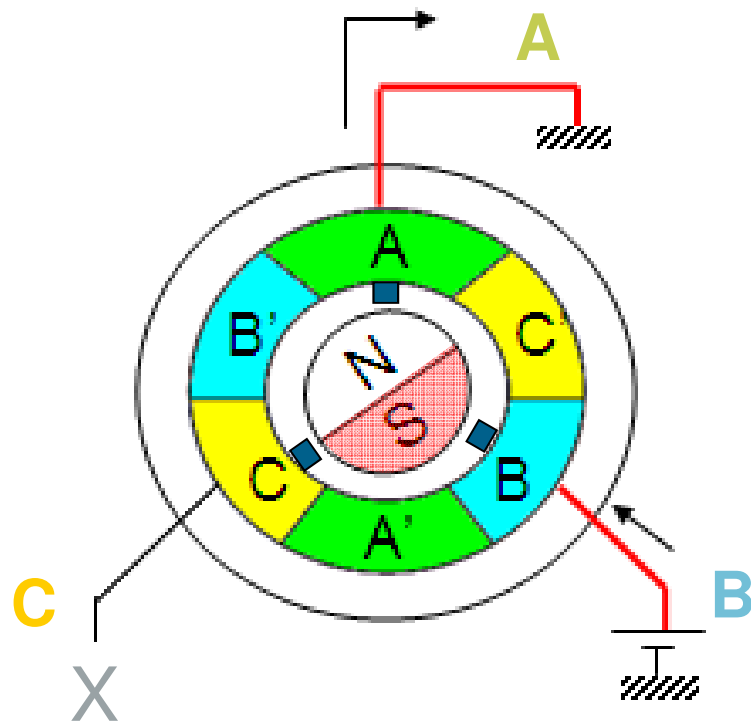


Phase A	Phase B	Phase C
$+V_{DCB}$	$-V_{DCB}$	NC
$+V_{DCB}$	NC	$-V_{DCB}$
NC	$+V_{DCB}$	$-V_{DCB}$
$-V_{DCB}$	$+V_{DCB}$	NC
$-V_{DCB}$	NC	$+V_{DCB}$
NC	$-V_{DCB}$	$+V_{DCB}$

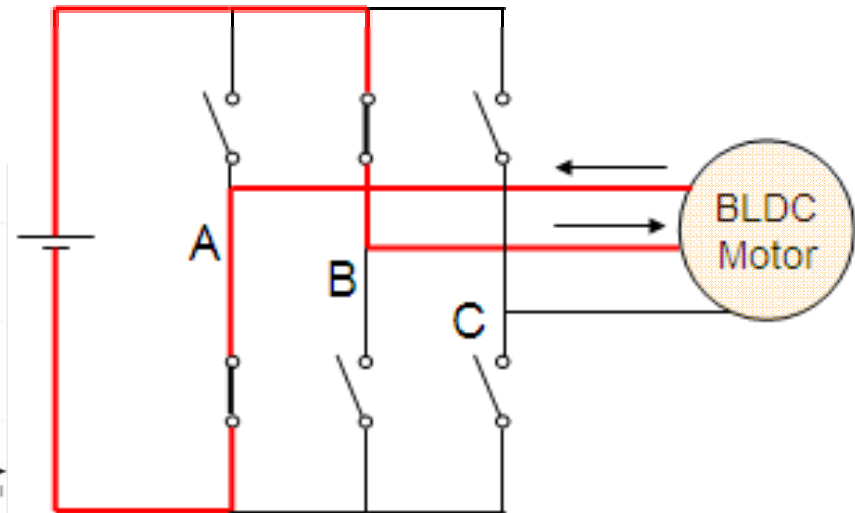
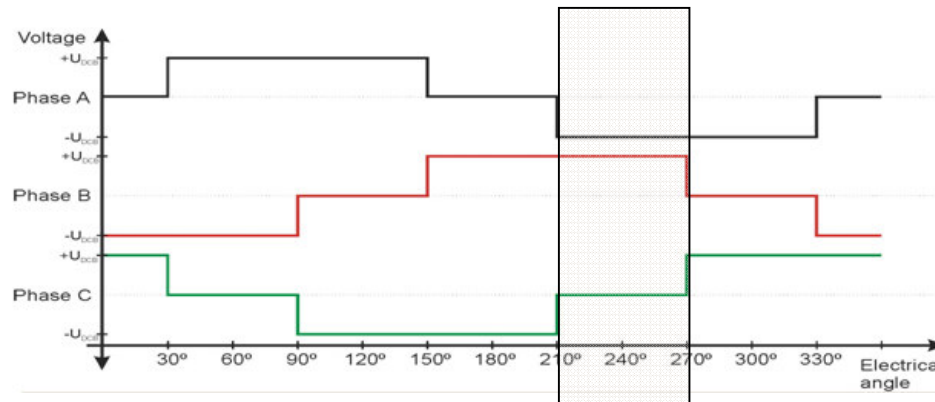


Animation part 5/6

Control of 3-Phase Inverter

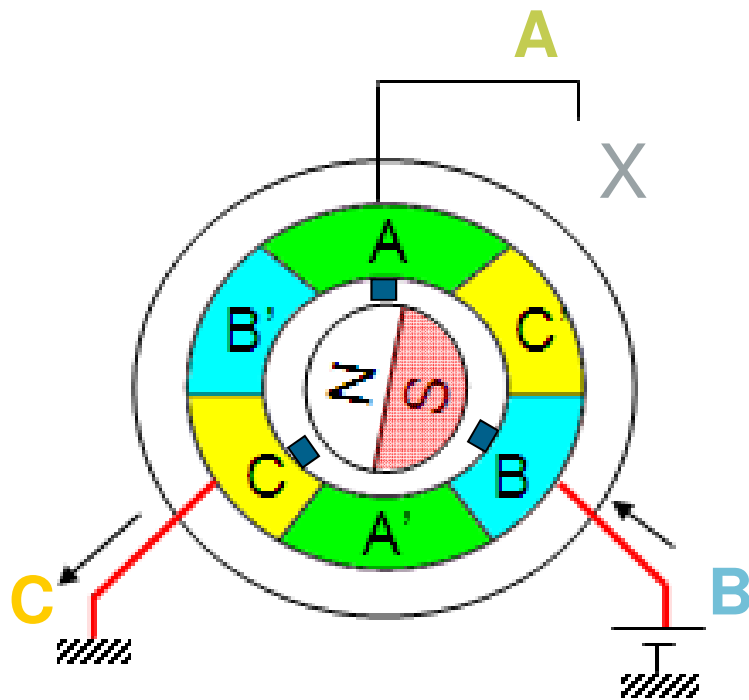


Phase A	Phase B	Phase C
$+V_{DCB}$	$-V_{DCB}$	NC
$+V_{DCB}$	NC	$-V_{DCB}$
NC	$+V_{DCB}$	$-V_{DCB}$
$-V_{DCB}$	$+V_{DCB}$	NC
$-V_{DCB}$	NC	$+V_{DCB}$
NC	$-V_{DCB}$	$+V_{DCB}$

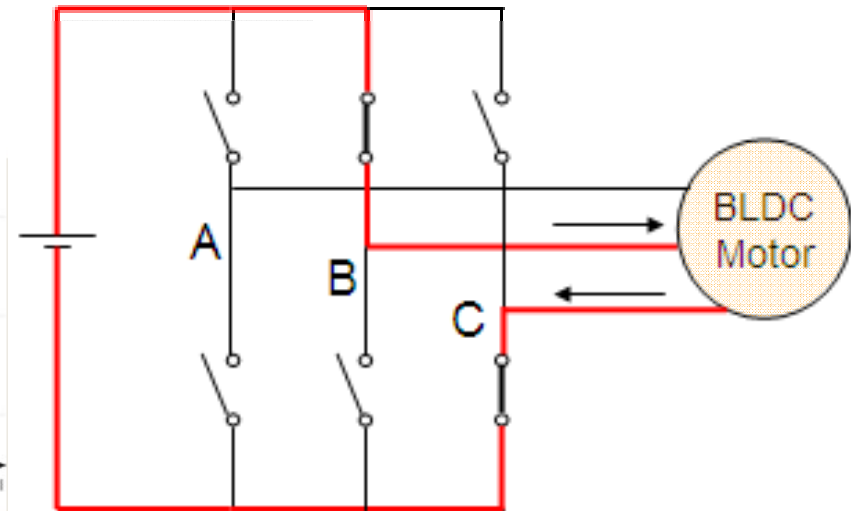
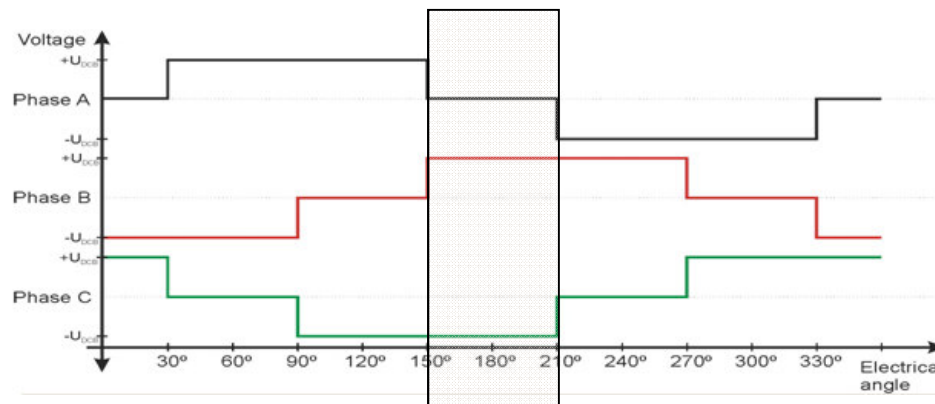


Animation part 4/6

Control of 3-Phase Inverter

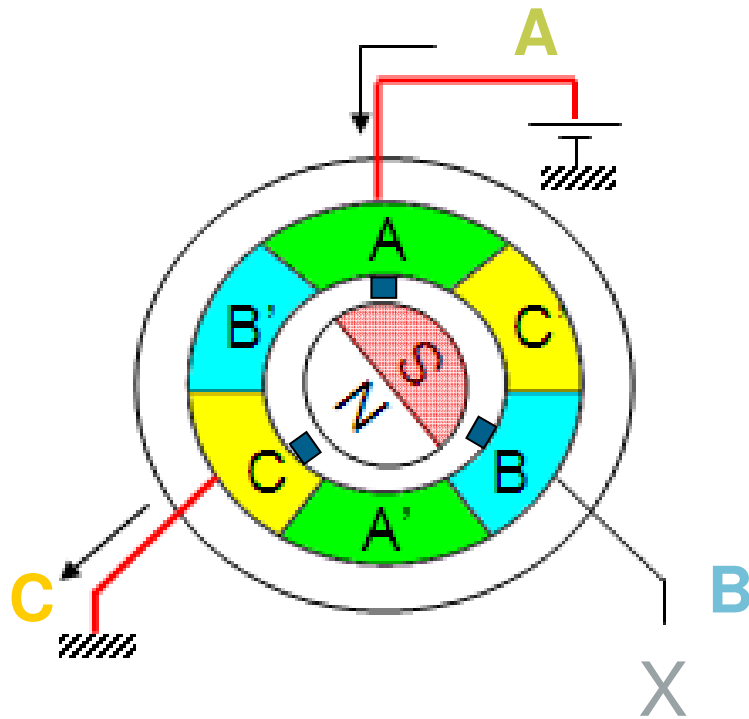


Phase A	Phase B	Phase C
$+V_{DCB}$	$-V_{DCB}$	NC
$+V_{DCB}$	NC	$-V_{DCB}$
NC	$+V_{DCB}$	$-V_{DCB}$
$-V_{DCB}$	$+V_{DCB}$	NC
$-V_{DCB}$	NC	$+V_{DCB}$
NC	$-V_{DCB}$	$+V_{DCB}$

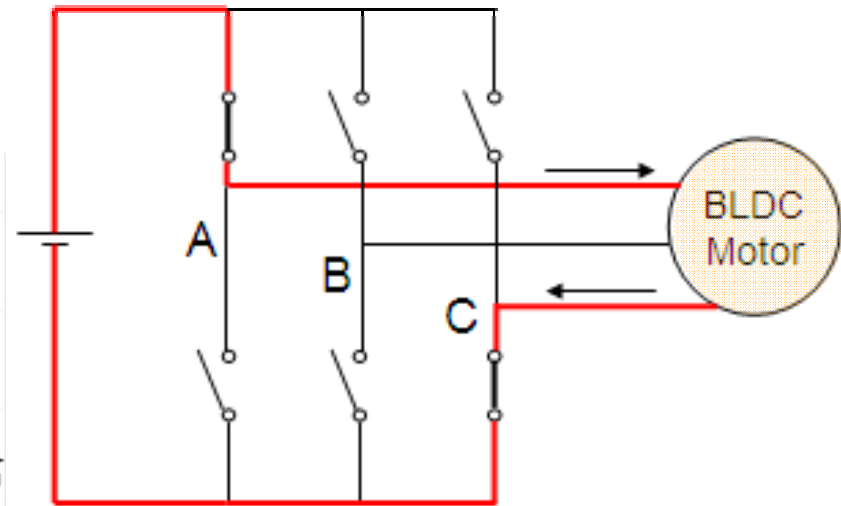
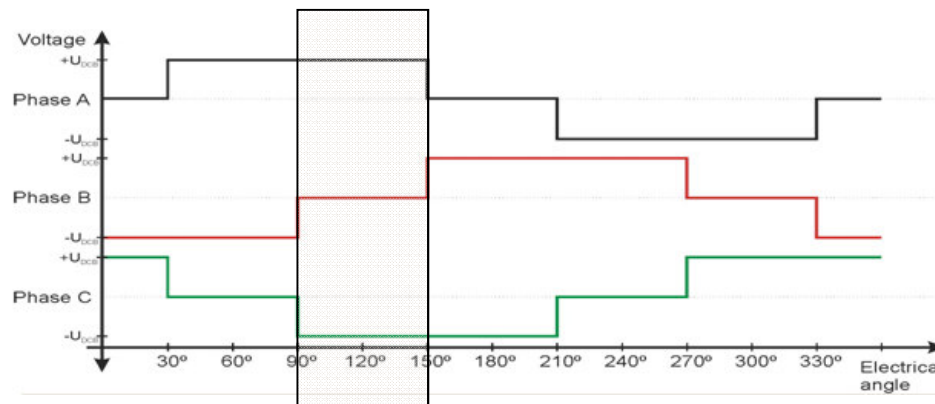


Animation part 3/6

Control of 3-Phase Inverter

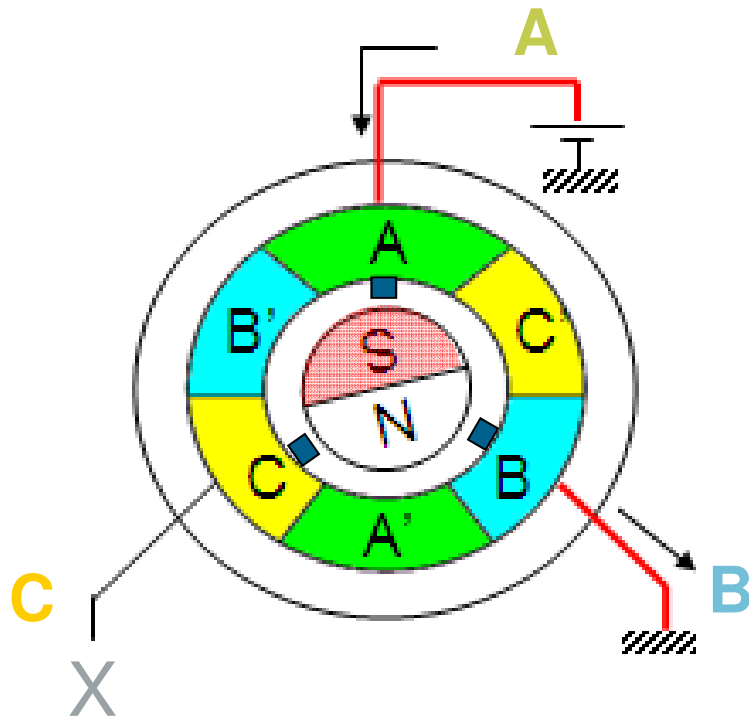


Phase A	Phase B	Phase C
$+V_{DCB}$	$-V_{DCB}$	NC
$+V_{DCB}$	NC	$-V_{DCB}$
NC	$+V_{DCB}$	$-V_{DCB}$
$-V_{DCB}$	$+V_{DCB}$	NC
$-V_{DCB}$	NC	$+V_{DCB}$
NC	$-V_{DCB}$	$+V_{DCB}$

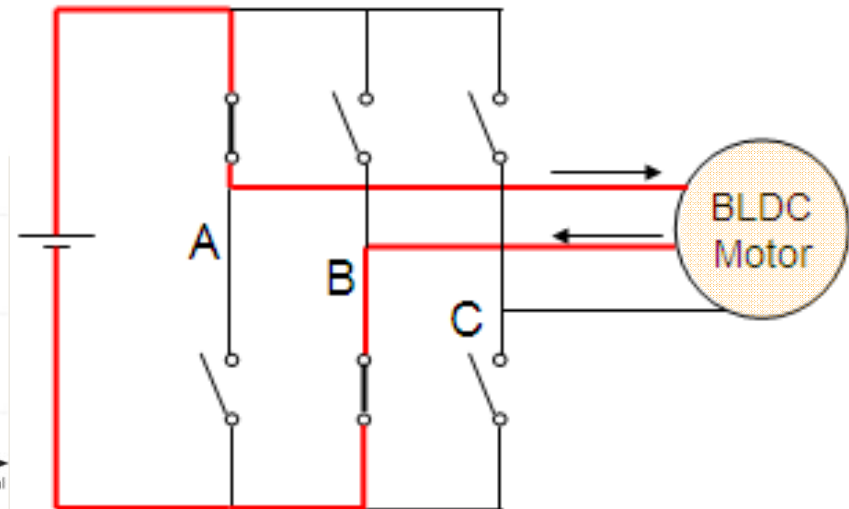
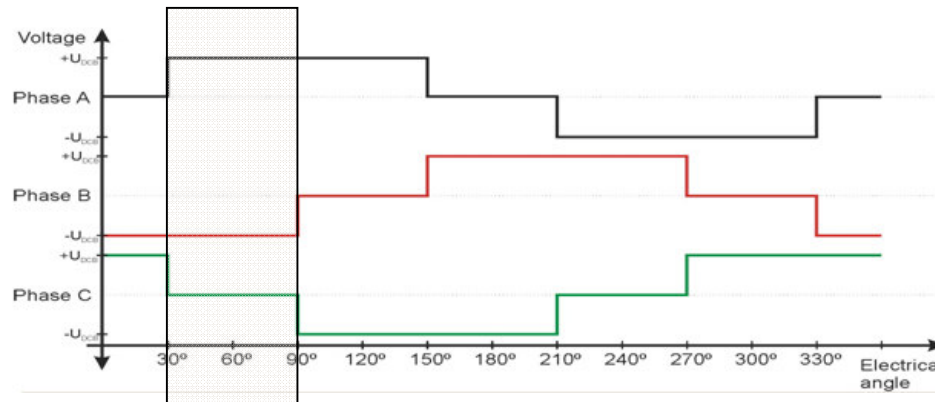


Animation part 2/6

Control of 3-Phase Inverter

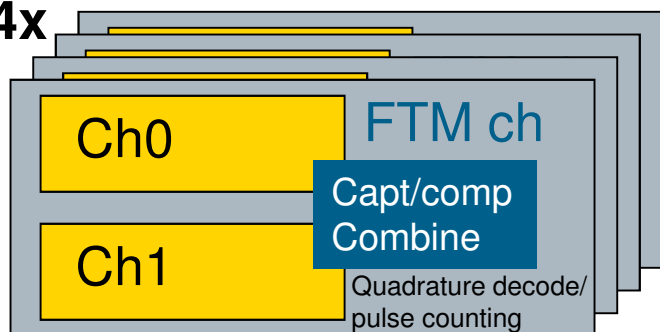


Phase A	Phase B	Phase C
$+V_{DCB}$	$-V_{DCB}$	NC
$+V_{DCB}$	NC	$-V_{DCB}$
NC	$+V_{DCB}$	$-V_{DCB}$
$-V_{DCB}$	$+V_{DCB}$	NC
$-V_{DCB}$	NC	$+V_{DCB}$
NC	$-V_{DCB}$	$+V_{DCB}$



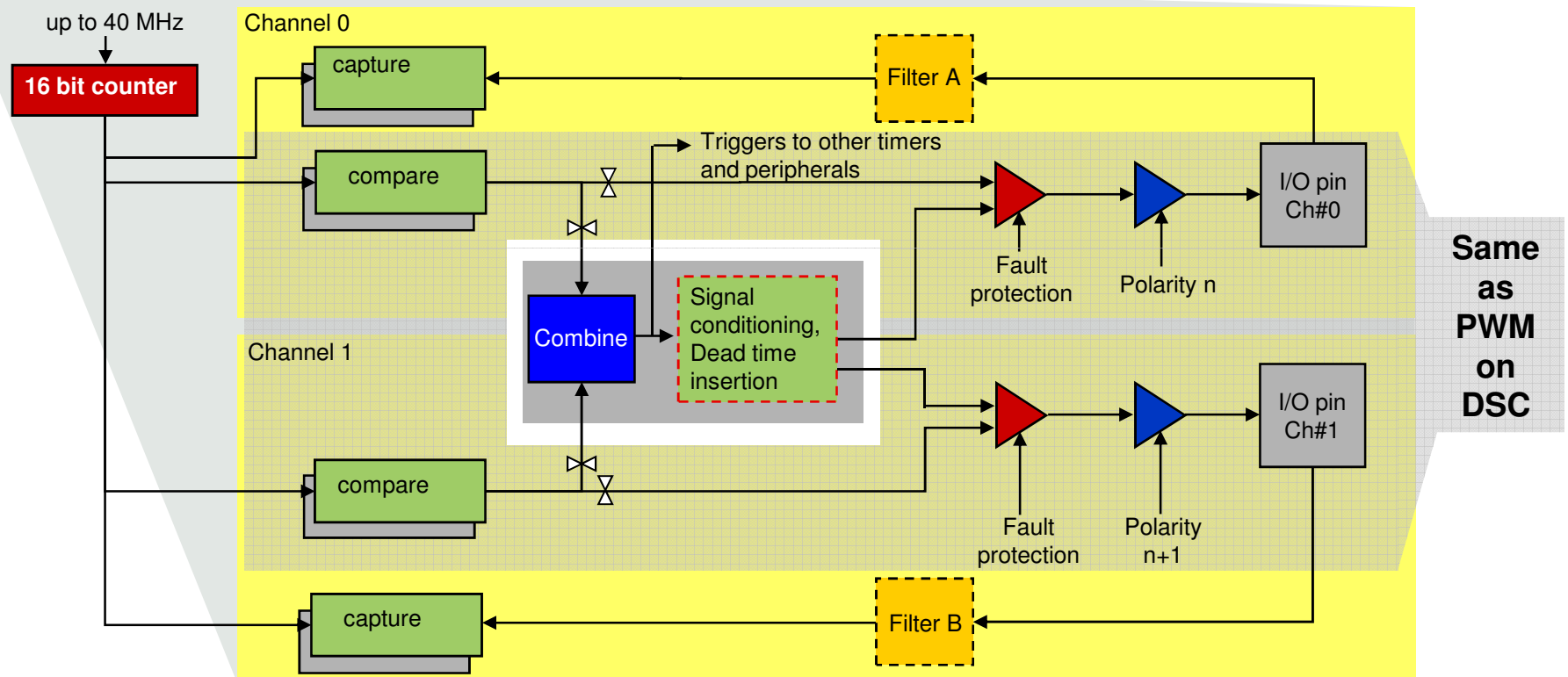
Animation part 1/6

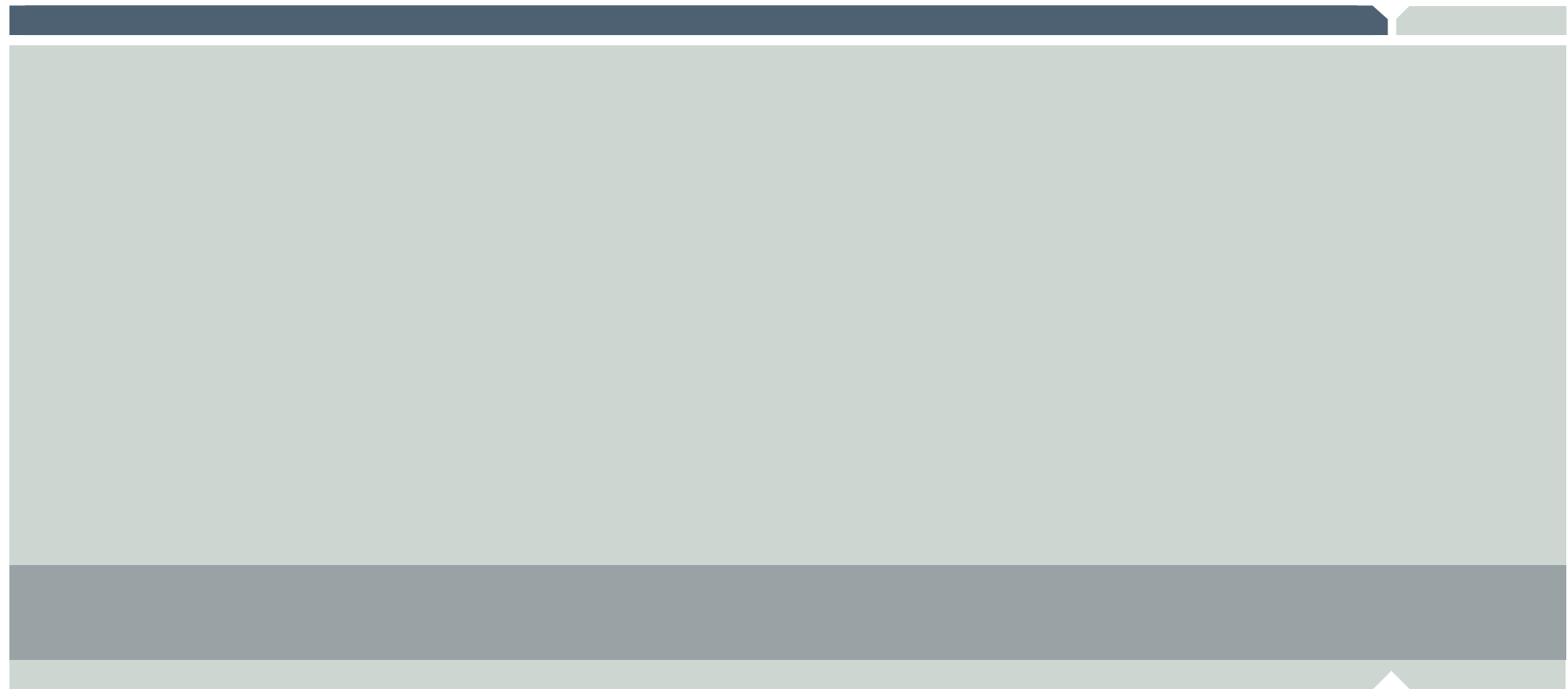
► 4x



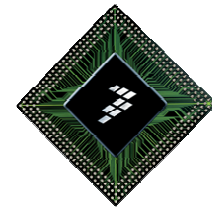
Flextimer on 9S08MP16

- Combining block provides flexibility without breaking compatibility
- Added functionality is optional within the architecture

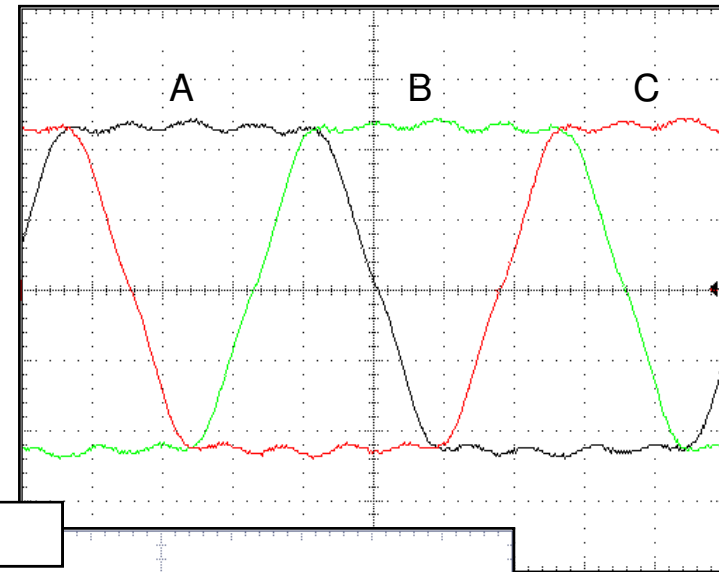
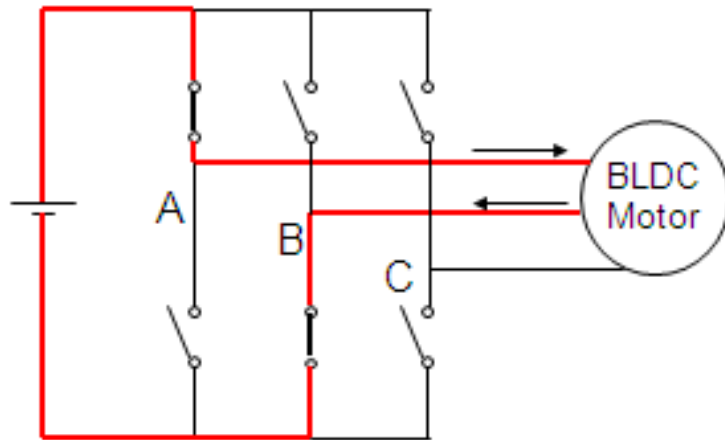




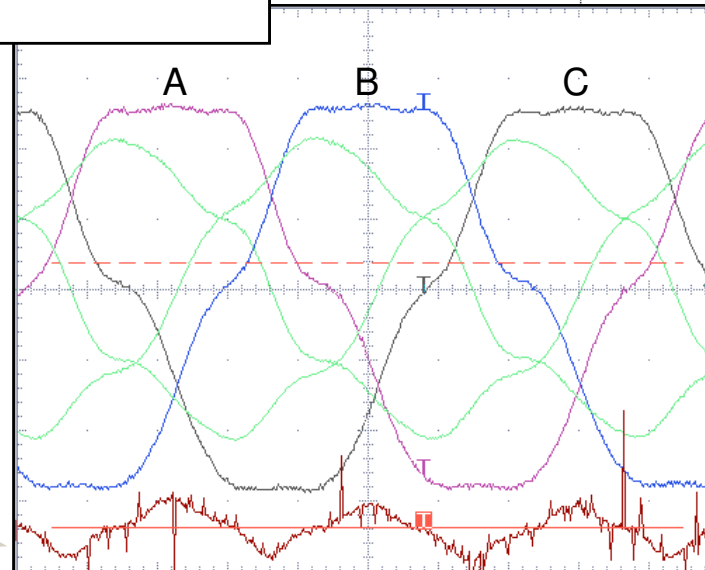
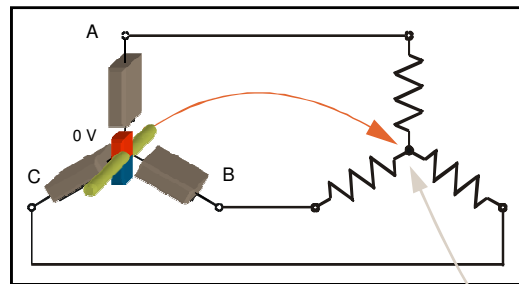
Back EMF Basics



Sensorless BLDC Motor Control Theory



- Trapezoidal Back-EMF



Sensorless BLDC Motor Control Theory

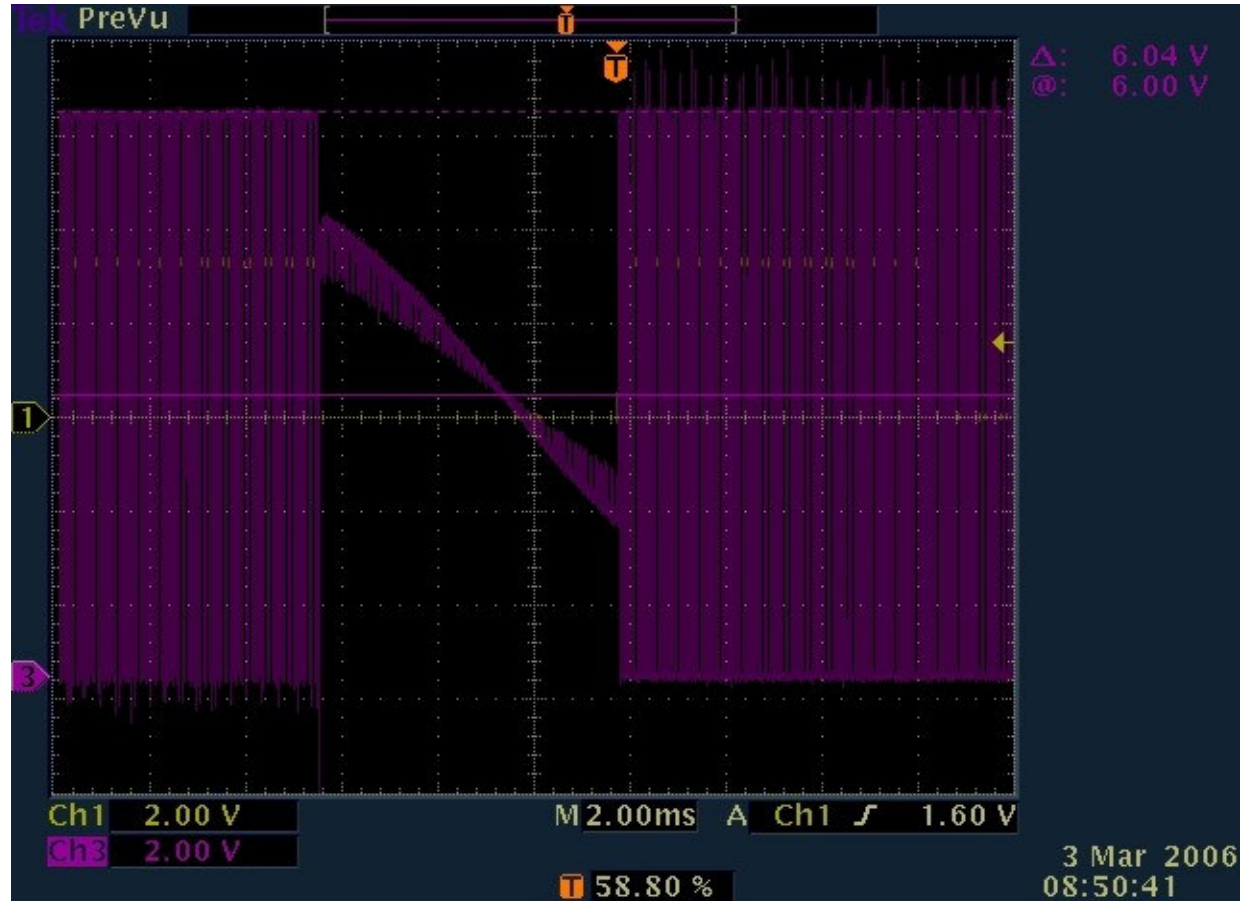
► Sensorless Back-EMF zero cross algorithm

- Sensing voltage on disconnected phase
 - When sensed voltage crosses half of DC bus voltage, the rotor is in middle between two commutations
 - The detection of this zero crossing allows detection of rotor position

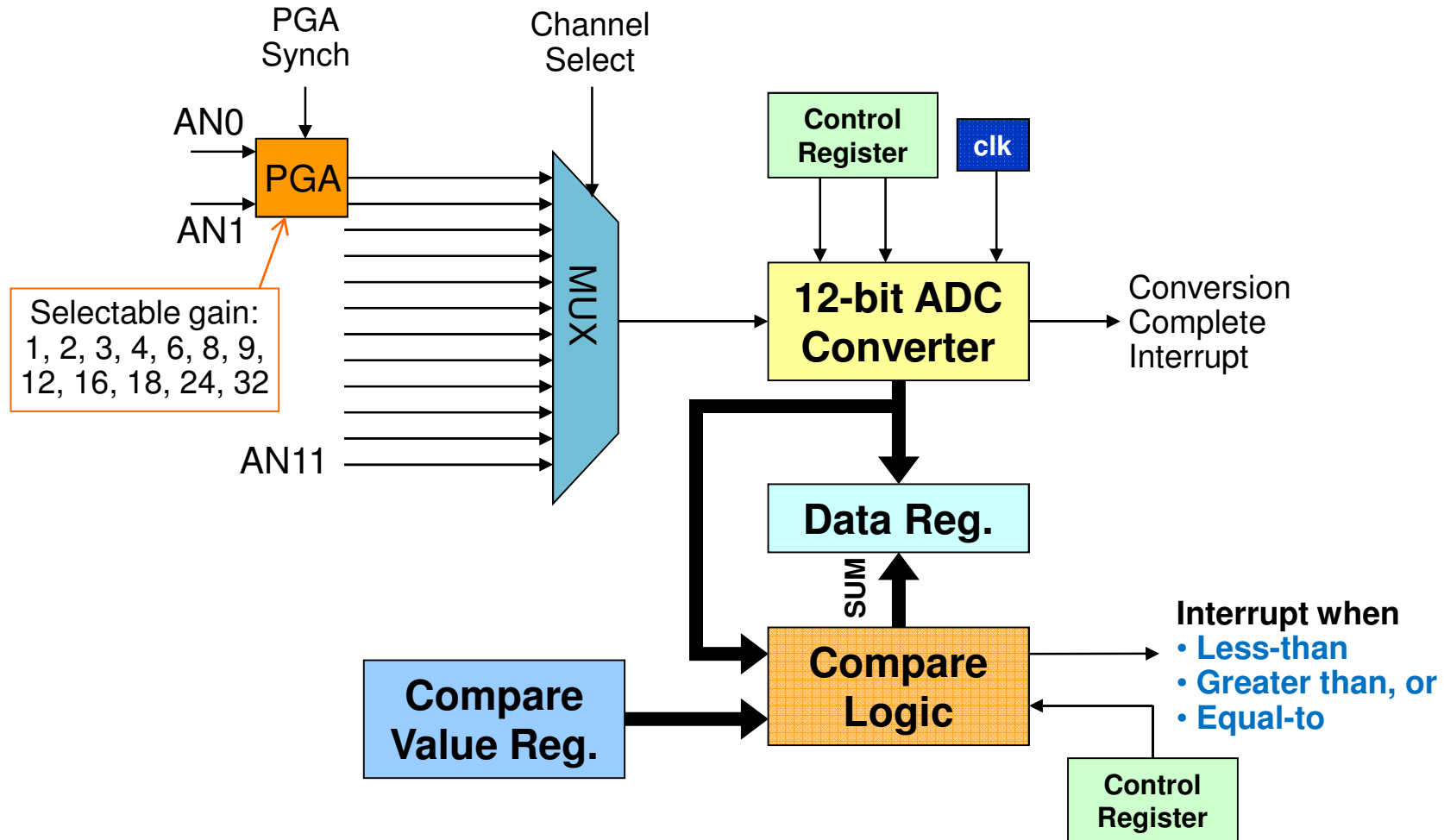


Sensorless BLDC Motor Control Theory

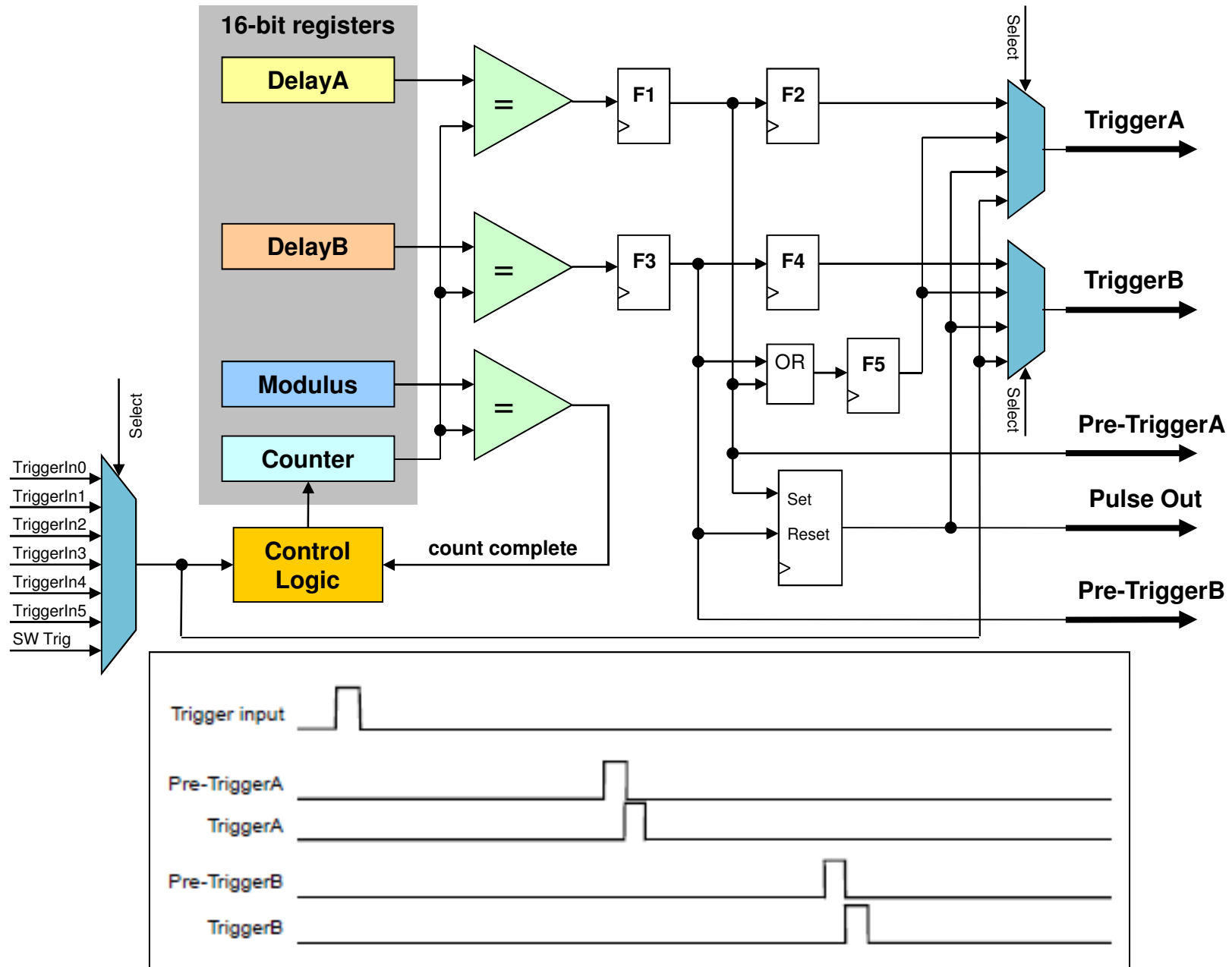
► Sensorless Back-EMF zero crossing - detail



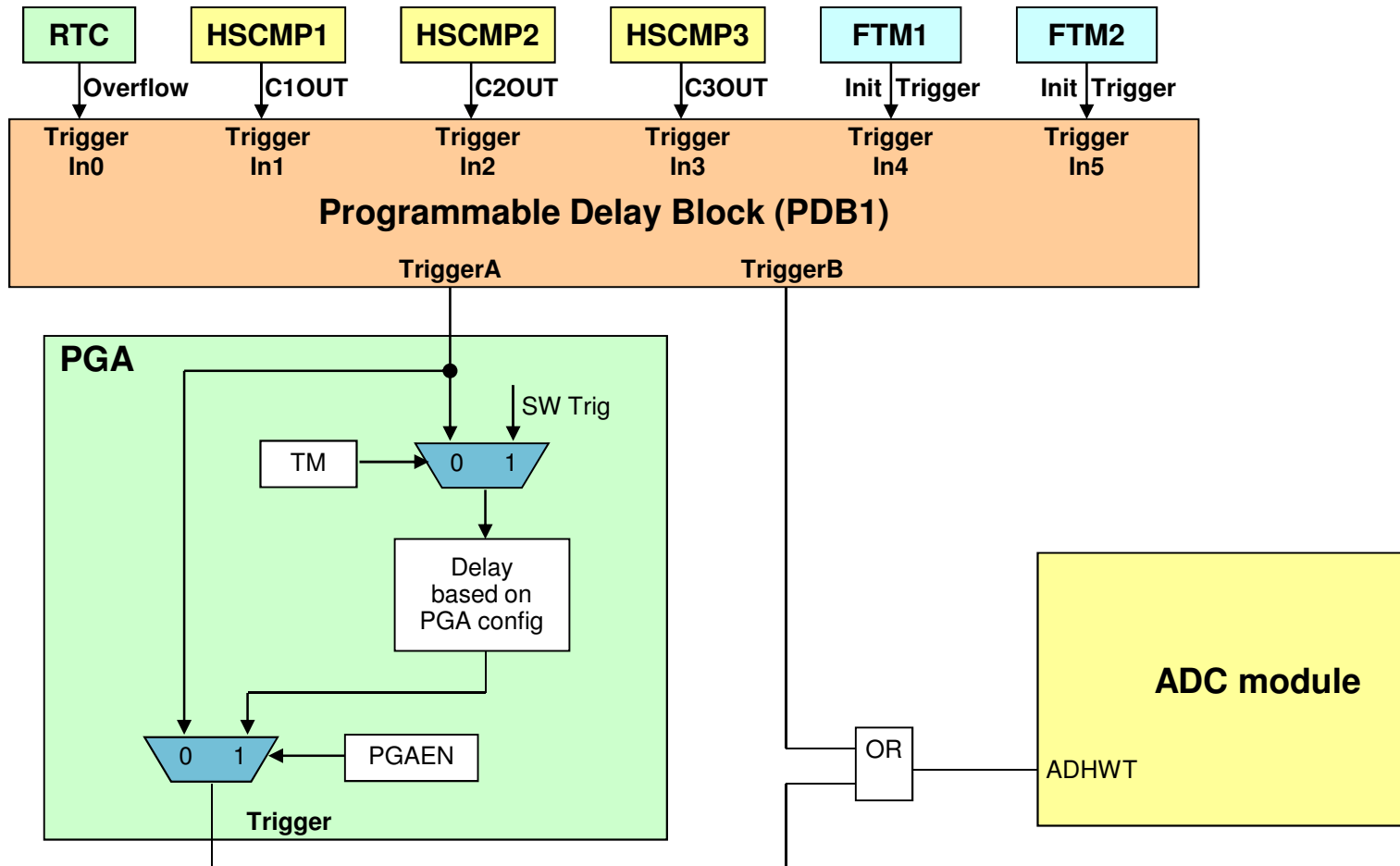
9S08MP16 ADC Structure



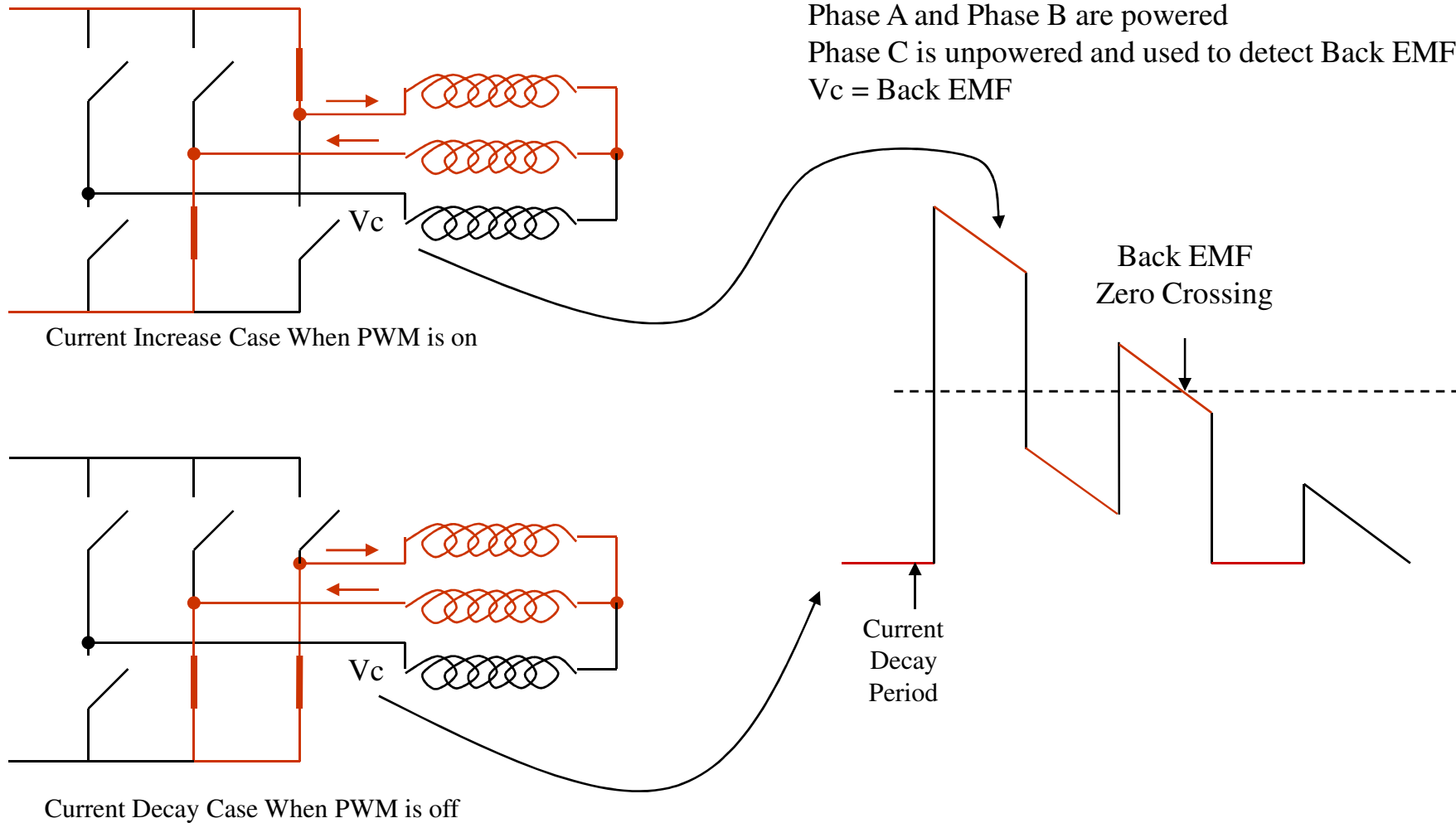
Programmable Delay Block on 9S08MP16



9S08MP16 ADC Synchronization Features

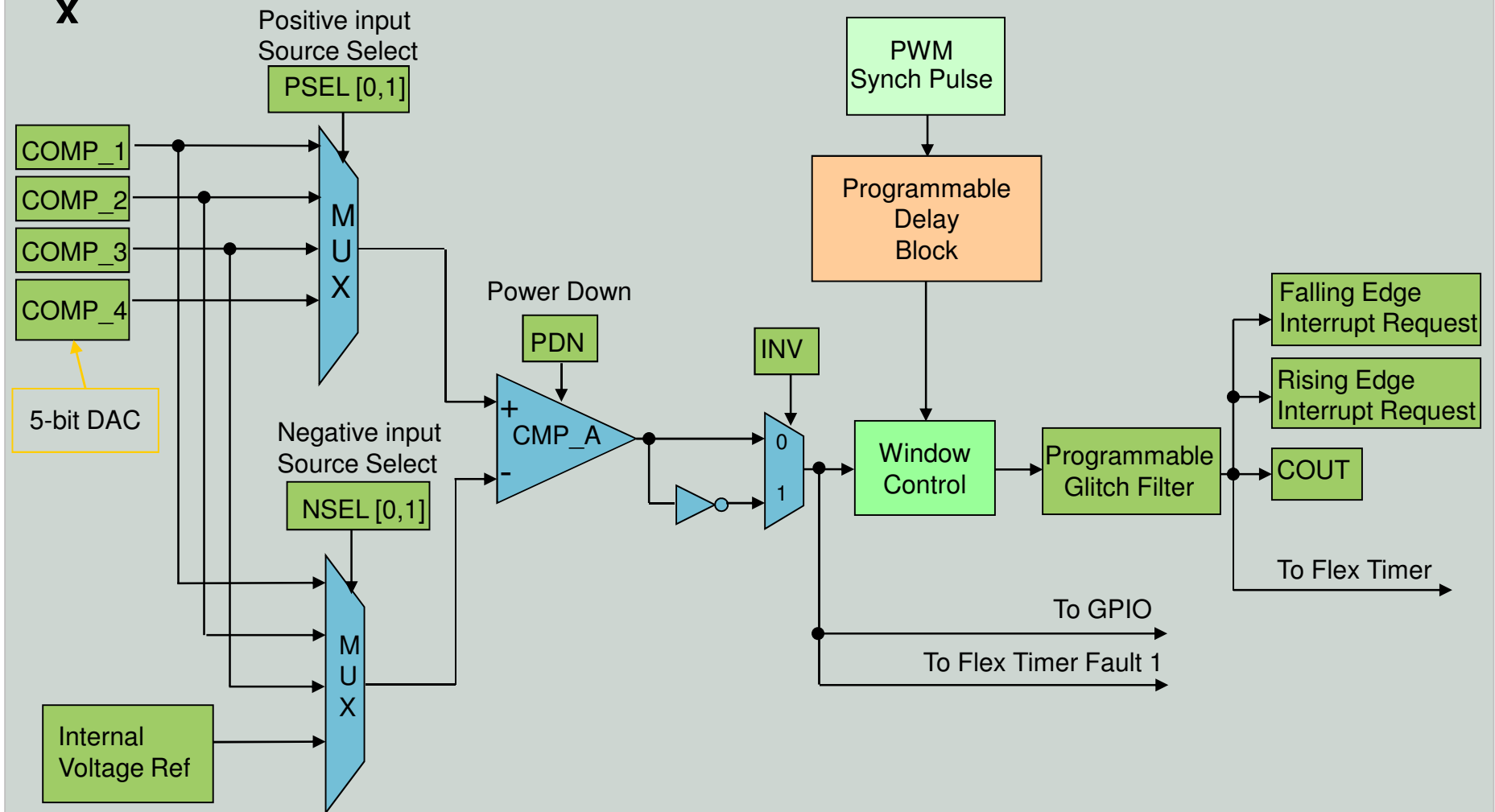


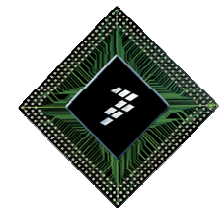
Comparator Window Sampling



MP16 Comparator with Window Sample

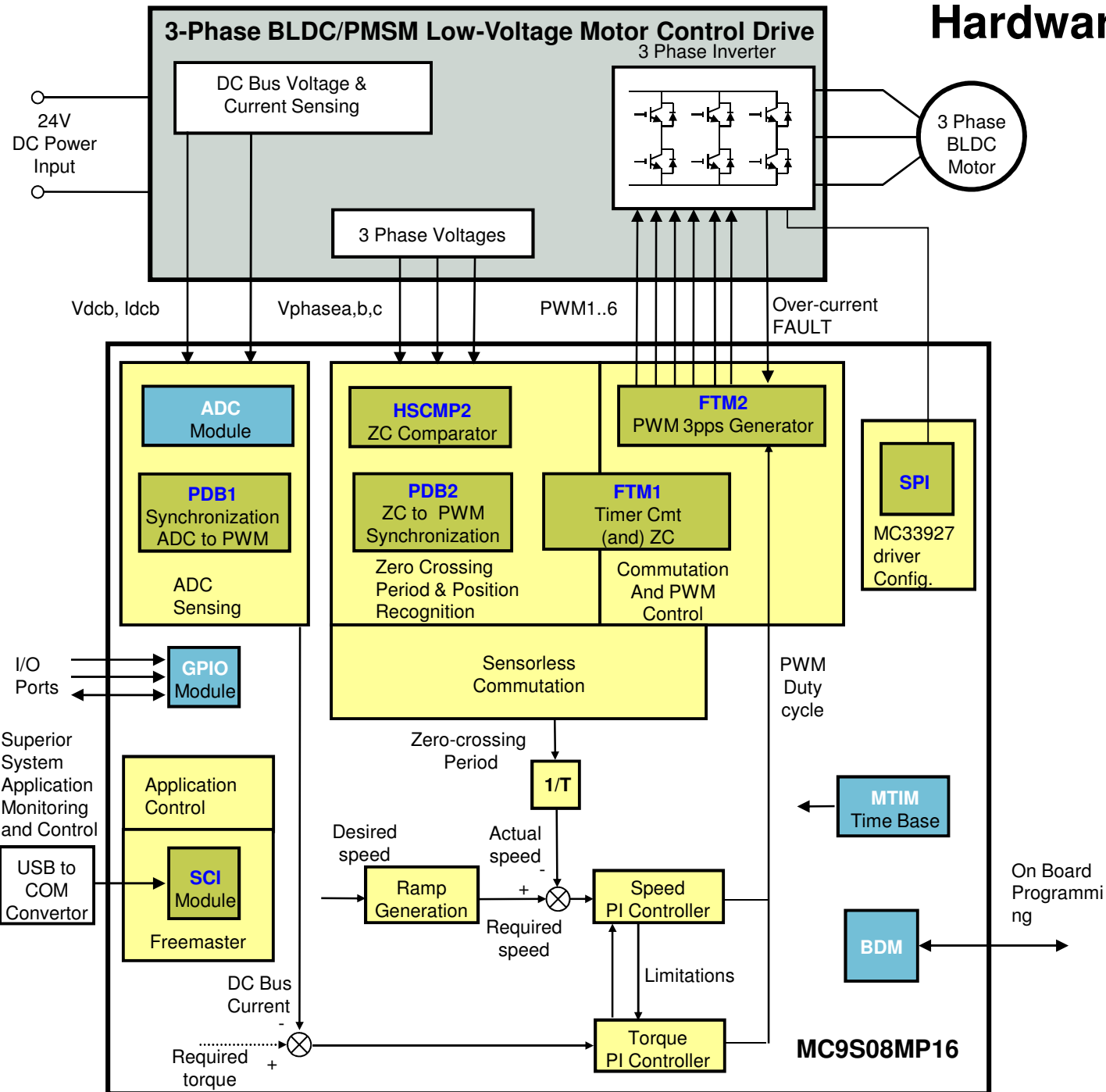
3
X



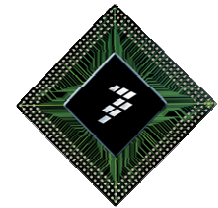


Hardware Overview

Hardware Overview



BLDC – Recommended Application: DRM117

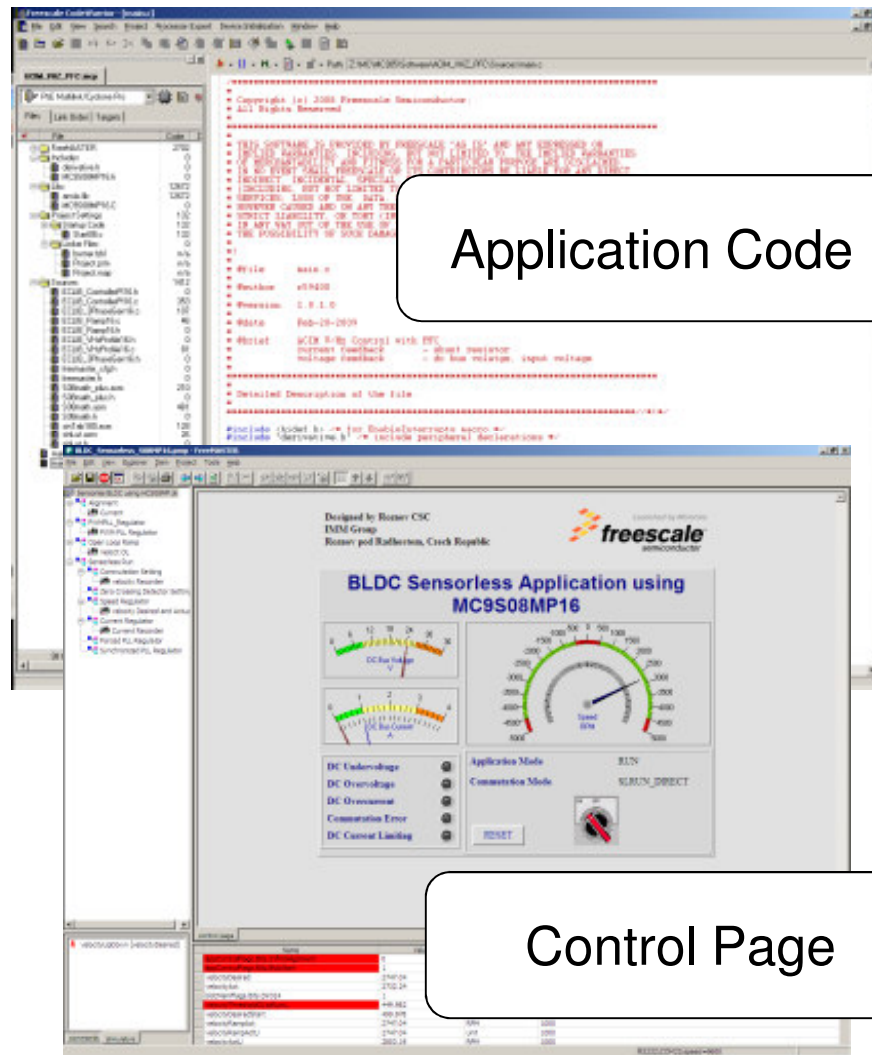


DRM117: 3-phase Sensorless BLDC Motor Control using MC9S08MP16.

- ▶ 3-phase trapezoidal BLDC motor control with 6-step commutation (60, 120 degree control).
- ▶ MP16 in-built high-speed comparator (HSCMP) detects the back-EMF voltage zero-crossing.
- ▶ **Three** sensorless synchronized commutation control algorithms incorporating:
 - Commutation instant calculated directly from the period between two back-EMF zero-crossings
— **Direct Drive.**
 - Commutation period synchronized with the back-EMF zero-crossing using a closed-loop according to a phase error
— **Synchronized PLL.**
 - Constant commutation period forced with the motor voltage controlled in a closed-loop according to a phase error
— **Forced PLL.**
- ▶ **Controlled acceleration and deceleration.**
- ▶ **Bidirectional** rotation.
- ▶ Both motor and generator modes. (4 Quadrant operation)
- ▶ Two PWM techniques possible: **Unipolar or Bipolar.**
- ▶ Software over-voltage and under-voltage protection.
- ▶ Hardware over-current protection.
- ▶ **FreeMASTER** control interface (start/stop, speed setup).
- ▶ **FreeMASTER** software monitor.
 - **FreeMASTER** software graphical control page (required speed, actual motor speed, start/stop status, DC-Bus voltage level, motor current, system status).
 - **FreeMASTER** software speed scope.



3-Phase Sensorless BLDC Using MC9S08MP16

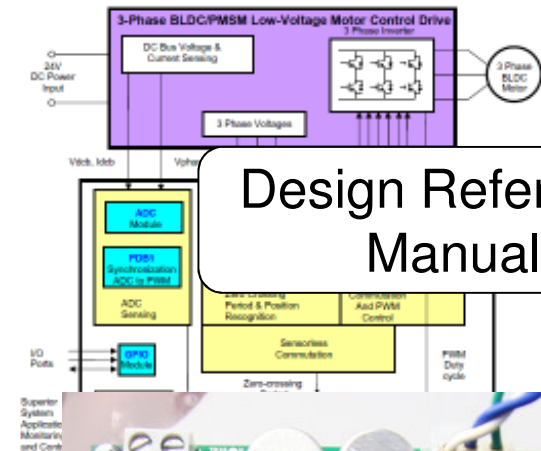


Application Code

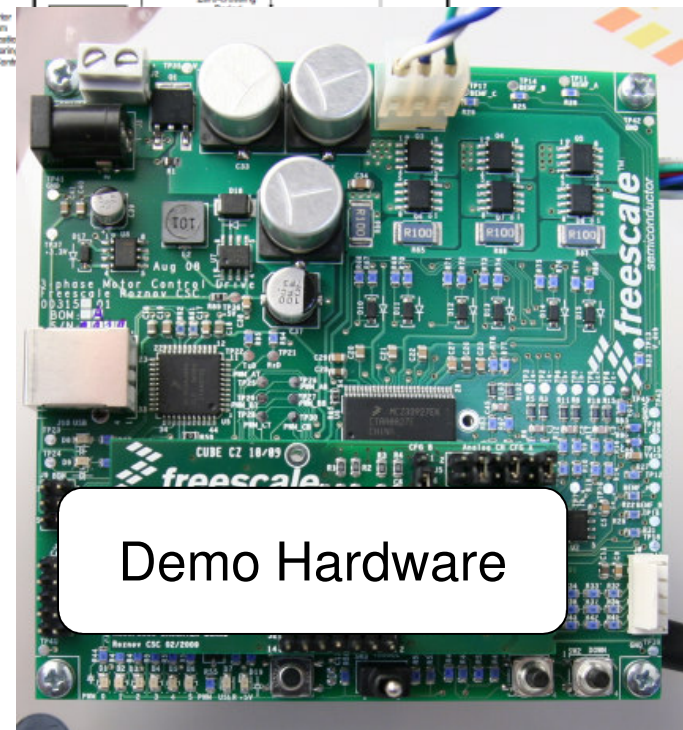
Control Page

3.3 3-phase Sensorless BLDC Drive using S08MP16

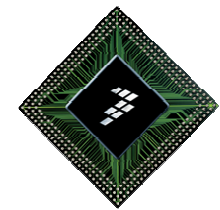
Figure 3-1 shows the system functional blocks and their subblocks. Most of the subblocks are implemented with the HCS08MP16 modules.



Design Reference Manual

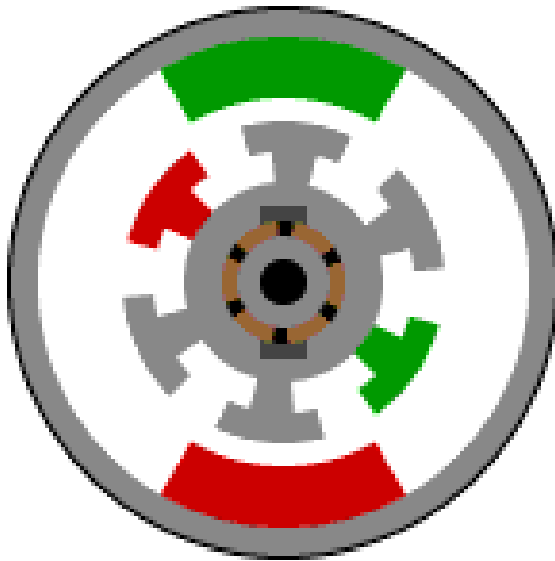


Demo Hardware

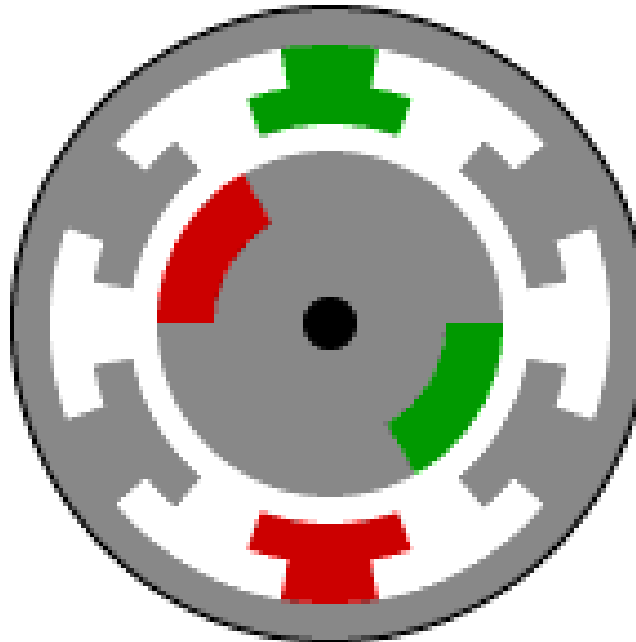


Why BLDC with MP16?

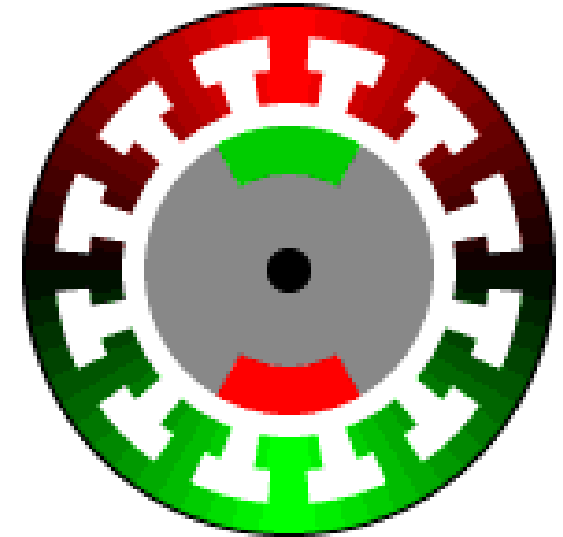
Motor Types



DC Motor



Brushless DC Motor
(BLDC)



Permanent Magnet
Synchronous Motor
(PMSM)

Brushed and Brushless Motors Comparison

Feature	Brushed DC motor	BLDC Motor
Commutation	+ Brushed commutation	- Electronic commutation
Maintenance	- Periodic maintenance is required	+ Less required due to absence of brushless
Noise/EMI	- Higher	+ Lower
Life	- Shorter	+ Longer
Speed/Torque	- Moderately Flat. Higher speeds produces higher friction and this reduces torque.	+ Flat
Speed Range	- Lower – Mechanical limitations by the brushes	+ Higher – No mechanical limitation
Acceleration	- Lower	+ Higher (Permanent Magnet rotor has less inertia)
Torque/Size Ratio	- Lower	+ Higher
Building Cost	+ Lower	- Higher – Permanent magnets
Control	+ Simple	- Complex and expensive
Control Requirements	+ A controller is required only when variable speed is desired	- A controller is always required

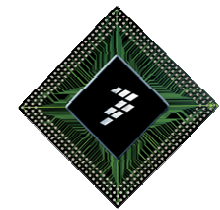
Six step control versus sinusoidal control

Six step control	Sinusoidal control
+ Simple PWM generation	– More complex PWM generation (sinewave has to be generated)
+ Simple MCU with the right peripherals can be used.	- DSP is required for calculations (Clark, Park, BEMF estimator)
– Ripple in the torque (stator flux jumps by 60°)	+ Smooth torque (stator flux rotates fluently)
– A little noise operation (due to ripple in the torque)	+ Very quiet
+ Simple sensor	– Requires sensor with high resolution
+ Direct measurement of BEMF Voltage	– BEMF estimation with observer

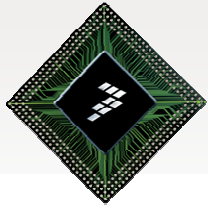
Why MP16 ?

- ▶ Compared with DSPs / DSCs
 - Easier to understand if you have experience with 8-bits.
 - Easy to port applications made for 8-bits MCU.
 - Cheaper.
 - Cheaper.

- ▶ Compared with regular 8-bits MCUs
 - Automatic Dead-time insertion.
 - Regular MCUs must use “software” PWM for dead-time insertion.
 - External or Internal Fault input.
 - Hardware turns off PWM outputs much faster.
 - LOAD_OK bit for PWM reload all channels at same time.
 - Avoids possible current glitches / short-circuit.
 - Better synchronization with PDB.
 - Much better control, lower CPU overhead.

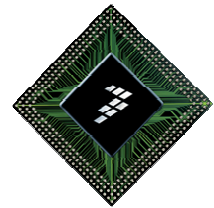


What do we know so far?



Summary

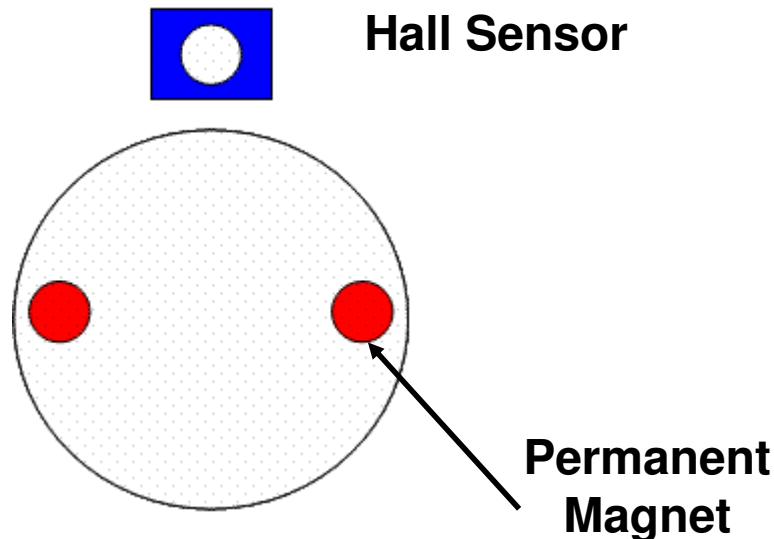
- ▶ BLDC motors provide a more **reliable** and **efficient** control than Brushed DC motors.
- ▶ Freescale's MC9S08MP16 MCU enables **low-cost** BLDC motor control:
 - ▶ Motor control **optimized** features: HSCMP, FTM, PDB.
 - ▶ Peripheral **integration**: lower system cost: PGA.
 - ▶ Lower **price** point than DSPs.
- ▶ Freescale provides a rich **enablement** set:
 - ▶ Reference **design**, with software, hardware and full documentation.
 - ▶ **FreeMaster**, **CodeWarrior** and development board.
 - ▶ Application notes, Data Sheets, Reference Manuals.



Controlling a BLDC Motor with Sensors

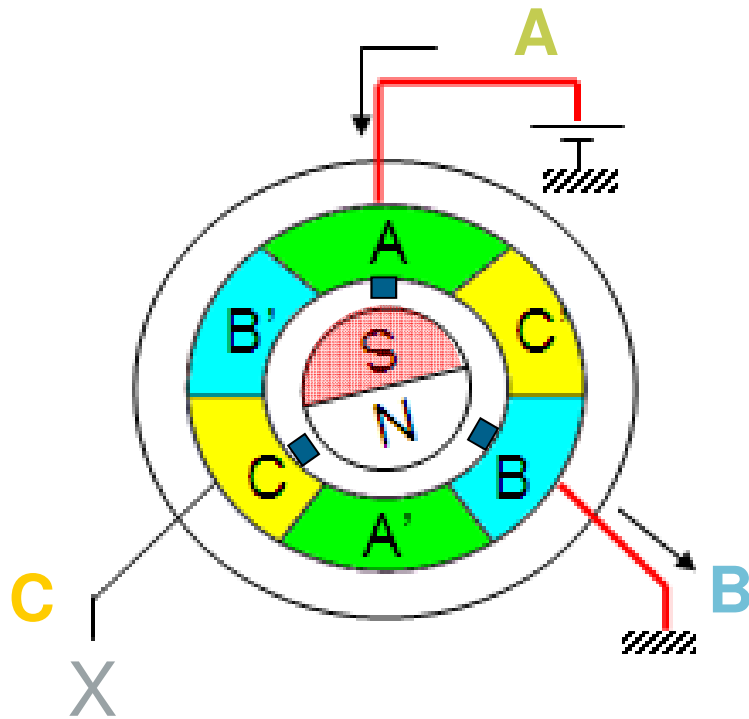
Sensor Example: Hall Effect Sensor

- ▶ A transducer that varies its output voltage in response to changes in magnetic field
- ▶ Used for proximity switching, positioning, speed detection and current sensing applications
- ▶ In motor control applications they are used as *on/off* switches

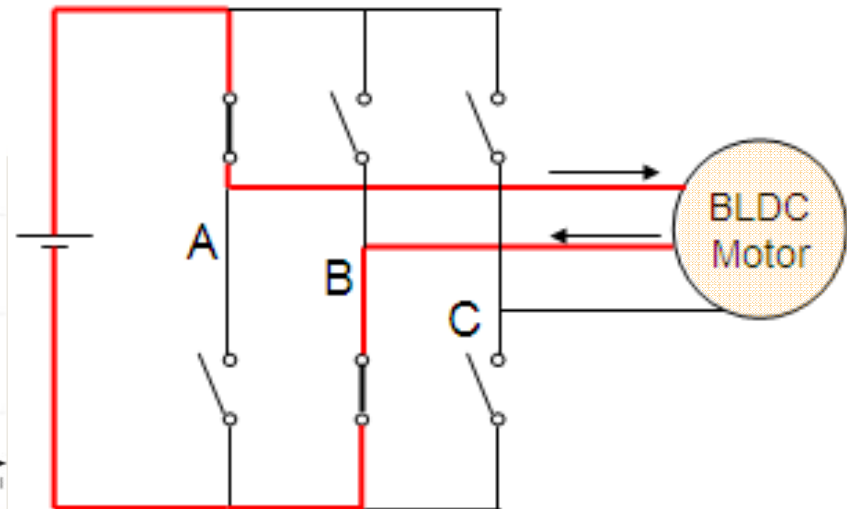
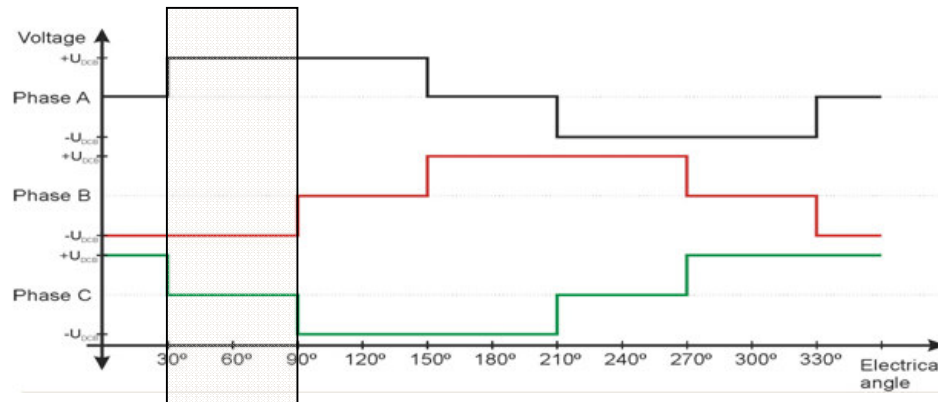


Every time a magnetic field is sensed, a change in voltage can be detected

Control of 3-Phase Inverter

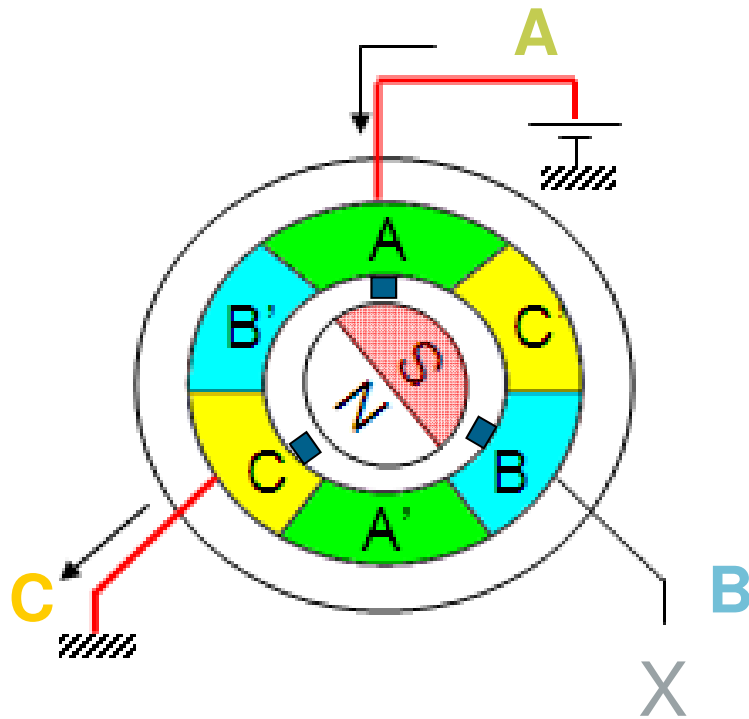


Phase A	Phase B	Phase C	Hall Sensors		
			A	B	C
$+V_{DCB}$	$-V_{DCB}$	NC	1	0	0
$+V_{DCB}$	NC	$-V_{DCB}$	1	1	0
NC	$+V_{DCB}$	$-V_{DCB}$	0	1	0
$-V_{DCB}$	$+V_{DCB}$	NC	0	1	1
$-V_{DCB}$	NC	$+V_{DCB}$	0	0	1
NC	$-V_{DCB}$	$+V_{DCB}$	1	0	1

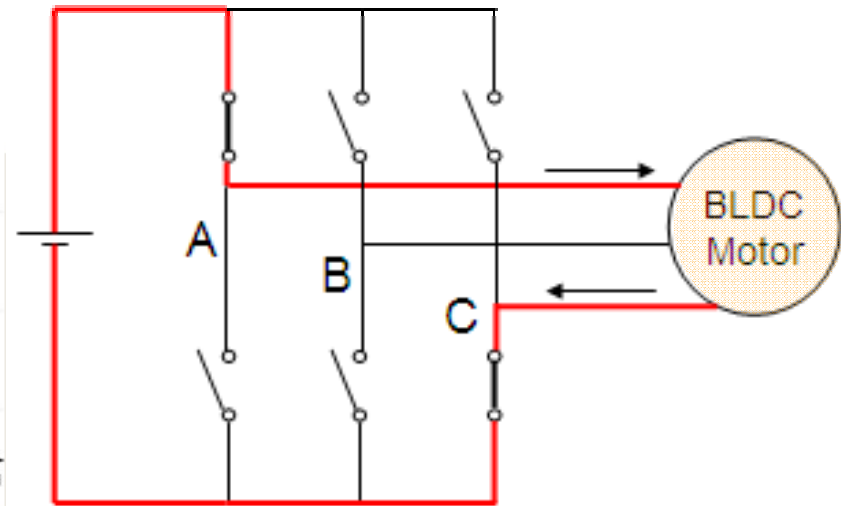
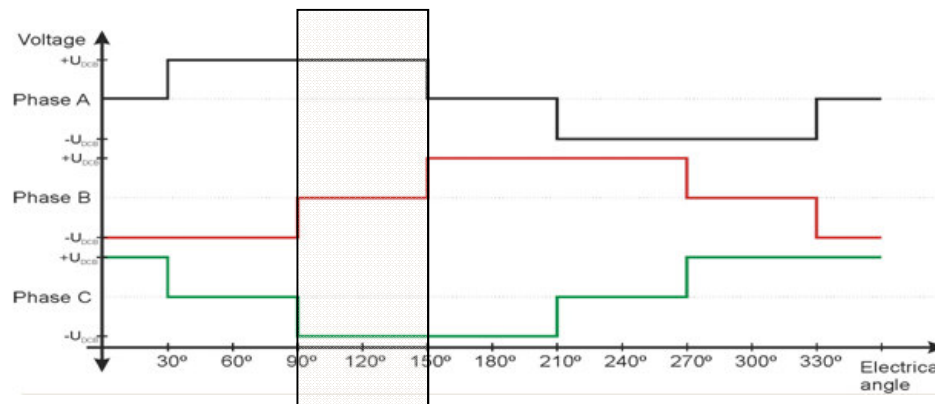


Animation part 1/6

Control of 3-Phase Inverter

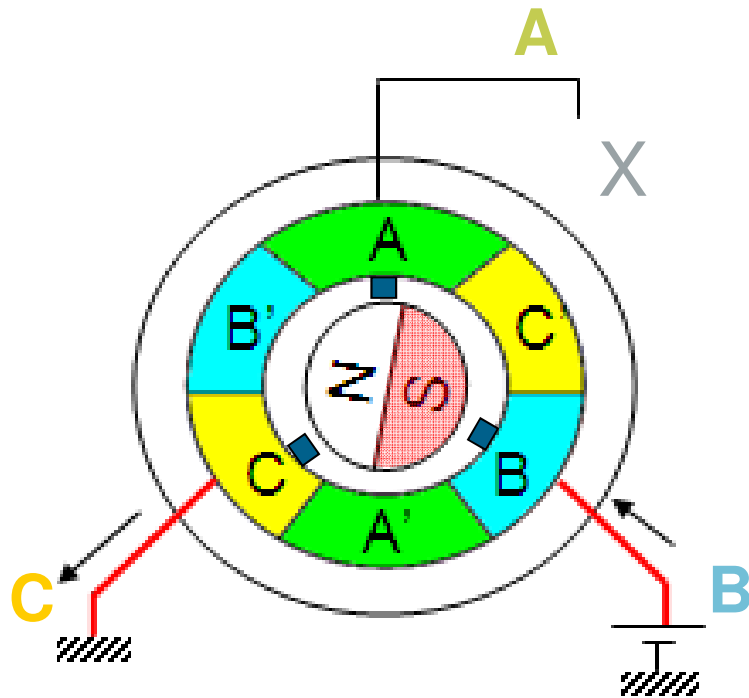


Phase A	Phase B	Phase C	Hall Sensors		
			A	B	C
$+V_{DCB}$	$-V_{DCB}$	NC	1	0	0
$+V_{DCB}$	NC	$-V_{DCB}$	1	1	0
NC	$+V_{DCB}$	$-V_{DCB}$	0	1	0
$-V_{DCB}$	$+V_{DCB}$	NC	0	1	1
$-V_{DCB}$	NC	$+V_{DCB}$	0	0	1
NC	$-V_{DCB}$	$+V_{DCB}$	1	0	1

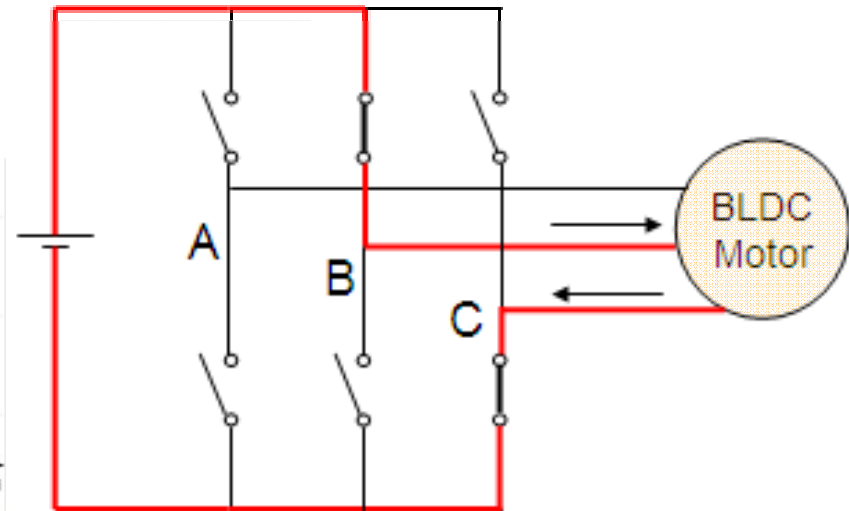
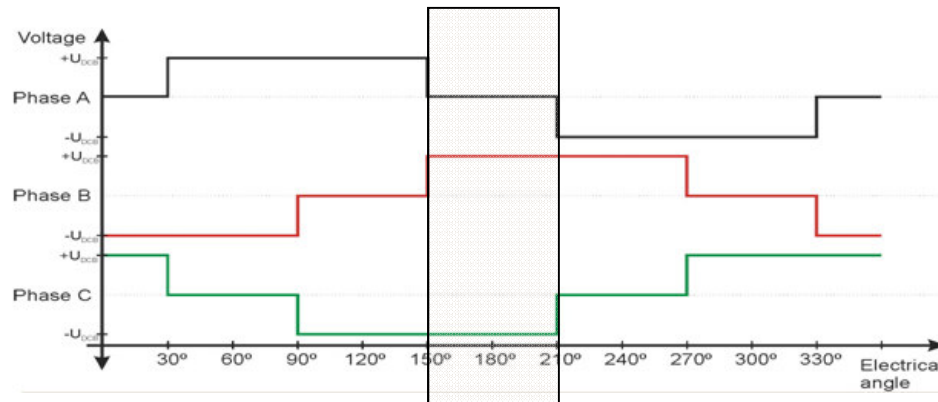


Animation part 2/6

Control of 3-Phase Inverter

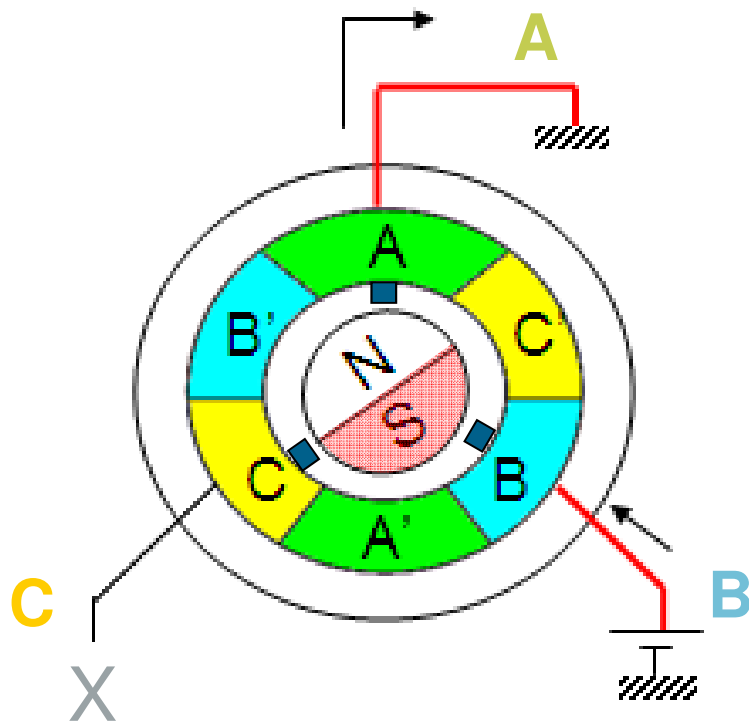


Phase A	Phase B	Phase C	Hall Sensors		
			A	B	C
$+V_{DCB}$	$-V_{DCB}$	NC	1	0	0
$+V_{DCB}$	NC	$-V_{DCB}$	1	1	0
NC	$+V_{DCB}$	$-V_{DCB}$	0	1	0
$-V_{DCB}$	$+V_{DCB}$	NC	0	1	1
$-V_{DCB}$	NC	$+V_{DCB}$	0	0	1
NC	$-V_{DCB}$	$+V_{DCB}$	1	0	1

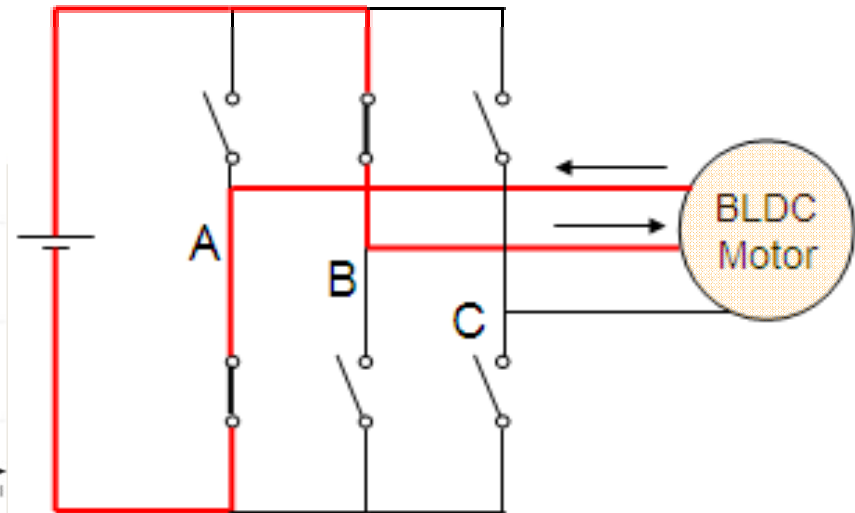
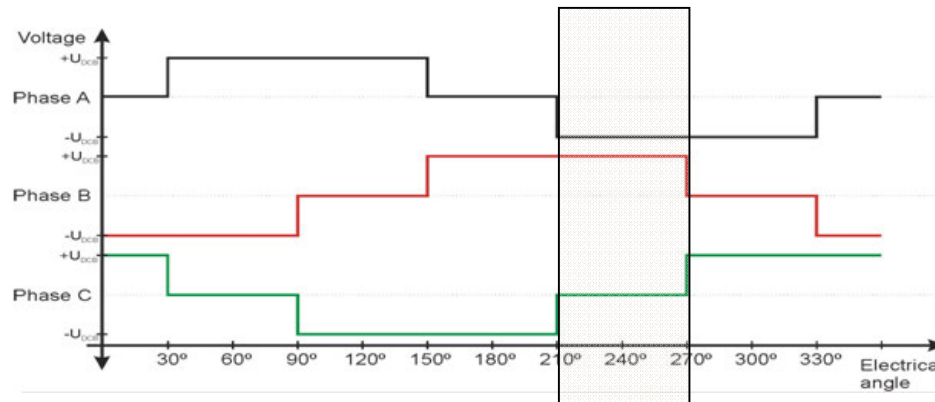


Animation part 3/6

Control of 3-Phase Inverter

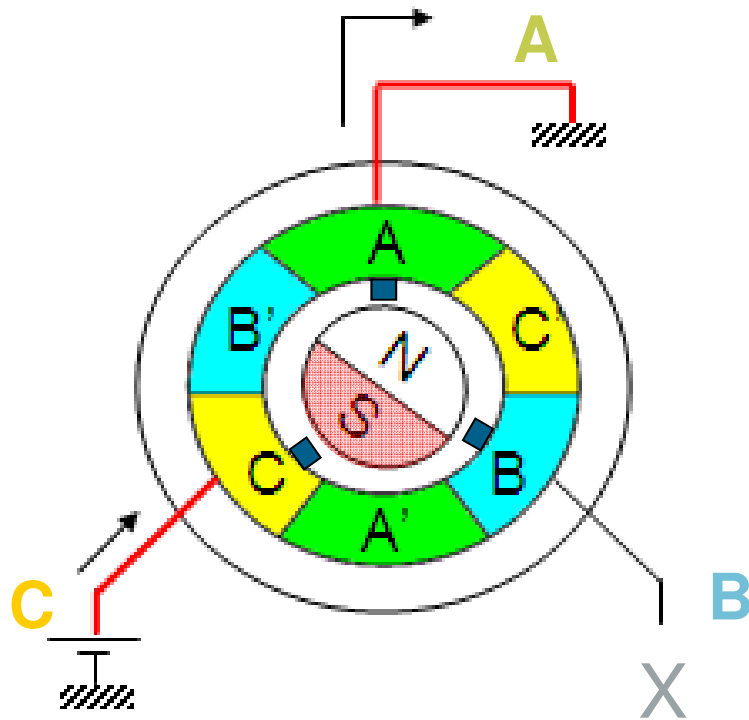


Phase A	Phase B	Phase C	Hall Sensors		
			A	B	C
$+V_{DCB}$	$-V_{DCB}$	NC	1	0	0
$+V_{DCB}$	NC	$-V_{DCB}$	1	1	0
NC	$+V_{DCB}$	$-V_{DCB}$	0	1	0
$-V_{DCB}$	$+V_{DCB}$	NC	0	1	1
$-V_{DCB}$	NC	$+V_{DCB}$	0	0	1
NC	$-V_{DCB}$	$+V_{DCB}$	1	0	1

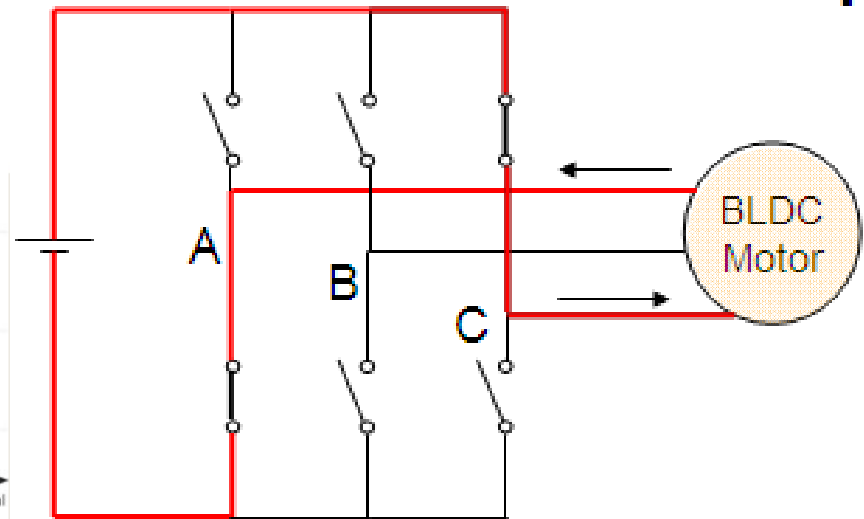
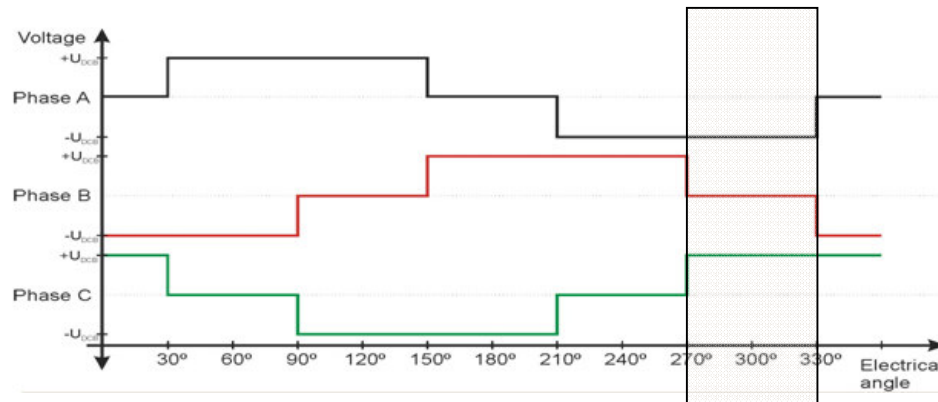


Animation part 4/6

Control of 3-Phase Inverter

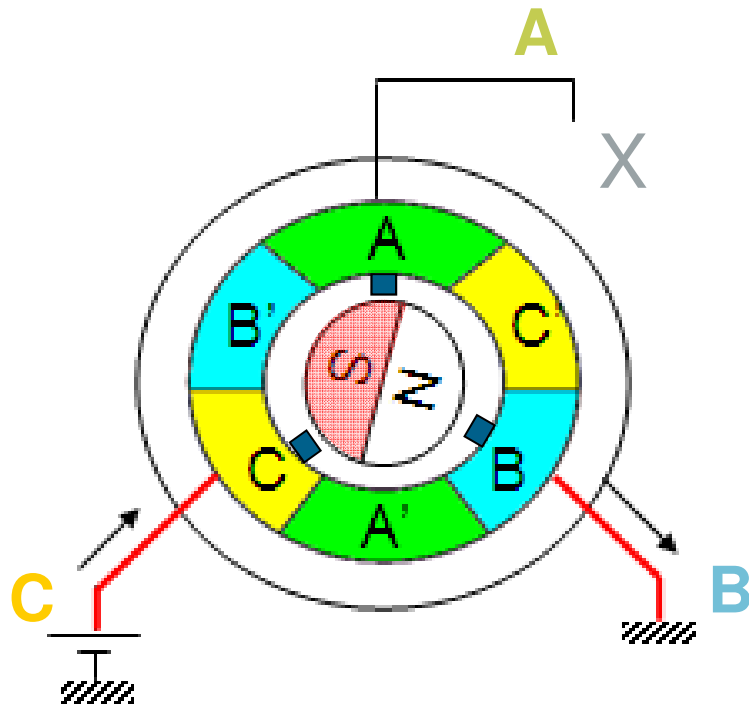


Phase A	Phase B	Phase C	Hall Sensors		
			A	B	C
$+V_{DCB}$	$-V_{DCB}$	NC	1	0	0
$+V_{DCB}$	NC	$-V_{DCB}$	1	1	0
NC	$+V_{DCB}$	$-V_{DCB}$	0	1	0
$-V_{DCB}$	$+V_{DCB}$	NC	0	1	1
$-V_{DCB}$	NC	$+V_{DCB}$	0	0	1
NC	$-V_{DCB}$	$+V_{DCB}$	1	0	1

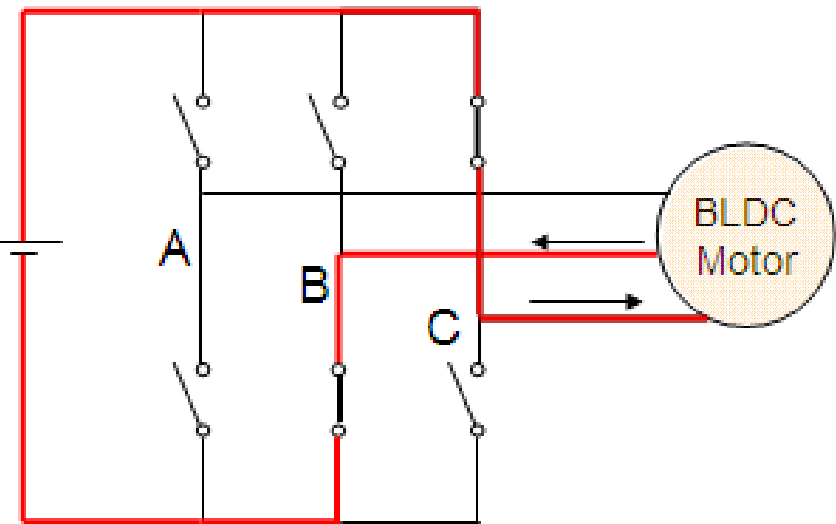
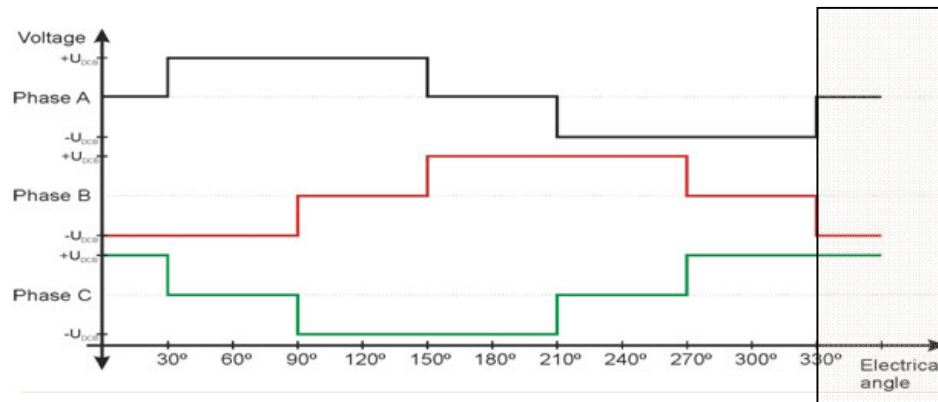


Animation part 5/6

Control of 3-Phase Inverter

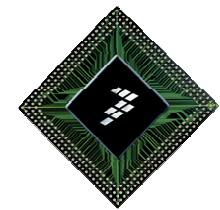


Phase A	Phase B	Phase C	Hall Sensors		
			A	B	C
$+V_{DCB}$	$-V_{DCB}$	NC	1	0	0
$+V_{DCB}$	NC	$-V_{DCB}$	1	1	0
NC	$+V_{DCB}$	$-V_{DCB}$	0	1	0
$-V_{DCB}$	$+V_{DCB}$	NC	0	1	1
$-V_{DCB}$	NC	$+V_{DCB}$	0	0	1
NC	$-V_{DCB}$	$+V_{DCB}$	1	0	1

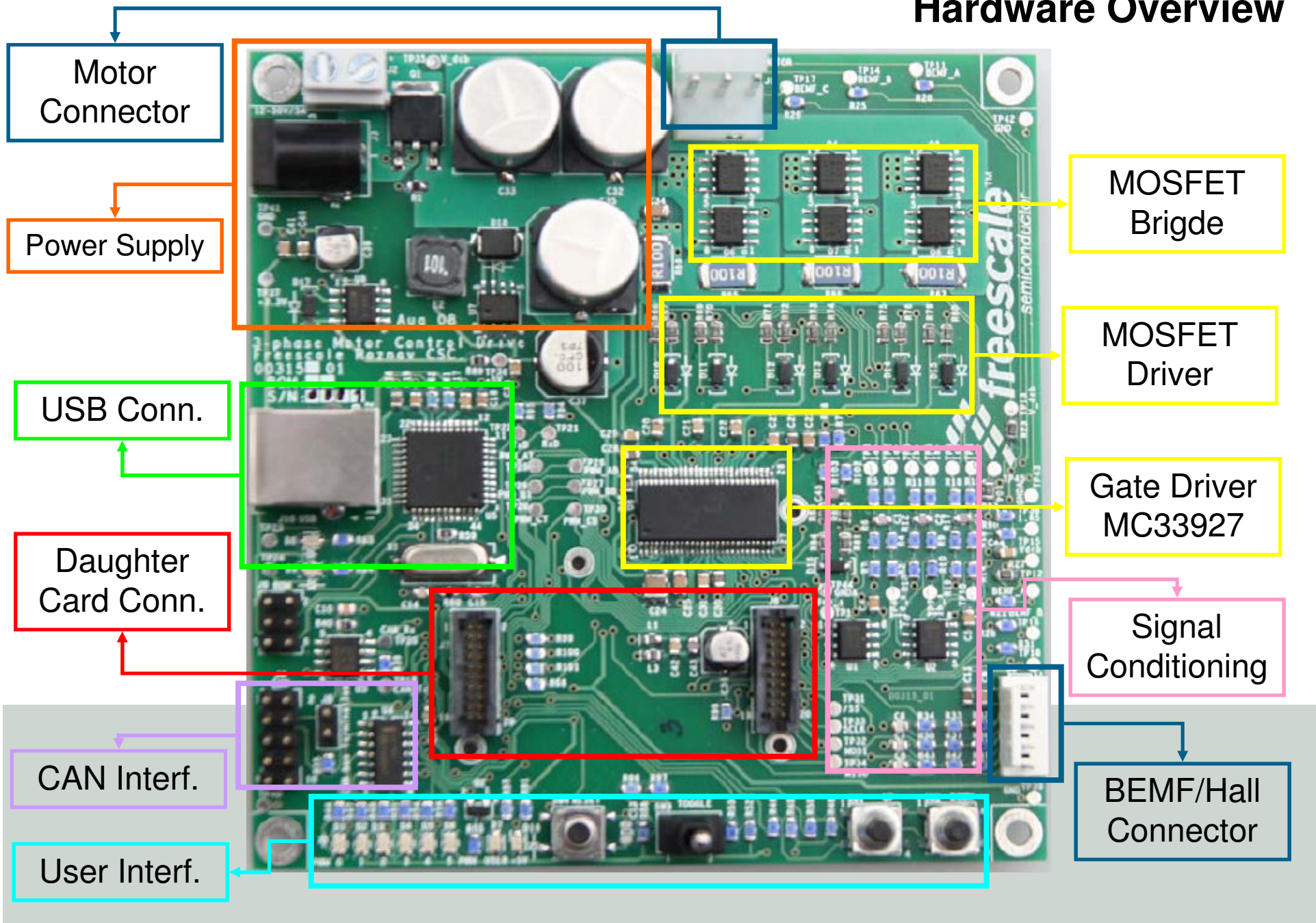


Animation part 6/6

Mini Hands-on: How Sensors Behave (Lab 0)

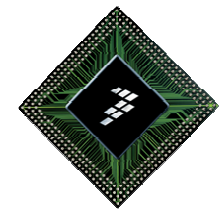


Hardware Overview



Read Sensors Show Their Values

1. Open *CodeWarrior for MCU version 6.3*
2. Click on *File > Open*
3. Browse for the *Labs* folder
4. Open *lab0_bldctraining.mcp*
5. Click *F5* or the *debug* symbol 
6. Wait for the debug prompt to open, then click *Ok* to flash the device.
7. Make sure the encoder/sensor connector is properly plugged
8. Wait for the programming to finish and click the *Run* button (or *F5*) when finished
9. Observe the LED in the board
10. Rotate the motor manually and see how the LED change with each step of the motor

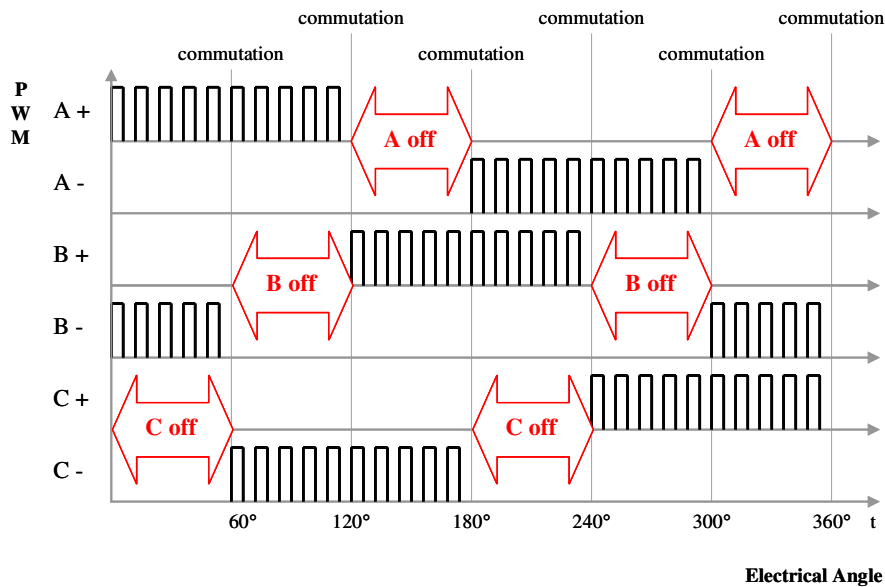


Electronic Motor Commutation

PWM Modes used for BLDC Motor Operation

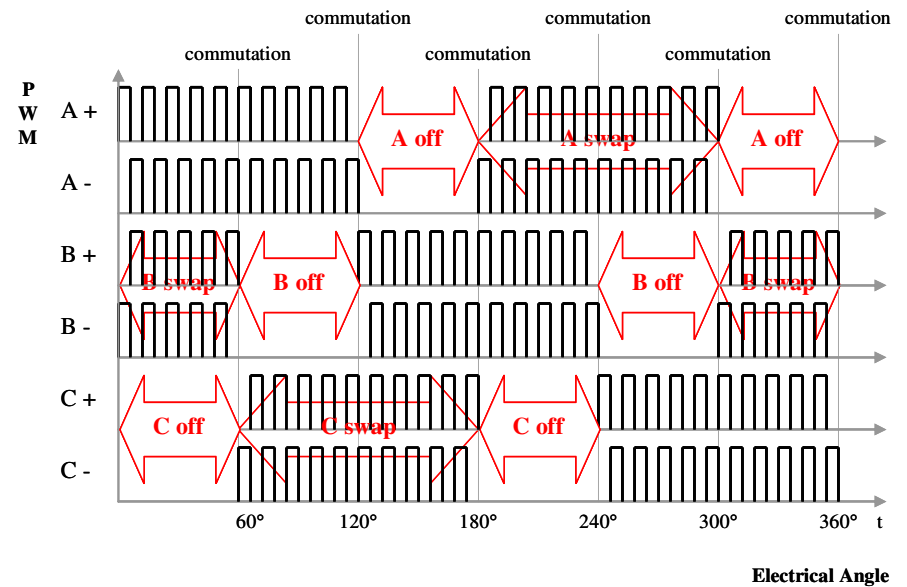
Independent Mode

- ▶ Simple implementation
- ▶ Masking of PWM channel required

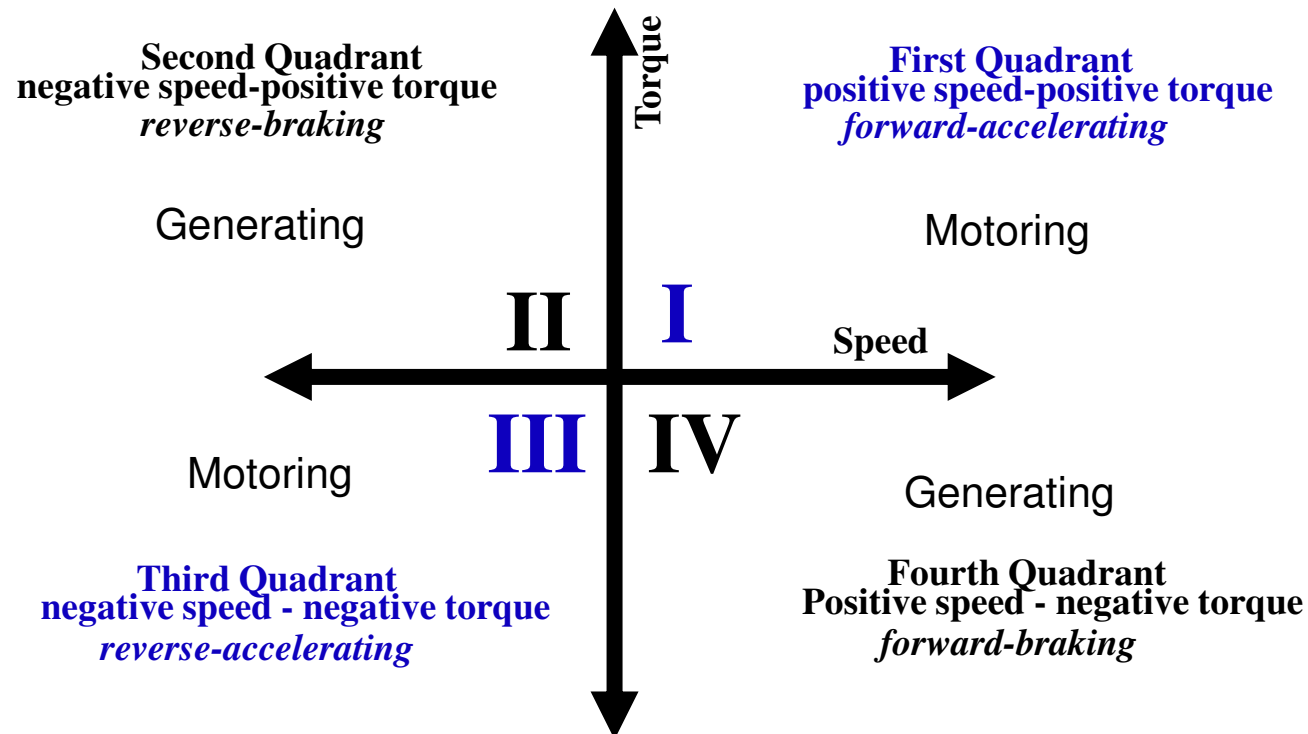


Complementary Mode

- ▶ Required sophisticated PWM support
- ▶ Mask and Swap of PWM channel required
- ▶ Allows energy recuperation

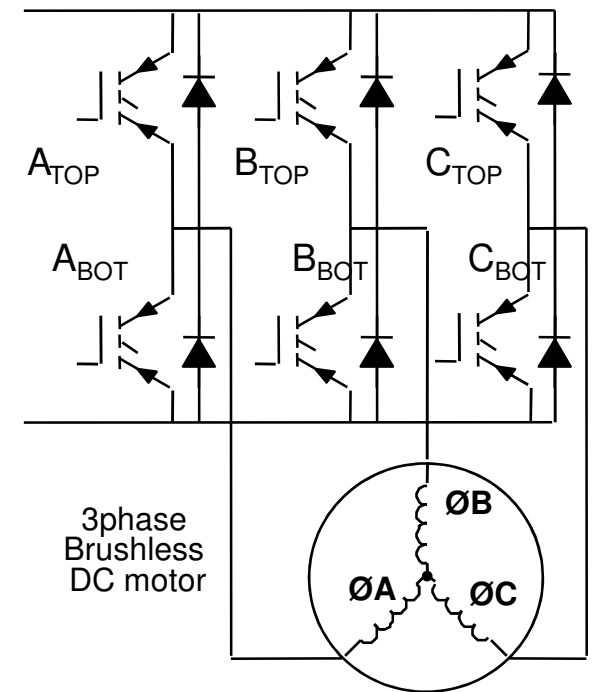
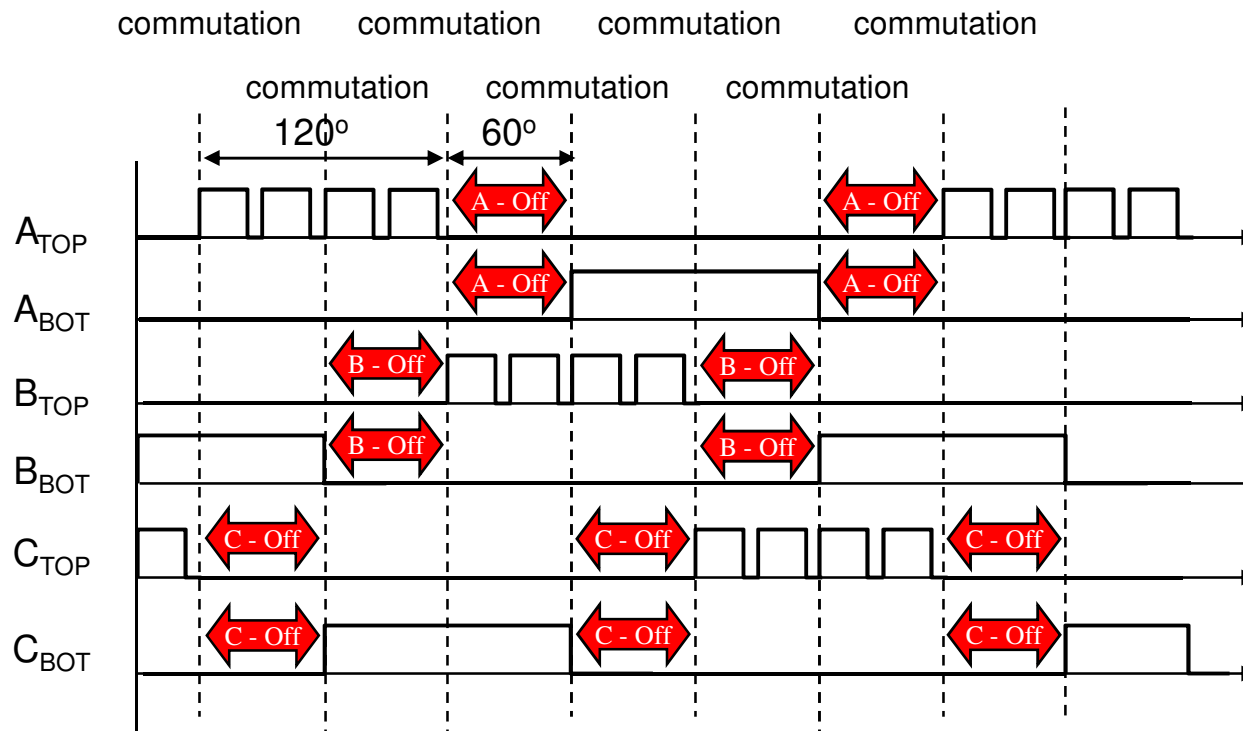


Quadrants of Operation



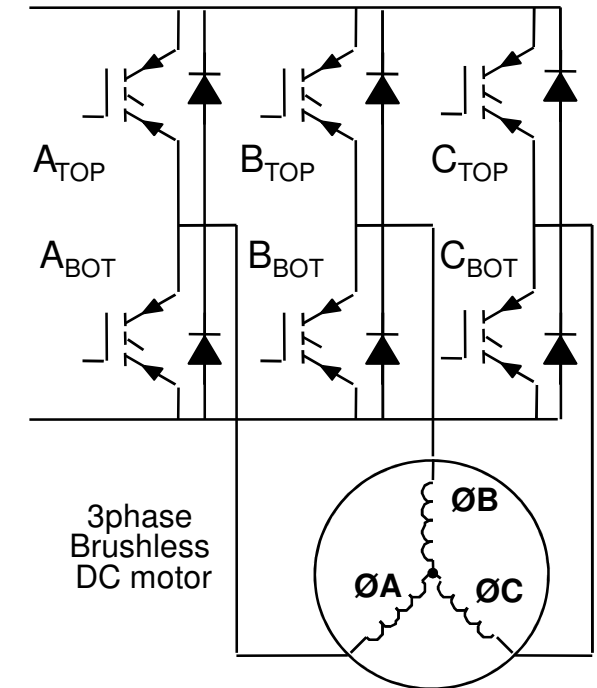
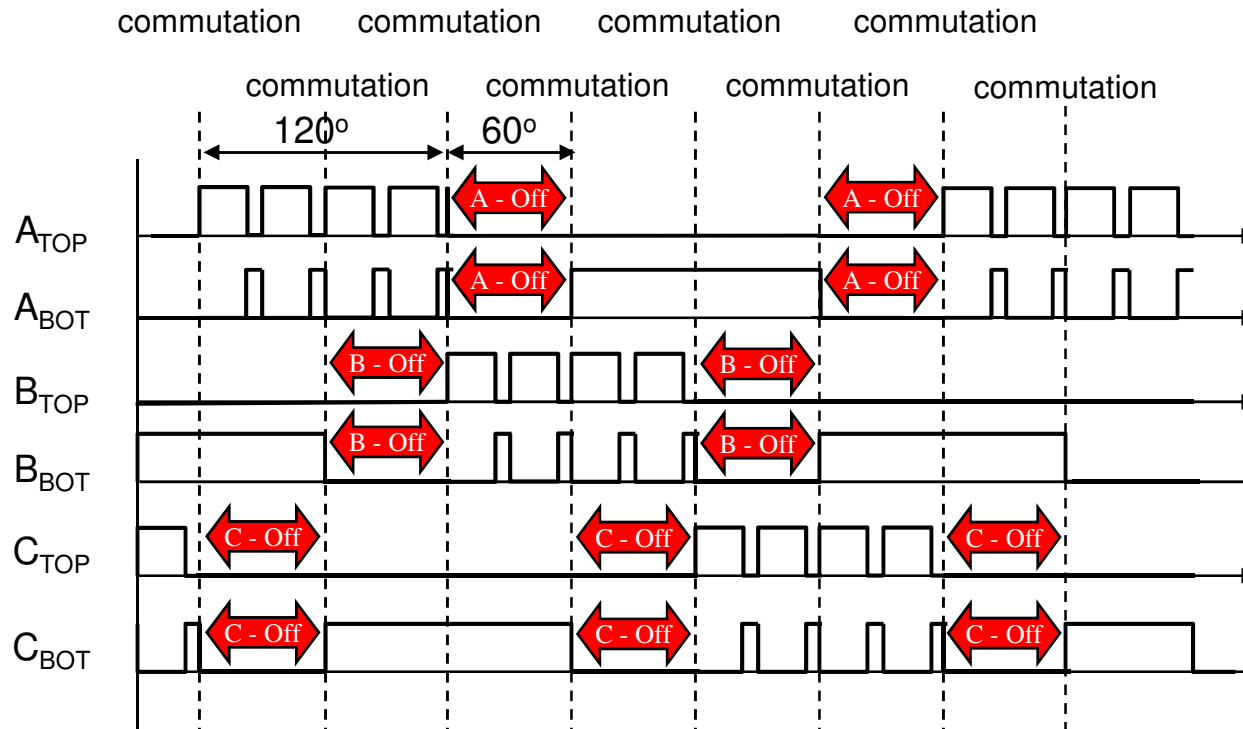
Unipolar BLDC Commutation

- ▶ No energy recuperation
- ▶ Simple implementation



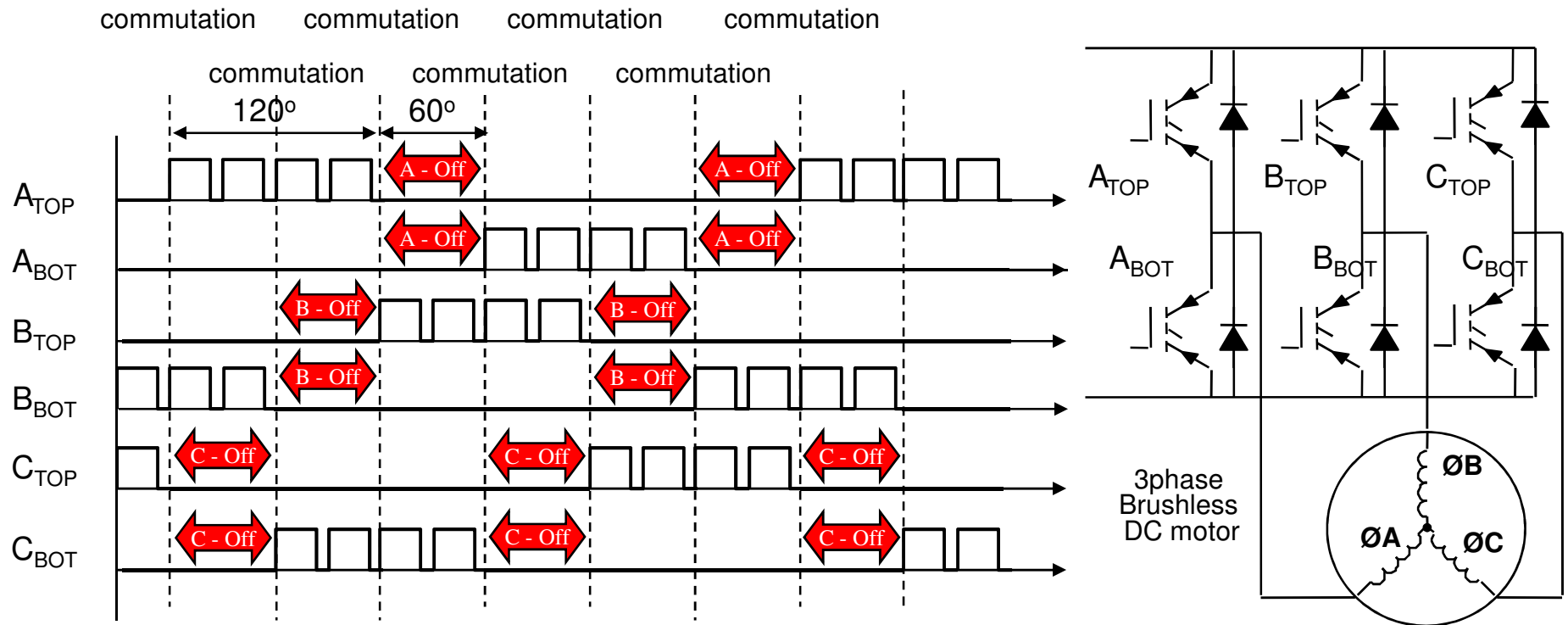
Unipolar BLDC Commutation (3 Complementary PWM pairs)

- ▶ Allows energy recuperation
- ▶ *Bottom* is opposite of *Top*
- ▶ Requires sophisticated PWM support



Bipolar BLDC Commutation (6 independent PWMs)

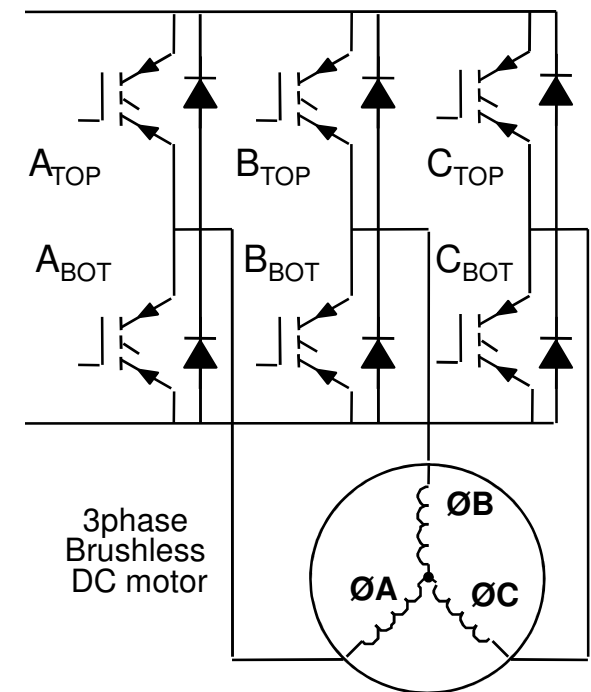
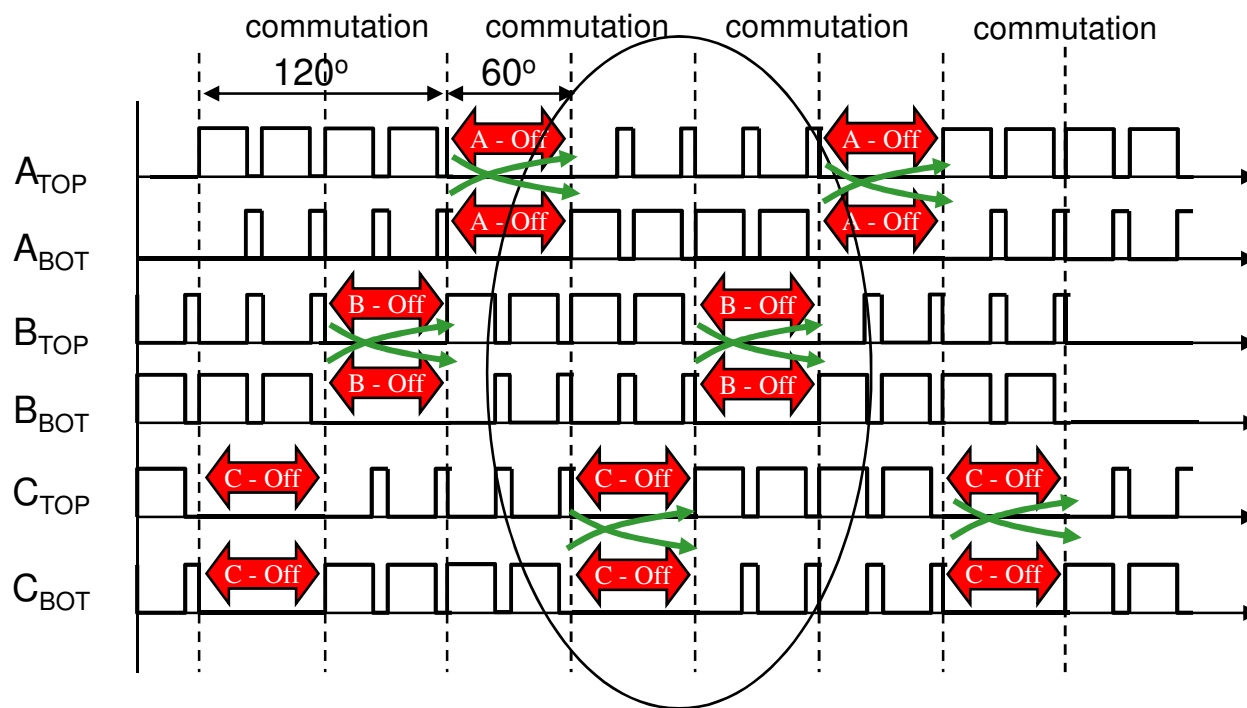
- ▶ No energy recuperation
- ▶ Simple implementation when 6 PWM available



Bipolar BLDC Commutation (3 Complementary PWM pairs)

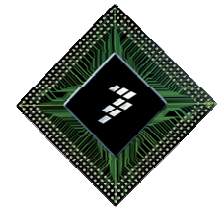
- ▶ Allows energy recuperation
- ▶ *Bottom* is opposite of *Top*
- ▶ Requires sophisticated PWM support

commutation commutation commutation commutation



Swap

Hands-on: Run the BLDC with Sensors Demo (Lab 1)



MC9S08MP16/12

Core / Temp / Package

► Industrial Version:

- 50MHz (25MHz bus), -40to105C, 28SOIC, 32LQFP, 48LQFP

► Automotive Version:

- 40MHz (20MHz bus), -40to125C, 48LQFP

► 2.7V to 5.5V operating range

Memory

► 16KB Flash / 1KB RAM

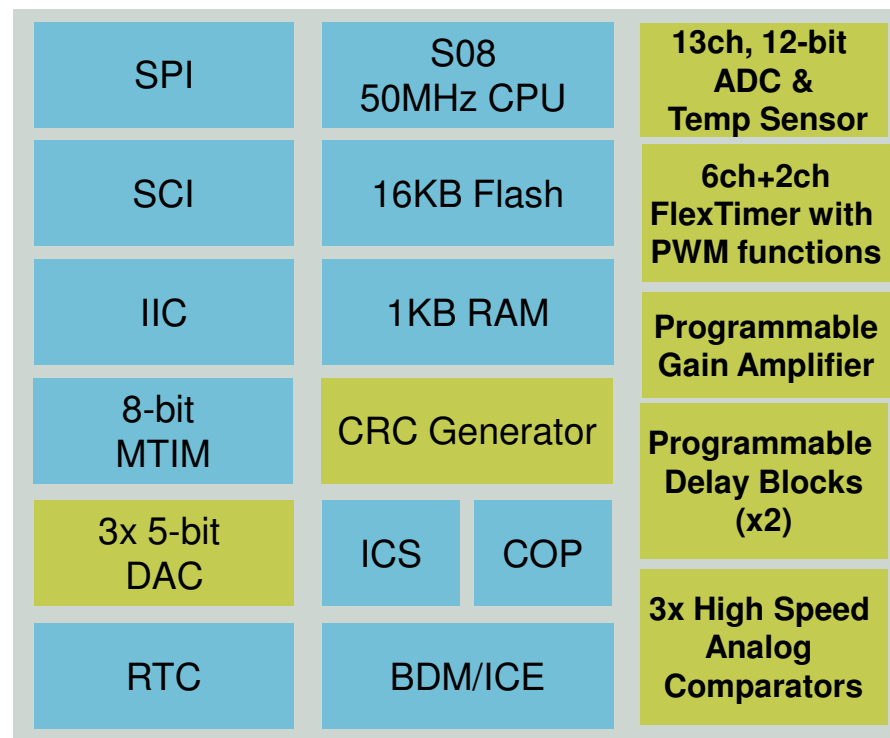
► 12KB Flash / 512B RAM

Features

- 2x FlexTimers (6ch + 2ch) – automatic fault protection, supports up to 50MHz CLK, selectable input capture, output compare, edge- or center-aligned PWM; dead time insertion
- 3 Analog Comparators – h/w sample trigger from PWM module allowing comparison at any point in cycle
- 2x Programmable Delay Blocks (PDB)
- 12-ch 12-bit ADC – 3.5 uS conversion, h/w trigger from PWM module allowing conversion at any point in cycle
- Programmable Gain Amplifier (PGA)
- 8-bit Modulo Timer Module (MTIM)
- LIN SCI, SPI, IIC
- 3x 5-bit DAC used as a 32 tap voltage reference
- RTC with periodical timer interval interrupt
- Software Programmable Internal Clock Source
- 3x low power modes & peripheral CLK gating
- Power Management Controller (PMC)
- 3x 8-bit KBI
- POR / LVI – supports 4 interrupt priority levels
- Background Debug Mode Interface/ICE

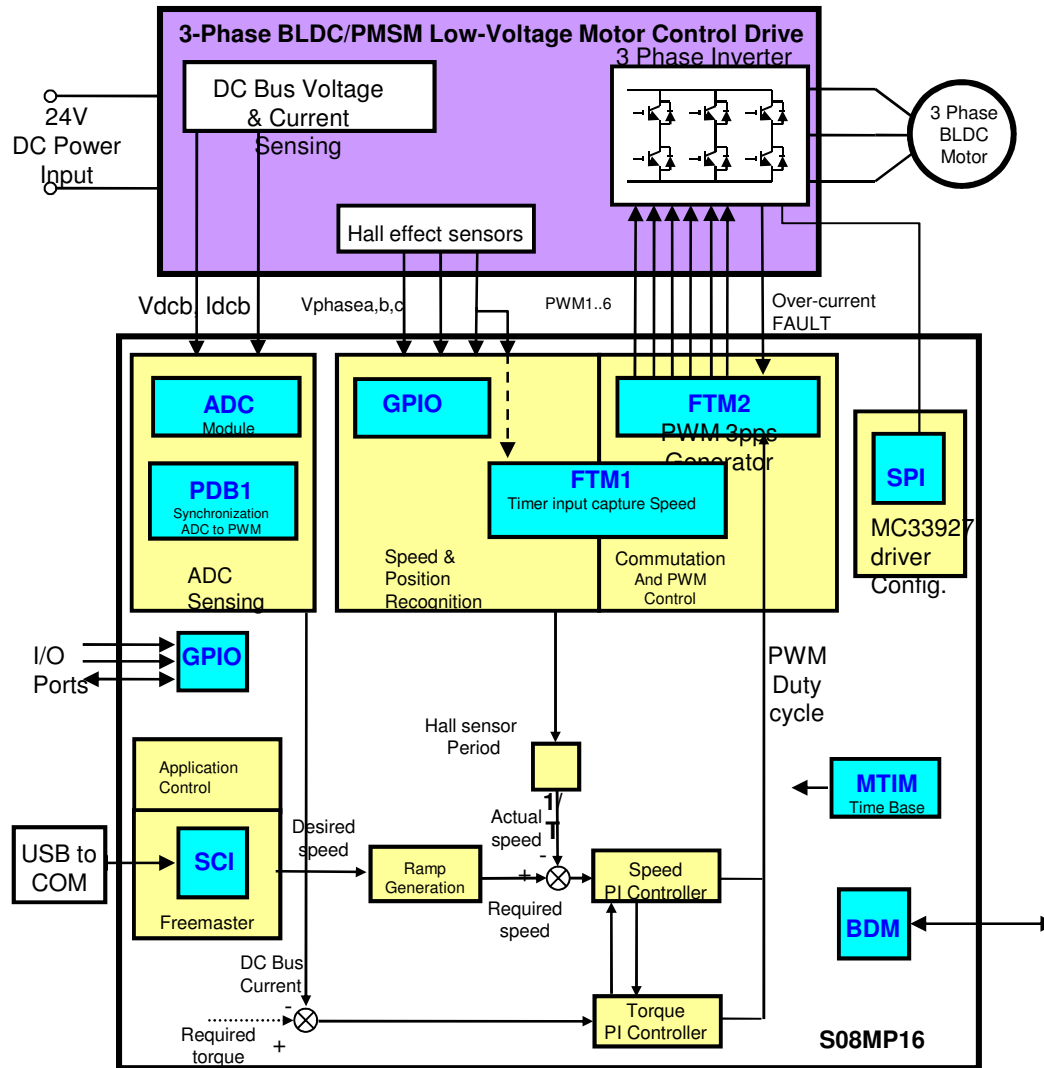
System Protection

- Cyclic Redundancy Check Generator (CRC)
- Watchdog Timer with Independent Clock Source



Device	Package	RAM	ADC	PGA	IIC	I/O
MC9S08MP16	48LQFP	1KB	13ch	1	1	40
MC9S08MP16	32LQFP	1KB	12ch	1	1	25
MC9S08MP16	28SOIC	1KB	8ch	1	1	22
MC9S08MP12	28SOIC	512B	8ch	-	-	22
S9S08M16E2MLF	48LQFP	1KB	13ch	1	1	40

BLDC with Sensors Block Diagram



Run the Demo

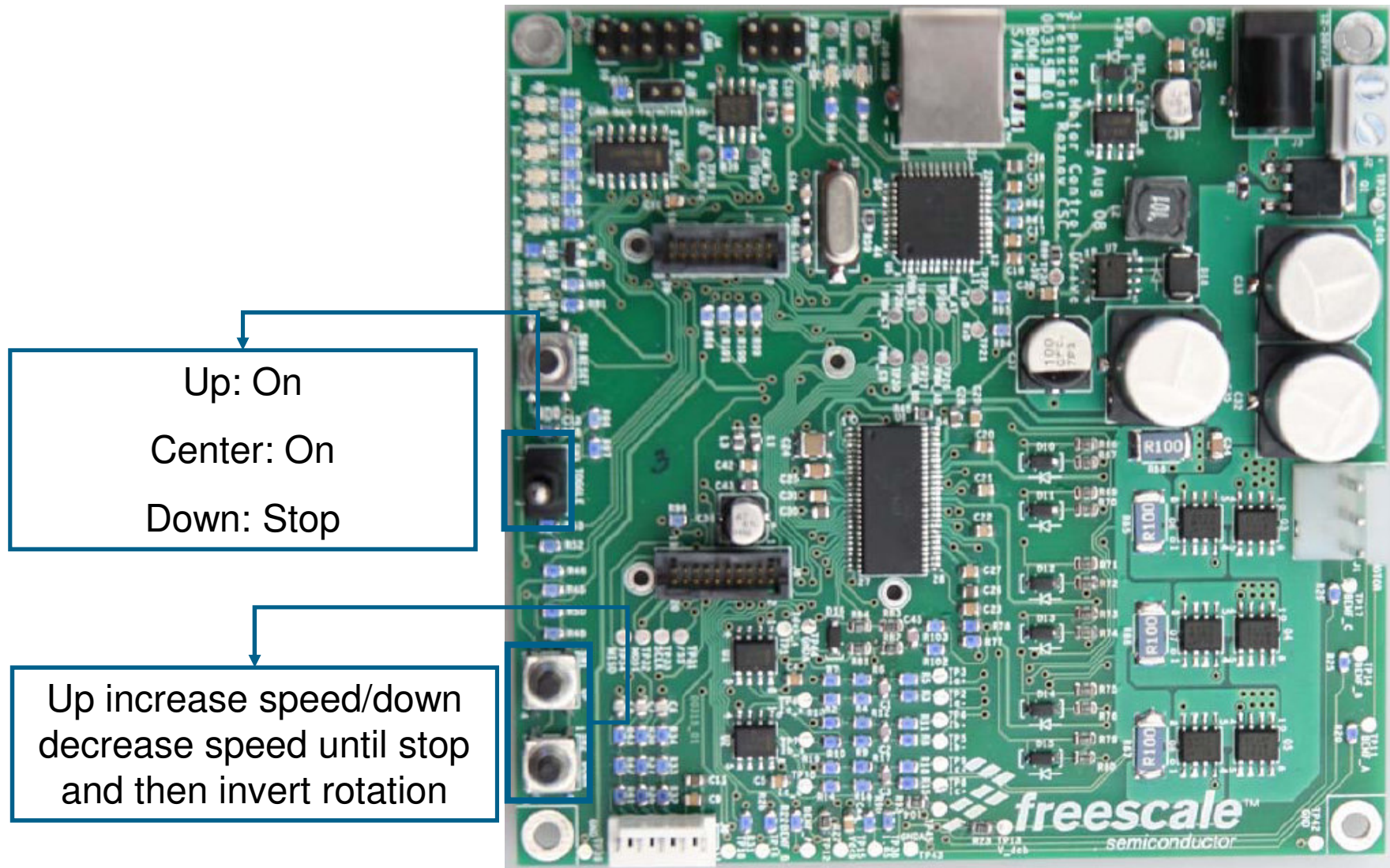
1. Open *CodeWarrior for MCU version 6.3*
2. Click *File > Open*
3. Browse for the *Labs* folder
4. Open *lab1_bldctraining.mcp*
5. Click *F5* or the *debug* symbol
6. Wait for the debug prompt to open and click *ok* to flash the device.
7. Make sure the encoder/sensor connector and the motor connector are properly plugged
8. Wait for the programming to finish and click the *run* button (or *F5*) when finished

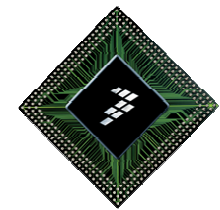
Run the Demo

1. Open *CodeWarrior* for *MCU* version 6.3
2. Click *File > Open*
3. Browse for the *Lab1*
4. Open *lab1_bldct*
5. Click *F5* or the *debug* button
6. Wait for the debug tool to load and click ok to flash the device.
7. Make sure the encoder and the motor connector are properly connected
8. Wait for the programming to finish and click the *run* button (or *F5*) when finished

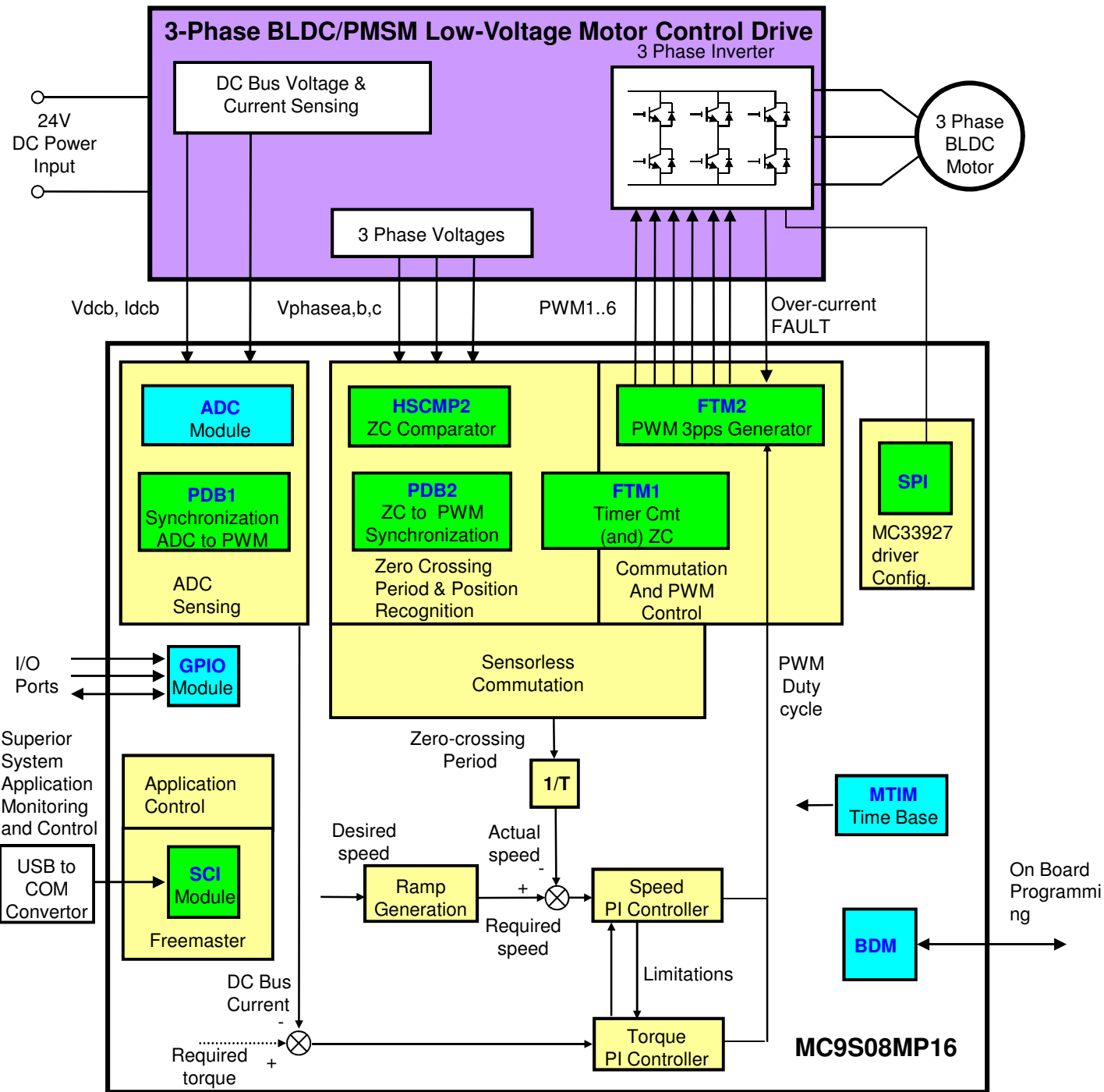
WAIT!!!

Before going forward make sure no obstacles are near the motor shaft ... specially your badges. We do not want you tangled up in there!!!





Hands-on: The BLDC Sensorless Demo

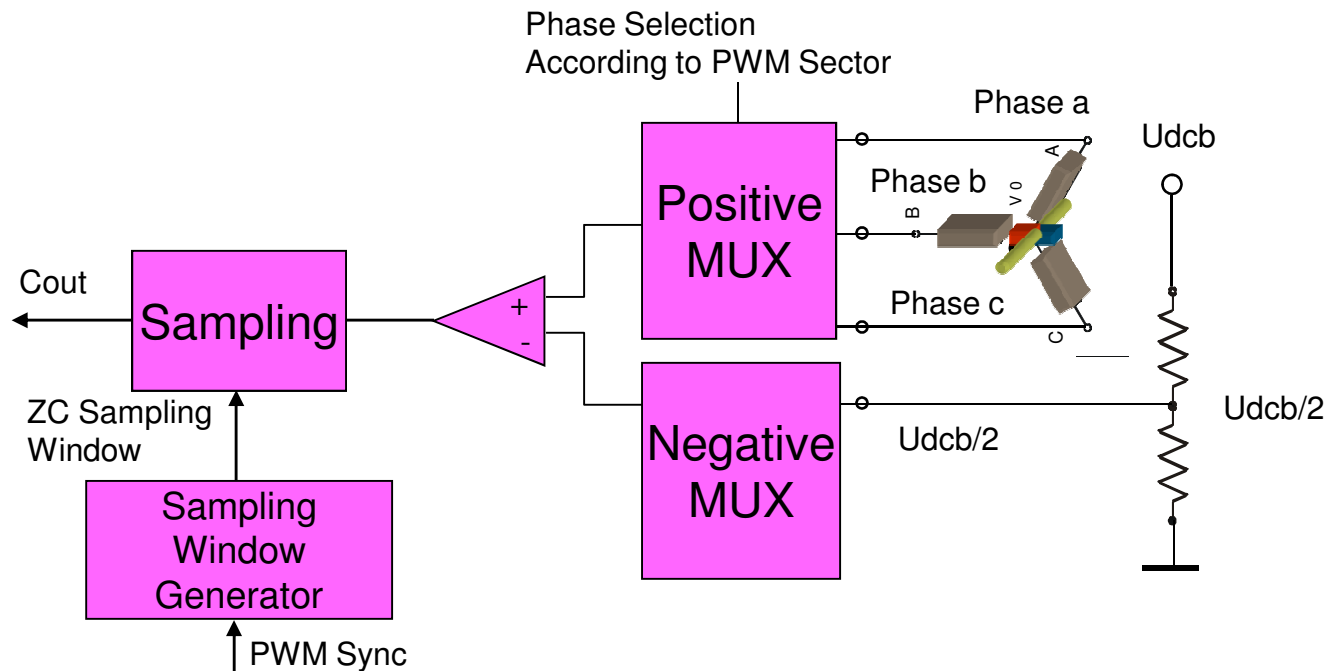


FTM

Similar Peripheral to MCF51AG 128

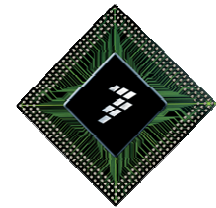
Back-EMF Zero-Crossing Sensing Circuit

► $\frac{1}{2} U_{DCB}$ reference

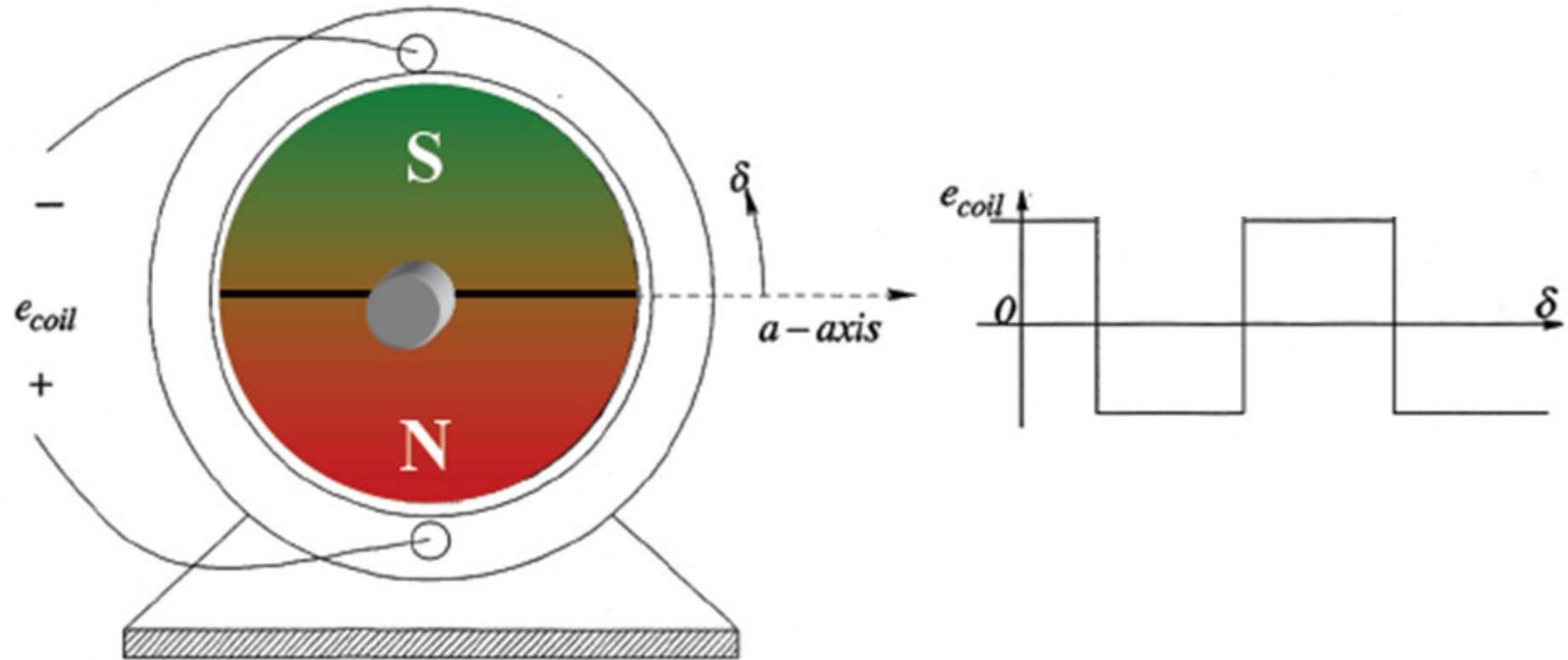




Practical Sensorless Motor Control: How is BEMF Used?

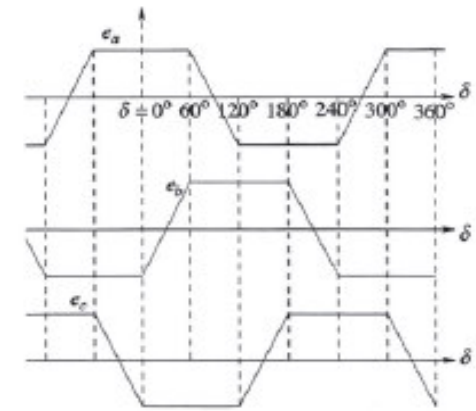
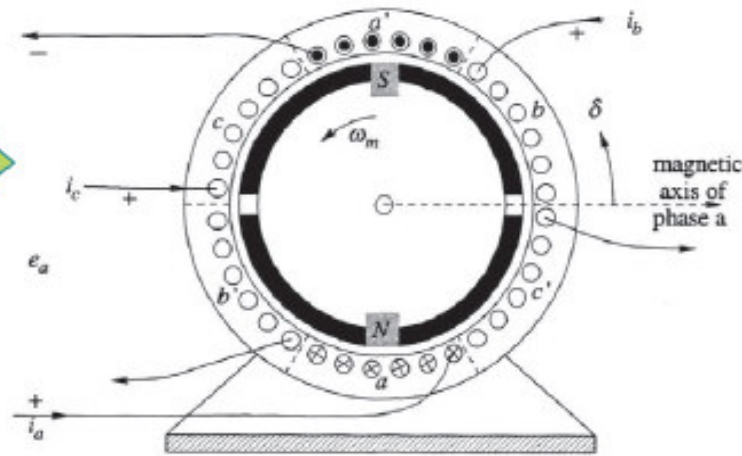


Back EMF in a Single Loop of Wire

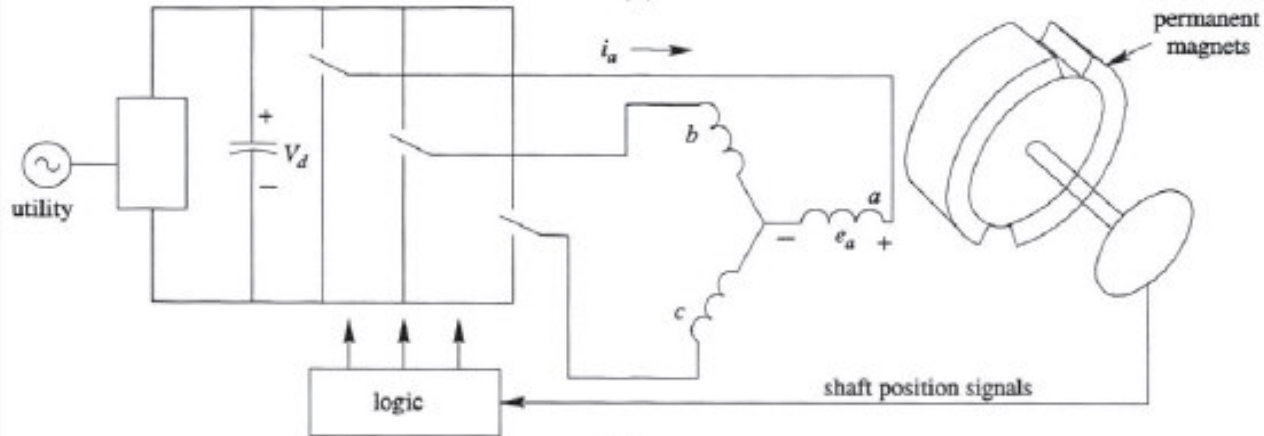


Back EMF in a Multi-turn Winding

Uniform
winding
density



Induced emf in the three phases.



(b)

Source: *Electric Drives, an Integrative Approach*,
by Ned Mohan, University of Minn. Printing Services, 2000

Sensorless BLDC Motor Control Theory

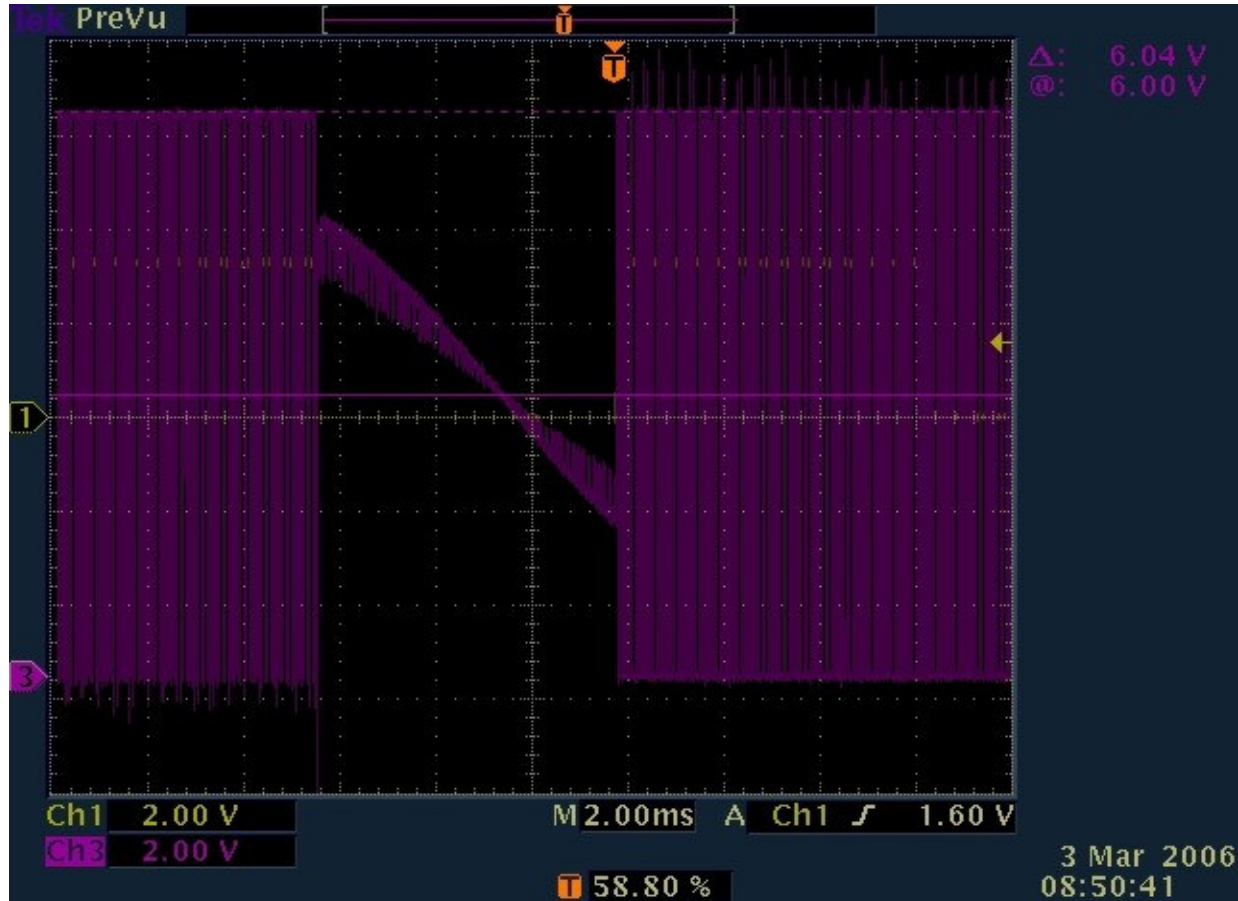
► Sensorless Back-EMF Zero-Cross Algorithm

- Sensing voltage on disconnected phase
 - When sensed voltage crosses half of DC bus voltage, the rotor is in middle between two commutations
 - The detection of this zero crossing allows to detect rotor position

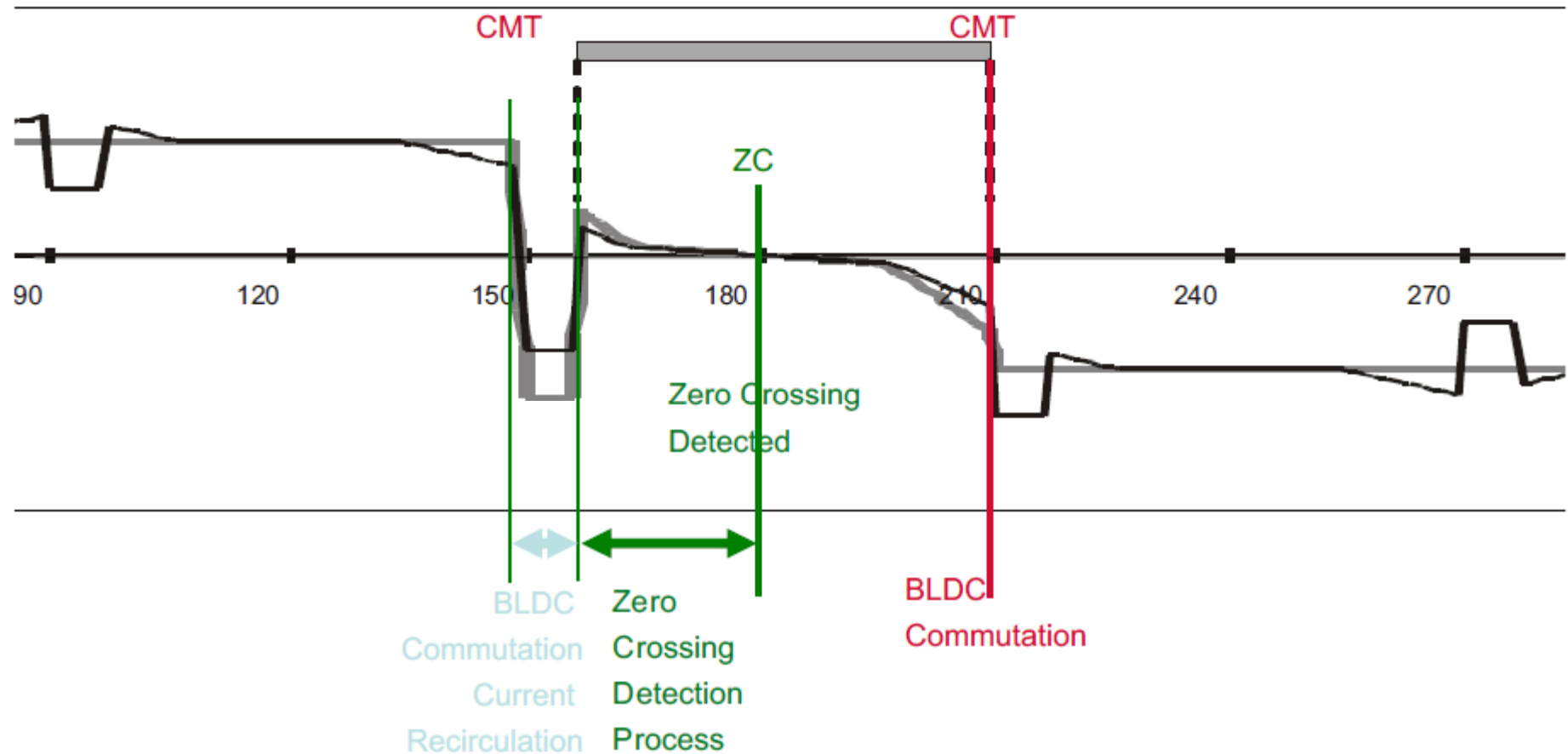


Sensorless BLDC Motor Control Theory

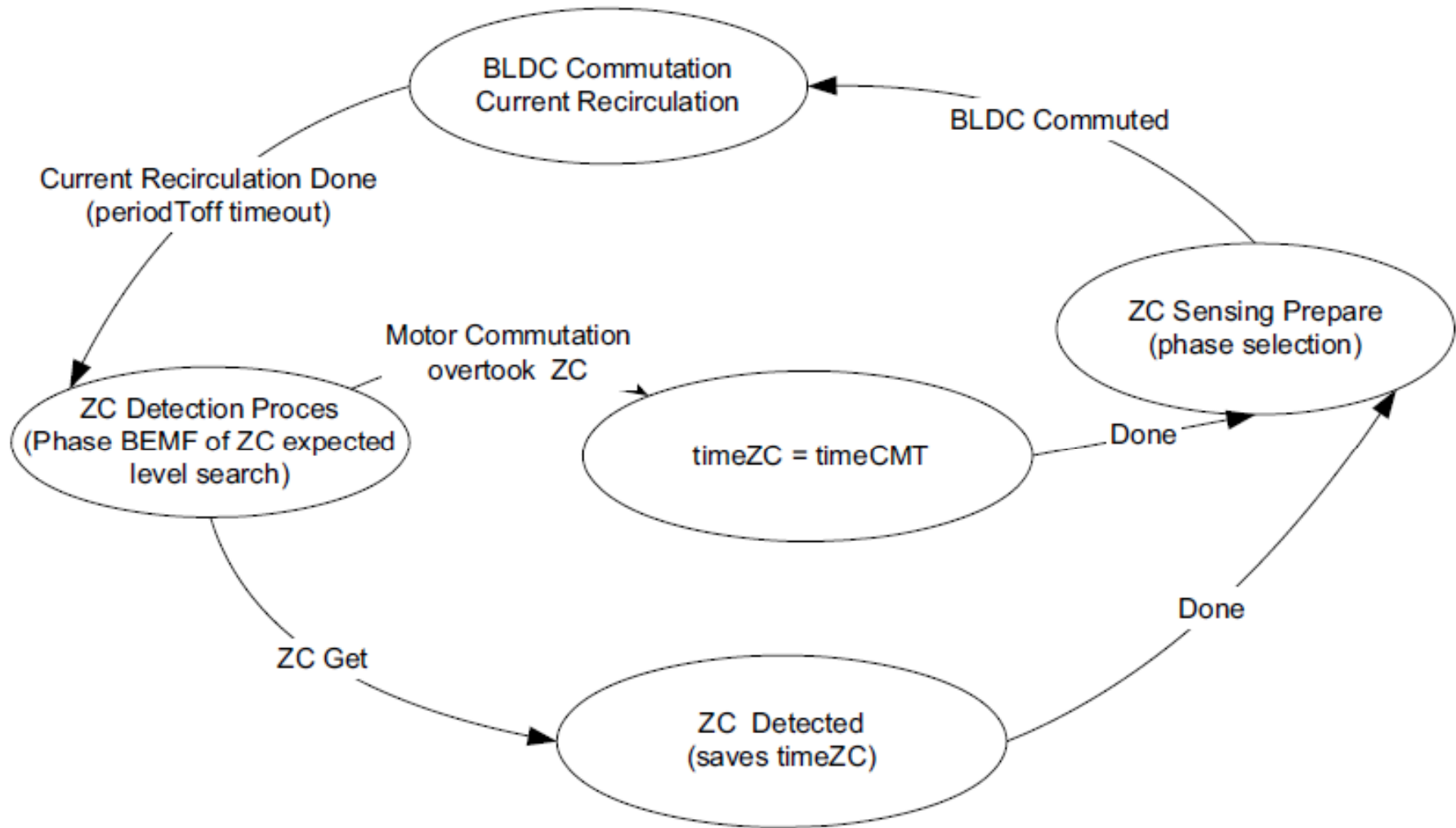
► Detail: Sensorless Back-EMF Zero-Crossing

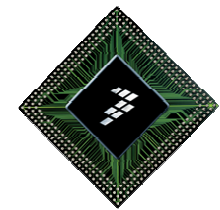


BEMF Detection States



Software Implementation





Open Loop Startup on a Sensorless Application

BEMF Facts

- ▶ Must detect BEMF to spin the motor in sensorless mode.
- ▶ In order to generate BEMF, the rotor must be spinning.
- ▶ The faster the rotor is the higher is the BEMF voltage.
- ▶ So how to start the motor from zero speed?



▶ Classic Chicken or the egg question

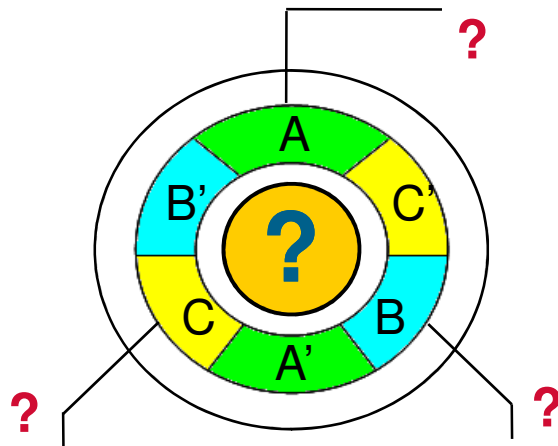


Open Loop Startup is the answer!

Rotor Alignment

- ▶ For the required **higher** initial torque, the rotor position **must** be known to apply the best next commutation.
- ▶ Unsure and without sense of the rotor position, **rotor alignment** is required
- ▶ One of the simplest techniques is to **force** a known position.
- ▶ The rotor position is stabilized by applying PWM signals to **only two** motor phases (no commutation).

(The current controller keeps the current within predefined limits.)



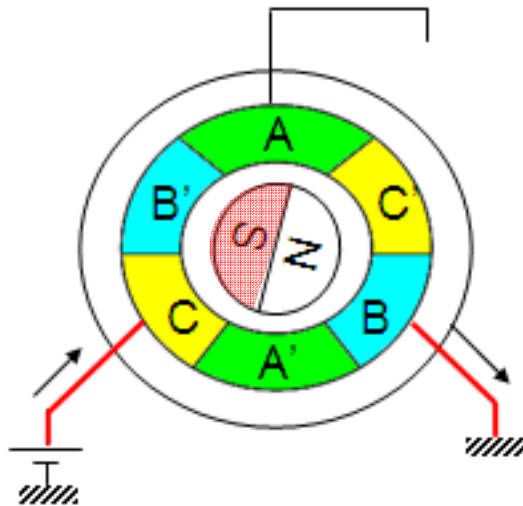
Phase A	Phase B	Phase C
$+V_{DCB}$	$-V_{DCB}$	NC
$+V_{DCB}$	NC	$-V_{DCB}$
NC	$+V_{DCB}$	$-V_{DCB}$
$-V_{DCB}$	$+V_{DCB}$	NC
$-V_{DCB}$	NC	$+V_{DCB}$
NC	$-V_{DCB}$	$+V_{DCB}$

Animation part 1/2

Rotor Alignment

- ▶ For the required **higher** initial torque, the rotor position **must** be known to apply the best next commutation.
- ▶ Unsure and without sense of the rotor position, **rotor alignment** is required
- ▶ One of the simplest techniques is to **force** a known position.
- ▶ The rotor position is stabilized by applying PWM signals to **only two** motor phases (no commutation).

(The current controller keeps the current within predefined limits.)

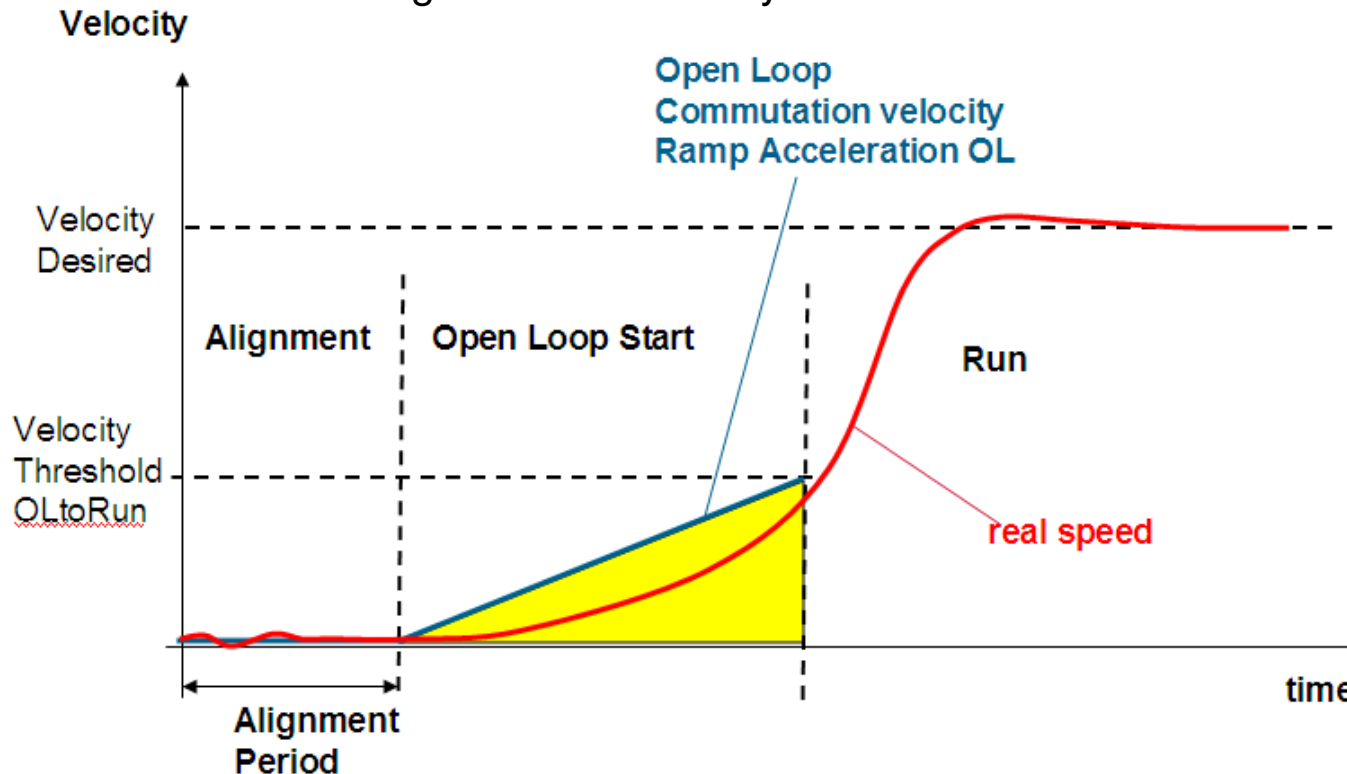


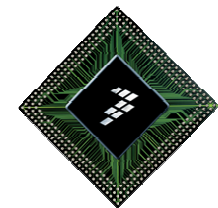
Phase A	Phase B	Phase C
$+V_{DCB}$	$-V_{DCB}$	NC
$+V_{DCB}$	NC	$-V_{DCB}$
NC	$+V_{DCB}$	$-V_{DCB}$
$-V_{DCB}$	$+V_{DCB}$	NC
$-V_{DCB}$	NC	$+V_{DCB}$
NC	$-V_{DCB}$	$+V_{DCB}$

Animation part 2/2

Openloop Startup

- ▶ Commutation starts the **open-loop** without rotor position feedback.
- ▶ Period is controlled by a **linear velocity ramp**.
- ▶ Rotor and stator flux need to be in an approximately **90 degree relation** to maintain synchronization.
- ▶ **Torque** is lower and not constant.
- ▶ Needs to be a **short state** at a very low speed where the back-EMF is too small so the zero-crossing cannot be reliably detected.
- ▶ so the zero-crossing cannot be reliably detected.





Hands-on: Sensorless App

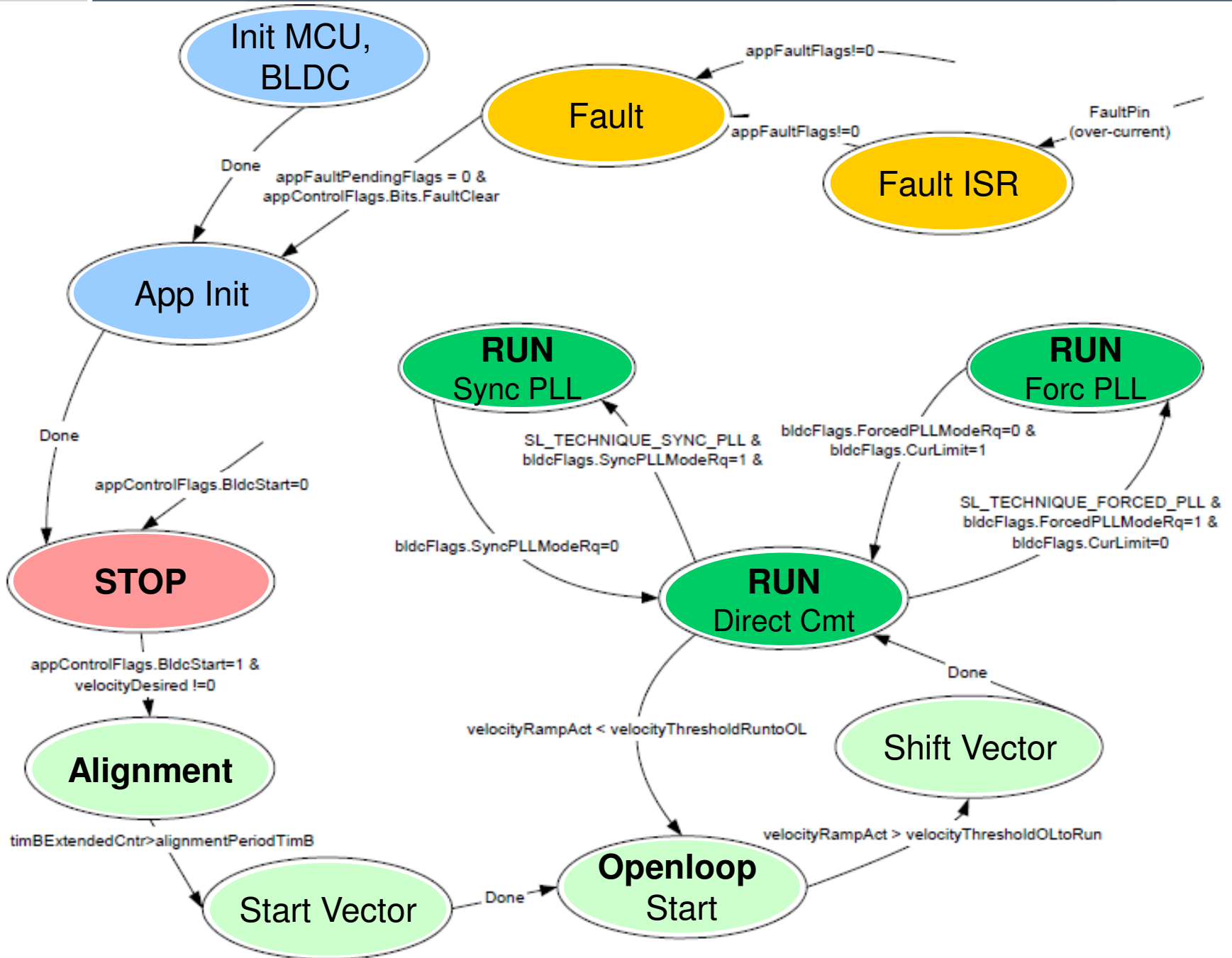
- ▶ Please refer to the Lab Guide for instructions on how to proceed.
- ▶ For this first lab, please follow only the first 6 steps on the Lab Part1 session.

► Please refer to the instructions on how to proceed.

► For this first lab, please follow the 5 steps on the Lab Part1 session.

WAIT!!!

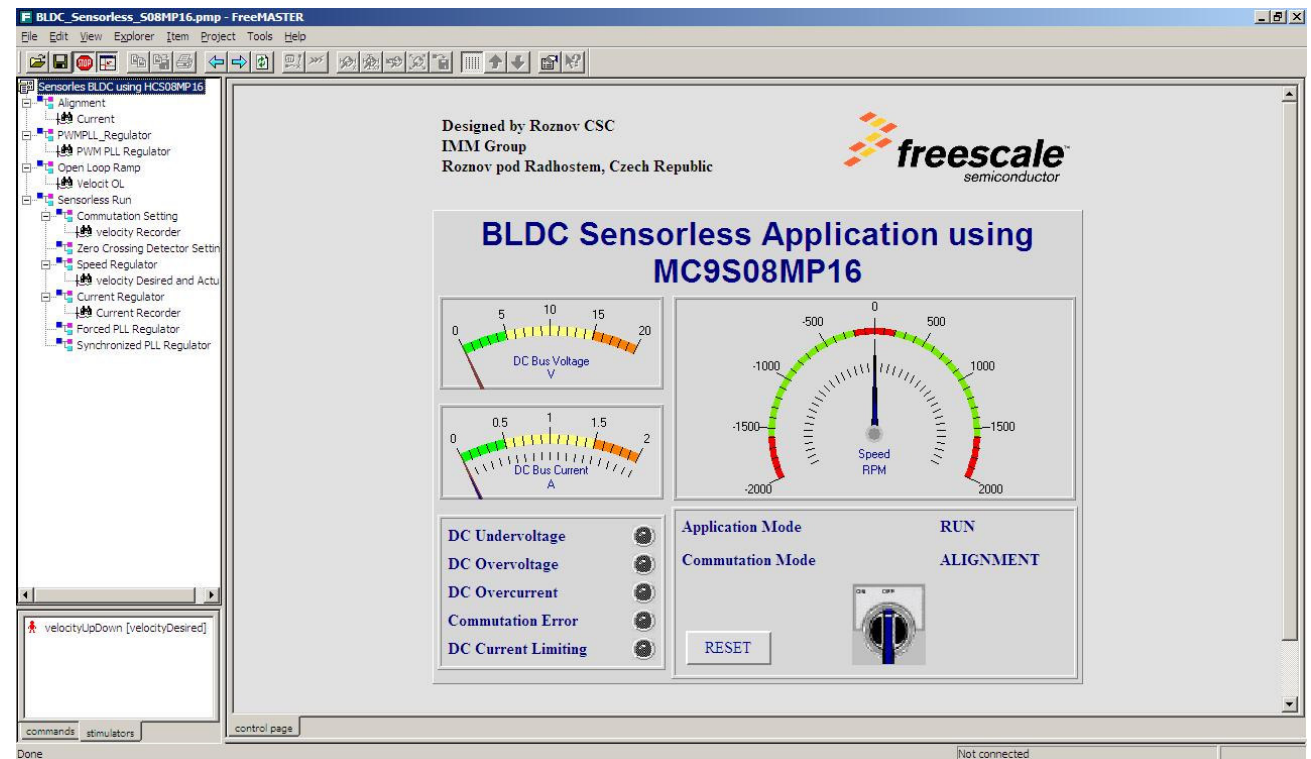
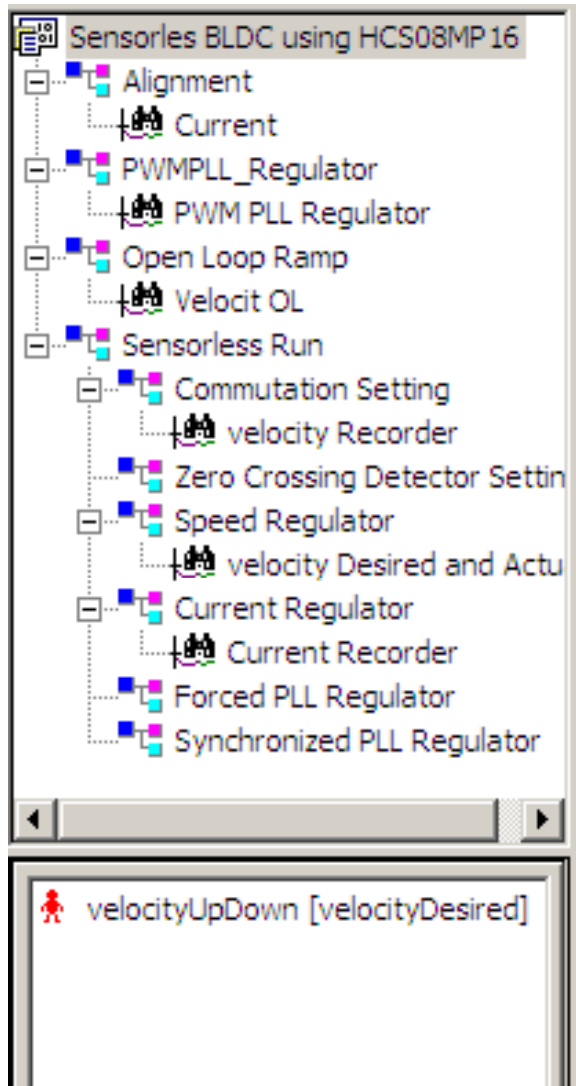
Before going forward make sure no obstacles are near the motor shaft ... specially your badges. We do not want you tangled up in there!!!



FreeMASTER

► FreeMASTER software provides:

- Debugging, diagnostic and demonstration tool for the development of algorithms and applications
- Very useful for tuning the application
- RS-232 serial port, JTAG, BDM or USB communication with PC



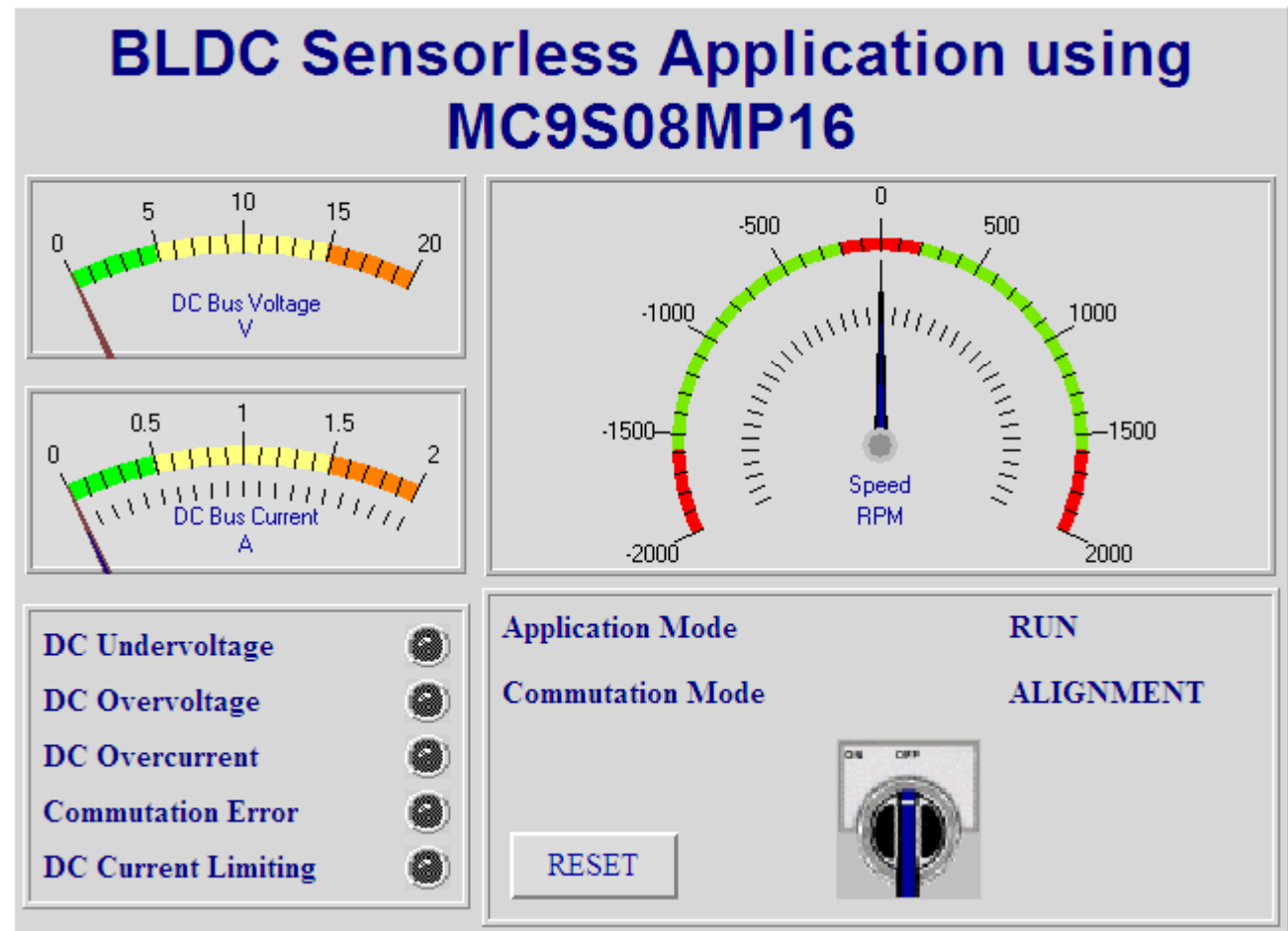
- ▶ Control page is a graphical user interface (GUI) for the 3-phase sensorless BLDC control

- ▶ Actions supported:

- Setting the required speed
- Switch running motor *on/off*

- ▶ Control page displays:

- Actual and required speed
- DC-bus current and voltage
- Application (fault) status



► Please refer to the instructions on how to proceed.

WAIT!!!

► For this lab, follow the lab Part1 from 8 to 18.

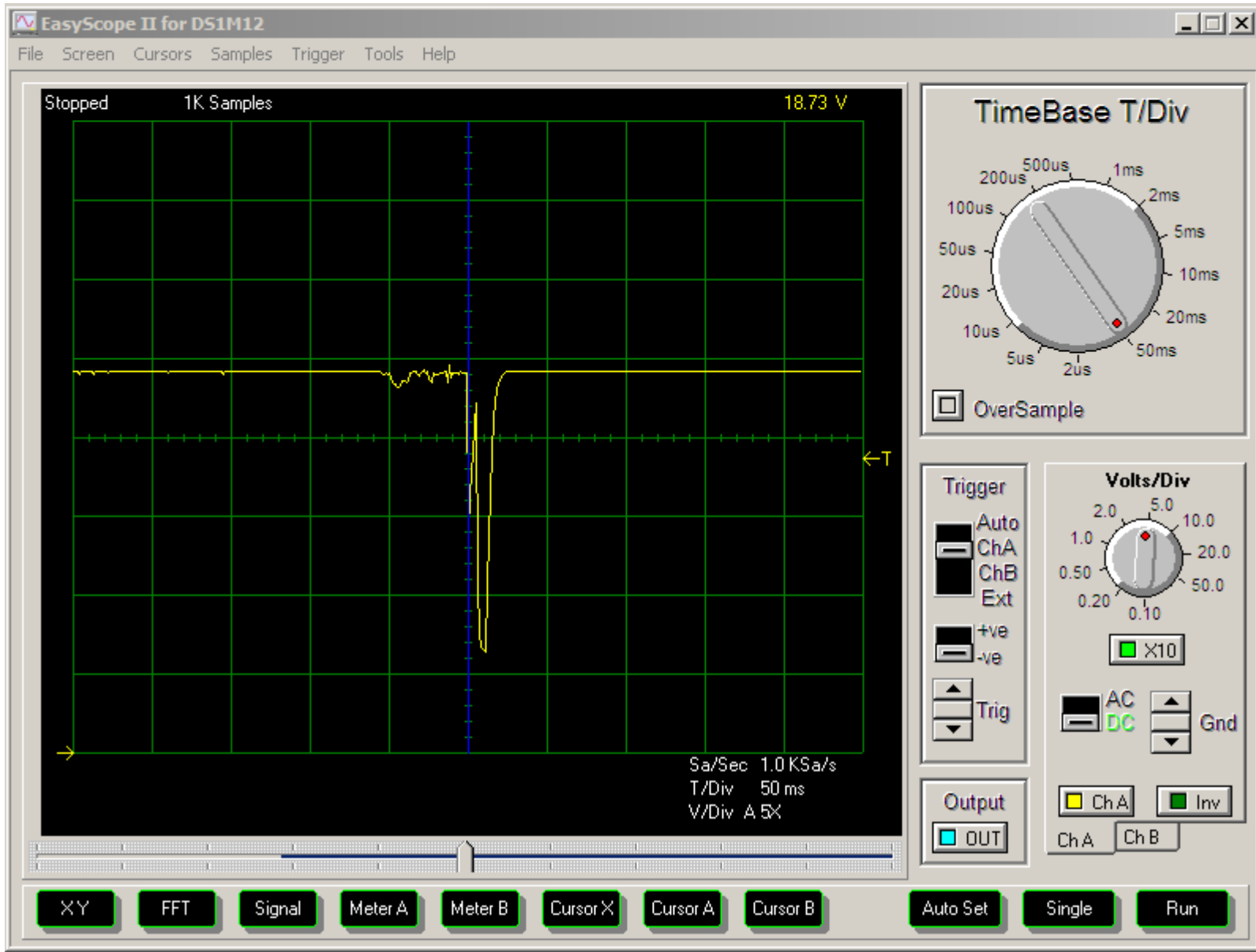
Before going forward make sure no obstacles are near the motor shaft ... specially your badges. We do not want you tangled up in there!!!

- ▶ Please refer to the **Lab Guide** for instructions on how to proceed.
- ▶ For this lab, follow the remaining steps on Lab Part1 from 8 to 18.

Acceleration Comments

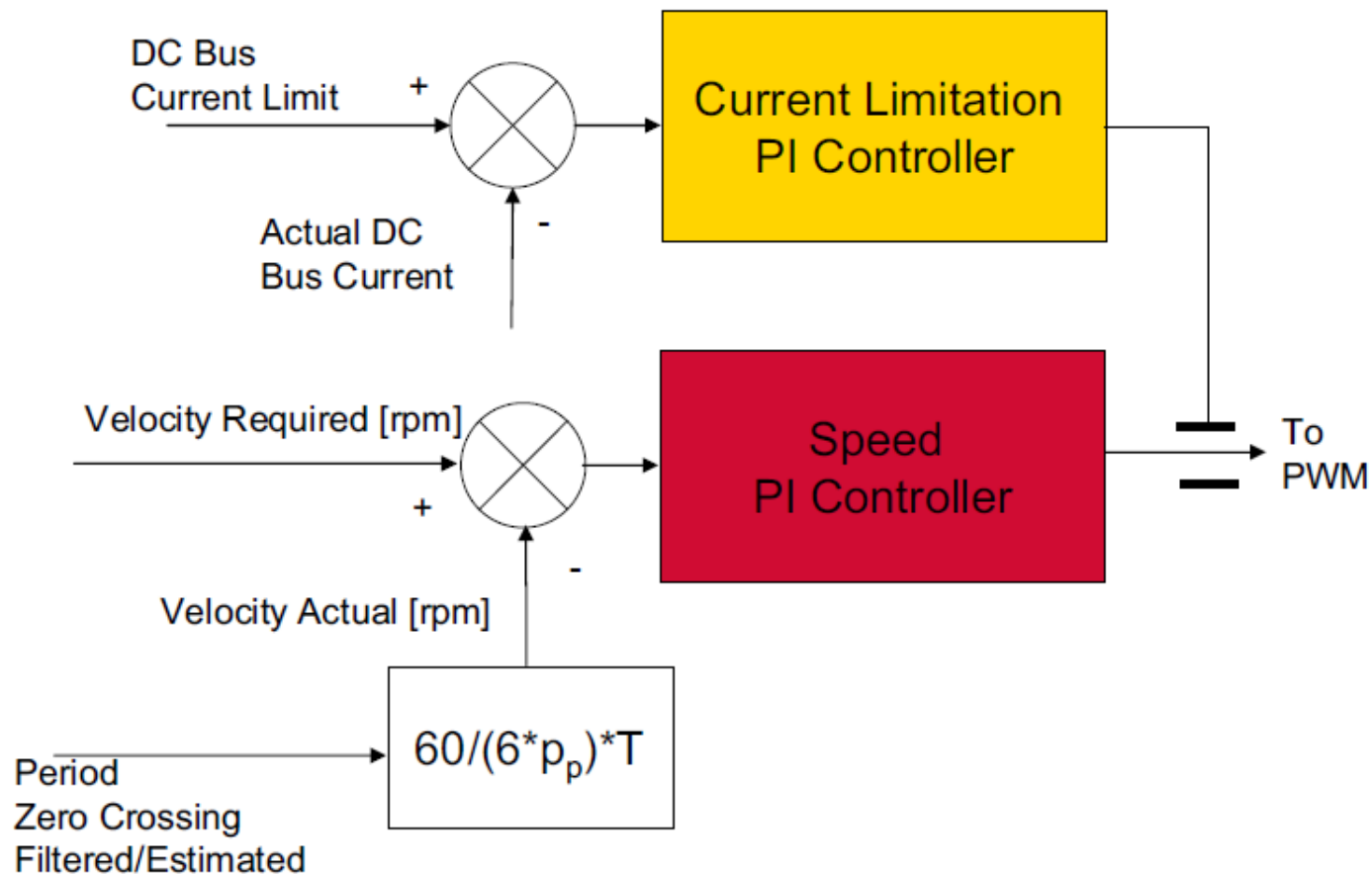
- ▶ Did you observe any unexpected behavior?
- ▶ How would you improve it?
- ▶ What other underlying effects does this behavior have?

DC Bus Voltage

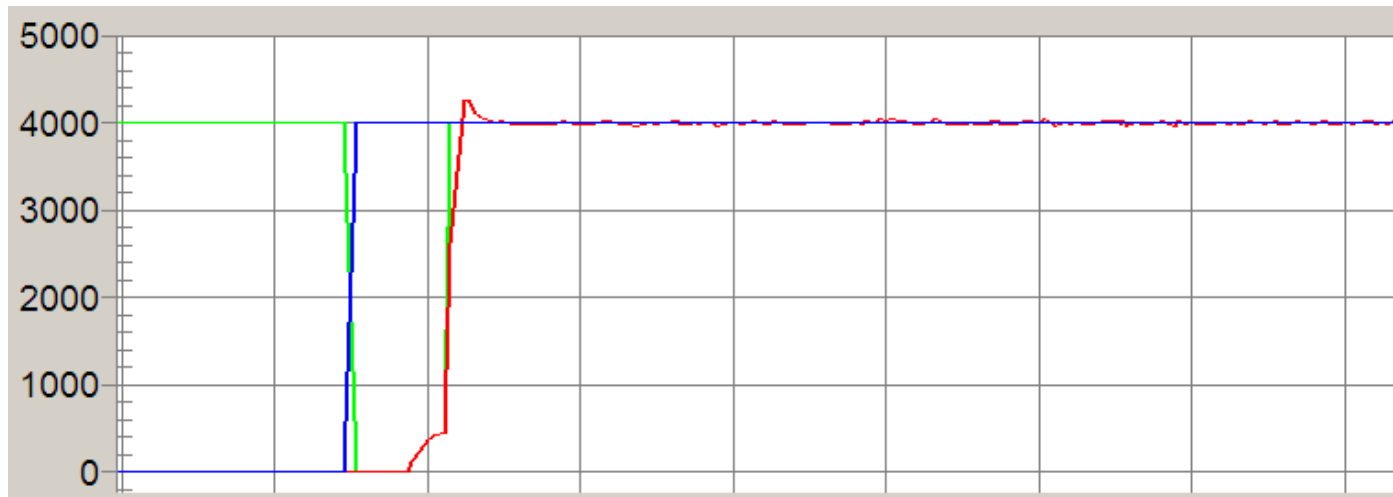


Speed Controller with Current (Torque) Limitation

- ▶ Speed is controlled using the zero-crossing period feedback provided.
- ▶ The outer current regulator limits the motor current.
This provides the torque limitation in order to limit the maximal motor current.
- ▶ The speed regulator controls the 3-phase power stage PWM.

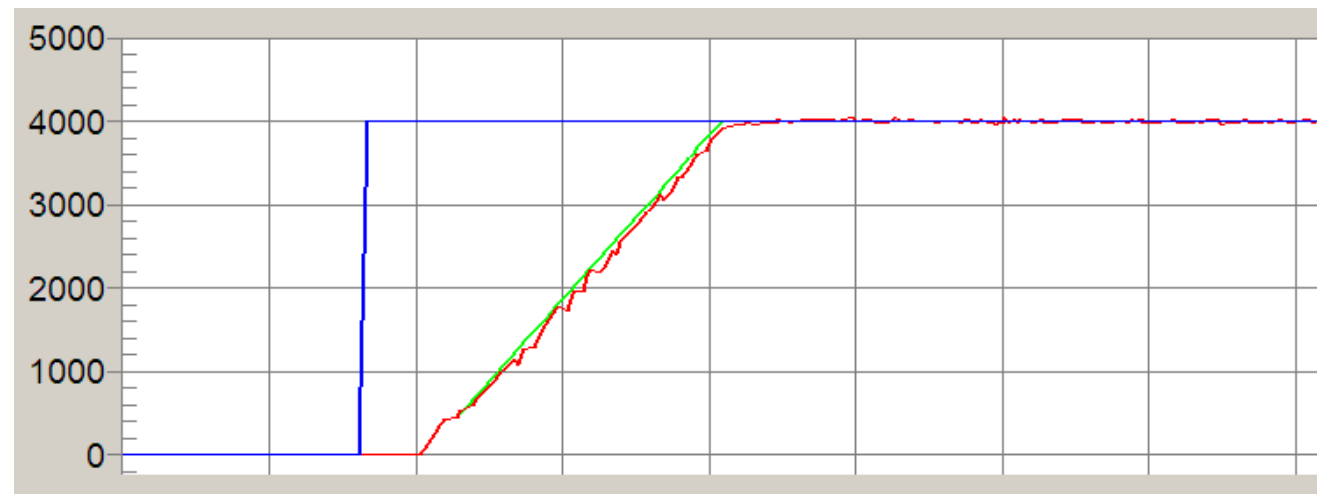


Add a Speed Ramp



- We need to add a Speed Ramp before the PI controller.

```
/* function using the acceleration Ramp */
velocityRampAct = ECLIB_Ramp16(rampAccelCl, velocityDesired, velocityRampAct);
```



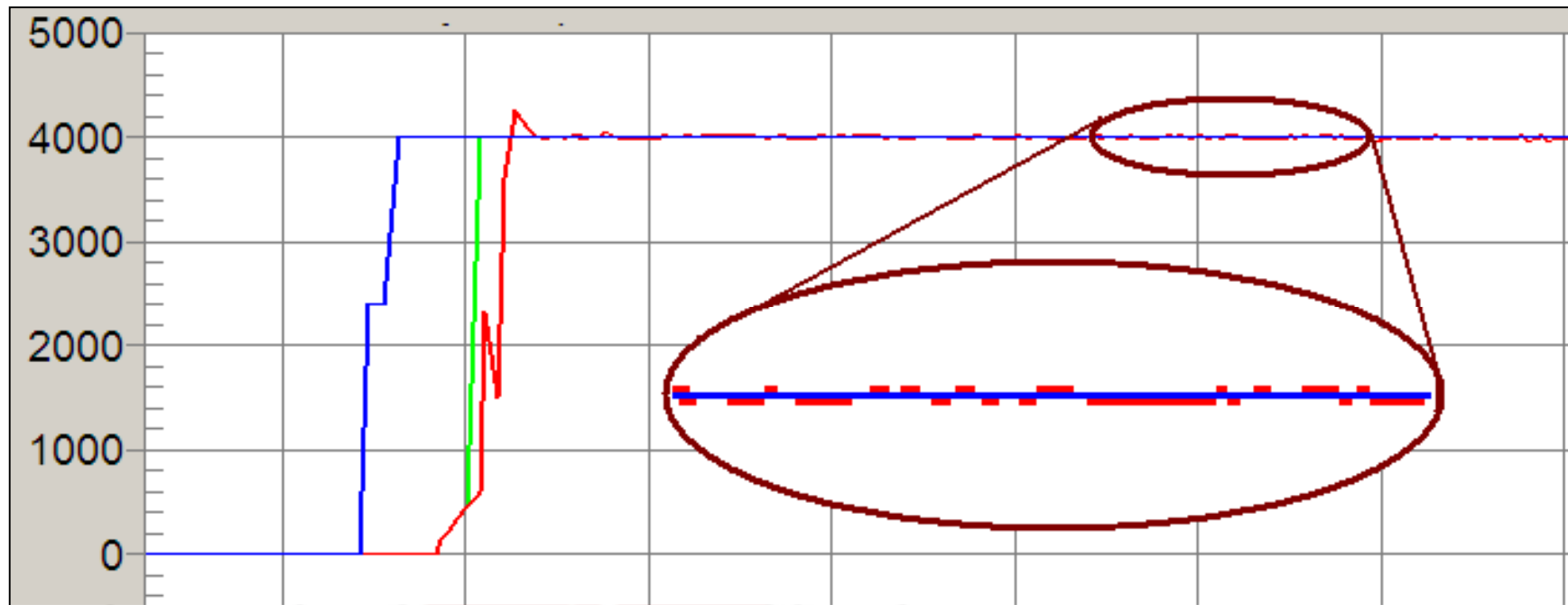
- ▶ Please refer to the **Lab Guide** for instructions on how to proceed.
- ▶ For this lab, follow the **Lab Parts 2 and 3**.

► Please refer to the instructions on how to proceed.

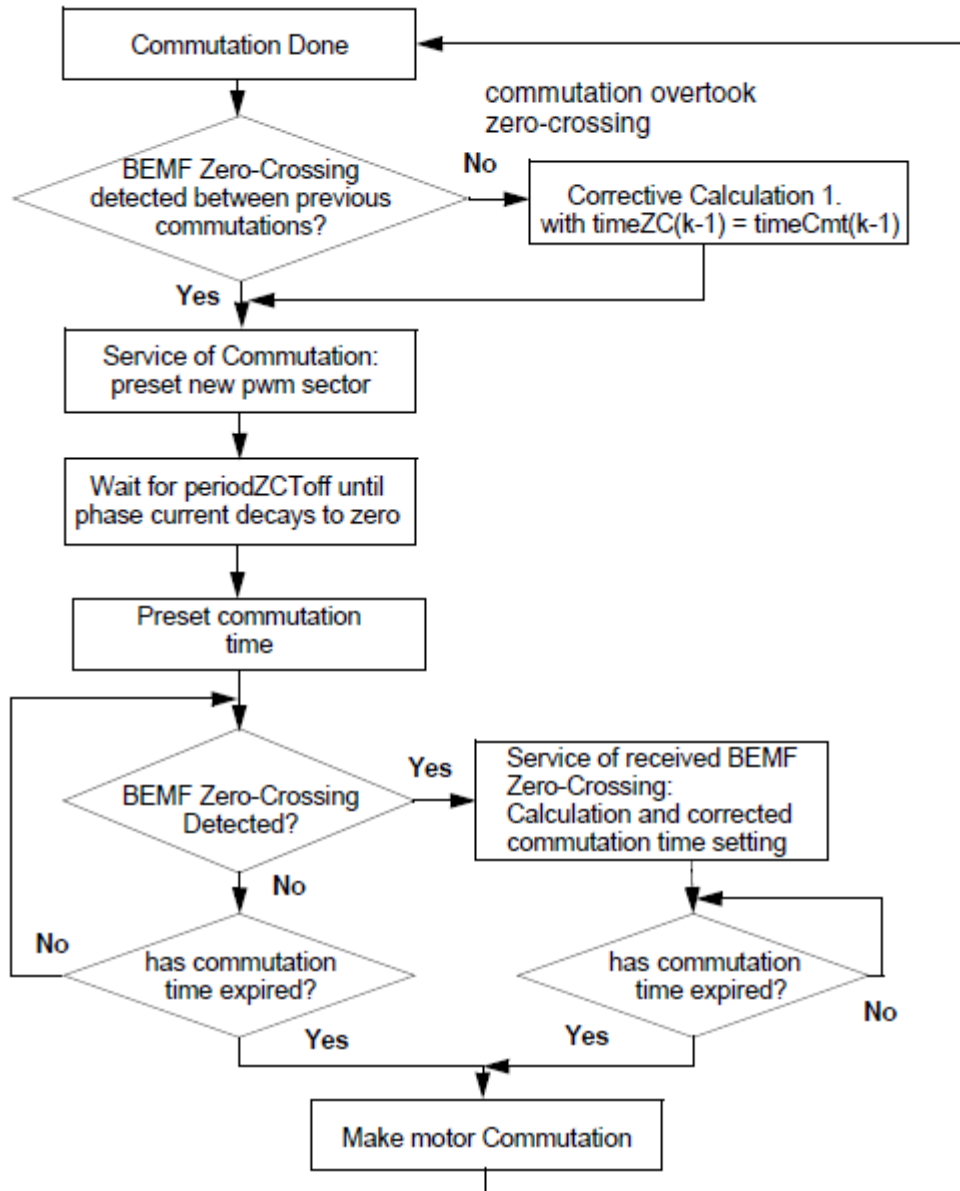
WAIT!!!

► For this lab, follow the instructions on how to proceed. **Before going forward make sure no obstacles are near the motor shaft ... specially your badges. We do not want you tangled up in there!!!**

Constant Speed Control

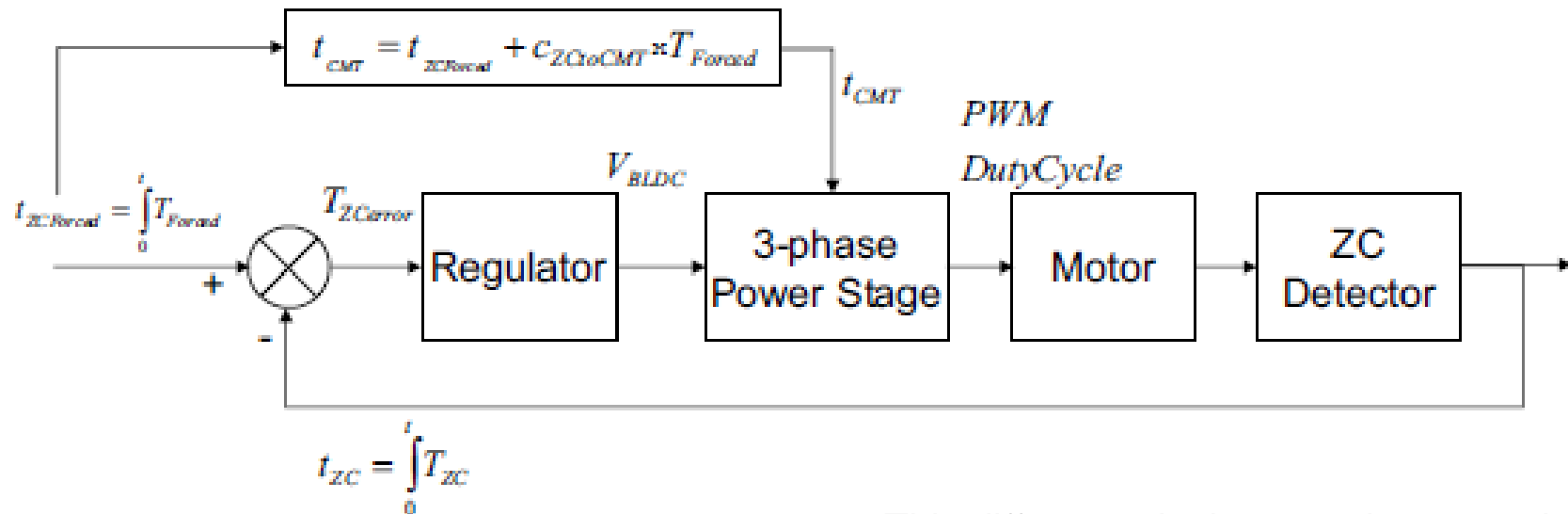


Run - Direct Commutation



- CMT time depends on the time difference between the last 2 Zero-crossings.
- In case Zero-cross is not detected properly, the last value is used.
- Very good technique if speed change is required very often.
- The Speed controller uses the PWM duty-cycle to control.
- Zero-crossing time will be automatically changed.
- Any small error in measurement can cause changes in motor speed.

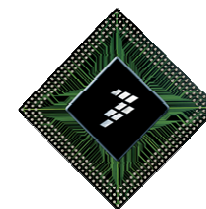
Forced Phase-Locked-Loop (PLL) — Forced Cmt



- Forced PLL uses a **constant Commutation time** (T_{cmt}).
- As we know the speed required, we know the **Tbemf required**.
- The **difference** between the desired Tbemf and the measured Tbemf is calculated.
- This difference is then used to control the **PWM duty-cycle**, that will change back the Zero-crossing time.
- If the Tbemf difference **is too high**, the control might have problems, so the application changes to **Run-Direct** mode.
- This mode is much **more stable** on **constant** speed applications.

- ▶ Please refer to the **Lab Guide** for instructions on how to proceed.
- ▶ For this lab, follow the **Lab Part 4**.

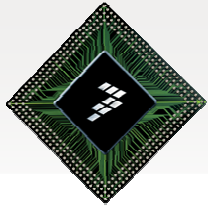
- ▶ Please refer to the **Lab Guide** for instructions on how to proceed.
- ▶ For this lab, follow the **Lab Part 4**.



Summary

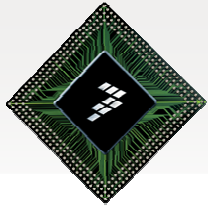
Control comparison table: with Sensors versus Sensorless

<i>Feature</i>	With Sensors	Sensorless
Connection issues	- Possible (add connectors and cables to the system)	+ No sensor connections.
Low Speed	+ Good low speed operation, position feedback from Zero	- Openloop startup needed, no feedback on lower speeds.
Maximum Speed	+ Can get to higher speed ranges	- Maximum speed limited due to BEMF reading time.
Control	+ Easier	- More complex (BEMF, Openloop startup)
Torque control	+ Good on all operation speeds	- Lower on Openloop startup
Cost	- Higher (sensors, cables, connectors)	+ Lower



Summary

- ▶ BLDC motor control schemes go from simple and able to be handled by simple generic microcontrollers to complex PWM switching mechanisms. Although higher-end microcontrollers may seem better suited for these tasks, a complex timer with a simpler CPU is enough to handle most of the more complicated motor control features, leaving the level of mathematical control to the CPU.
- ▶ With an adequate motor control-enabled timer, PWM switching schemes are equally simple to integrate, it is just a matter of choosing the right scheme for the application.
- ▶ Acceleration ramps allow motor control applications a simple way to maintain DC bus current low during start-up. Ramps can be easily tuned to provide the best start-up time for the motor depending on the amount of expected initial torque.



Freescale Product Longevity Program

- ▶ The embedded market needs **long-term product support**
- ▶ Freescale has a longstanding track record of **providing long-term production support** for our products
- ▶ Freescale offers a **formal product longevity program** for the market segments we serve
 - For the automotive and medical segments, Freescale will make a broad range of program devices available for a minimum of **15 years**
 - For all other market segments in which Freescale participates, Freescale will make a broad range of devices available for a minimum of **10 years**
 - **Life cycles** begin at the time of launch
- ▶ A list of participating **Freescale products** is available at: www.freescale.com/productlongevity

