MICROCHIP

Microchip Technology Inc.

dsPIC®

Digital Signal Controller (DSC)

Motor Control Workshop In a Box

WIB Agenda

- Microchip's Corporate Overview
- dsPIC® DSC Overview
- dsPIC DSC Motor Control Peripherals
- BLDC Motor Control Algorithms
- Labs

WIB Agenda

LAB Sessions:

- *Lab 1* – **Programming a dsPIC® DSC Using the PICDEM™ MC LV Board**
- *Lab 2* – **Running BLDC Motor with Forced Commutation**
- *Lab 3* – **Running Sensored BLDC Motor with GPIO**
- *Lab 4* – **Running BLDC Motor with MCPWM**
- *Lab 5* – **Running Closed-loop BLDC Motor**
- *Lab 6* – **Running Sinusoidal BLDC Motor**
- *Lab 7* – **BLDC Operation with Phase Advance**
- *Lab 8* – **Running Sensorless BLDC Motor**

Overview

dsPIC® DSC

- Architecture Overview
- Motor Control Family Overview
- Motor Control Peripherals

BLDC Motor Theory

- Motor Theory
- BLDC Motor Construction
- Position Sensing

Overview (continued)

BLDC Motor Algorithms:

- Forced Commutation Operation
  - What is Commutation?
  - Commutating a BLDC with no position feedback
- Six Step Control (120° Conduction)
  - BLDC Position Sensing
  - Synchronizing Commutation with Position
- Six Step Sensored Algorithm

Overview (continued)

BLDC Motor Algorithms

- Variable Speed BLDC Motor Control
  - **Using MCPWM for Variable Speeds**
  - **Commutation using Override Control**
- Closed Loop Speed Control of a BLDC
  - **PID Implementation with dsPIC**
  - **Measuring Speed of a BLDC Motor**

Overview (continued)

BLDC Motor Algorithms

- Sinusoidal Control (180° Conduction)
  - **Target Motors for Sinusoidal Control**
  - **Sinusoidal Voltage Generation**
  - **BLDC Commutation using Sinusoidal Voltages**
- Phase Advance Commutation
  - **Scheduling BLDC Motor Commutation**

Overview (continued)

BLDC Motor Algorithms

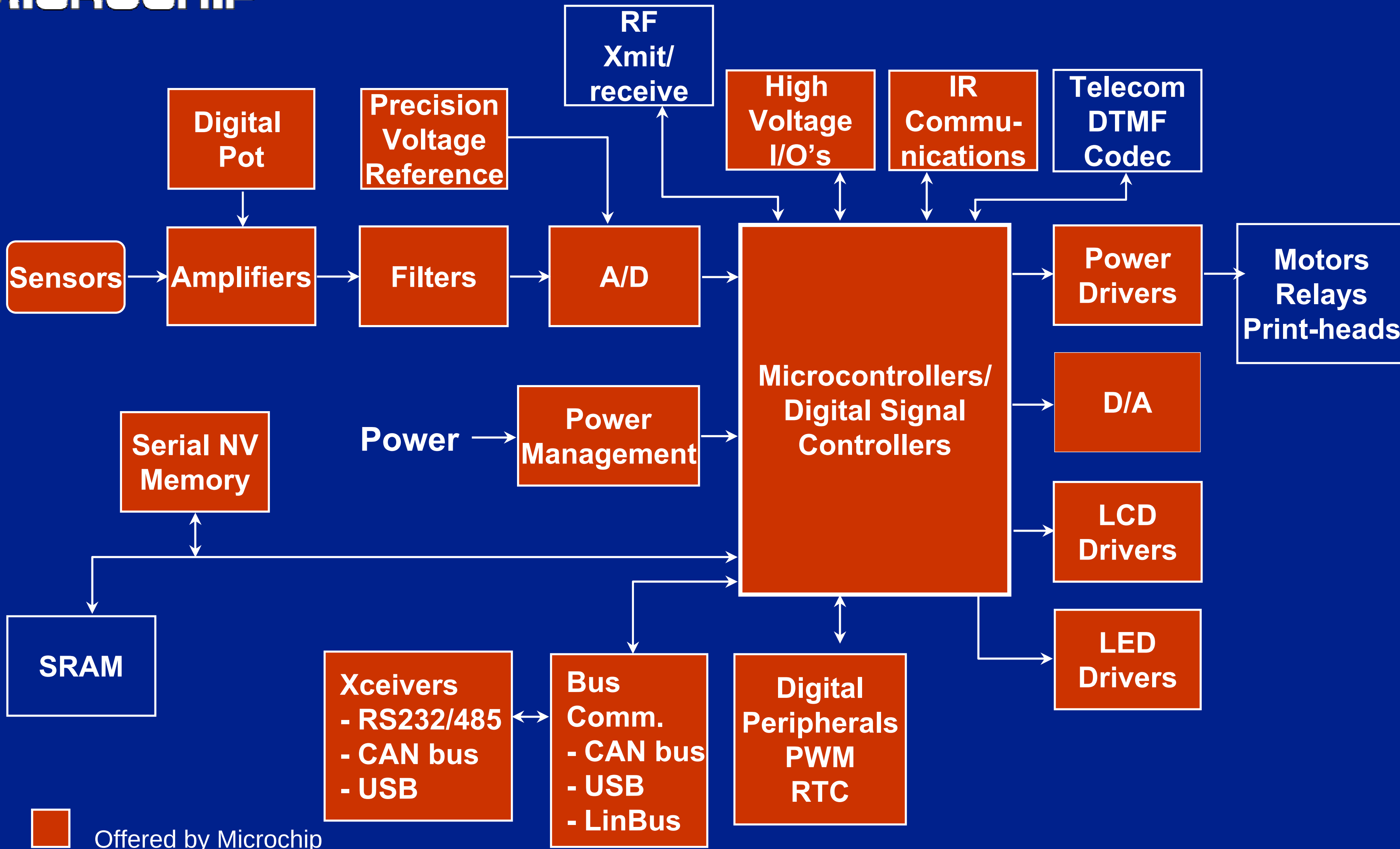
- Sensorless BLDC Motor Control
  - **Why Sensorless?**
  - **BEMF Zero Crossing Detection Technique**
  - **Hardware Implementations for Detecting BEMF**
  - **Implementing Sensorless BLDC with dsPIC® DSC**
  - **Introduction to SMTI for Tuning Parameters**
  - **Parameter Tuning Exercise using SMTI**

Microchip's Corporate Overview

Microchip

- **Leading semiconductor manufacturer:**
  - **of high-performance, field-programmable, 8-bit & 16-bit RISC Microcontrollers**
  - **of Analog & Interface products**
  - **of related Memory products**
  - **for high-volume embedded control applications**
- **$850M in product sales**
- **More than 4,200 employees**
- **Headquartered near Phoenix in Chandler, AZ**
- "The Silicon Desert"

Where Are We Today?



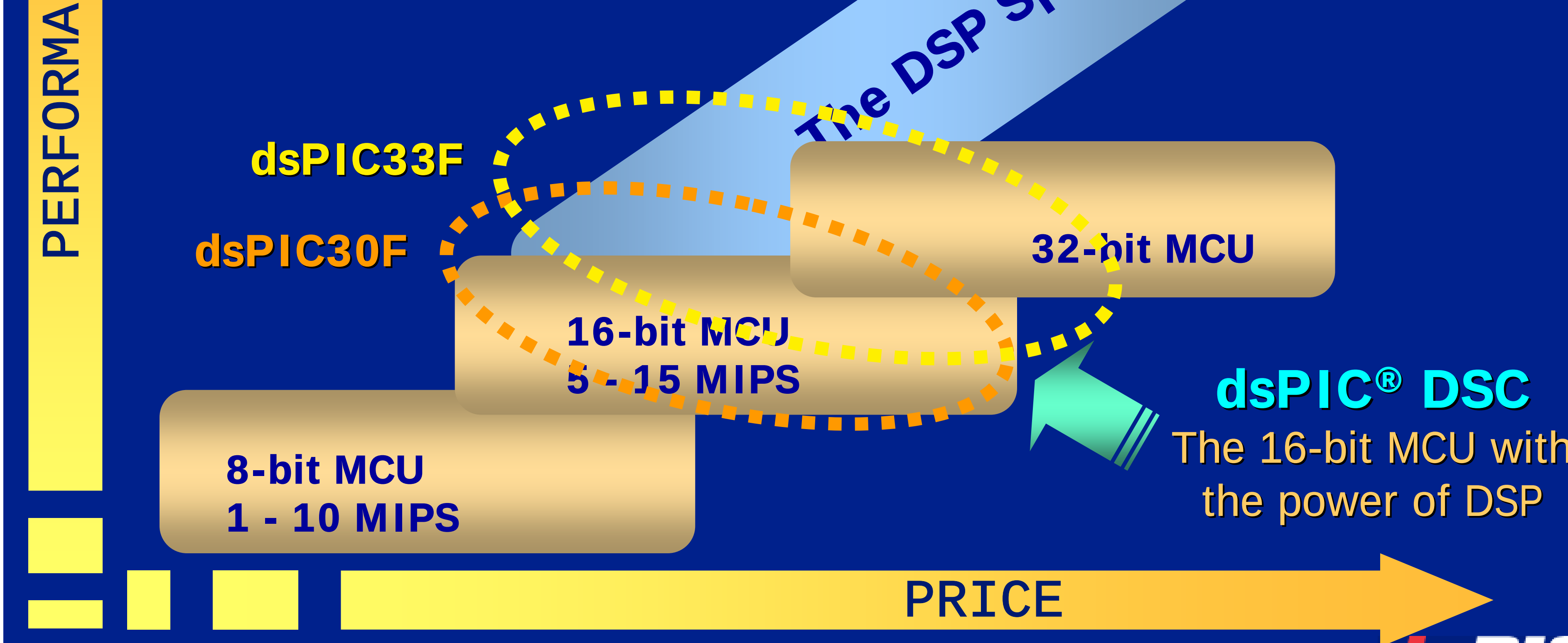
Digital Signal Controller

dsPIC® DSC Overview

What is DSC ?

Digital Signal Control

Embedded Control
+
Digital Signal Processing



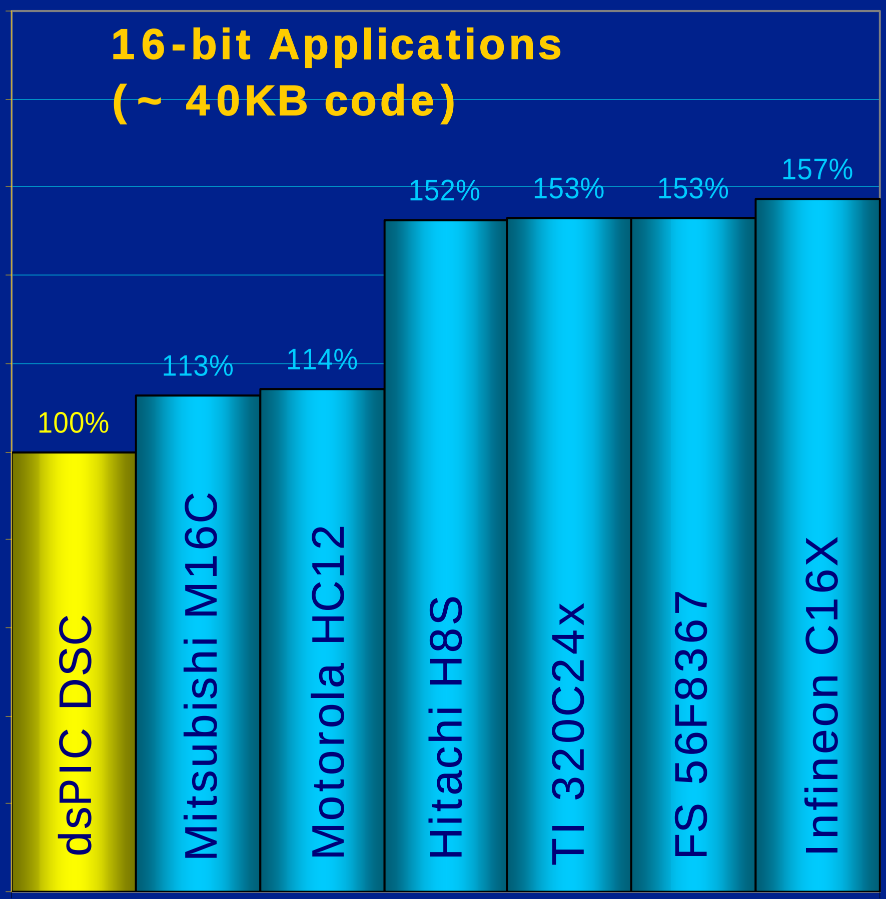
dsPIC® DSC Family: Architected from Scratch

- Seamlessly integrates a DSP and an MCU
- MCU look and feel, easy to use
- Competitive DSP performance
- Optimized for C compiler
- Fast, deterministic, flexible interrupts
- Excellent RTOS support

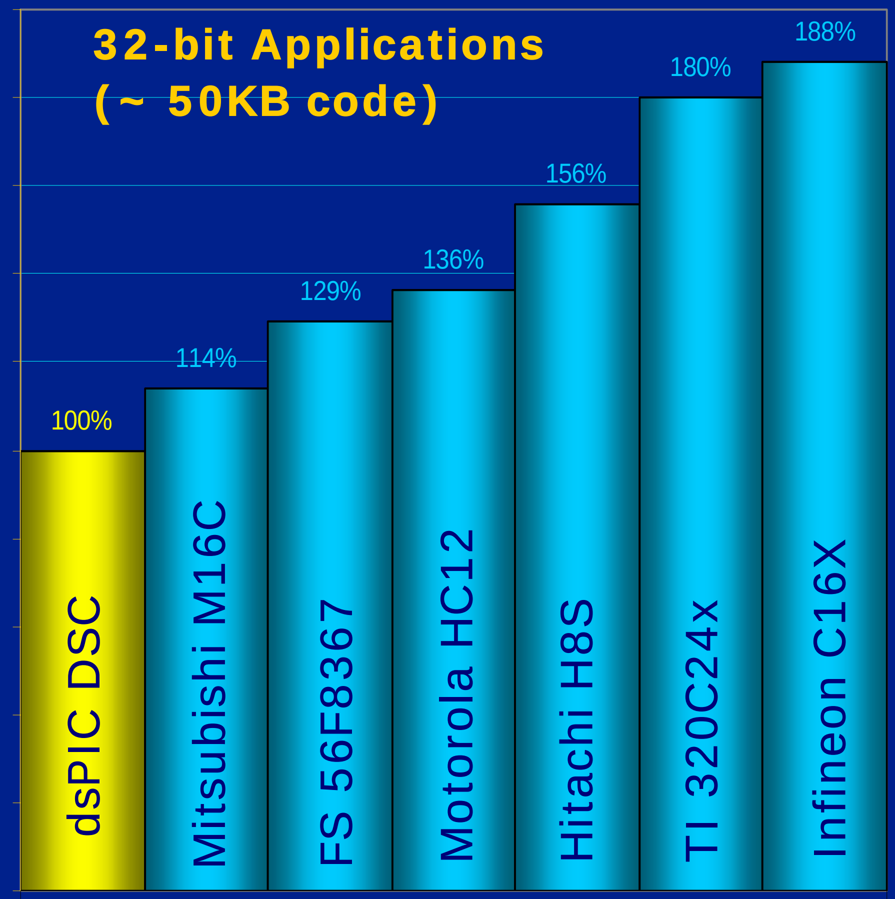
dsPIC® DSC Highly Optimized C compiler

Control Centric Benchmarks

Relative Code Size



MPLAB® C30 v1.30



MPLAB C30 v1.30

Digital Signal Controller

dsPIC30F Products

Power Conversion & Motor Control Family

Product dsPIC® DSC	Pins	Flash KB	SRAM Bytes	EE Bytes	Timer 16-bit	Input Cap	Output Comp/ Std PWM	Motor Cntrl PWM	A/D 10-bit 1.0 Msps	Quad Enc	UART	SPI™	I²C™	CAN
dsPIC30F2010	28	12	512	1024	3	4	2	6	6 ch	Yes	1	1	1	-
dsPIC30F3010	28	24	1024	1024	5	4	2	6	6 ch	Yes	1	1	1	-
dsPIC30F4012	28	48	2048	1024	5	4	2	6	6 ch	Yes	1	1	1	1
dsPIC30F3011	40	24	1024	1024	5	4	4	6	9 ch	Yes	2	1	1	-
dsPIC30F4011	40	48	2048	1024	5	4	4	6	9 ch	Yes	2	1	1	1
dsPIC30F5015	64	66	2048	1024	5	4	4	8	16 ch	Yes	1	2	1	1
dsPIC30F5016	80	66	2048	1024	5	4	4	8	16 ch	Yes	1	2	1	1
dsPIC30F6010	80	144	8192	4096	5	8	8	8	16 ch	Yes	2	2	1	2

- **Brushless DC Motor Control**
- **AC Induction Motor Control**
- **Switch Reluctance Motor Control**
- **UPS, Inverters and Power Supplies**

- **Appliances**
- **Power Tools**
- **Automotive**
- **Industrial**

dsPIC33F Products

Power Conversion & Motor Control Family

Product	PINS	Flash (KB)	RAM (KB)	Dual Port RAM (KB)	TIMER	IC	OC PWM	MCPWM	10-bit ADC 1.1 Msps # ADC, # Ch	UART	SPI™	I2C™	CAN	QEI
dsPIC33FJ64MC506	64	64	6	2	9	8	8	8	1 ADC, 16 Ch	2	2	1	1	1
dsPIC33FJ64MC508	80	64	6	2	9	8	8	8	1 ADC, 18 Ch	2	2	1	1	1
dsPIC33FJ64MC510	100	64	6	2	9	8	8	8	1 ADC, 24 Ch	2	2	1	1	1
dsPIC33FJ64MC706	64	64	14	2	9	8	8	8	2 ADC, 16 ch	2	2	2	1	1
dsPIC33FJ64MC710	100	64	14	2	9	8	8	8	2 ADC, 24 ch	2	2	2	2	1
dsPIC33FJ128MC506	64	128	6	2	9	8	8	8	1 ADC, 16 Ch	2	2	2	1	1
dsPIC33FJ128MC510	100	128	6	2	9	8	8	8	1 ADC, 24 Ch	2	2	2	1	1
dsPIC33FJ128MC706	64	128	14	2	9	8	8	8	2 ADC, 16 ch	2	2	2	1	1
dsPIC33FJ128MC708	80	128	14	2	9	8	8	8	2 ADC, 18 Ch	2	2	2	1	1
dsPIC33FJ128MC710	100	128	14	2	9	8	8	8	2 ADC, 24 ch	2	2	2	2	1
dsPIC33FJ256MC510	100	256	14	2	9	8	8	8	1 ADC, 16 ch	2	2	2	1	1
dsPIC33FJ256MC710	100	256	28	2	9	8	8	8	2 ADC, 24 ch	2	2	2	2	1

dsPIC® DSC Architecture Summary

dsPIC® DSC Architecture

- Main Features
  - Single core integrating an MCU & a DSP
  - Modified Harvard Architecture
  - Data is 16-bit wide
  - Instruction is 24-bit wide
  - Linear program memory up to 12 MB
  - Linear data (RAM) up to 64 kB
  - True DSP capability
  - Many integrated peripherals

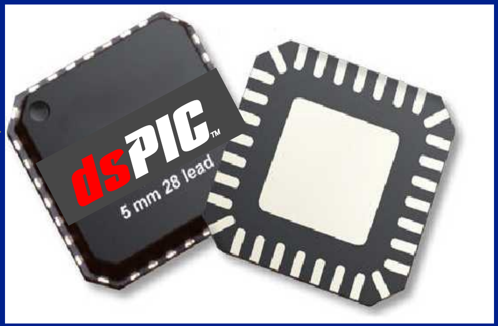
dsPIC® DSC Architecture

- Main Features (continued)
  - 16 x 16-bit working register array
  - Software stack
  - Fast, deterministic interrupt response
  - Three operand instructions: C = A + B
  - Extensive addressing modes
  - DMAC w/ dual port SRAM - 8 channels for peripherals

dsPIC® DSC Operating Parameters

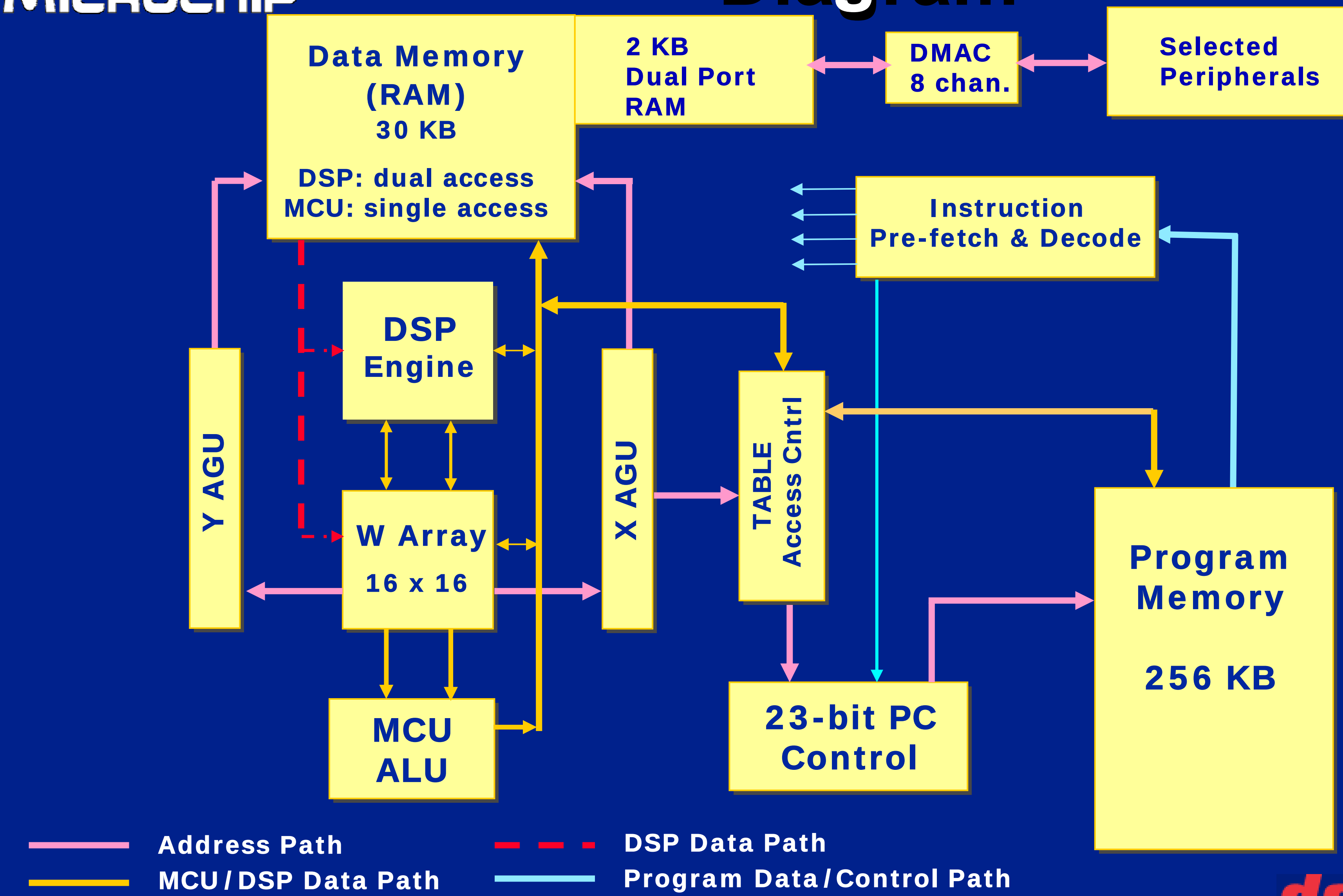
Feature	dsPIC30F	dsPIC33F
Operating Speed:	DC to 30 MIPS	DC to 40 MIPS
VDD:(VDC)	2.5 to 5.5	3.0 to 3.6
Temp:	-40°C to +125°C	-40°C to +85°C
Program Memory:	Flash	Flash
Data Memory:	SRAM, EEPROM	SRAM, Self-write Flash

- Package sizes
  - 18-pin SO & SP
  - 28-pin SO, SP and QFN
  - 40-pin SP; 44-pin TQFP, QFN
  - 64-, 80- and 100- pin TQFP



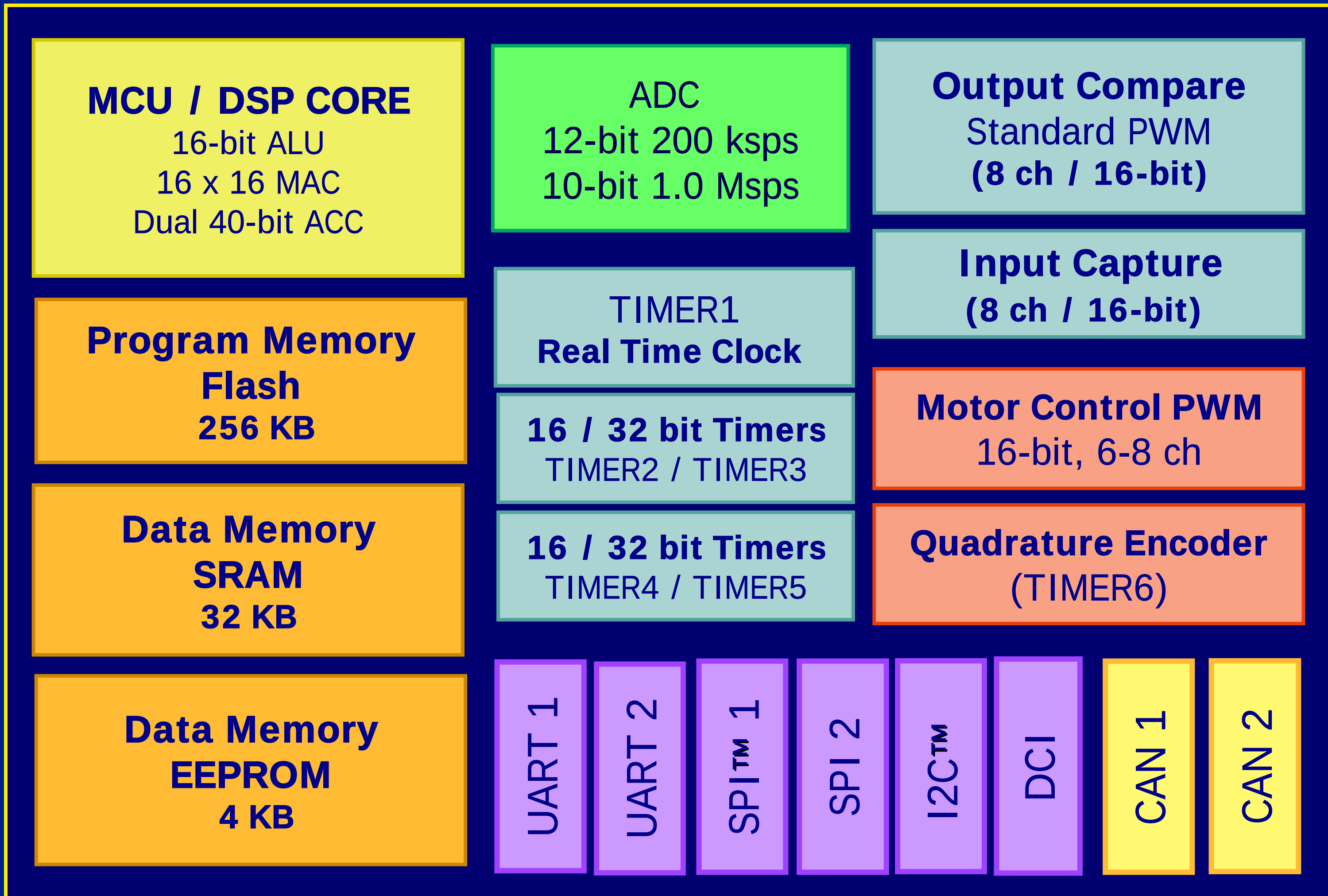
28 lead QFN: 6 x 6 x 0.9 mm

dsPIC® Architecture Block Diagram



dsPIC® DSC Peripherals Overview

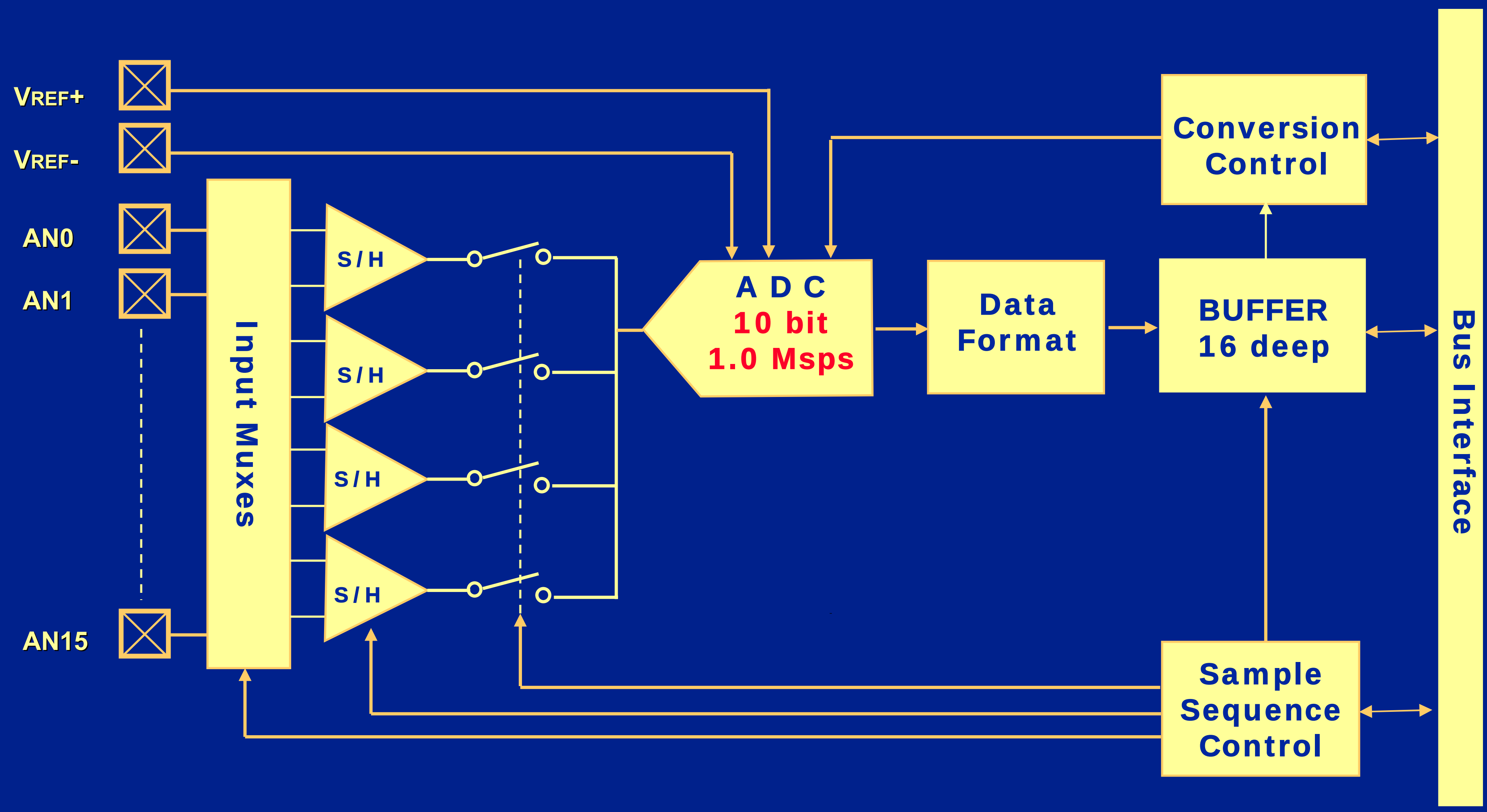
dsPIC Peripherals



A/D Converter

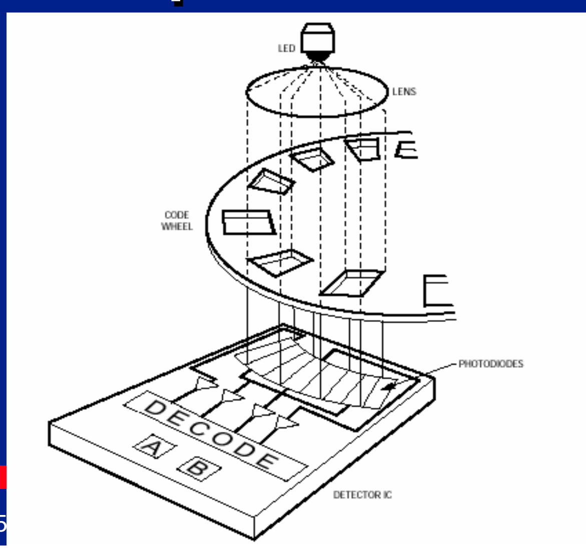
- 10-bit High Speed A/D
  - 10 bit resolution with ± 1 LSB accuracy
  - 1 Msps conversion rate
  - Up to 16 input channels, 4 S/H Amplifiers
  - Synchronization to the MCPWM time base
- 12-bit A/D
  - 12 bit resolution with ± 1 LSB accuracy
  - 200 ksps conversion rate
  - Up to 16 input channels, single S/H amplifier

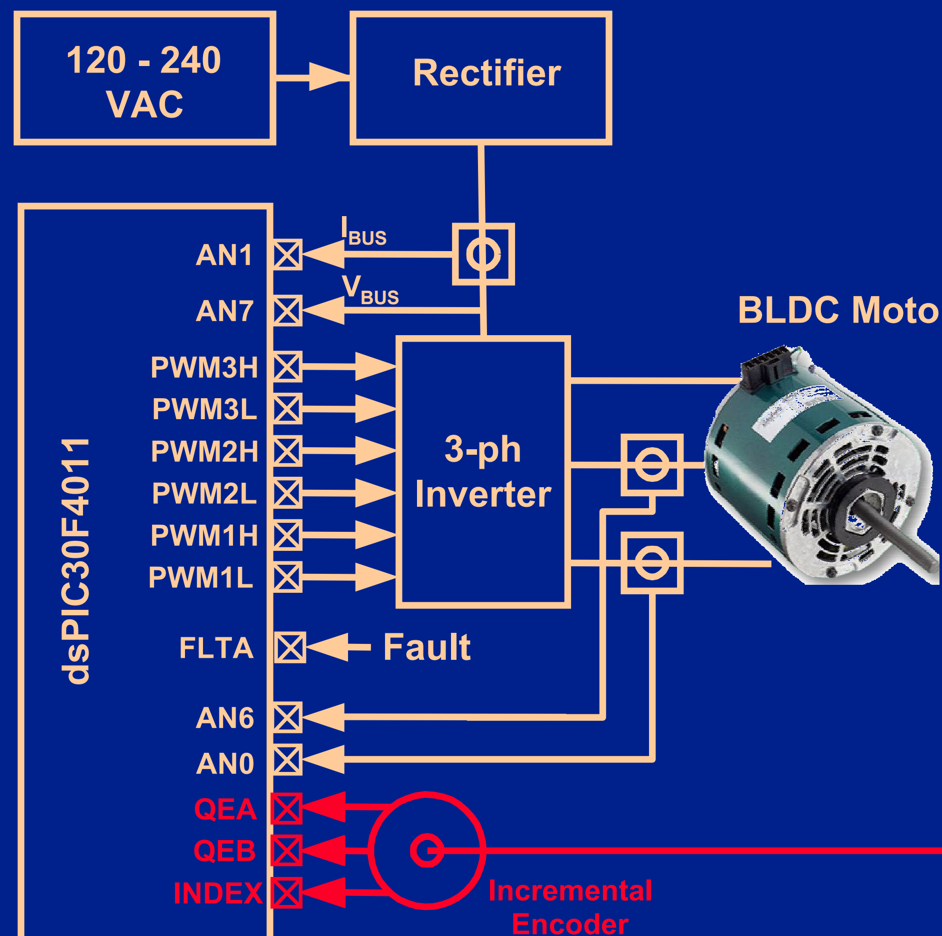
10-bit A/D Converter



Quadrature Encoder Interface

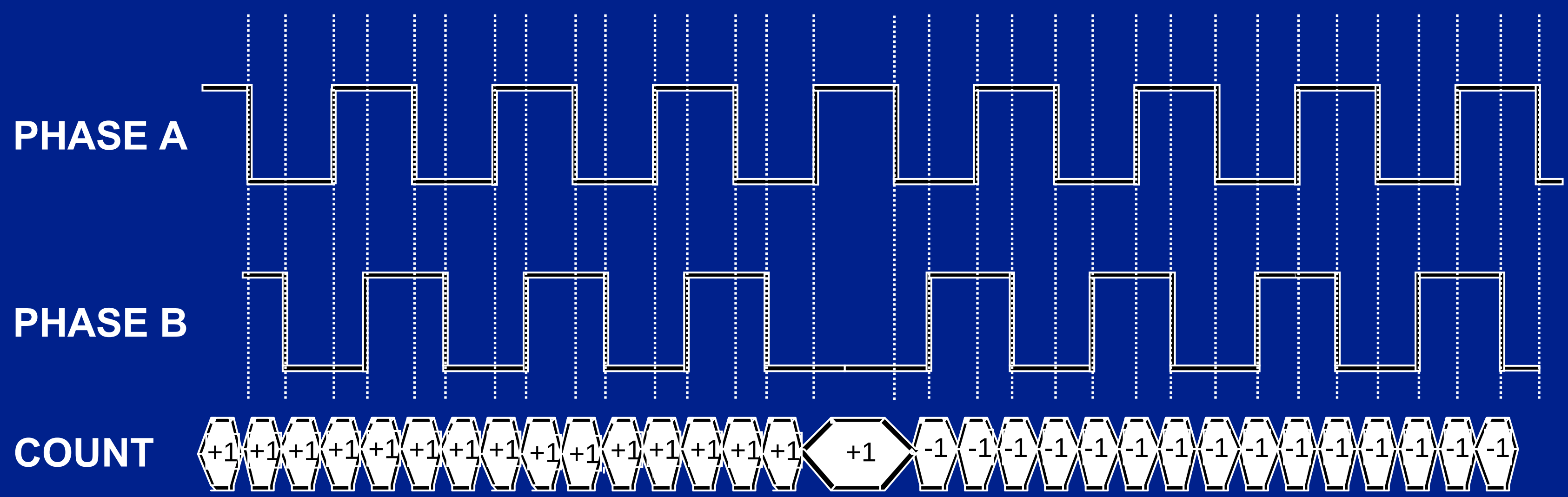
- QEI Module senses motor speed and position
- Three Input Quadrature Encoder
  - Phase A
  - Phase B
  - INDEX signals
- 16-bit position counter



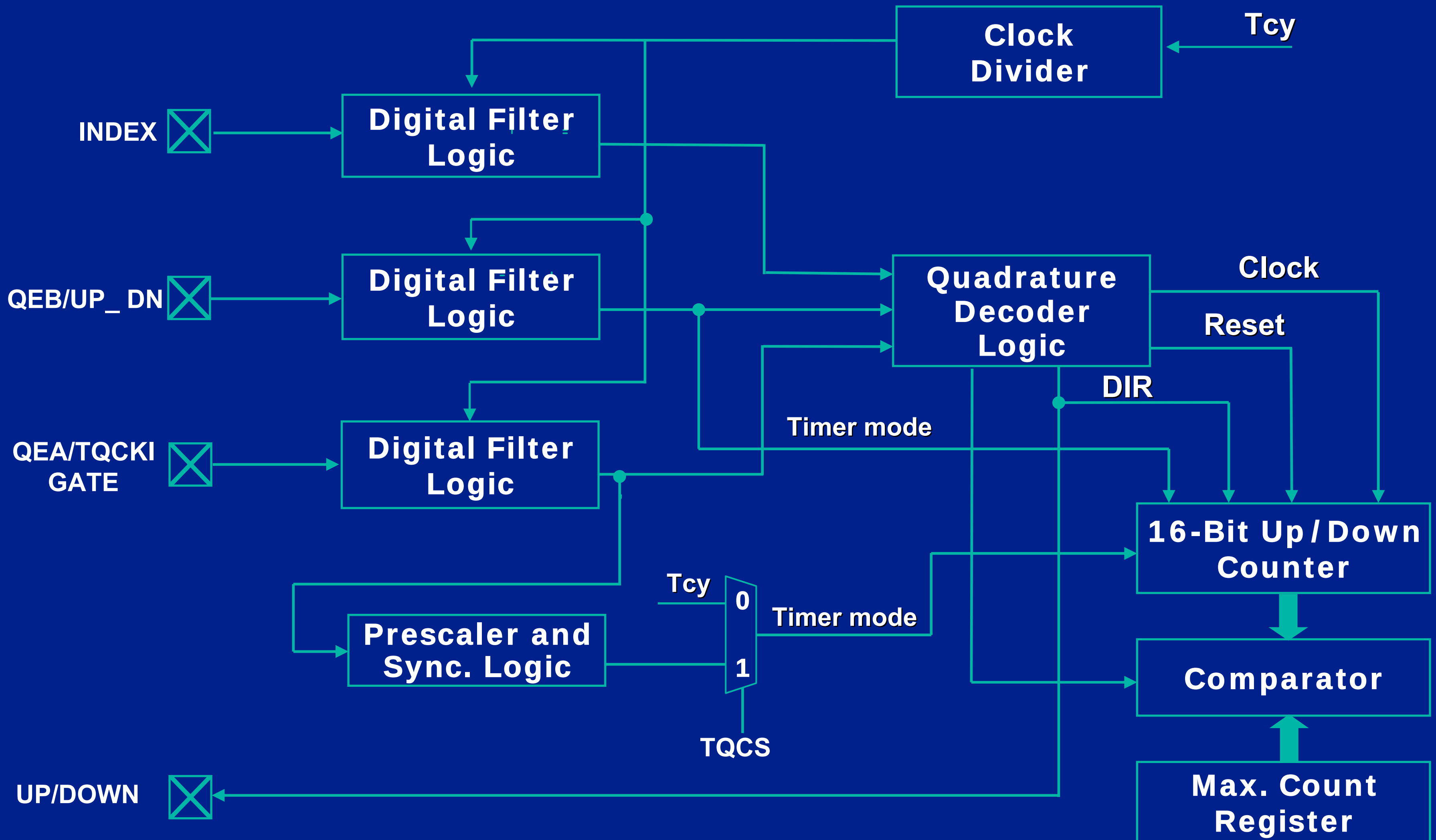


Quadrature Timing Diagram

- **State machine determines relative phase at each edge**



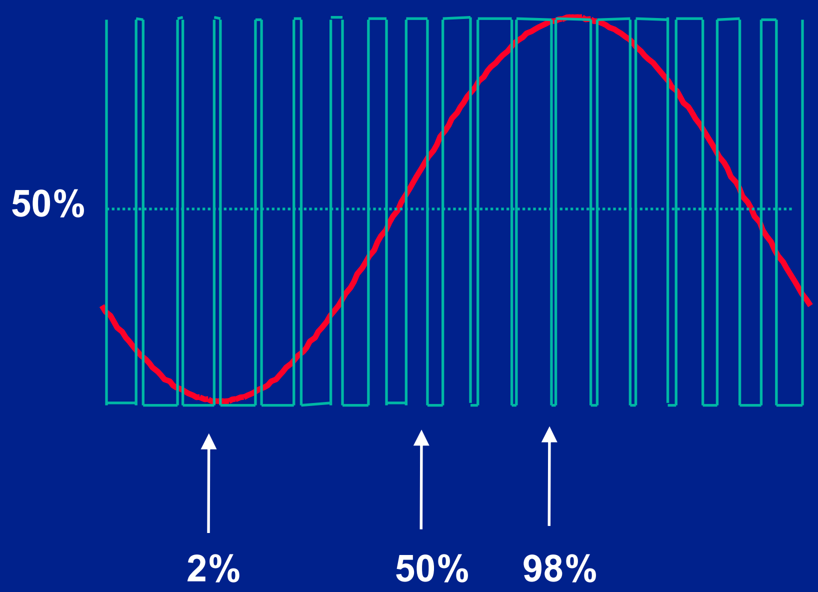
QEI Block Diagram



Digital Signal Controller

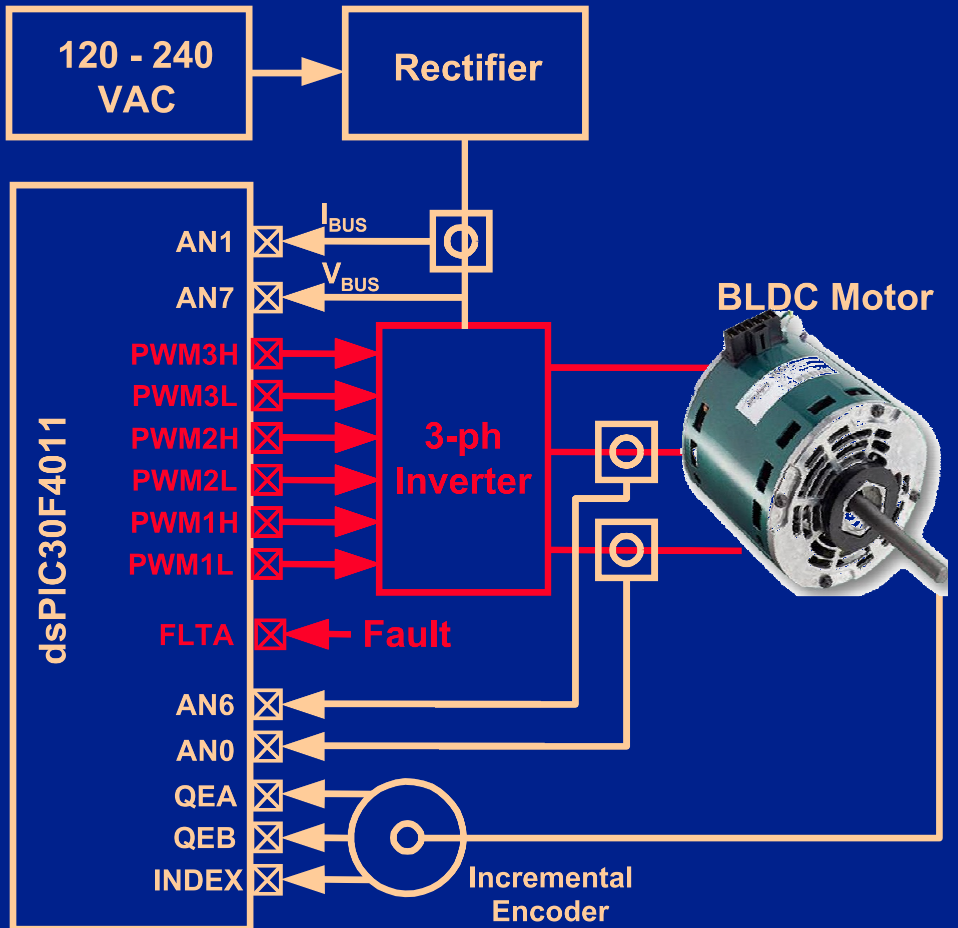
Pulse Width Modulation

- Allows fixed DC Input, AC output.
- Output voltage is PWM
- Motor integrates PWM voltage and produces sinusoidal current with small ripple at carrier frequency
- Minimal power loss in power transistors



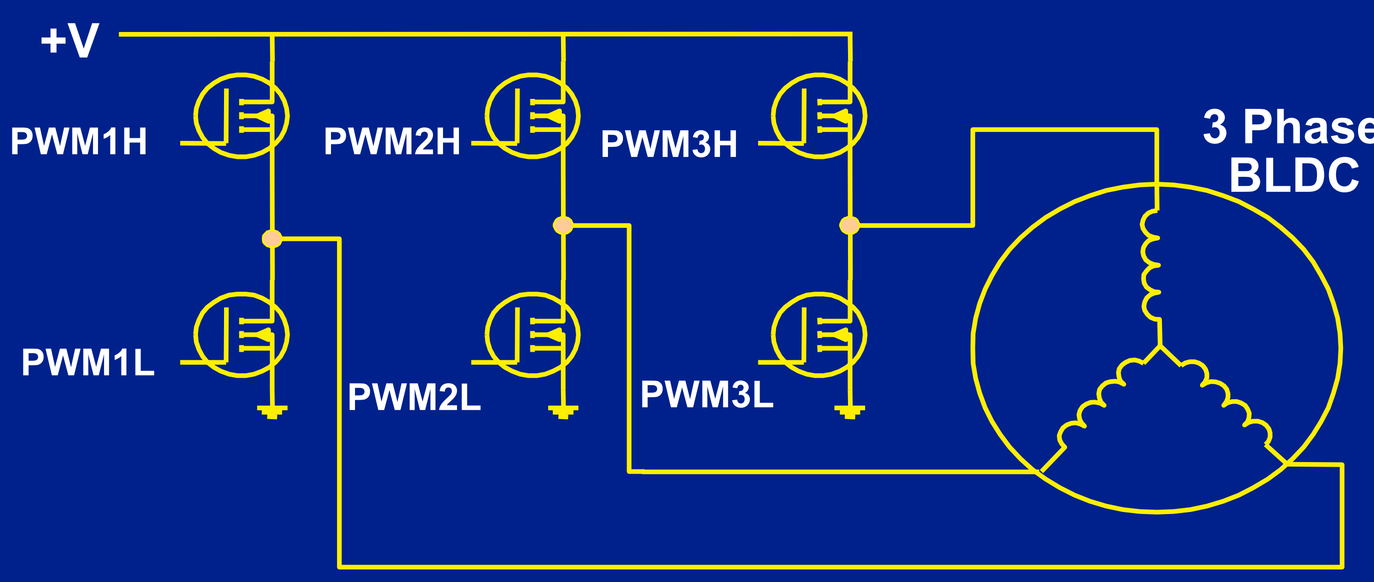
Motor Control PWM Module

- PWM Module drives motor
- Up to Four PWM generators
- Several options allow PWM to drive many circuit types
  - AC Motors
  - DC motors
  - Power supplies
- High frequency @ more bits = better control of motor operation
- Fault detection for safe operation

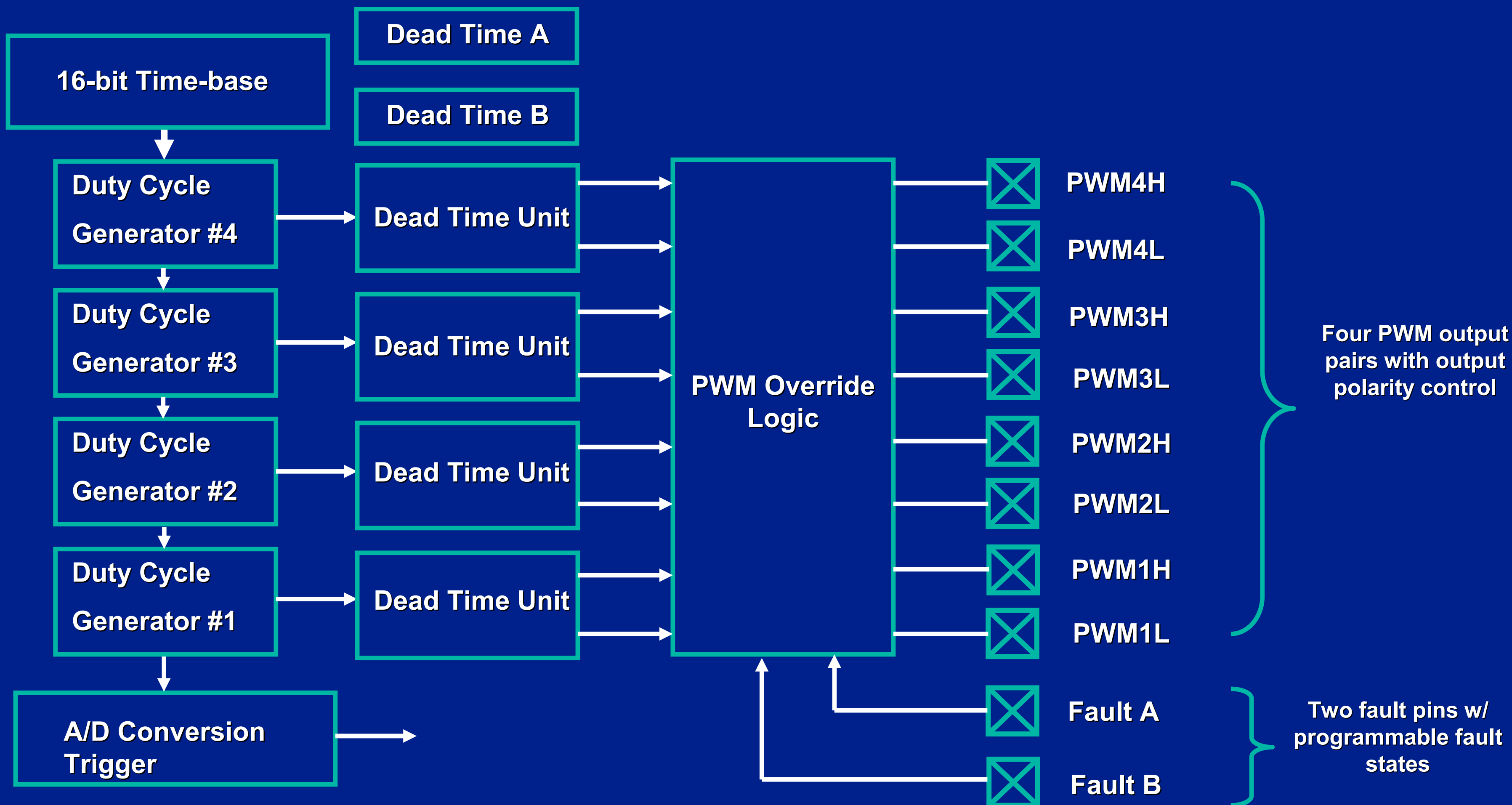


PWM with Inverter

- High Frequency Carrier
- Duty Cycle Varied Over Time to Generate a Lower Frequency Signal

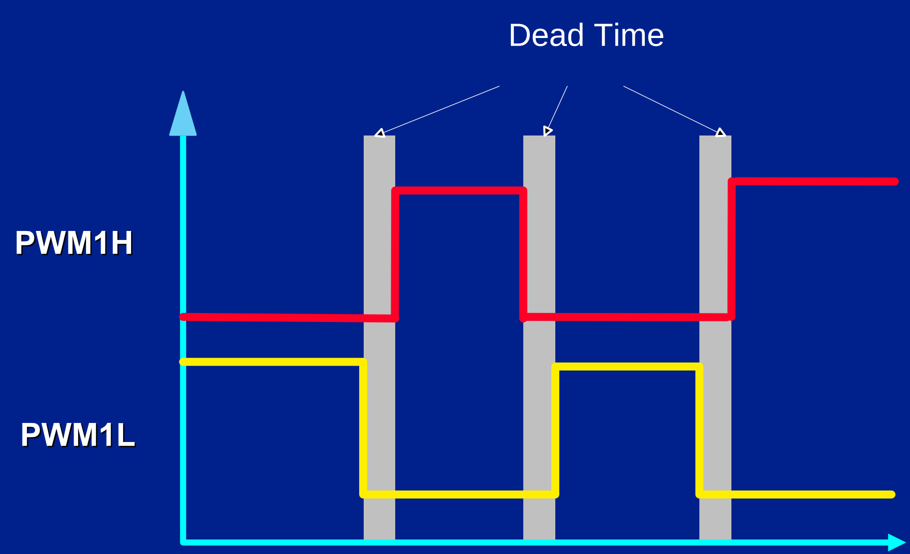


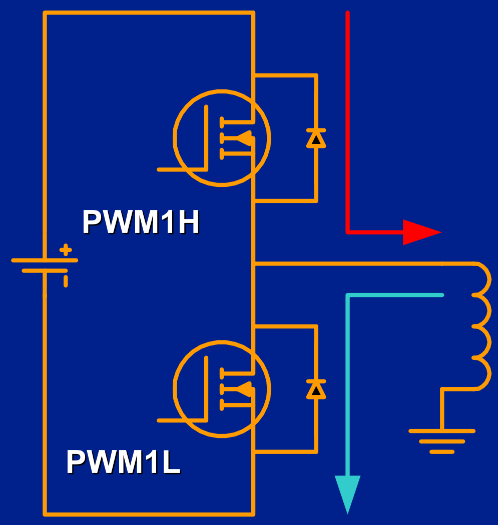
Motor Control PWM Block Diagram



Motor Control PWM

- Dead Time Insertion Example
  - Shoot Through is Prevented Automatically

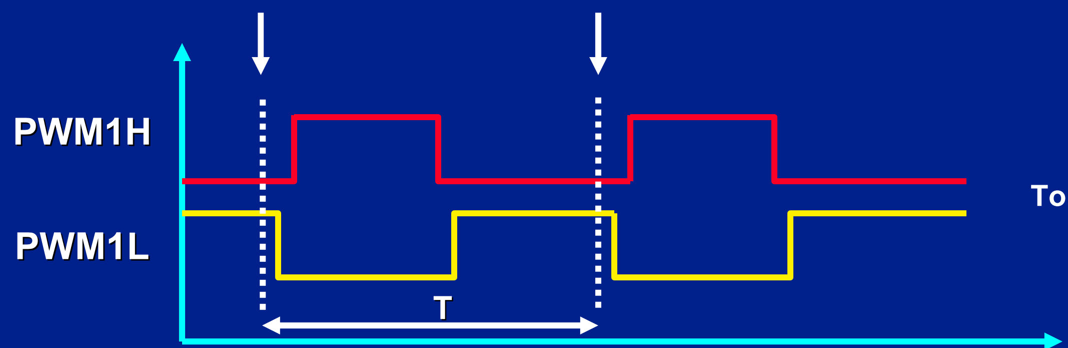


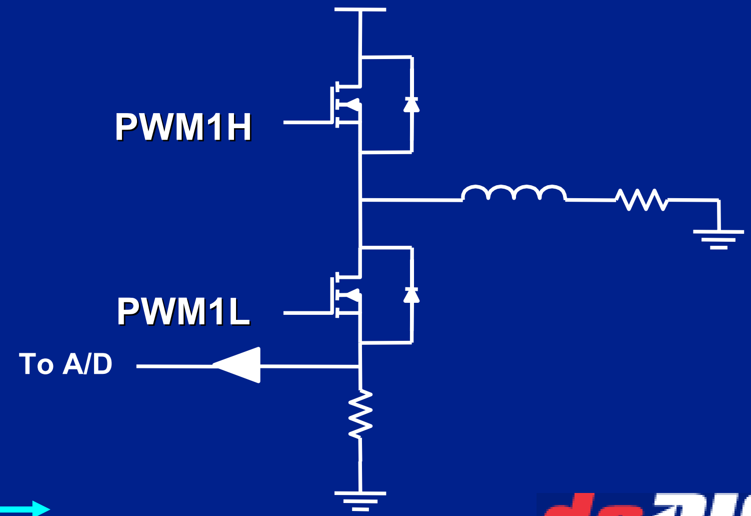


MCPWM A/D Synchronization

- SEVTCMP register sets A/D conversion start time in PWM cycle
- Ensure A/D properly captures shunt current
- Can also use to minimize control loop update delay

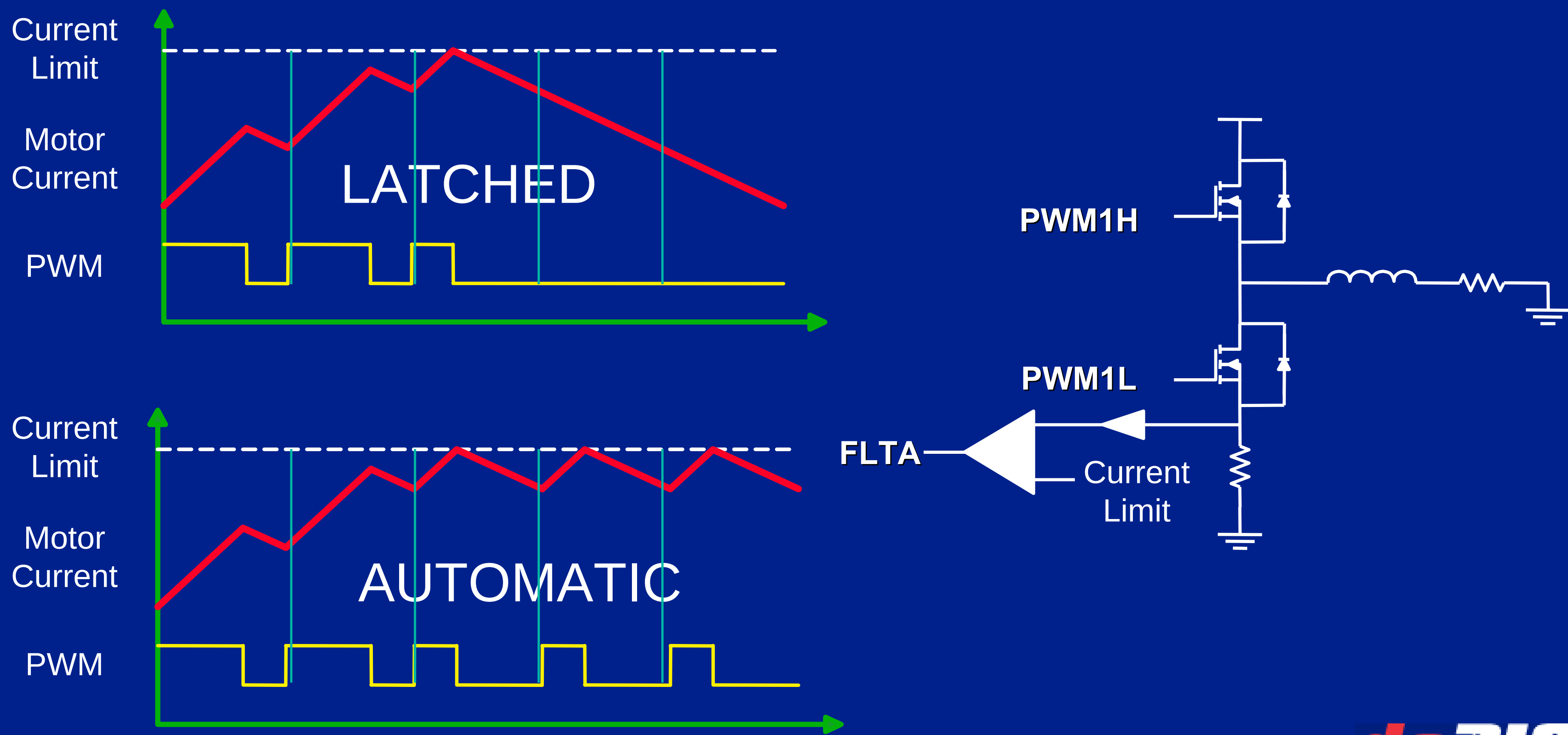
Trigger conversion at end of bottom transistor on-time





MCPWM Fault Inputs

- Automatic or latched fault protection
- Fault condition overrides all other pin control



MCPWM Override Control

- OVDCON (override control) register
  - Used for motor commutation
  - I/O pin can be PWM, active, or inactive
  - POVD =0, I/O pin is controlled manually
  - POUT bits set pin state for manual control
  - If Program is halted, PWM pins are turned OFF

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
POVD4H	POVD4L	POVD3H	POVD3L	POVD2H	POVD2L	POVD1H	POVD1L
bit15	14	13	12	11	10	9	bit8

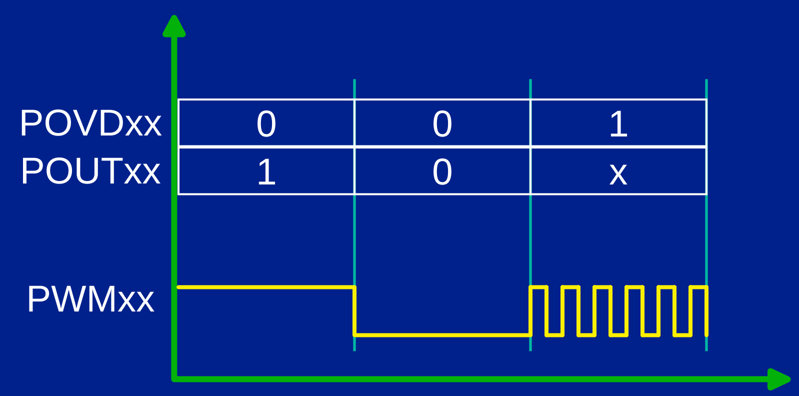
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
POUT4H	POUT4L	POUT3H	POUT3L	POUT2H	POUT2L	POUT1H	POUT1L
bit7	6	5	4	3	2	1	bit0

MCPWM Override Control

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
POVD4H	POVD4L	POVD3H	POVD3L	POVD2H	POVD2L	POVD1H	POVD1L
bit15	14	13	12	11	10	9	bit8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
POUT4H	POUT4L	POUT3H	POUT3L	POUT2H	POUT2L	POUT1H	POUT1L
bit7	6	5	4	3	2	1	bit0

POVDx	POUTx	PWM Output	Inactive Output	Active Outpu t
0	**1**			✓
0	**0**		✓	
1	**x**	✓		



Lab 1. Programming a dsPIC® DSC Using the PICDEM™ MCLV Board

Objectives of Lab1

- Getting to know the hardware in front of you
- Where are the Labs located?
  - C:\WIB\Lab1\Lab1.mcw
- How to load the lab projects
- Programming the dsPIC® DSC devices
- Running the program on dsPIC DSC

You should have….

1) MPLAB® IDE V7.20 or higher installed
2) Complete MPLAB ICD 2 setup. R20 or Latest Rev.
3) PICDEM™ MCLV board
4) 24V power supply for the board
5) Hurst (NTDynamo®) BLDC motor with
   - Power cable (4 wires with white square connector) and
   - Hall sensor cable (5 wires with 8-pin inline black connector)

LAB 1

- What we will do:
  - Configure board hardware connections
  - Open a workspace in MPLAB® IDE
  - Compile or build a simple first project in MPLAB IDE
  - Follow a procedure to program the dsPIC using MPLAB ICD 2
  - Follow a procedure to run the program using MPLAB ICD 2

Training board

28-pin MC dsPIC30F

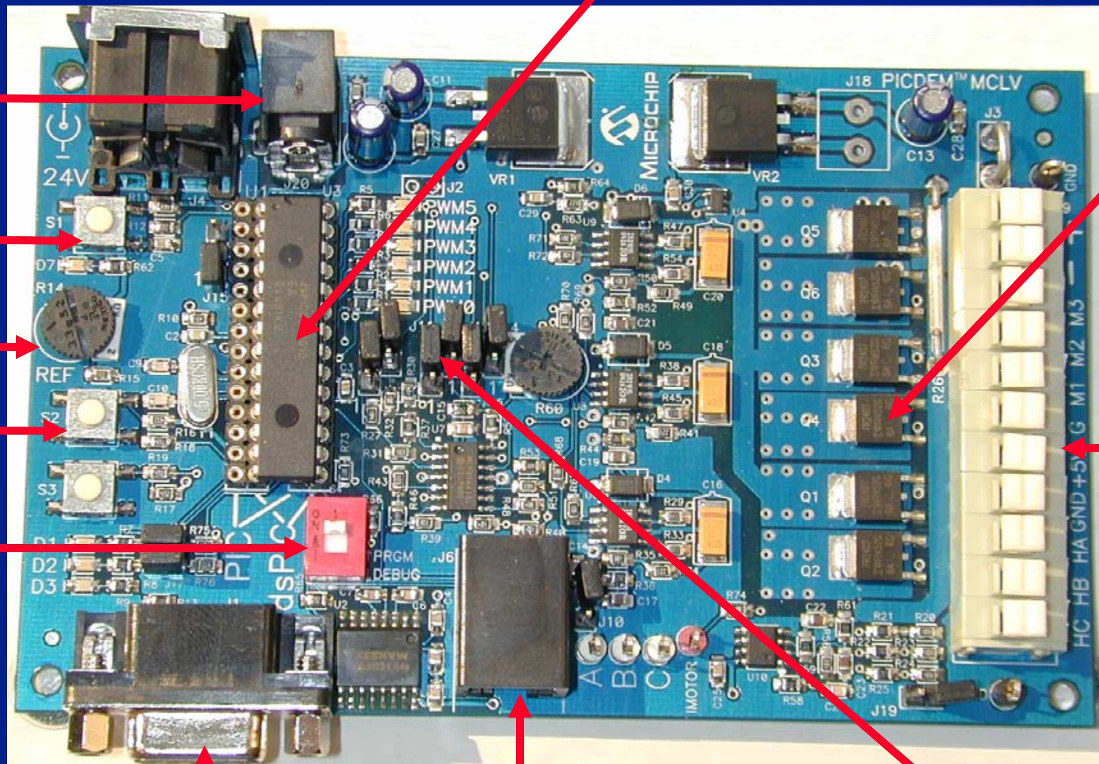
Power

Reset

Pot

S2

DIP Switch



3-Phase Inverter

Motor connector

Serial connector

MPLAB® ICD 2 connector

Jumpers
J7, J8, J11, J12, J13, J14

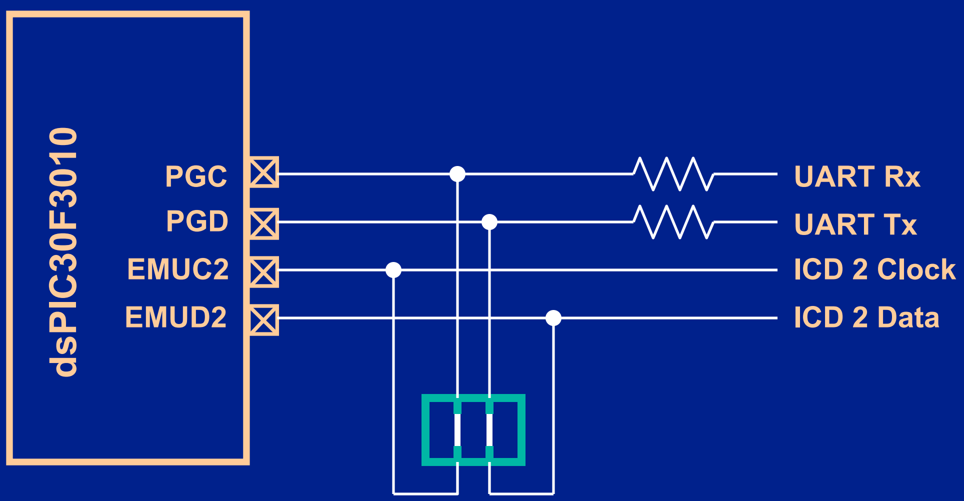
Default Jumper Settings

- The Jumper settings are printed on the under side of the PICDEM™ MCLV board
- Turn board over to view and set Jumper settings.
- Use "dsPIC® DSC Sensored" setting for Lab 1
- Keep Potentiometer REF(R14) and R60 in center position

dsPIC® DSC Sensored Settings

Jumper	Position
J7	NC
J8	NC
J11	NC
J12	NC
J13	NC
J14	NC
J15	NC
J10	NC
J16	1-2
J17	1-2
J19	1-2

Debug / Program DIP Switch



S2 Position	Function
Closed	PRGM
Open	DEBUG

Lab1

- Instructions for Lab1:
  - On PICDEM™ MCLV board, move DIP switch to “PRGM” position
  - Connect power to PICDEM MCLV board
  - Open MPLAB® IDE by double clicking on icon
  - In MPLAB, select “File -> Open Workspace”
  - Browse to “C:\WIB\Lab1\Lab1.mcw”
  - Select “Lab1.mcw” and open workspace

Continued...

Lab1 (contd.)

- **Instructions for Lab1:**
  - **In MPLAB® IDE, Select “Project -> Build All”**
  - **IF NO errors then ...**
  - **In MPLAB IDE, Select “Debugger -> Program” to program dsPIC® DSC**
  - **On MCLV board, move DIP switch to “DEBUG” position**
  - **In MPLAB IDE, Select “Debugger -> Run”**
  - **Press S2 on PICDEM™ MCLV board and PWM LEDs will be blinking**

Lab1 Results

- Follow Lab1 for programming and running software:
  - Before programming dsPIC® DSC, move DIP to “PRGM” position
  - Before running, move DIP to “DEBUG” position
- Each lab has a already created workspace in the appropriate folder
- Use the created workspace for each lab

BLDC Motor Introduction

I. Basic Motor Theory

- **What is a Motor?**

A Motor Converts Electrical Energy to Mechanical

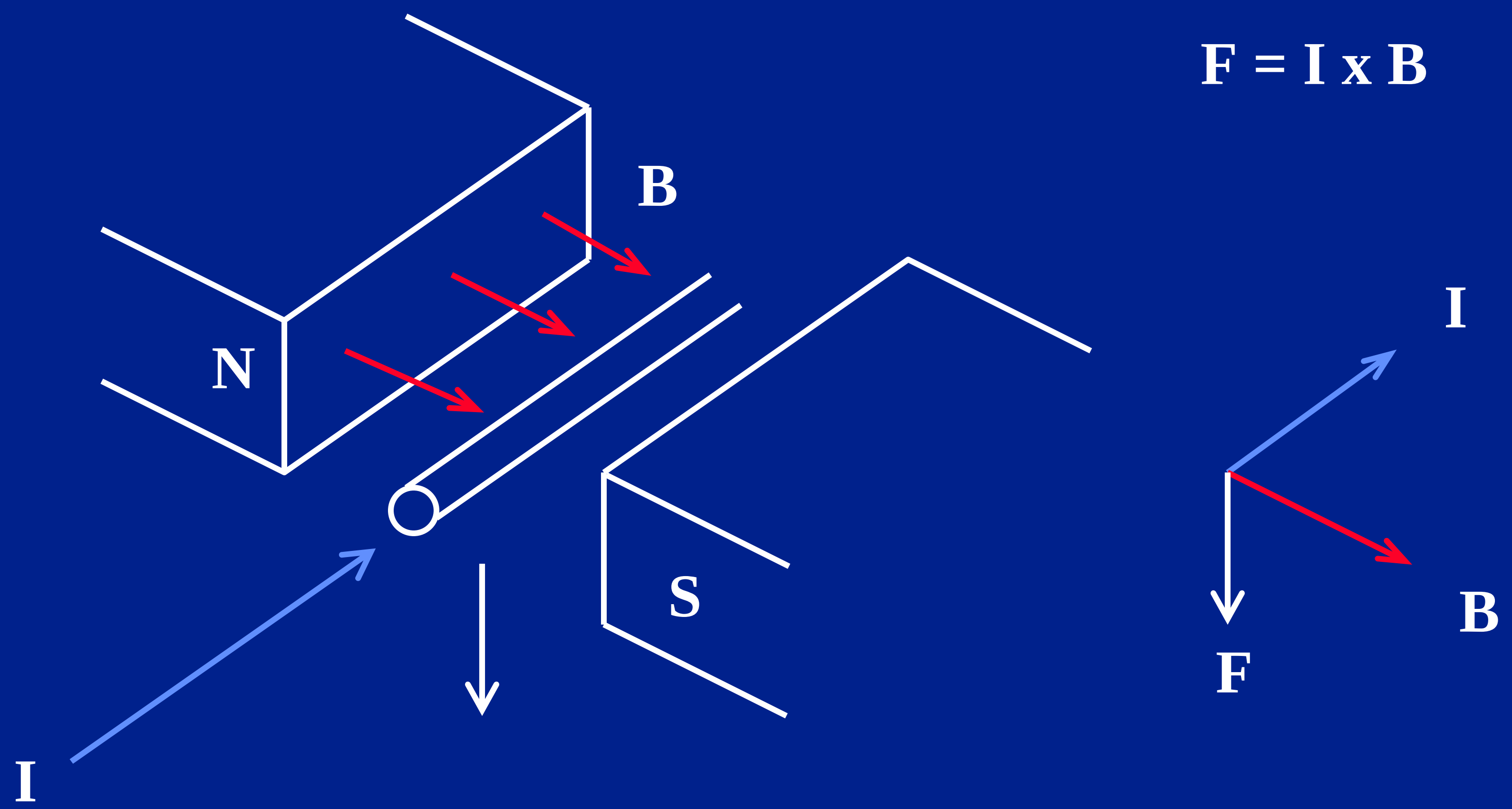
How?

Force is developed when charge moves through a magnetic field

$$F = I \times B$$

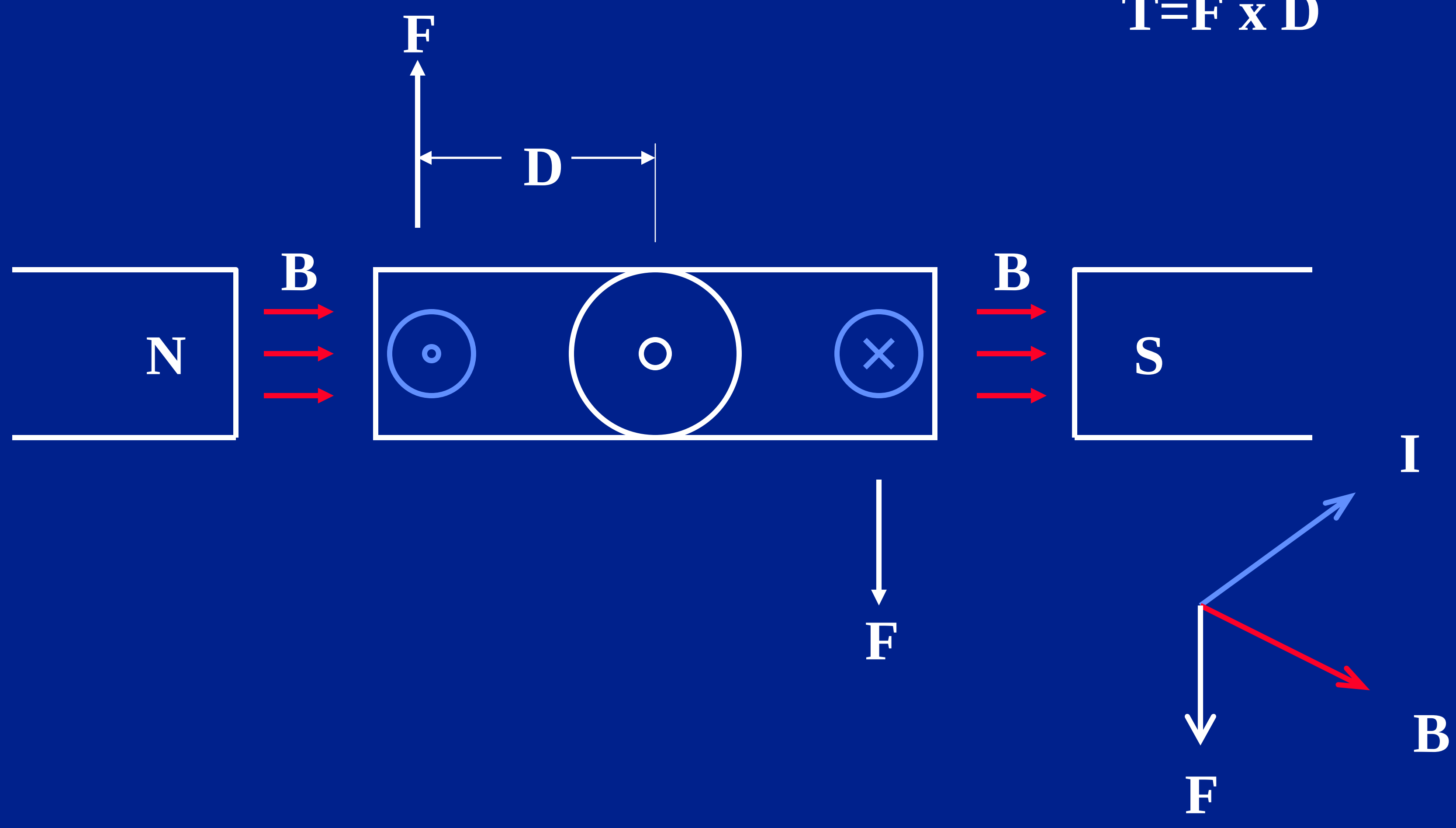
Left Hand Rule

$F = I \times B$



Motor Torque

T=F x D

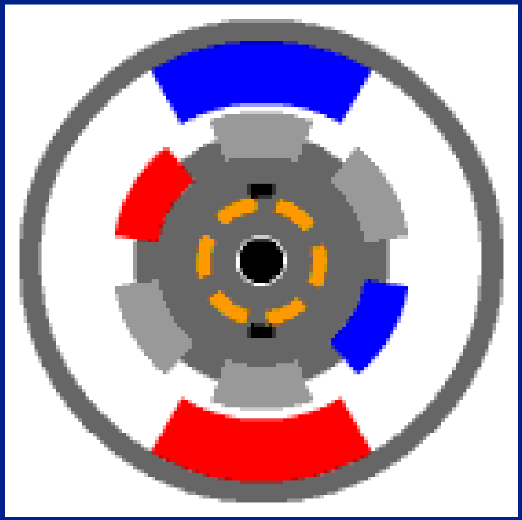


Digital Signal Controller

DC Motor Torque

- Summary
- Torque = Force * Distance
  - F = I x B
  - T = (I x B) * D
  - When B and D are constant $T = K * I_A$
  - When field is wound $B = K * I_F$
  - In wound DC motors Torque and Flux B can be controlled independently

DC Motor



- Red is North Polarization
- Blue is South Polarization
- Opposite Polarities attract
- Rotor will rotate until North is aligned with South
- Just before alignment, commutator contacts and energize next winding
- Spark is generated when the commutator change windings

The Brushless DC Motor (BLDC)

- An inside out brushed DC motor with electronic commutation
- A modern, much improved, version of the traditional brushed DC motor
- Field, which has relatively low loss, is generated on the rotor using permanent magnets
- Armature, which causes the majority of the loss, is on the stator which has good cooling

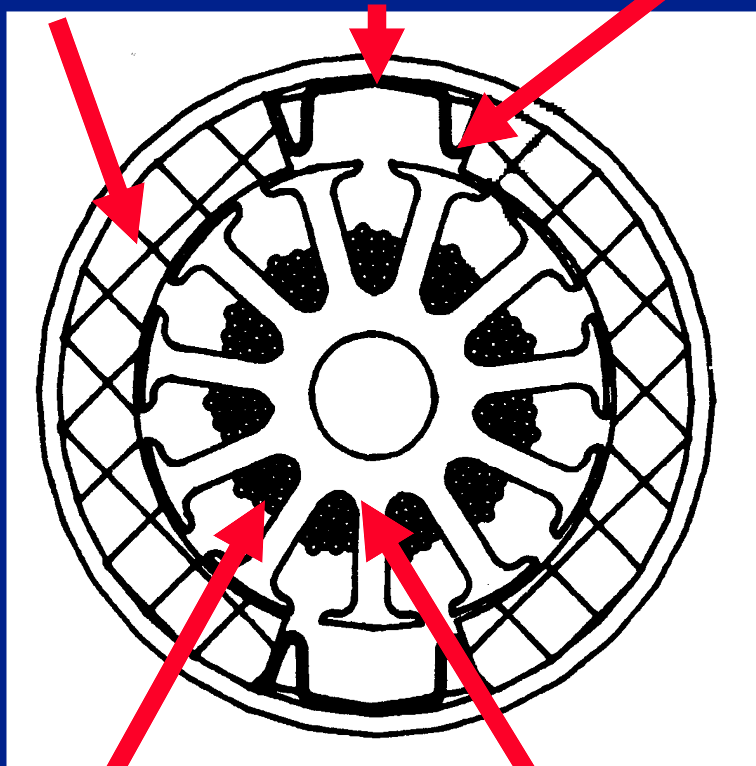
Brushed & Brushless DC Motor Construction

PERMANENT MAGNET BRUSHED DC MOTOR

Permanent Magnet

Stator

Brushes



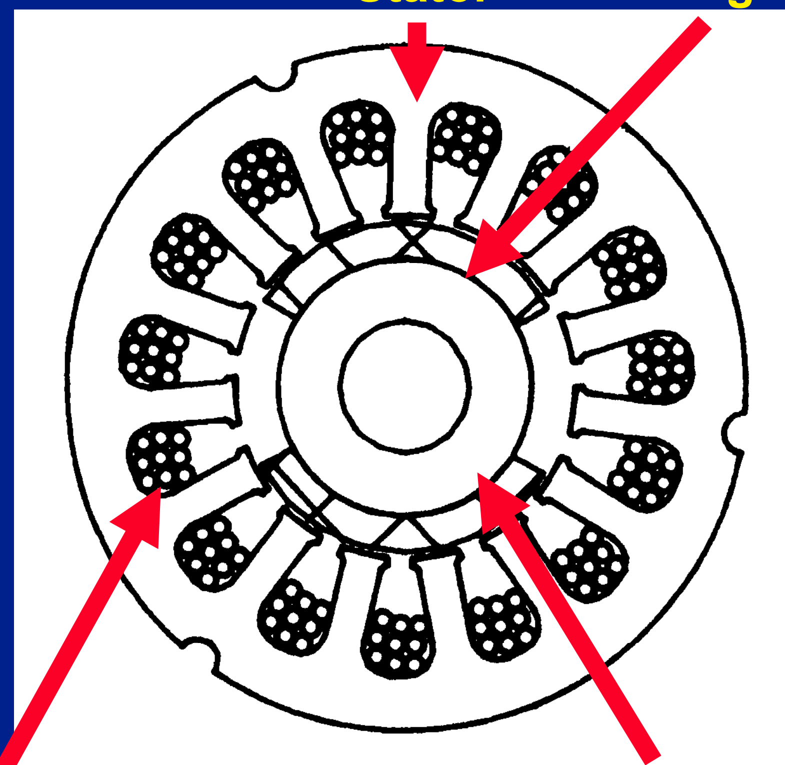
Windings

Rotor

PERMANENT MAGNET BRUSHLESS DC MOTOR

Stator

Permanent Magnet

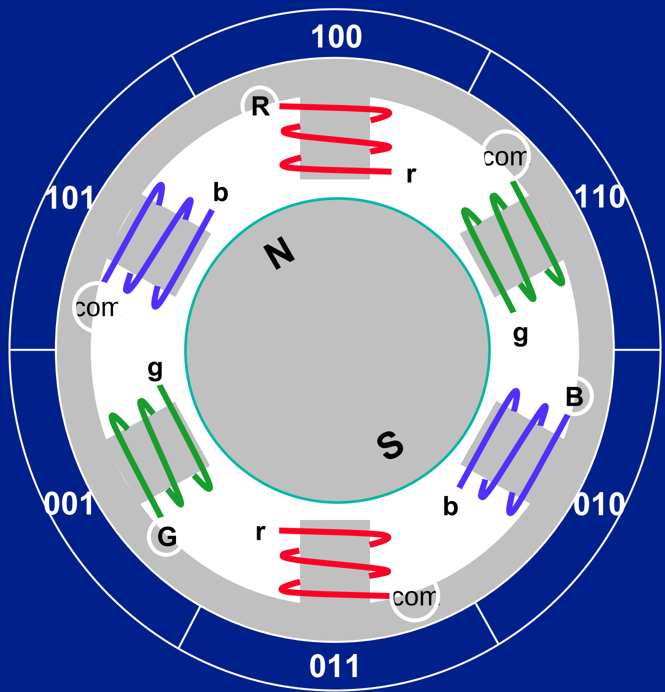


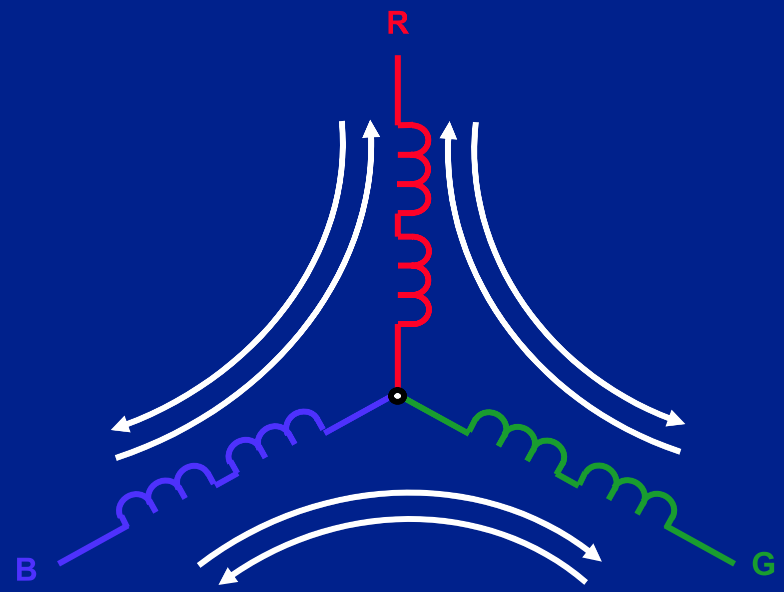
Windings

Rotor

Digital Signal Controller

Brushless DC Motor Energization





BLDC Advantages Over Bushed DC Motor

- High Efficiency
- More Reliable – No Brushes to Maintain
- Higher Speeds
- Higher Power/Size Ratio
- Heat is Generated in Stator – Easy to Remove
- Lower Inertia – No commutator
- Higher Acceleration Rates
- No Arcing on Commutator

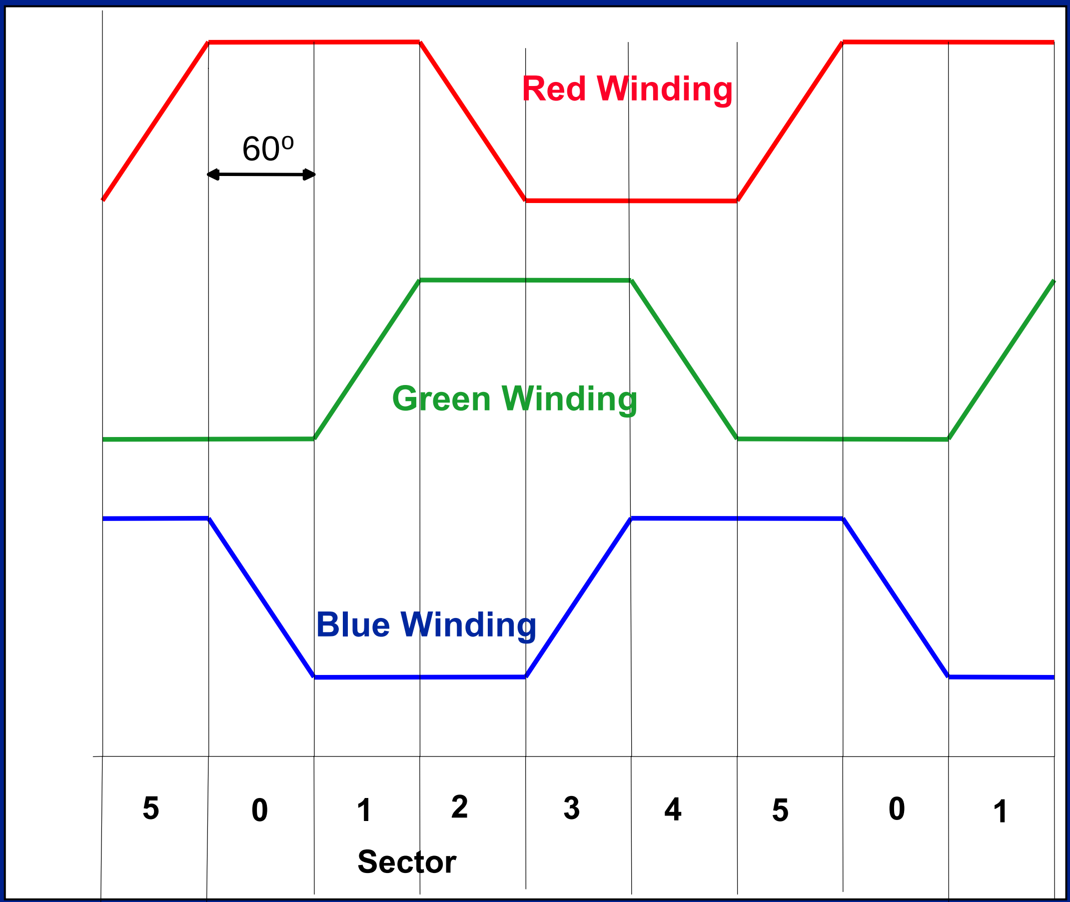
BLDC Control

- Mechanical commutator replaced by electronic switching
- BLDC is a synchronous motor
- Meaning that switching must be synchronized to rotor position

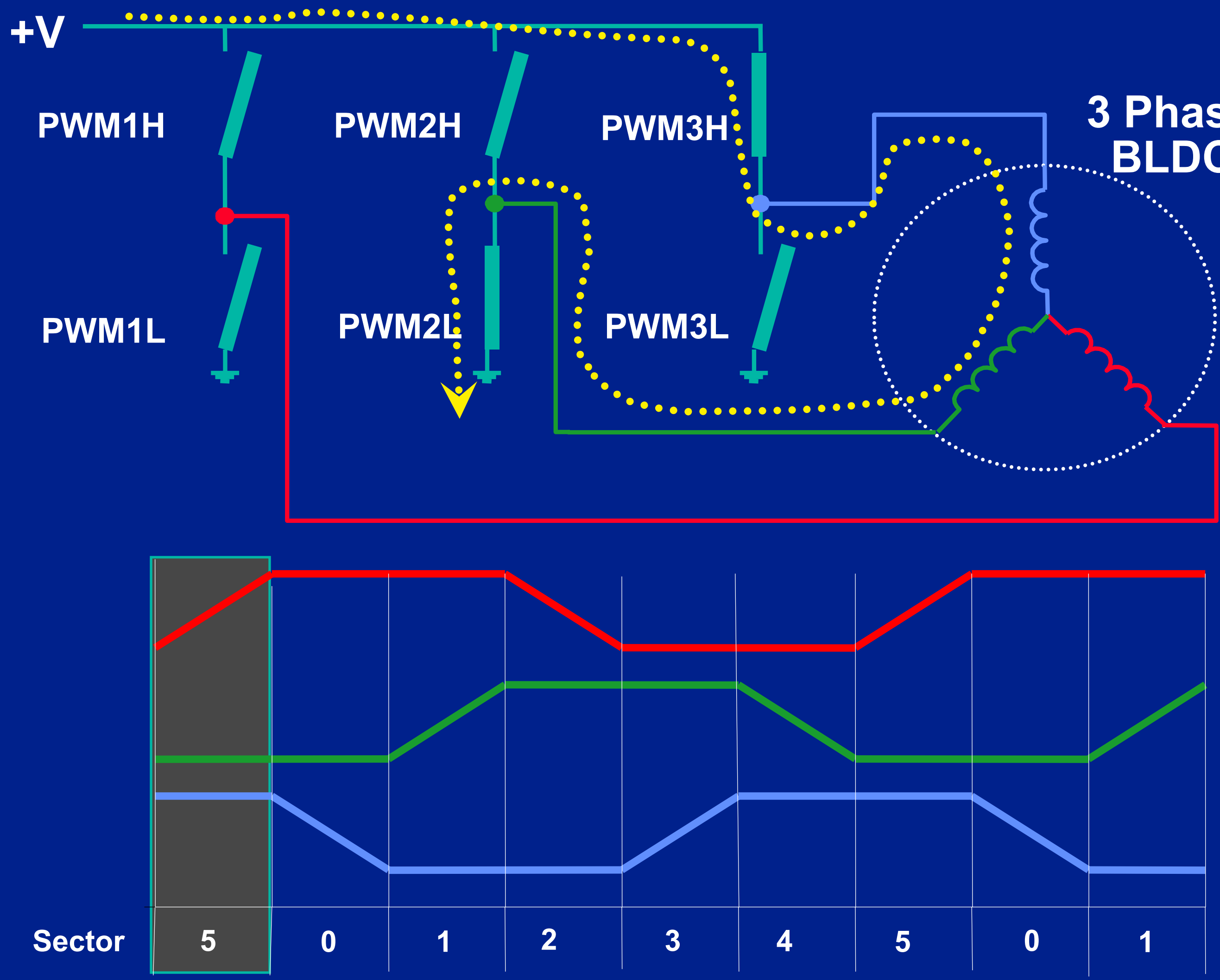
Lab 2. Running a BLDC Motor with Forced Commutation

Running a BLDC Motor with Forced Commutation

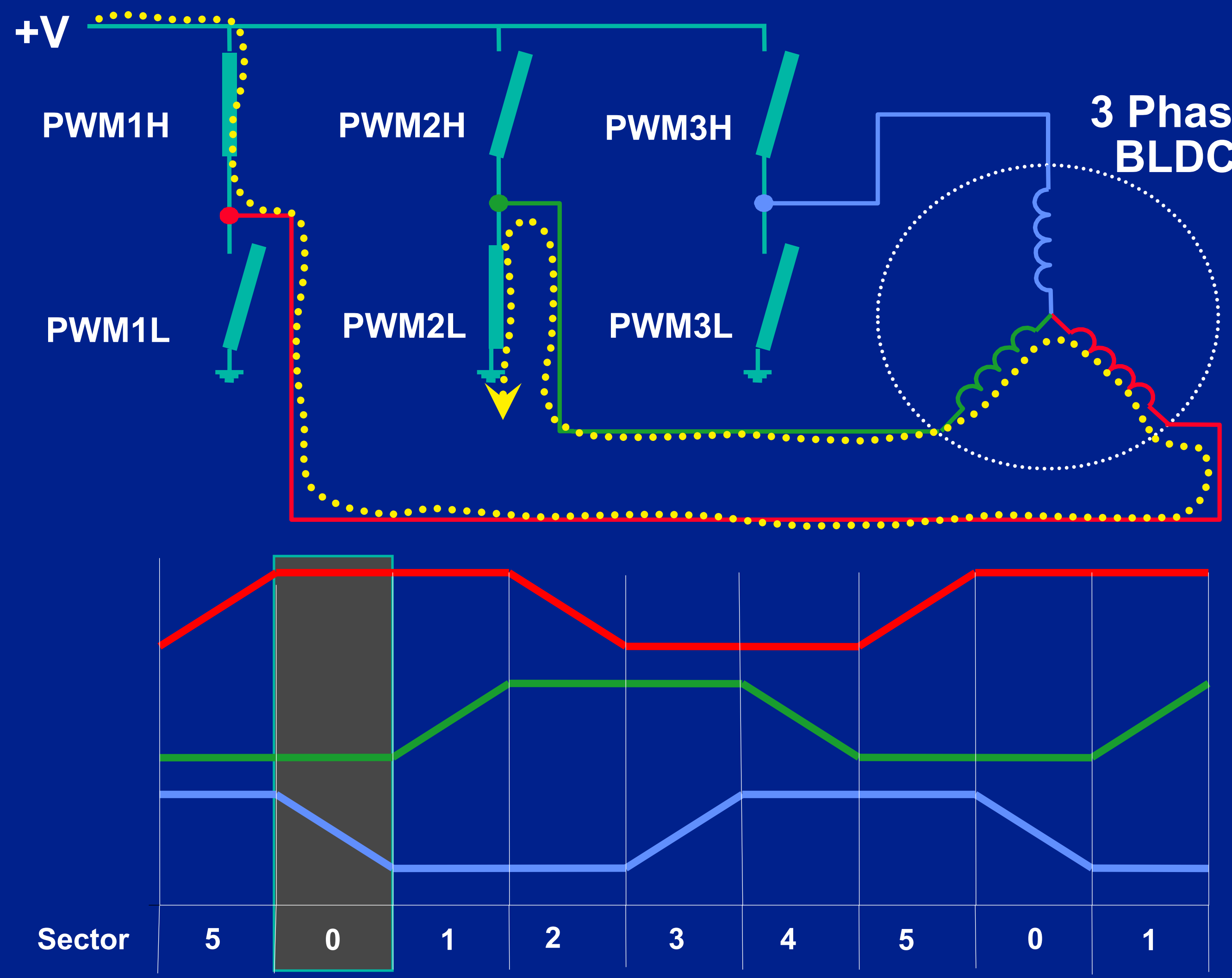
- Consider sector 5
- Blue Winding = 24V
- Green Winding = 0V
- Red Winding = OFF
- Delay for a short time
- Repeat process for all 6 sectors.
- Revolving Electrical field will cause rotor to rotate



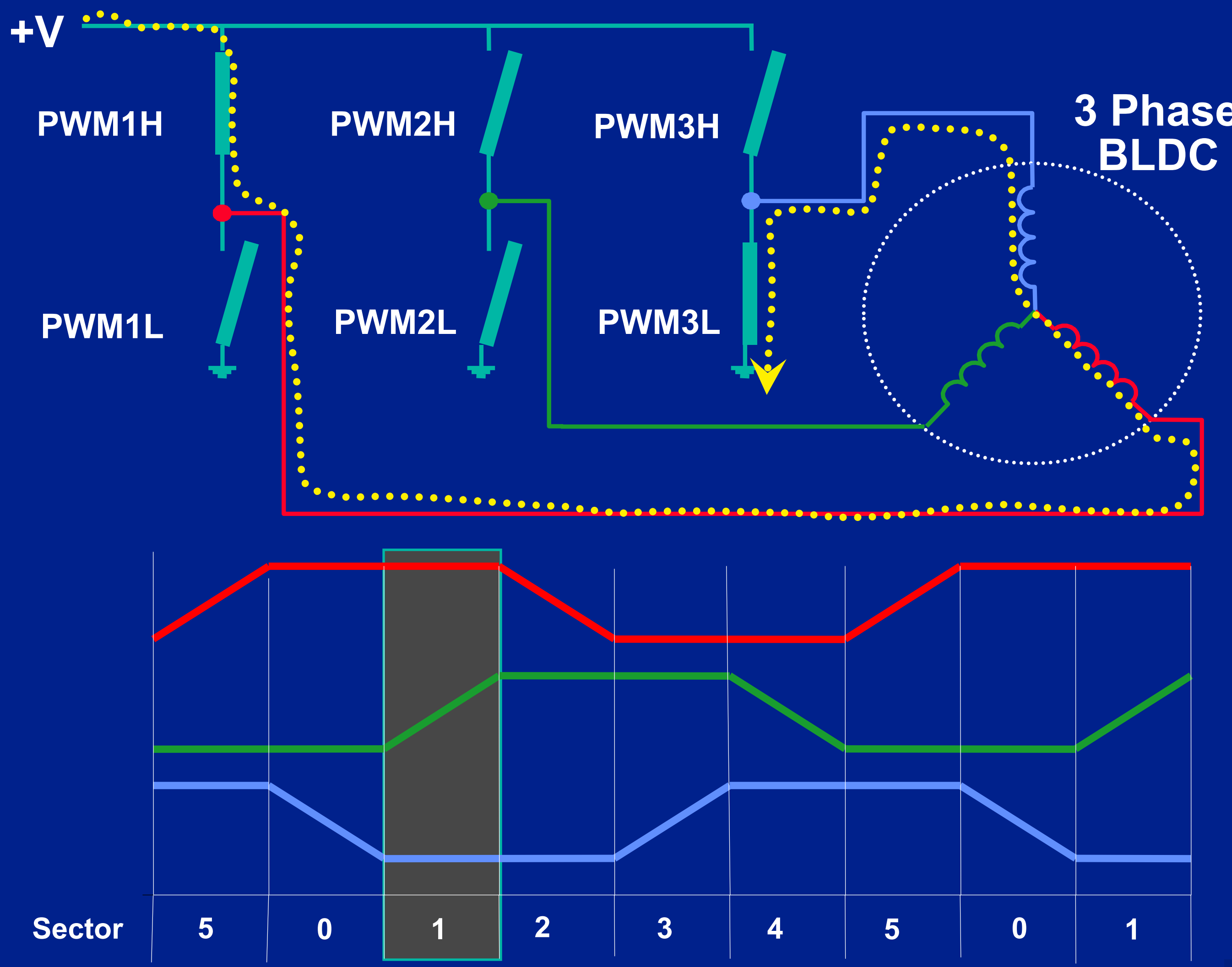
Six-Step Commutation with Inverter



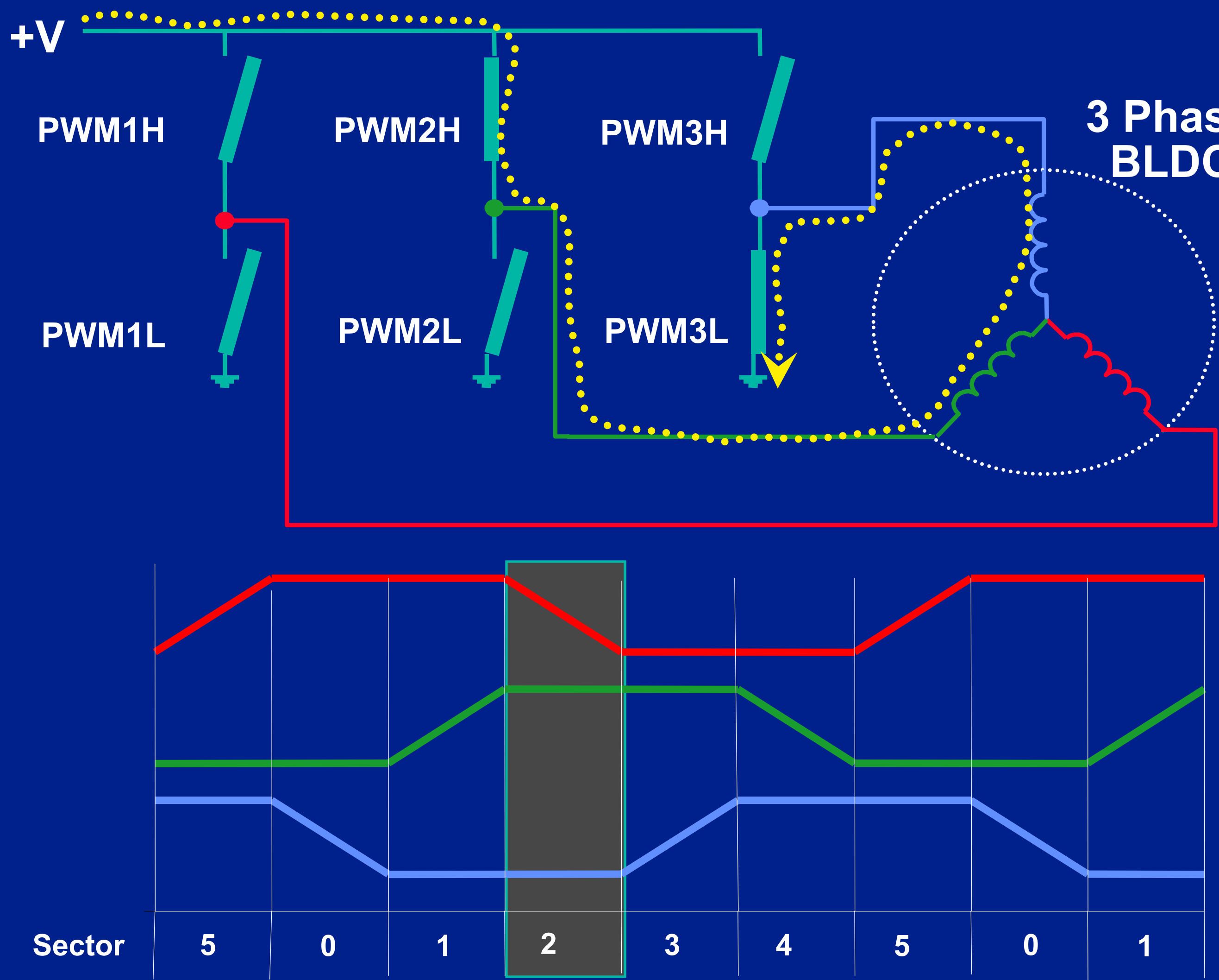
Six-Step Commutation with Inverter



Six-Step Commutation with Inverter



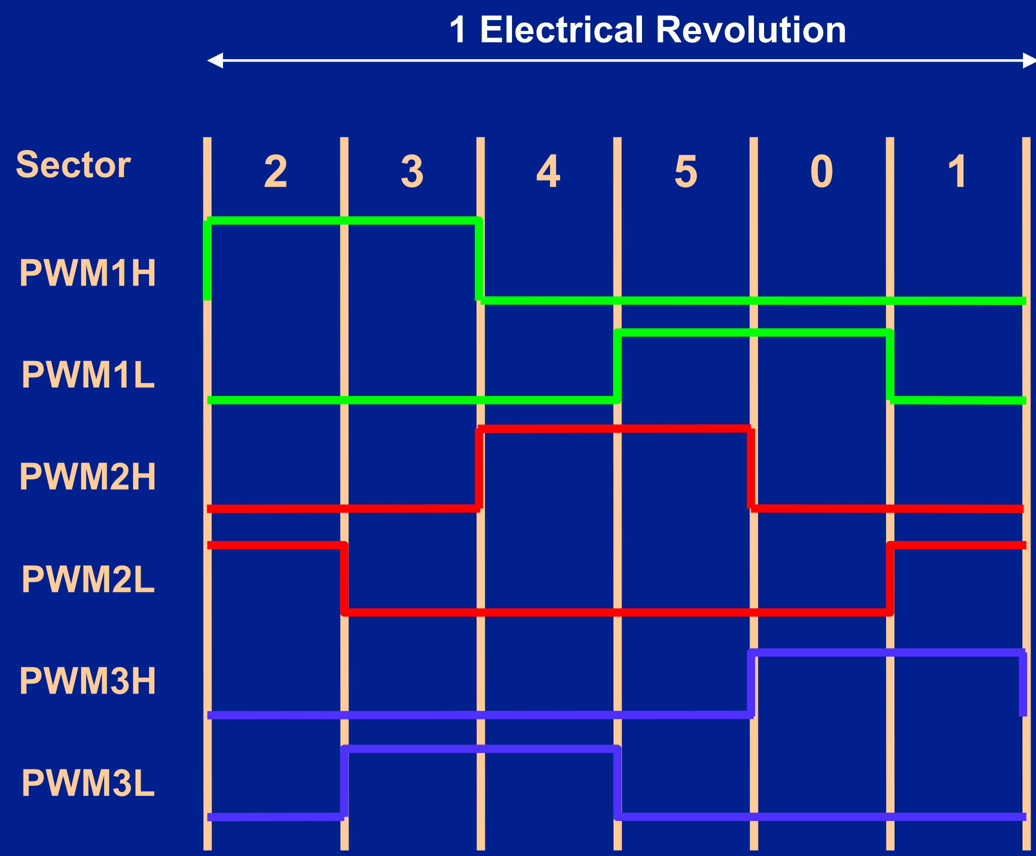
Six-Step Commutation with Inverter



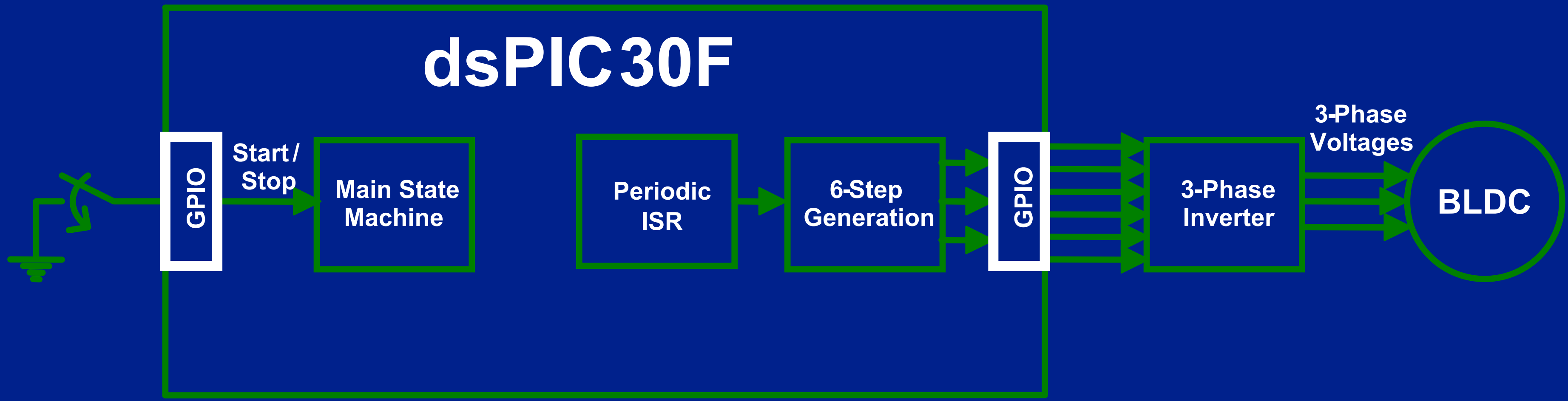
Motor Control PWM

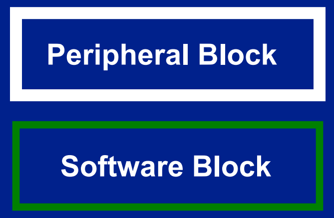
- **Using OVDCON for 6-Step Commutation**

Sector	OVDCON Value	
	POVD<7:0>	POUT<7:0>
0	00000000	00100001
1	00000000	00100100
2	00000000	00000110
3	00000000	00010010
4	00000000	00011000
5	00000000	00001001



Running a BLDC Motor with Forced Commutation





Lab2 – Running a BLDC Motor with Forced Commutation

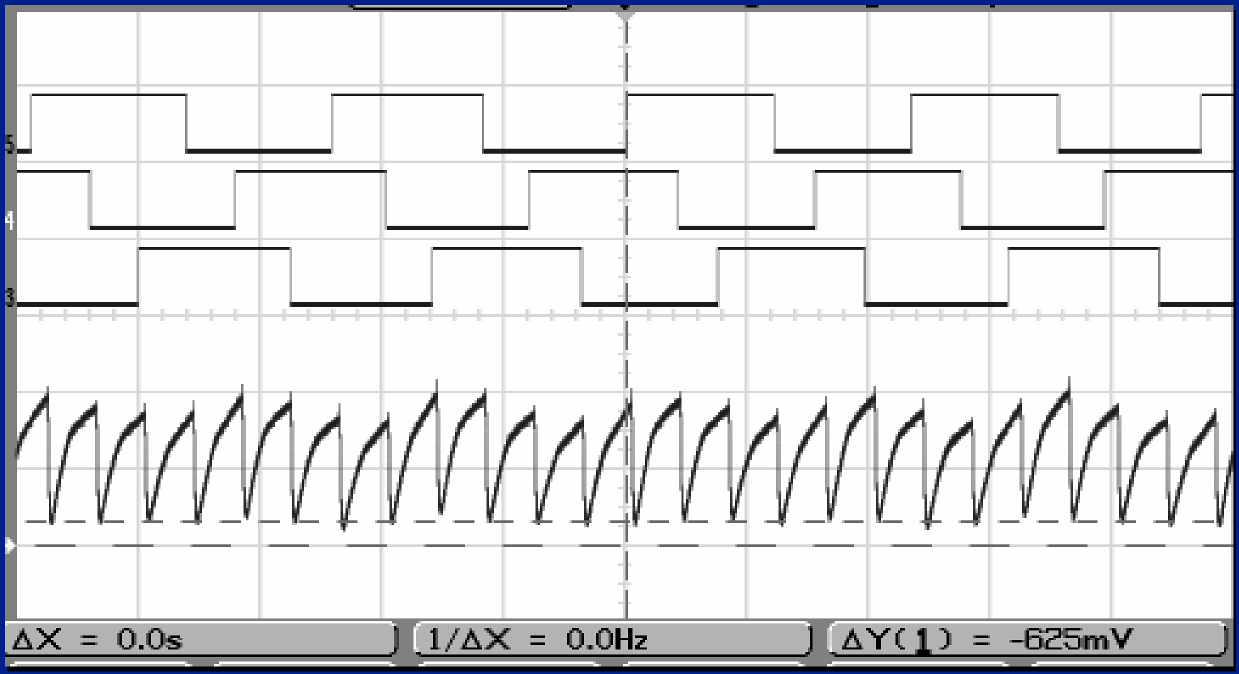
- Instructions for Lab2:
  - Use workspace “C:\WIB\Lab2\Lab2.mcw”
  - Follow Lab 1 instructions to:
    - Compile code
    - Program dsPIC® DSC
    - Run code

Continued...

Lab2 – Running a BLDC Motor with Forced Commutation

- Press S2 to start motor
- Notice that the motor is running rough and loud (almost screeching)
- Notice that the motor is getting warm.
- WHY?
- Press S2 to Stop the motor

Lab2 – Running a BLDC Motor with Forced Commutation



Fixed Speed
1000 RPM

4 Amps Peak

Details of Program

- **Use MPLAB® IDE to go thru sections of the code**

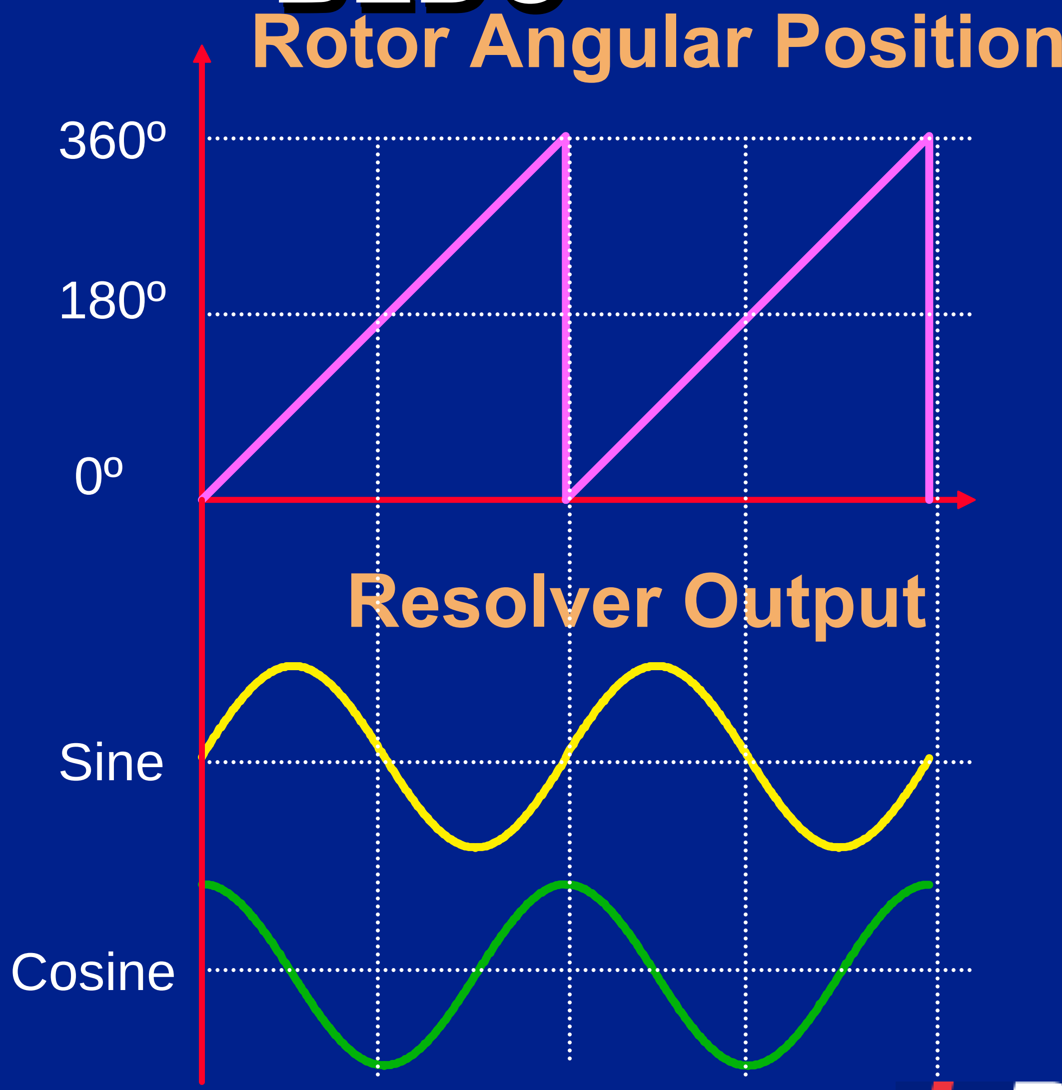
Lab2 Results

- First experience with a BLDC motor
- Understand “Six-Step Commutation” using the Override Feature of the dsPIC® DSC
- Spinning a BLDC motor without position sensing
- Very inefficient with high currents (up to 4 amps with no load)
- Understanding the need of position feedback

Sensing Position of a BLDC

Resolver

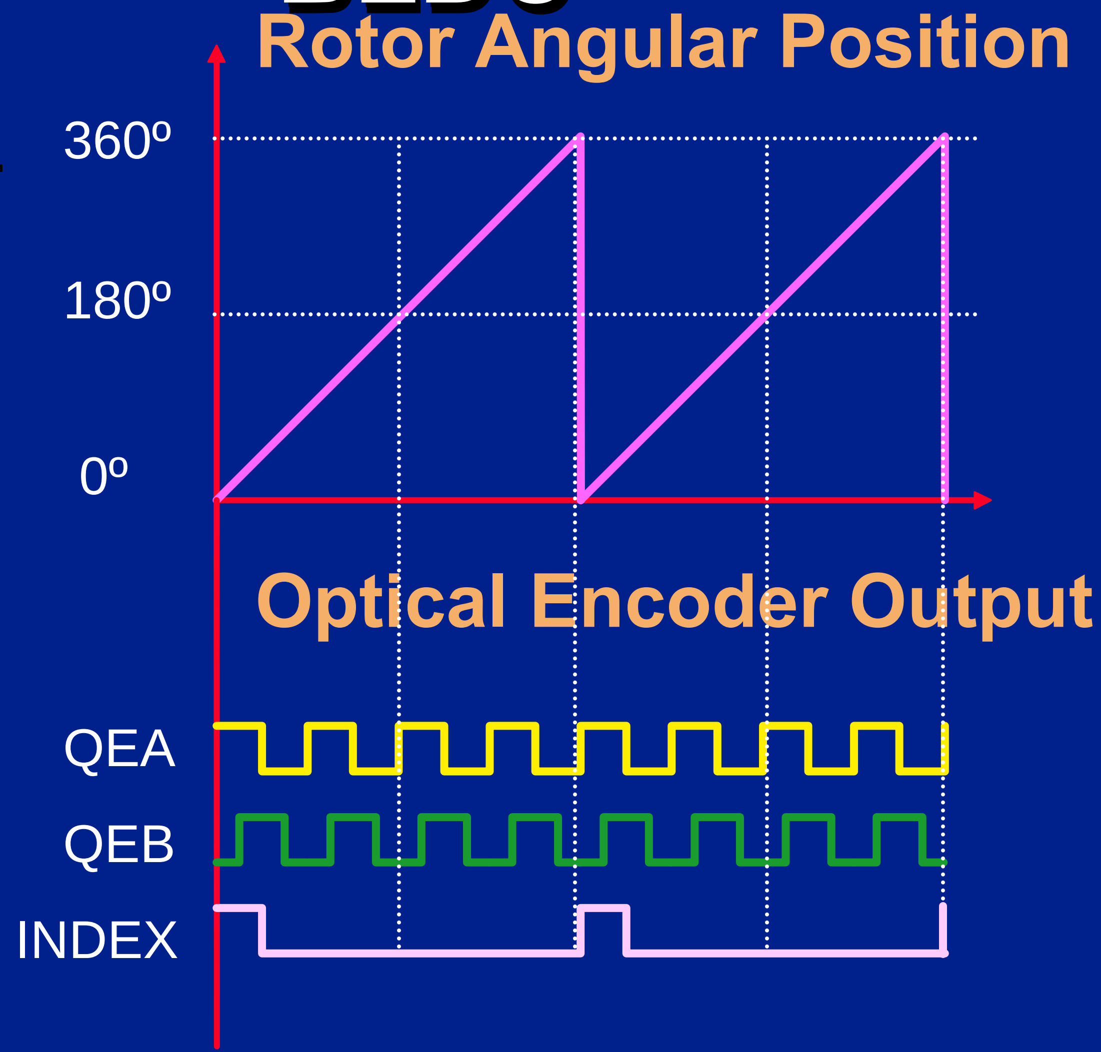
- Higher Resolution. (i.e. 1024 Different States per Rev)
- A/D Module + Processing Power
- Resolver Externally Mounted (More Expensive)
- Provides Absolute position feedback



Sensing Position of a BLDC

Optical Encoder

- High Resolution. (i.e. 500 Interrupts per Rev)
- Special QEI Module + Some Math
- Optical Encoder Externally Mounted (Expensive)
- Useful for servo applications due to resolution

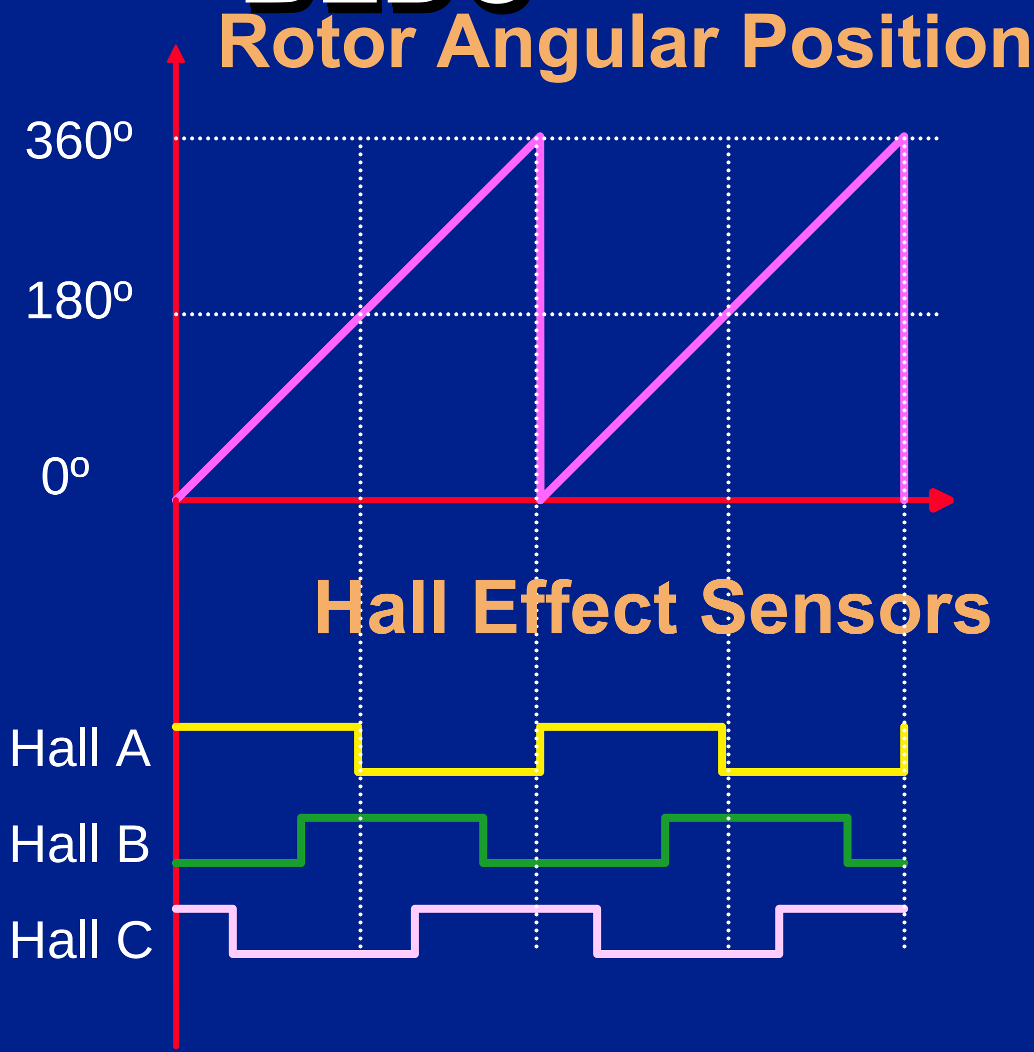


dsPIC Digital Signal Controller

Sensing Position of a BLDC

Hall Effect

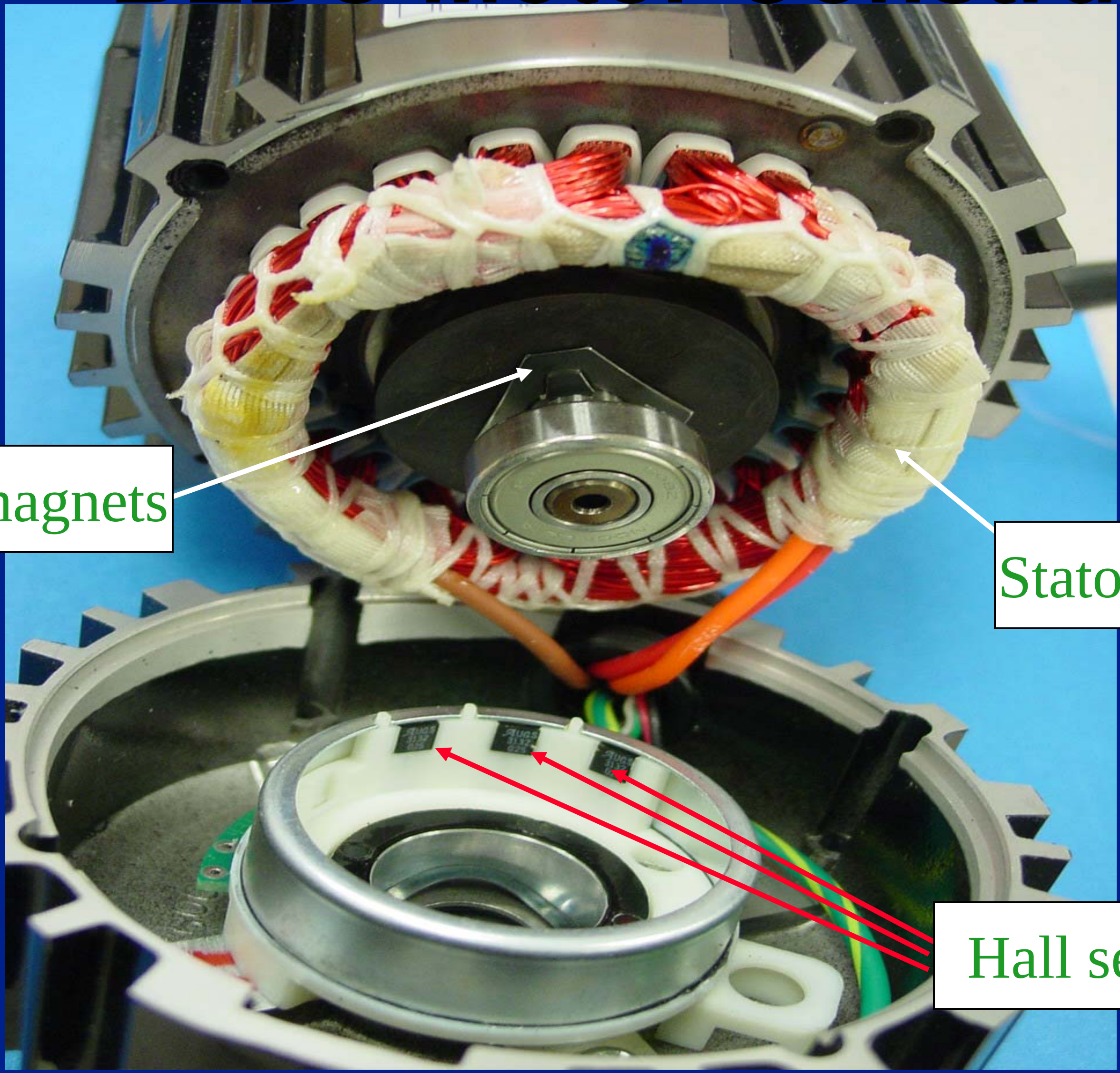
- Low Resolution (i.e. 30 Interrupts per Rev)
- Simple External Interrupt I/Os
- 1 to 3 Hall effect sensors (Less Expensive)
- Standard position sensing for low-cost applications



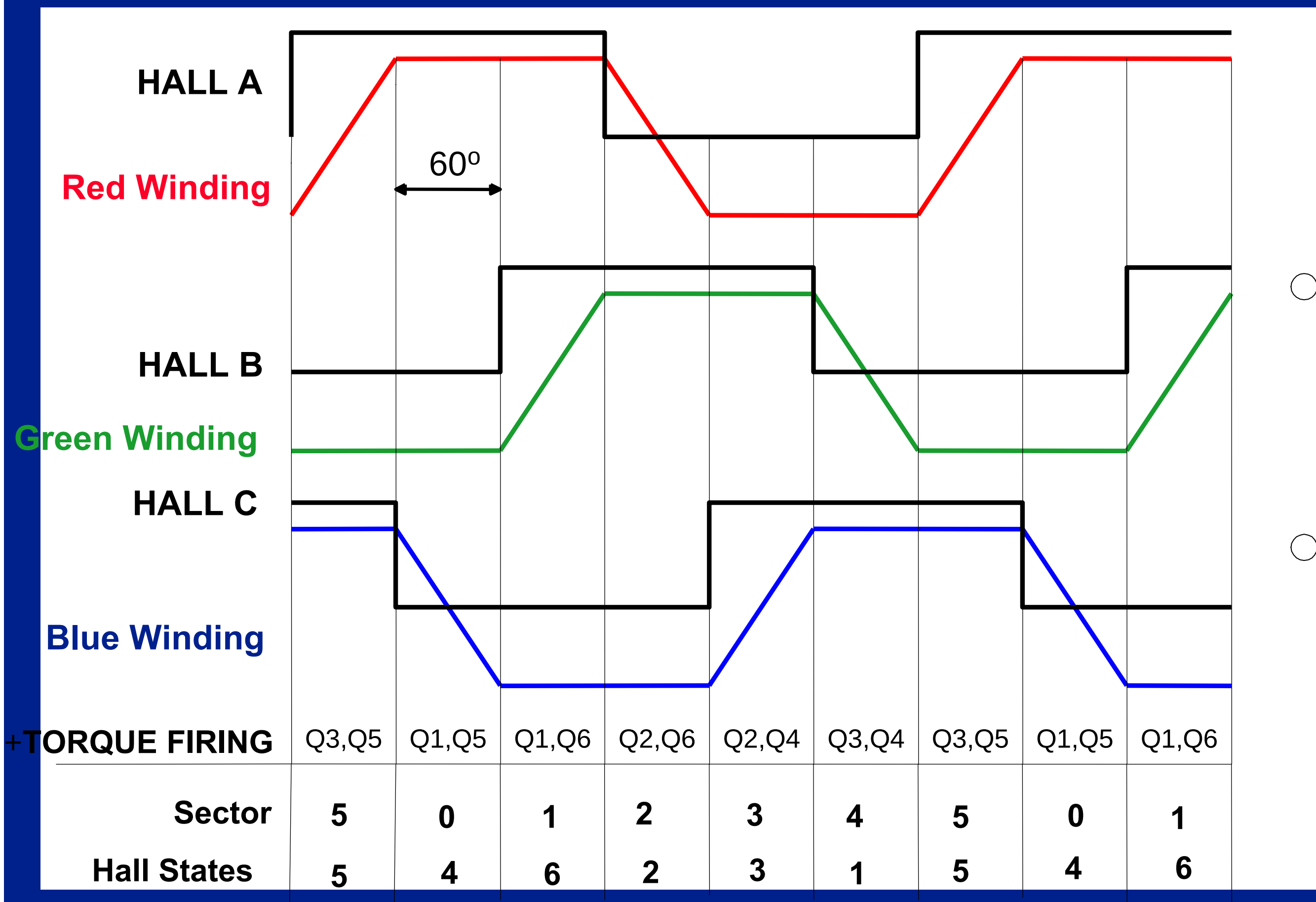
Standard BLDC Position Sensing

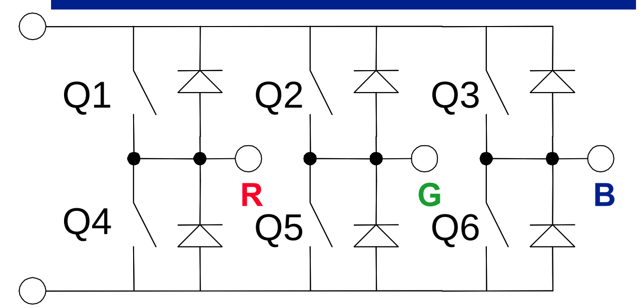
- A sensing disk is attached to the rotor which provides a ≈50% duty pattern aligned to the rotor magnets; the repetition rate of the pattern will follow the number of rotor poles
- The disk is monitored by three optical or hall sensors, displaced by the equivalent of 120°, located on the stator
- In the case of hall sensors, the rotor magnets themselves may be sensed directly

BLDC Motor Construction



Six Step BLDC Control





Typical Manufacturer's Table

Timing diagram for Hall Switches

Angle Degree														
	electr.	0	60	120	180	240	300	360	60	120	180	240	300	360
	mech.	0	15	30	45	60	75	90	105	120	135	150	165	180
S 1														
S 2														
S 3														
Motor Phase wires A		−	O	+	+	O	−	−	O	+	+	O	−	
Motor Phase wires B		+	+	O	−	−	O	+	+	O	−	−	O	
Motor Phase wires C		O	−	−	O	+	+	O	−	−	O	+	+	

Waveforms of Hall effect switches

Lab 3. Running Sensored BLDC Motor with OVDCON

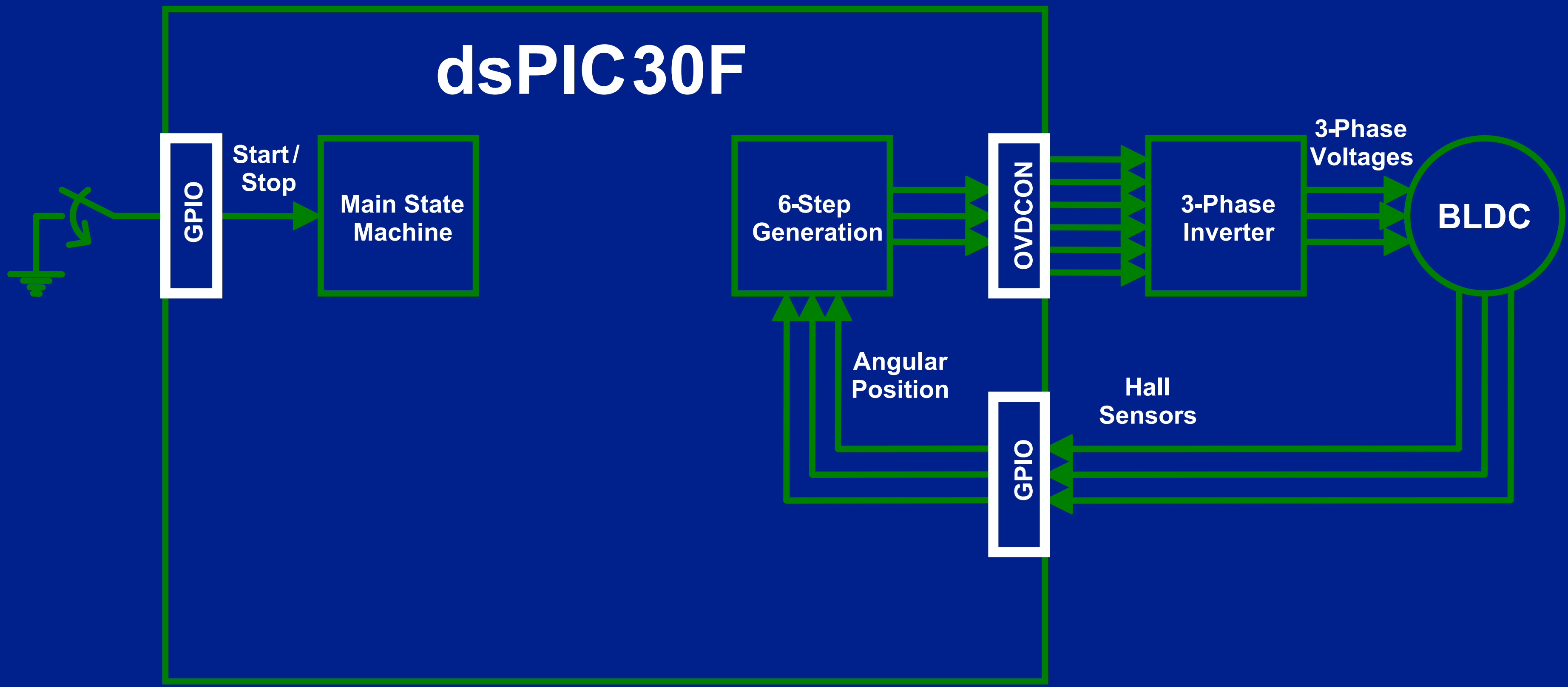
Six Step Sensored BLDC Control

- The 3 logic signals are decoded to determine which windings should be energized
- There are 6 valid states and 2 states that should never be seen (000, 111)
- Use Lookup table to drive the 3 windings, high or low or no-drive
- The 6 different valid states directly translate to the 6 different 60° electrical cycle sectors
- The states should only transition by one at a time. Missing transitions or invalid states should be detected for robust operation

Lab 3 Jumper settings

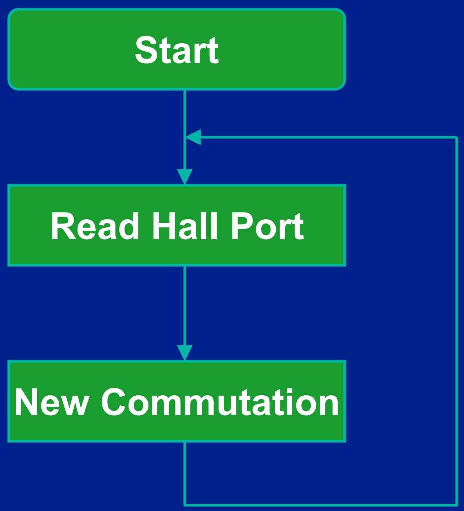
- Turn MCLV board over and refer to the jumper settings for "dsPIC® DSC Sensored"
- Keep Potentiometer REF(R14) and R60 in center position
- Connect Hall Sensors to the Motor (Black Connector)
- Connect Motor Windings (White Connector) to Motor

Running Sensored BLDC Motor with OVDCON



Running Sensored BLDC Motor with OVDCON

Control Technique:



- Remember motor is running at full speed, no PWM is used.

Lab3 – Running Sensored BLDC Motor with OVDCON

- Instructions for Lab3:
  - Use workspace "C:\WIB\Lab3\Lab3.mcw"
  - Follow Lab 1 instructions to:
    - Compile code
    - Program dsPIC
    - Run code

Continued...

Lab 3 - Running Sensored BLDC Motor with OVDCON

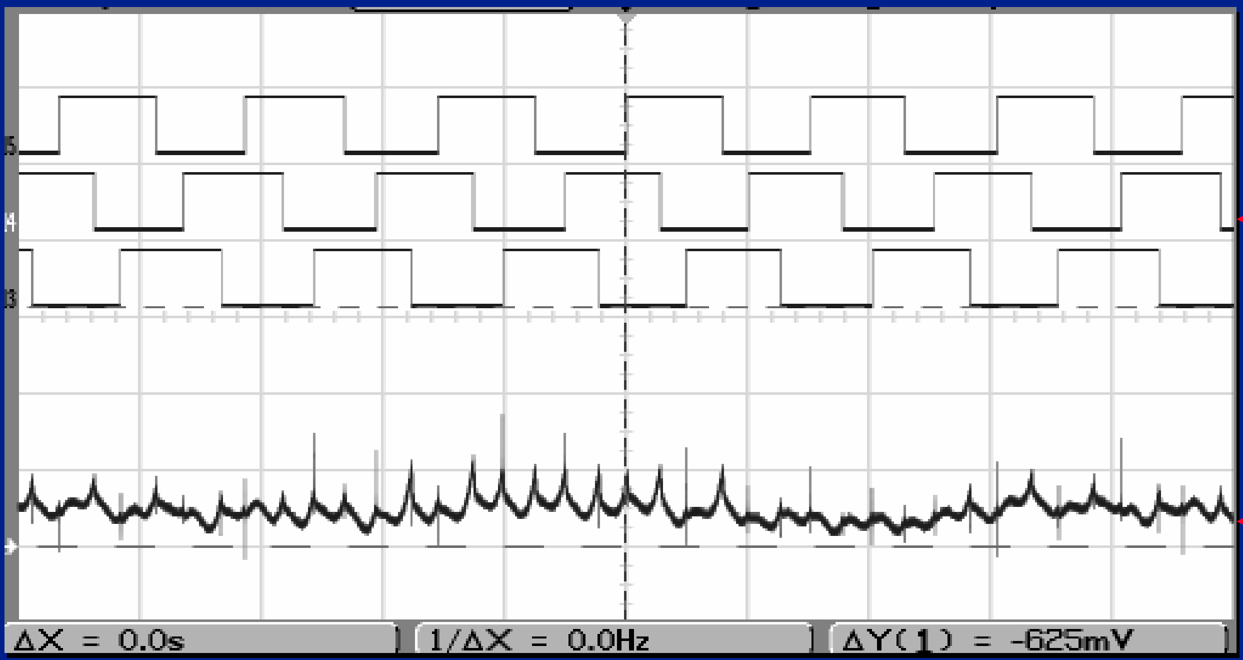
- Press S2 to start motor
- Notice that the motor is running very smoothly
- Notice that the motor does not get warm
- WHY?
- Press S2 to stop the motor

Lab 3 - Running Sensored BLDC Motor with OVDCON

Motor doesn't run?

- Hall sensors wires might be loose
- Check jumper settings: "dsPIC® DSC Sensored"
- Make sure that after programming the device you changed the DIP Switch from PRGM to DEBUG before hitting S2
- Make sure program is not halted in MPLAB® IDE
- Did you press S2?

Lab 3 - Running Sensored BLDC Motor with OVDCON



Maximum Speed
3800 RPM

200 mA Peak

Details of Program

- Use MPLAB® IDE to go thru sections of the code

Lab3 Results

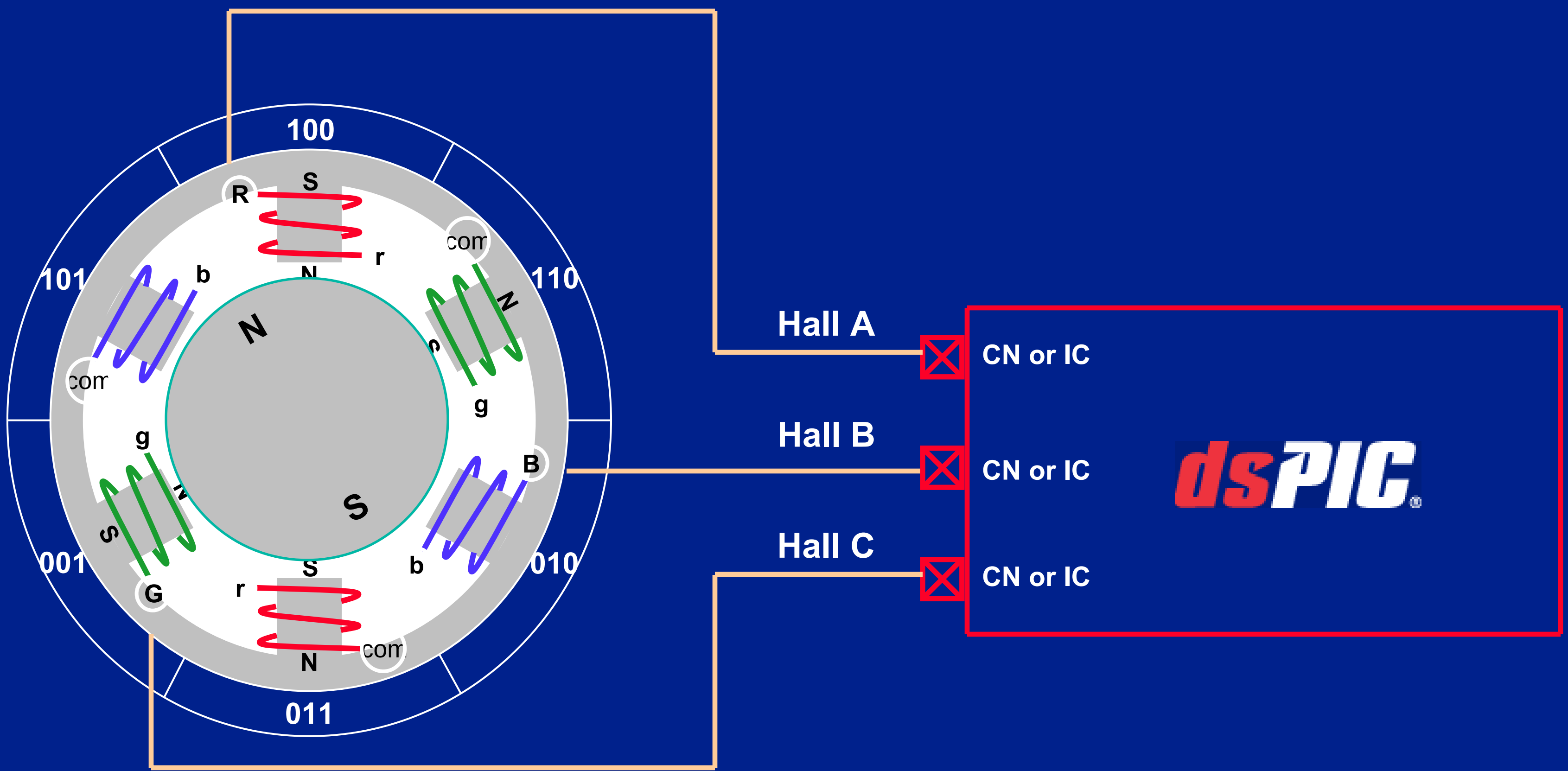
- Spinning a Sensored BLDC motor at full speed.
- Understanding how sensors position and BLDC motor efficiency are related.
- OVDCON will shut off the outputs if program execution is Halted, protecting the system HW
- With no PWM we have fixed motor speed.

Lab 4. Running Sensored BLDC Motor with MCPWM

Change Notification (CN)

- dsPIC® DSC has Change Notification inputs:
  - Detect digital changes on a specific input pin and generates an interrupt
  - Hall sensors A, B and C are connected to RB3, 4 and 5 or CN4, 5 and 6 respectively.
  - When CNxInterrupt occurs, all 3 Hall inputs are read and a lookup table is used to control the BLDC motor

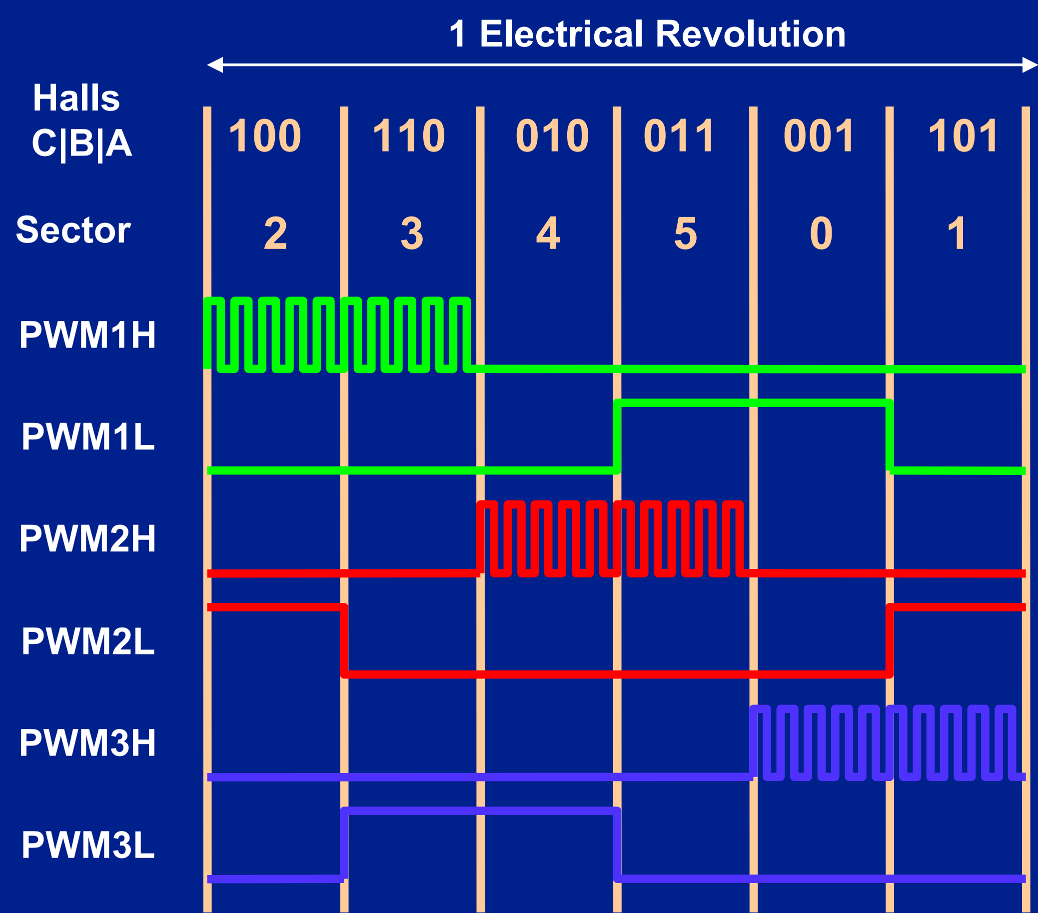
Hall Sensors Connection



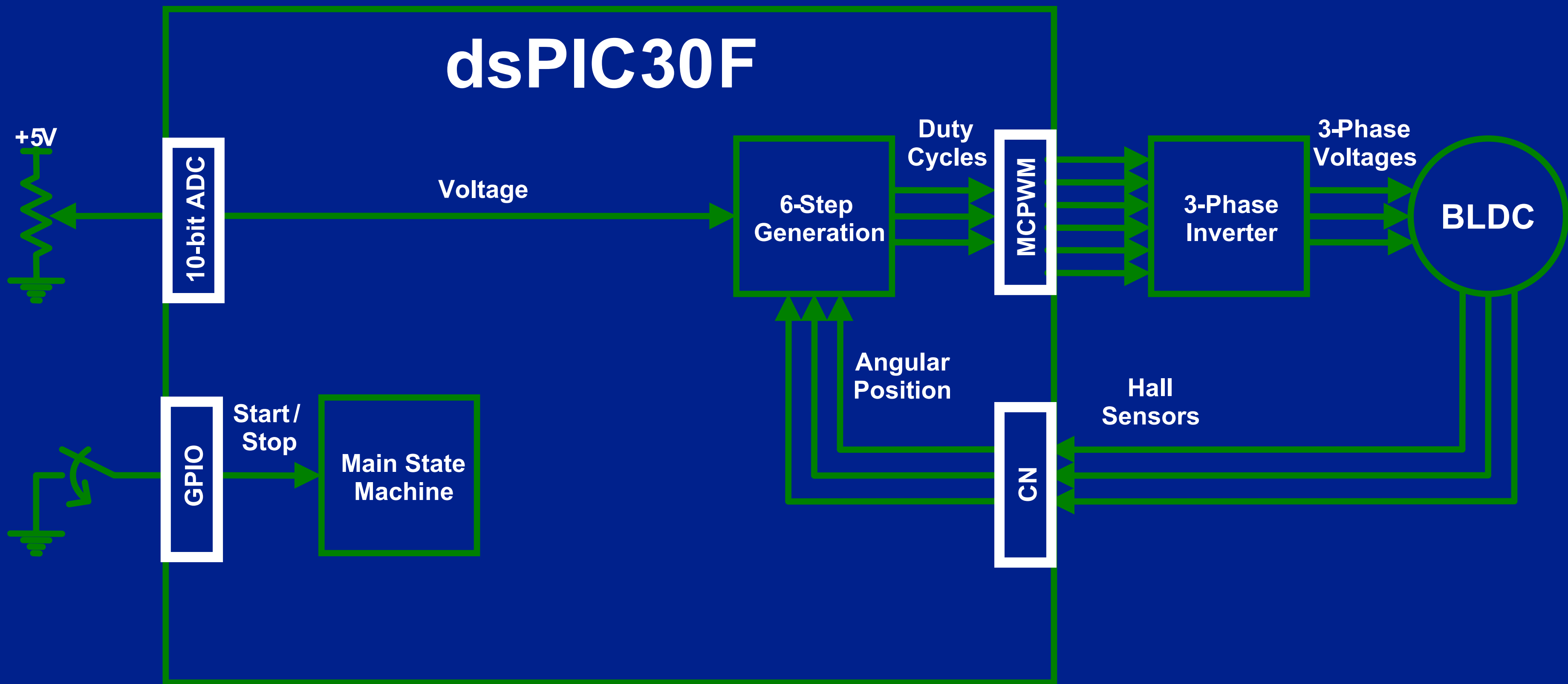
Motor Control PWM

- **Using OVDCON for PWM 6-Step commutation**

Halls C|B|A	OVDCON Value	
	POVD<7:0>	POUT<7:0>
000	00000000	00000000
001	00100000	00000001
010	00001000	00010000
011	00001000	00000001
100	00000010	00000100
101	00100000	00000100
110	00000010	00010000
111	00000000	00000000

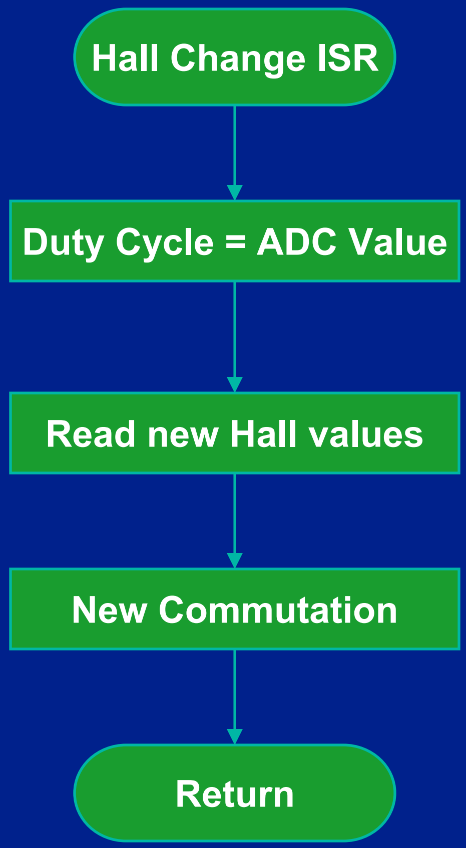


Running Sensored BLDC Motor with MCPWM



Running Sensored BLDC Motor with MCPWM

Control Technique:



Lab4 – Running BLDC Motor with MCPWM

- Instructions for Lab4:
  - Use workspace "C:\WIB\Lab4\Lab4.mcw"
  - Follow Lab 1 instructions to:
    - Compile code
    - Program dsPIC® DSC
    - Run code

Continued…

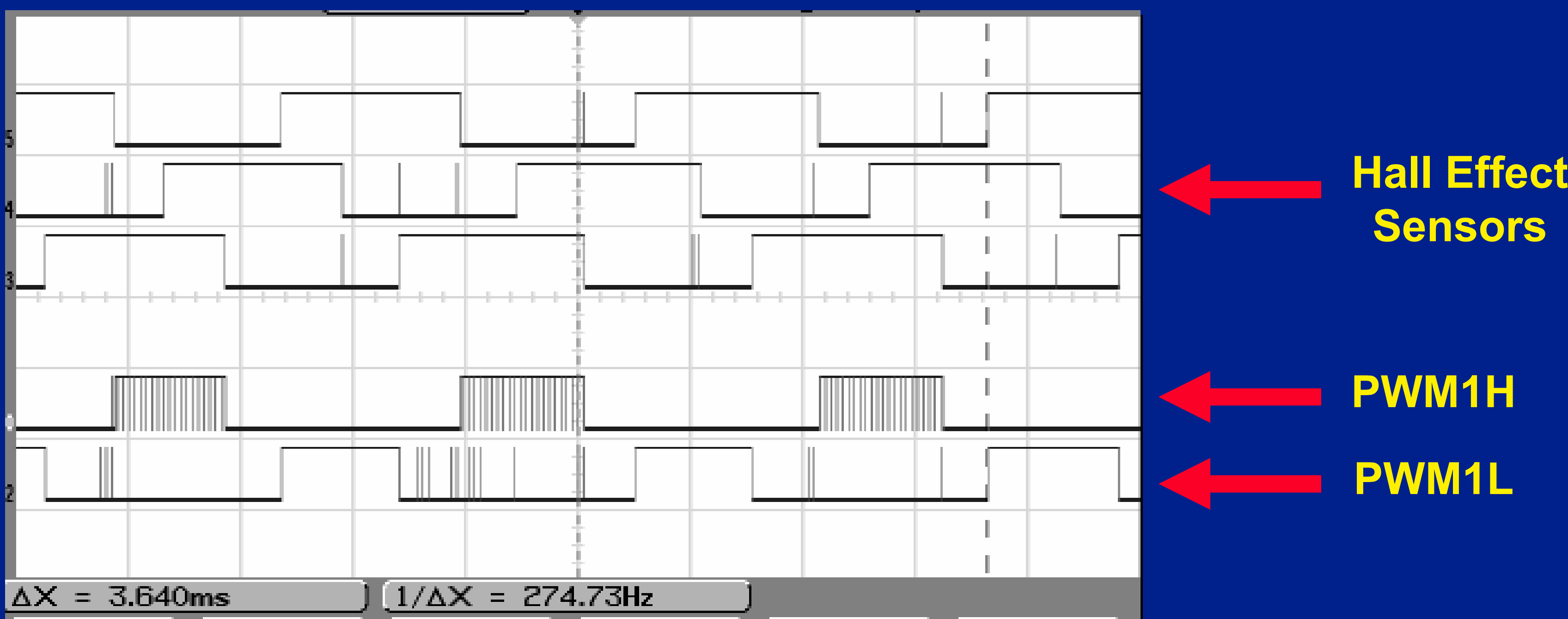
Lab4 – Running BLDC Motor with MCPWM

- Press S2 to start motor
- Use Pot to set the voltage of the motor
- Notice that the current consumption is very low and the motor does not get warm
- WHY?
- Press S2 to stop the motor

Lab4 – Running BLDC Motor with MCPWM

- Press S2 to STOP the motor
- Disconnect Phases Cable from Motor
- Move the Motor with your hands
- You can actually see the combination table looking at the LEDs from the board
- The intensity of the LEDs will depend on the POT value.

Lab4 – Running BLDC Motor with MCPWM



Lab4 Results

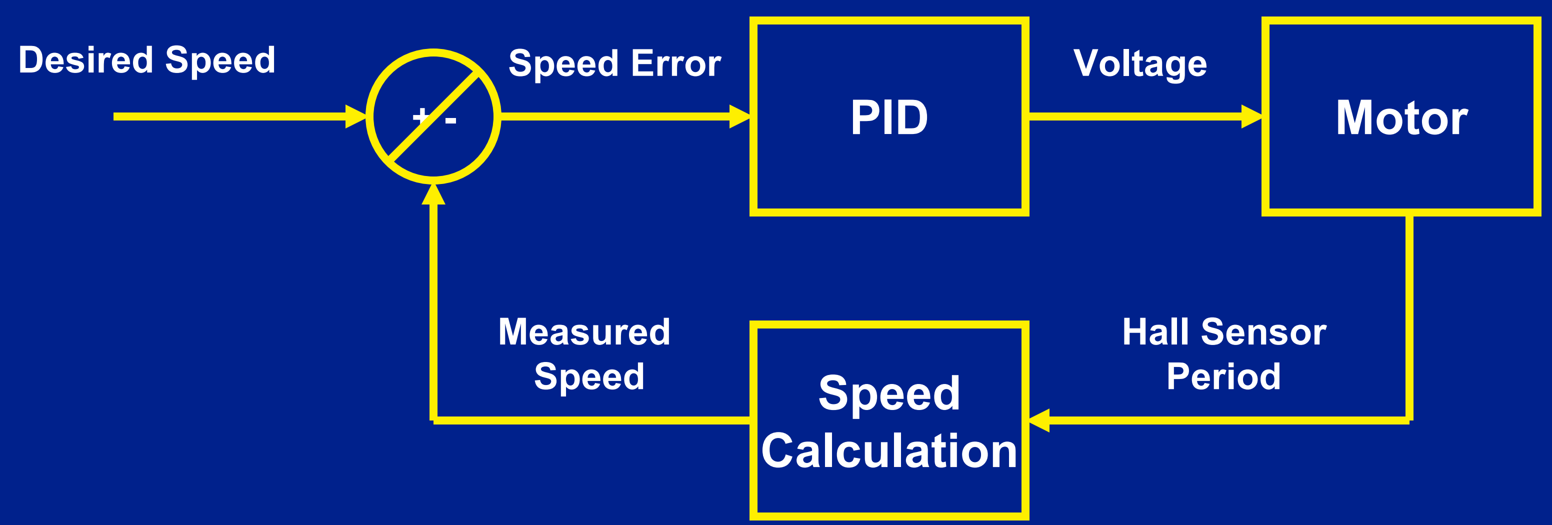
- Variable voltage using MCPWM module
- Maximum speed of 3600 RPM approx.
- BLDC Motor Speed will change if the Load Changes

Lab 5. Running Closed-Loop BLDC Motor

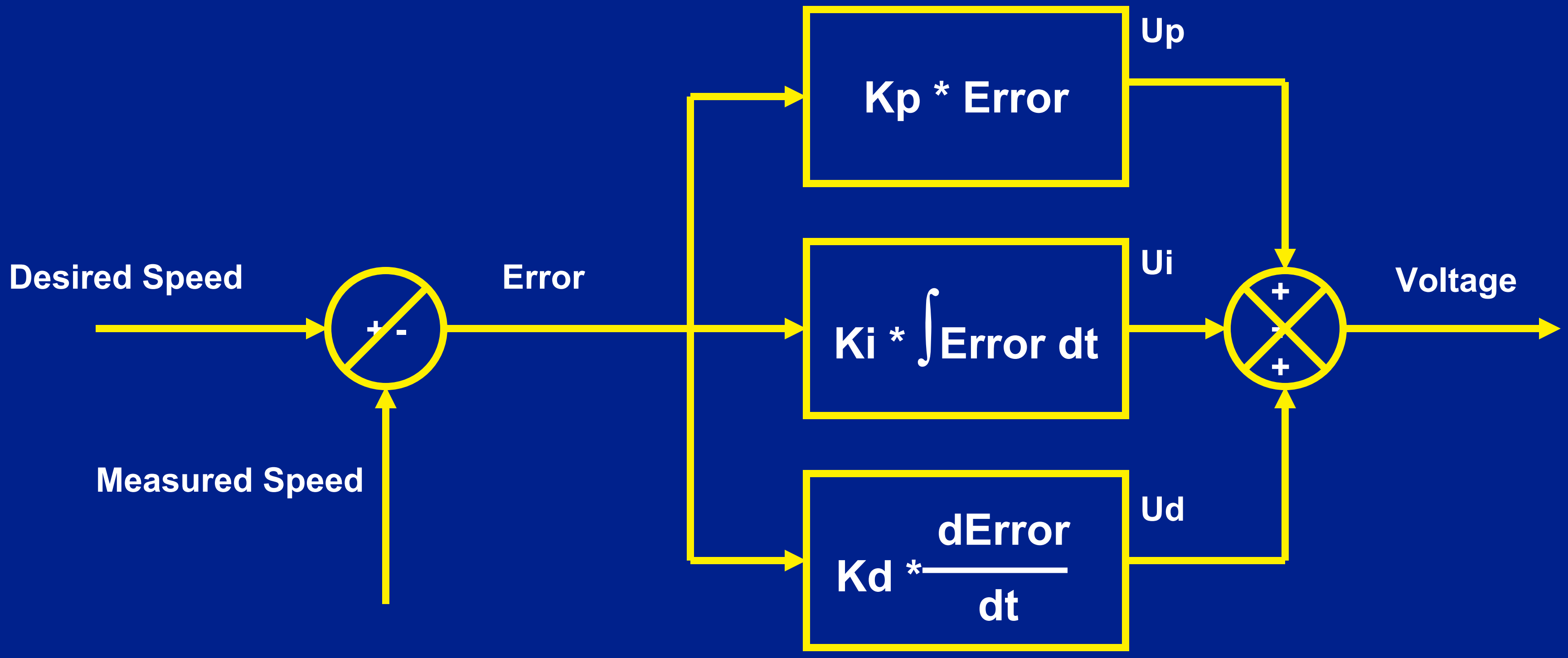
PI(D) Loop

- Proportional-Integral-Differential
- Set Point - Process Variable = Error
- Control Variable = Output
- $CV = Pe + I \int e\,dt + D\,de/dt$

Closed Loop


Desired Speed
+ -
Speed Error
PID
Voltage
Motor
Measured Speed
Speed Calculation
Hall Sensor Period

Digital PID



Up
Kp * Error
Desired Speed
+ -
Error
Ui
Ki * ∫Error dt
Voltage
+
-
+
Measured Speed
Ud
dError
Kd *
dt

Digital PID

- $Up(T) = Kp * Error(T)$
- $Ui(T) = Ki * Error(T) + Ui(T-1)$
- $Ud(T) = Kd * (Error(T) – Error(T-1))$

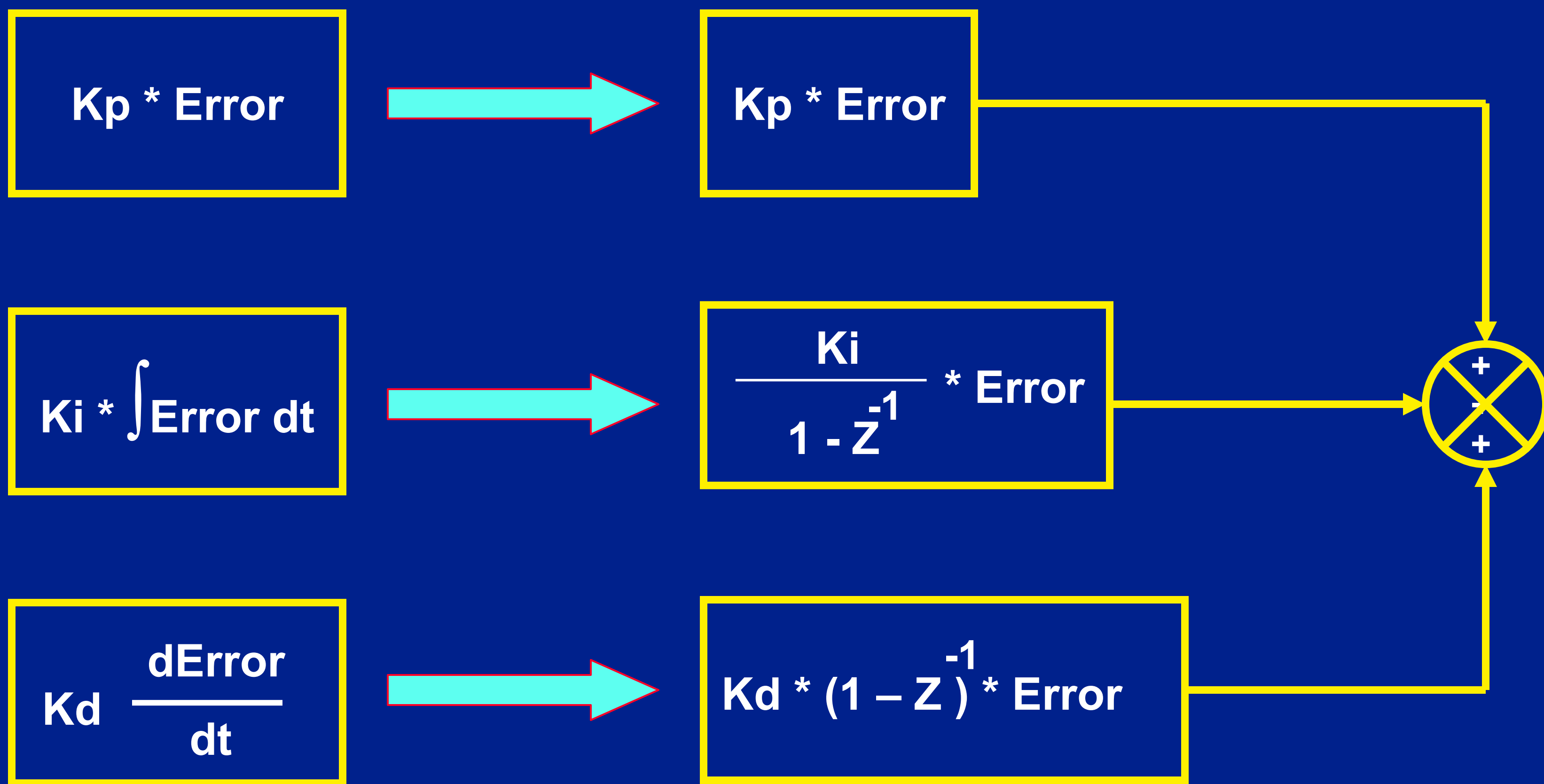
$$Kp * Error$$

$$Ki * \int Error\,dt$$

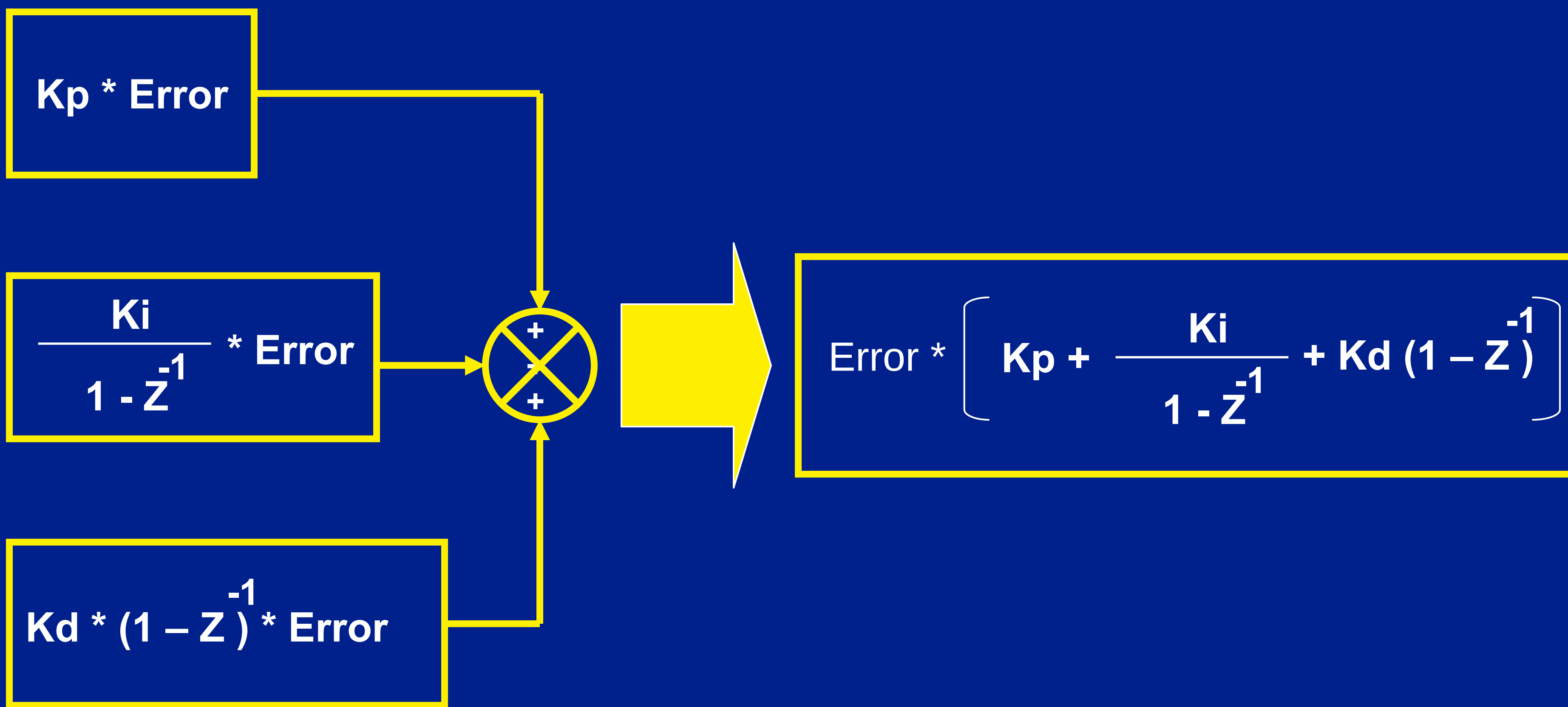
$$Kd * \frac{dError}{dt}$$

Voltage(T) = Up(T) + Ui(T) + Ud(T)

Optional Digital PID



Optional Digital PID



Optional Digital PID

$$\text{Error} * \left[Kp + \frac{Ki}{1 - Z^{-1}} + Kd\,(1 - Z^{-1}) \right] = \text{Controller Output}$$

$$\text{Error} * \left[\frac{Kp\,(1 - Z^{-1}) + Ki + Kd\,(1 - Z^{-1})^{2}}{1 - Z^{-1}} \right] = \text{Controller Output}$$

$$\text{Error} * \left[\frac{(Kp + Ki + Kd) + (-Kp - 2*Kd)\,Z^{-1} + Kd*Z^{-2}}{1 - Z^{-1}} \right] = \text{Controller Output}$$

Optional Digital PID

$$\text{Error} * \left[\frac{(Kp + Ki + Kd) + (-Kp - 2*Kd)\, Z^{-1} + Kd*Z^{-2}}{1 - Z^{-1}} \right] = \text{Controller Output}$$

$\text{Error} = \text{Error}(T)$ — Most Recent Error

$\text{Error} * Z^{-1} = \text{Error}(T-1)$

$\text{Error} * Z^{-2} = \text{Error}(T-2)$ — Least Recent Error

Optional Digital PID

$$\text{Error} * \left[\frac{(Kp + Ki + Kd) + (-Kp - 2*Kd)\, Z^{-1} + Kd*Z^{-2}}{1 - Z^{-1}} \right] = \text{Controller Output}$$

Controller Output (T) = Controller Output (T – 1)

+ Error (T) * K1

+ Error (T-1) * K2

+ Error (T-2) * K3

Where:

K1 = Kp + Ki + Kd

K2 = -Kp -2Kd

K3 = Kd

MAC Operation can be used!!!

MAC Class of DSP Instructions

- **Sample Instruction**
  - MAC   W4*W5, A, [W8]+=2, W4, [W10]-=6, W5, W13]

Source operand registers

Destination accumulator

X prefetch source

X prefetch destination

Y prefetch source

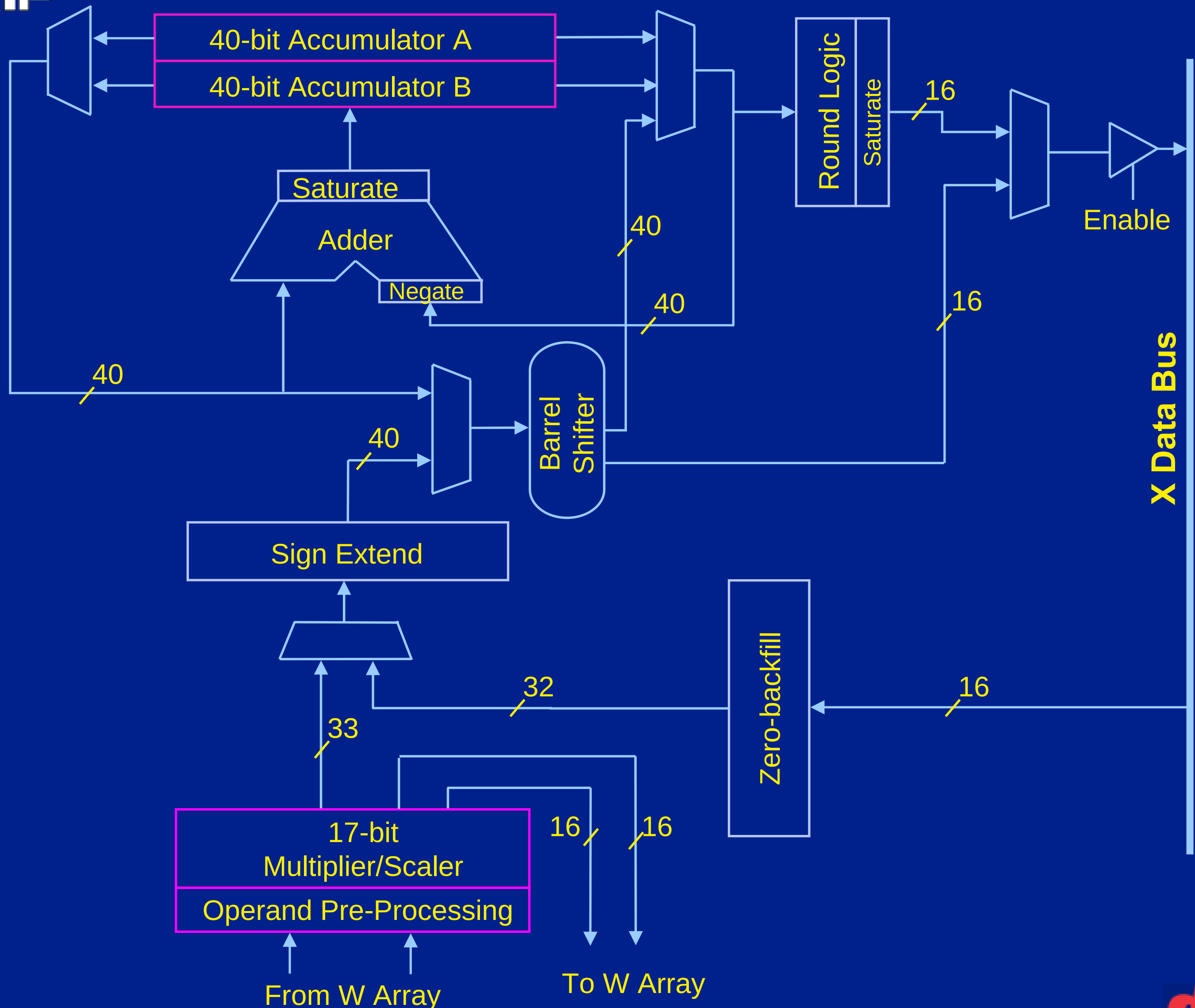
Y prefetch destination

Other Acc. Write-back destination

Basic Syntax

Optional Arguments

DSP Engine Block Diagram



dsPIC® Digital Signal Controller

ADC does support directly Fractional data format

- Scaling everything to -1….0…+1 makes the control-loop much easier to handle.

Word Value	*Integer Value*	*Fractional Value*
0x8000	-32768	-1.0
0xA000	-24576	-0.75
0xC000	-16384	-0.5
0xE000	-8192	-0.25
0x0000	0	0.0
0x2000	8192	+0.25
0x4000	16384	+0.5
0x6000	24576	+0.75
0x7FFF	32767	+0.999969

Measuring Motor Speed with Input Capture (IC)

dsPIC® DSC has Input Capture inputs:

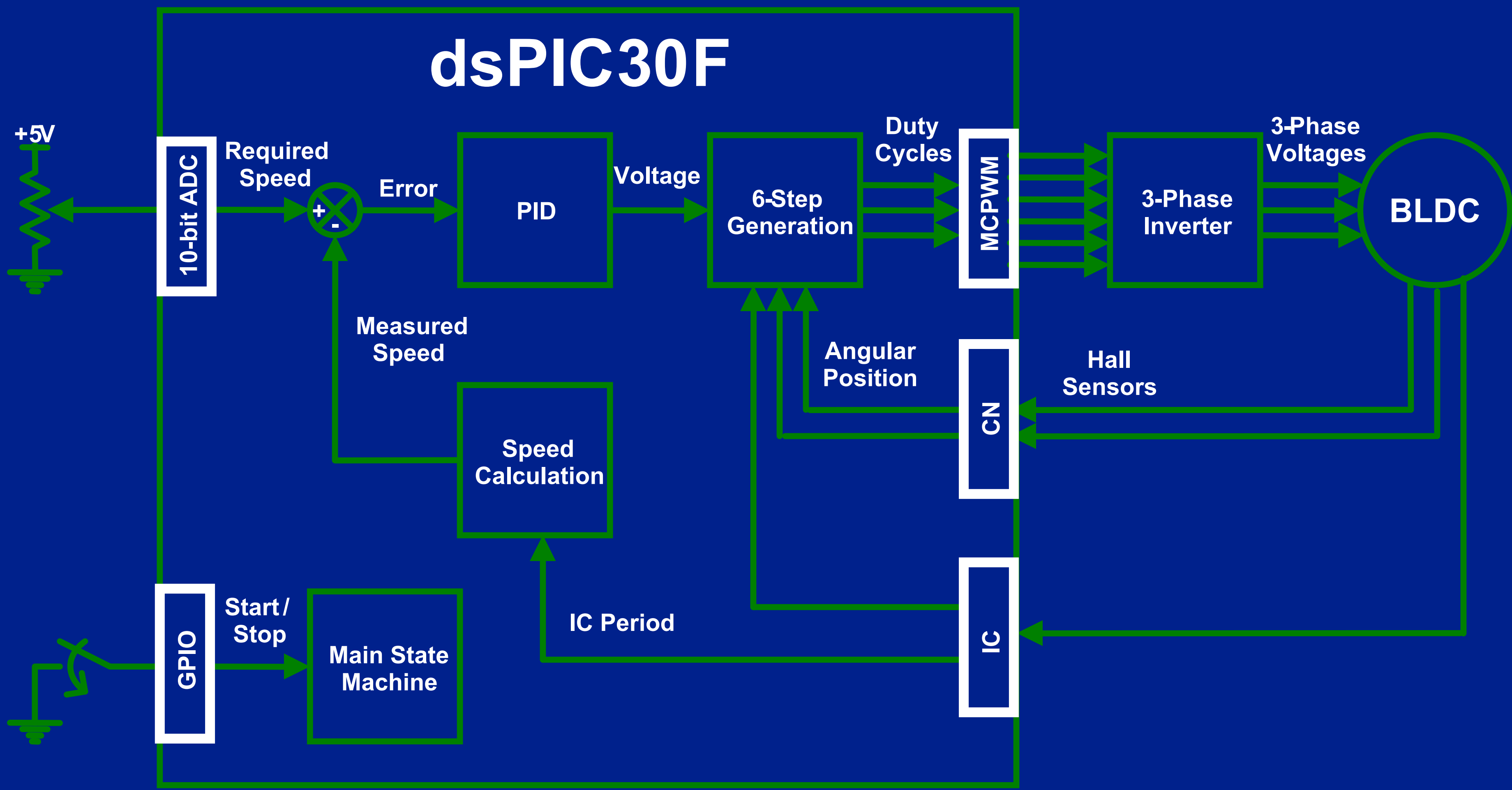
- The period from the IC Channel is used to measure the actual motor angular speed
- Detect digital changes on a specific input pin (Hall Sensor) and generates an interrupt
- One of the Hall effect sensors is connected to an IC Channel
- When ICxInterrupt occurs, the period between IC input transitions is buffered

Speed Calculation w dsPIC Engine

$$\text{Measured Speed} = \text{(Fractional Divide)}\ \frac{\text{Minimum Period}}{\text{IC Period}}$$

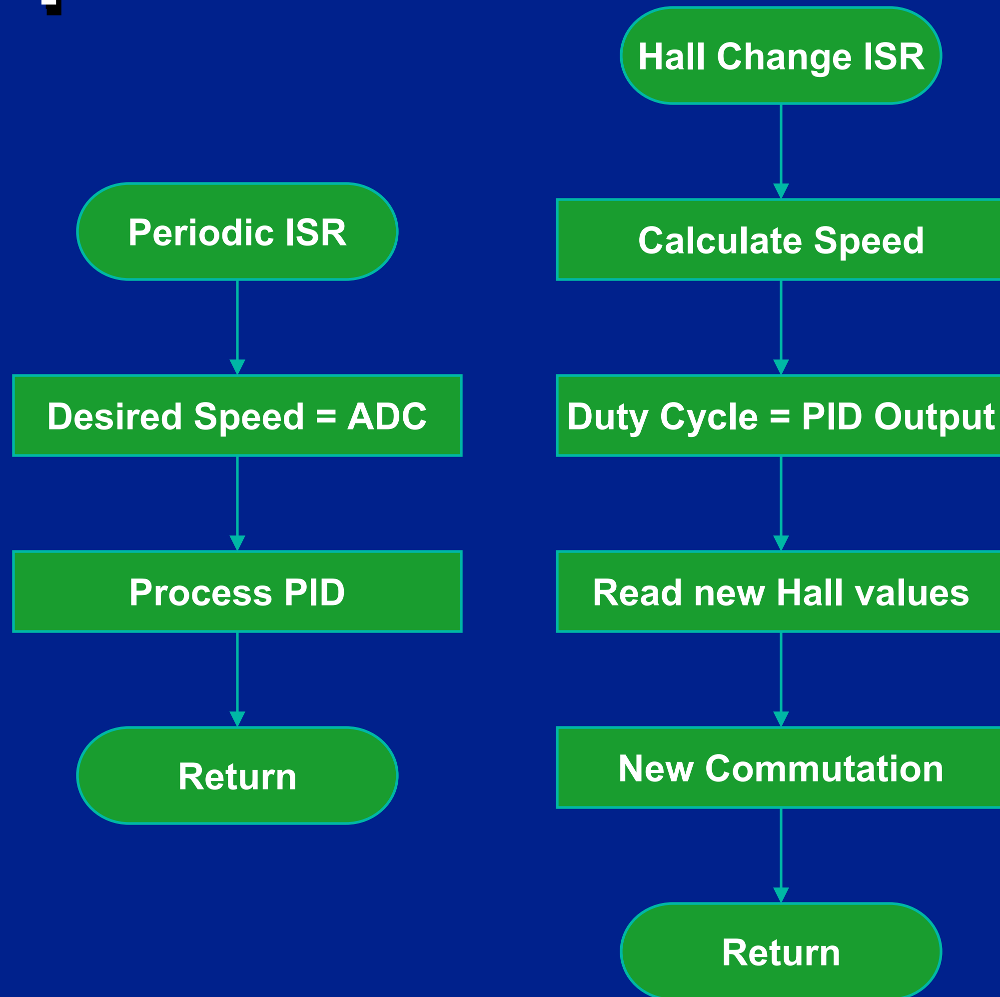
- Fast Speed Calculation using dsPIC® DSC Engine
- Small code size

Lab5 – Running Closed-Loop BLDC Motor



Lab5 – Running Closed-Loop BLDC Motor

Control Technique:



Lab5 – Running Closed-Loop BLDC Motor

- Instructions for Lab5:
  - Use workspace "C:\WIB\Lab5\Lab5.mcw"
  - Follow Lab 1 instructions to:
    - Compile code
    - Program dsPIC® DSC
    - Run code

Continued...

Lab5 – Running Closed-Loop BLDC Motor

- Press S2 to start motor
- Use Pot to set the Speed Reference of the motor
- Calculate speed of the motor
- Notice that the duty cycle is automatically adjusted to keep the same speed, even when changing the load
- WHY?
- Press S2 to stop the motor

Lab5 Results

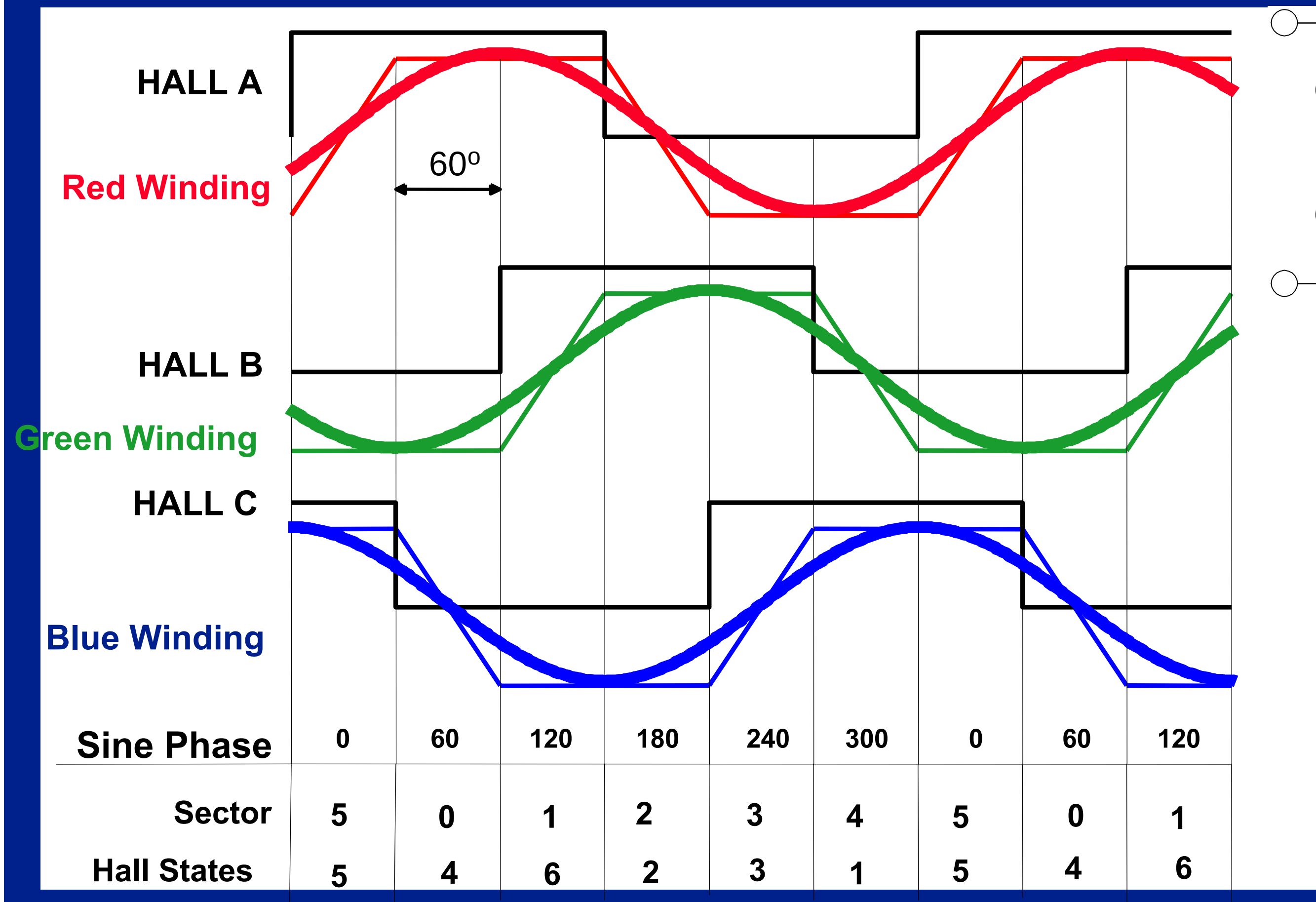
- Speed Control a Sensored BLDC motor
- Implementing a PID digital controller using DSP engine of a dsPIC® DSC
- Use dsPIC DSC's PWM, OVDCON, CN and IC feature to control the speed of the BLDC motor

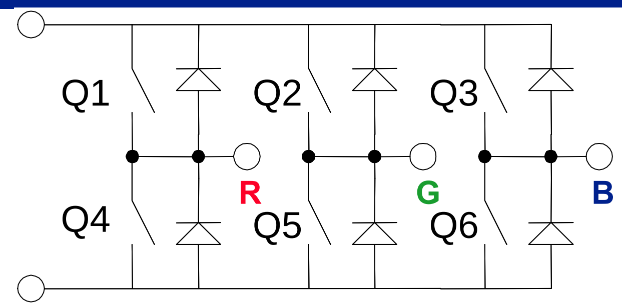
Lab 6. Running Sinusoidal BLDC Motor

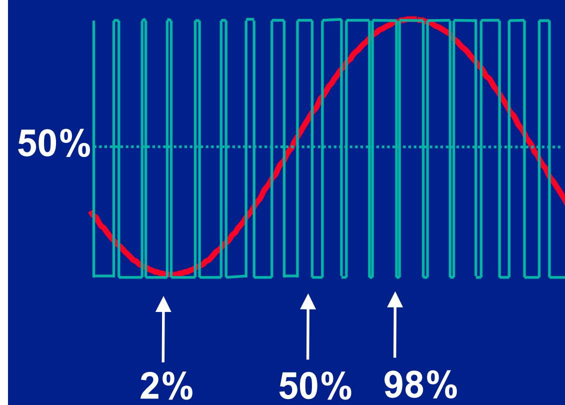
Lab6 – Running Sinusoidal BLDC Motor

- Used for reducing audible noise and reducing torque ripple
- Control technique used in Sinusoidal Back EMF motors, usually called Brushless AC
- Each hall effect sensor transition updates the sine phase
- The frequency of the generated sine wave depends on the motor actual speed
- The amplitude will depend on the speed controller output

Lab6 – Running Sinusoidal BLDC Motor

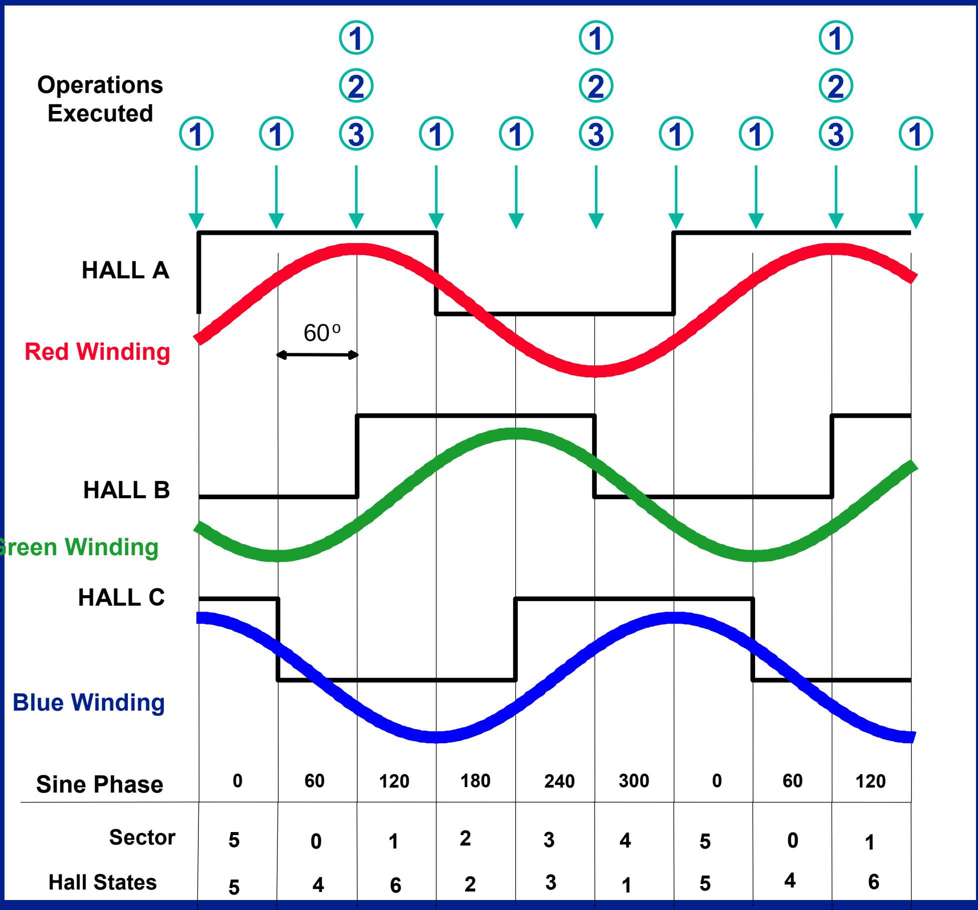






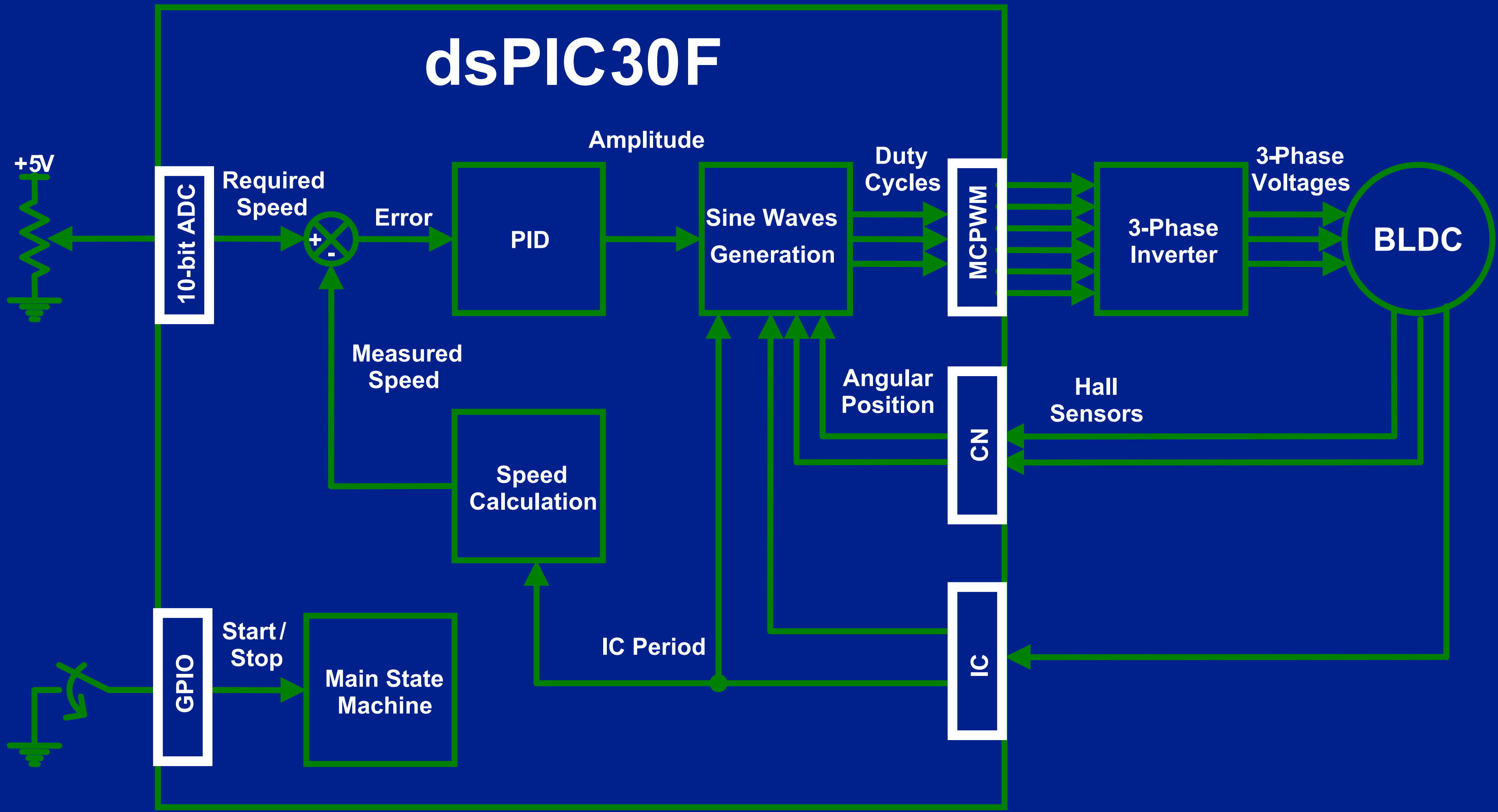
Digital Signal Controller

Lab6 – Running Sinusoidal BLDC Motor



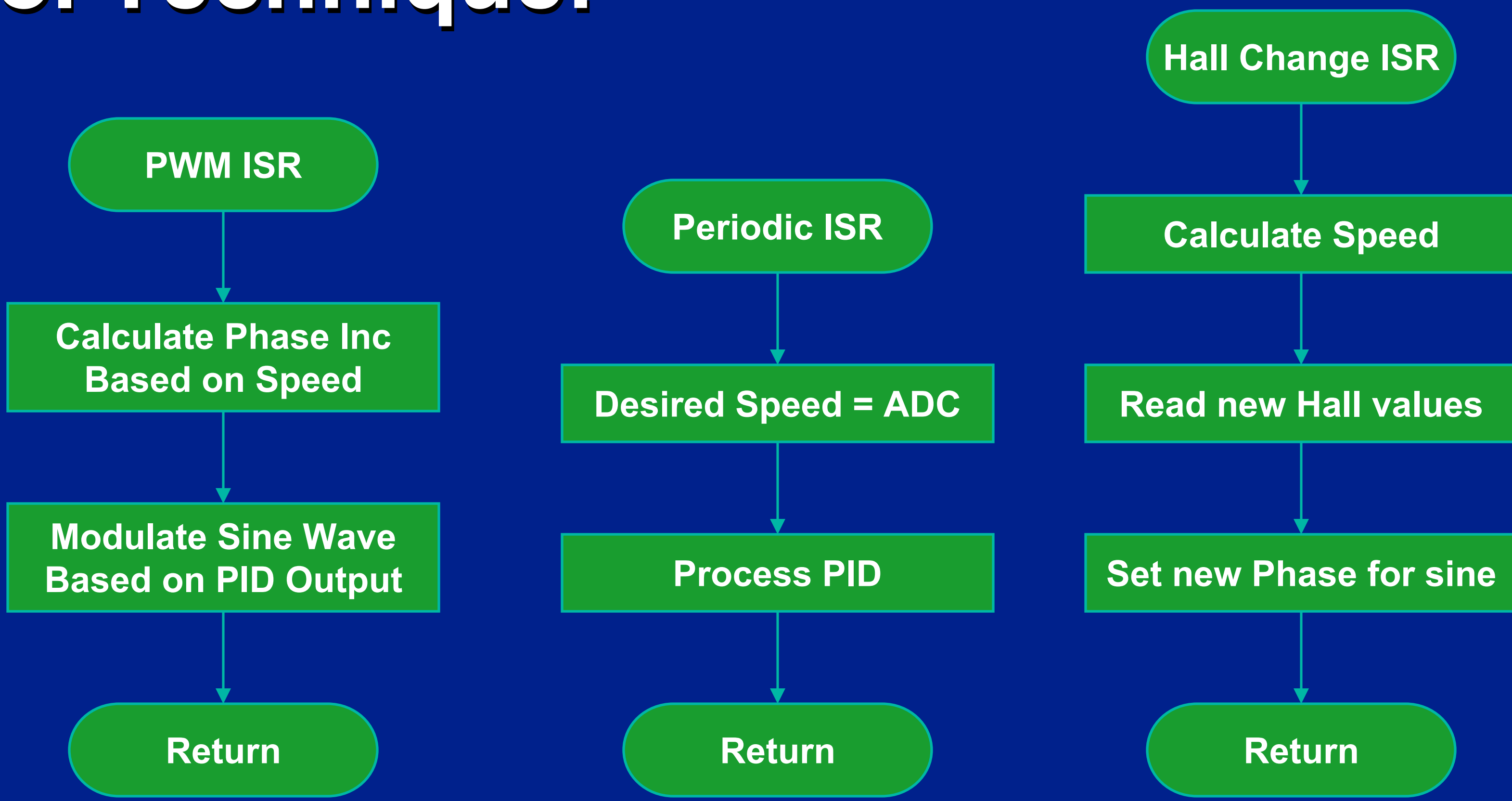
1. **Set Sine wave Phase according to new sector**
2. **Calculate period of one hall effect sensor using Input Capture value**
3. **Apply new sine wave period according to previous Hall effect period (Op 2)**

Lab6 – Running Sinusoidal BLDC Motor



Lab6 – Running Sinusoidal BLDC Motor

Control Technique:



Lab6 – Running Sinusoidal BLDC Motor

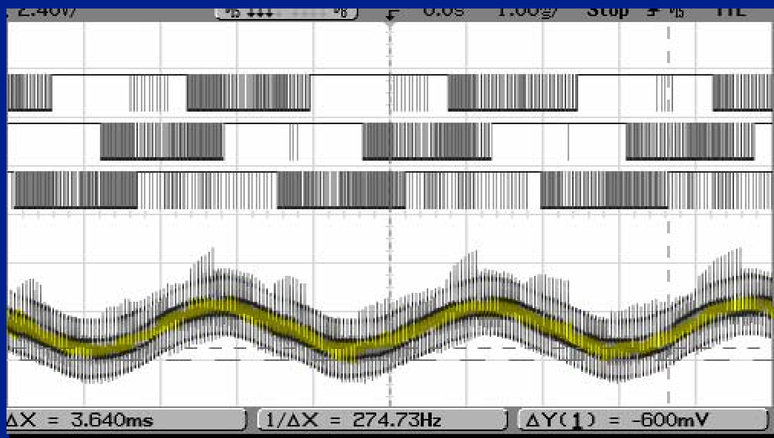
- Instructions for Lab6:
  - Use workspace “C:\WIB\Lab6\Lab6.mcw”
  - Follow Lab 1 instructions to:
    - Compile code
    - Program dsPIC® DSC
    - Run code

Continued...

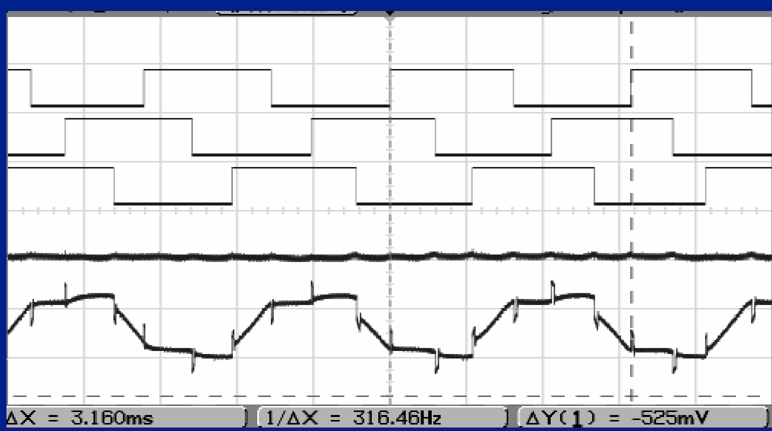
Lab6 – Running Sinusoidal BLDC Motor

- Press S2 to start motor
- Use Pot to set the Speed Reference of the motor
- Work with a partner to compare with previous Lab
- Notice that the noise from the motor has been significantly reduced by using sinusoidal control
- Press S2 to stop the motor

Lab6 – Running Sinusoidal BLDC Motor



Sinusoidal Phase Voltage



Six-Step Control
Trapezoidal Phase Voltage

Digital Signal Controller

Lab6 Results

- Sinusoidal control of a BLDC motor
- Reduced audible noise
- Reduced torque ripple
- CE003. Driving a BLDC with Sinusoidal Voltages using dsPIC30F.

Lab 7. BLDC Operation with Phase Advance

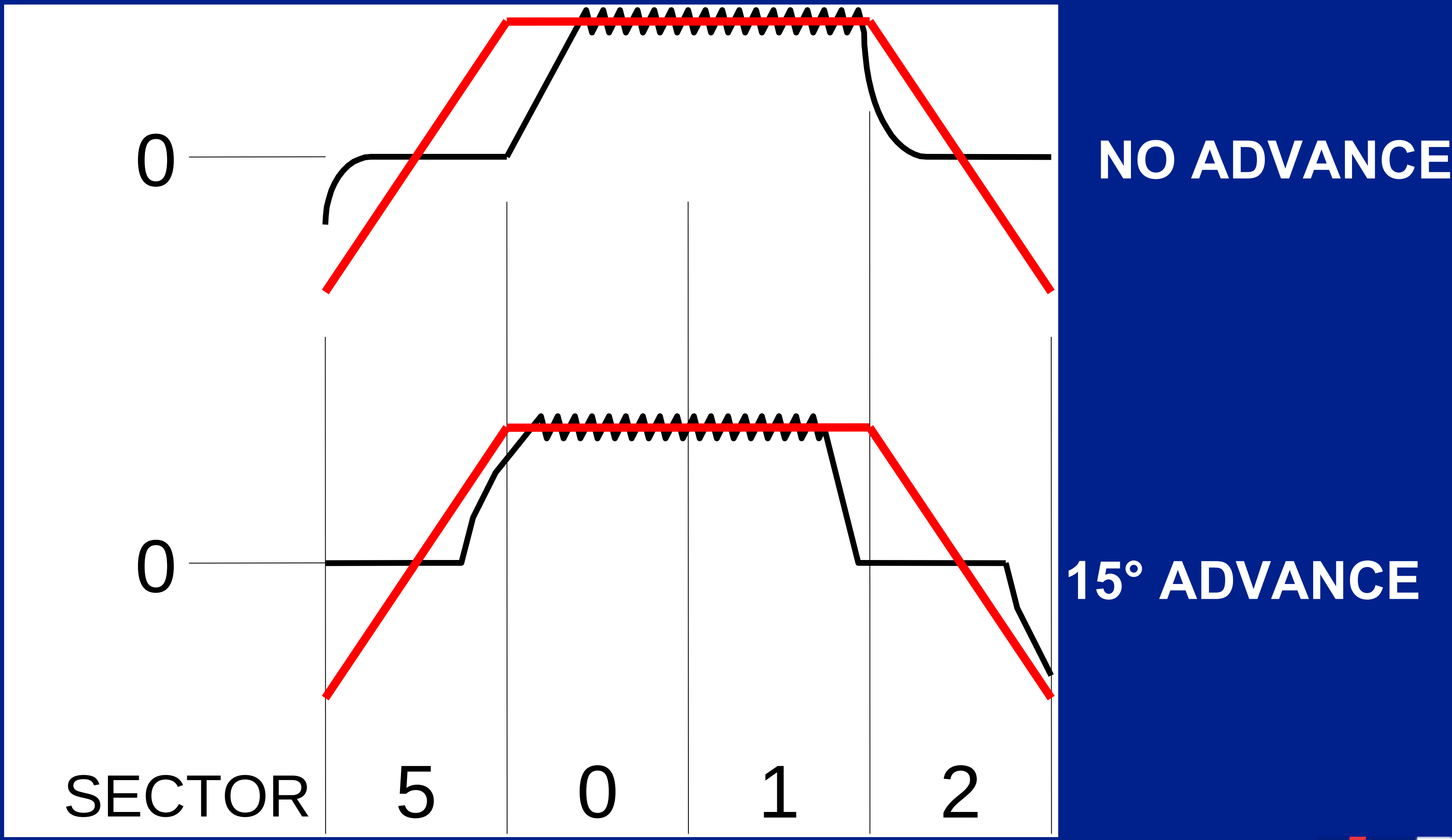
Lab 7. BLDC Operation with Phase Advance

- Drive voltages are shifted (Phase advanced) compared to back EMF
- Phase advance will produce an increase in the stator field, which increases the speed of the motor
- Phase shift will produce a negative field on the rotor, which will reduce the overall torque available in the motor
- For light loads, the speed is significantly increased using phase advance, sacrificing full load torque, efficiency and audible noise

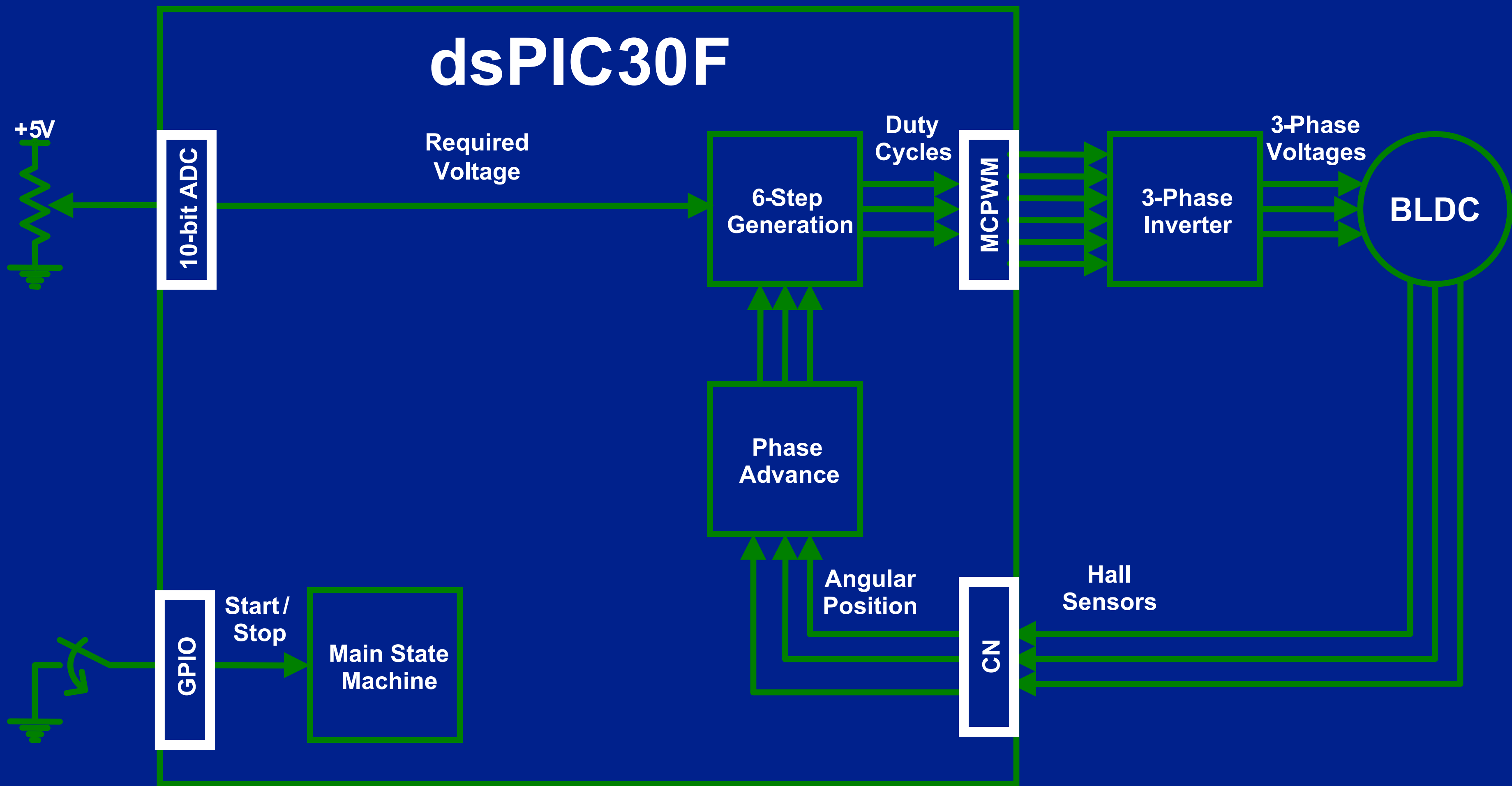
Lab 7. BLDC Operation with Phase Advance

- Consists on commutating the motor before the next hall effect sensor transition has occurred
- Knowing the motor speed, we can schedule a commutation with a timer, before the next hall effect sensor interrupt occurs
- Phase advance technique substantially increases speed range
- It also helps to compensate misalignments on the hall effect sensor

Plots showing the effect of Phase Advance at High Speed

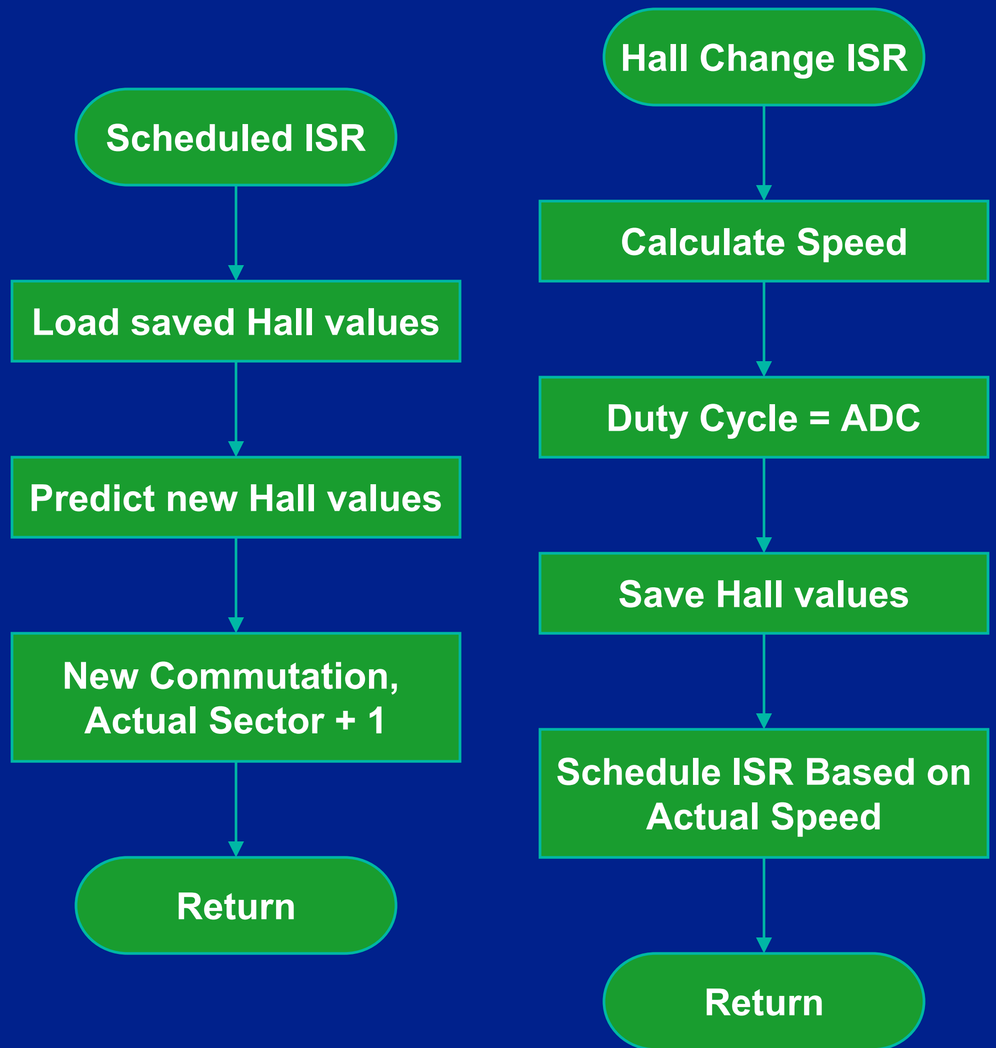


Lab7 – BLDC Operation with Phase Advance



Lab7 – BLDC Operation with Phase Advance

Control Technique:



Lab7 – BLDC Operation with Phase Advance

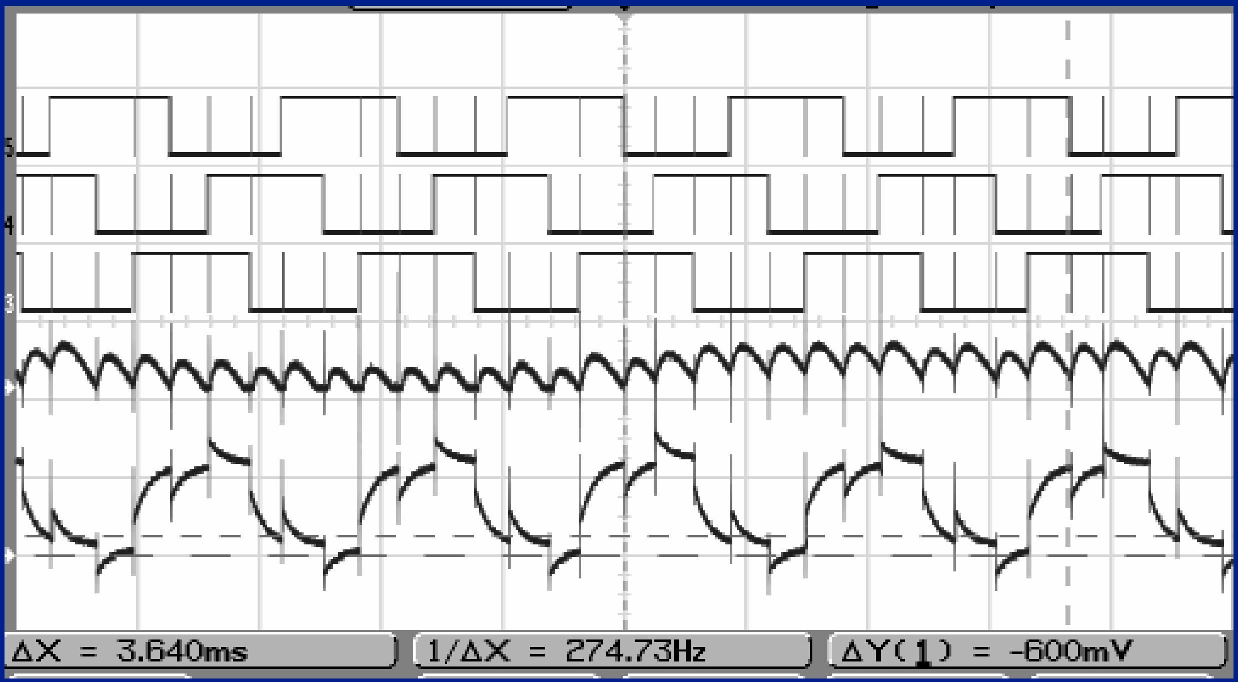
- Instructions for Lab7:
  - Use workspace "C:\WIB\Lab7\Lab7.mcw"
  - Follow Lab 1 instructions to:
    - Compile code
    - Program dsPIC® DSC
    - Run code

Continued...

Lab7 – BLDC Operation with Phase Advance

- Press S2 to start motor
- Use Pot to set the Voltage applied to the motor
- Notice that the maximum speed of the motor is extended using Phase Advance
- Although, motor is very noisy and current consumption is higher
- WHY?
- Press S2 to stop the motor

Lab7 – BLDC Operation with Phase Advance



Extended Speed: 6500 RPM

1 A Peak

Details of Program

- Use MPLAB® IDE to go thru sections of the code

Lab7 Results

- Phase advance control
- Extended speed range of up to 70% (motor dependent)
- Trade-off, current consumption and audible noise.

Sensorless BLDC Motor

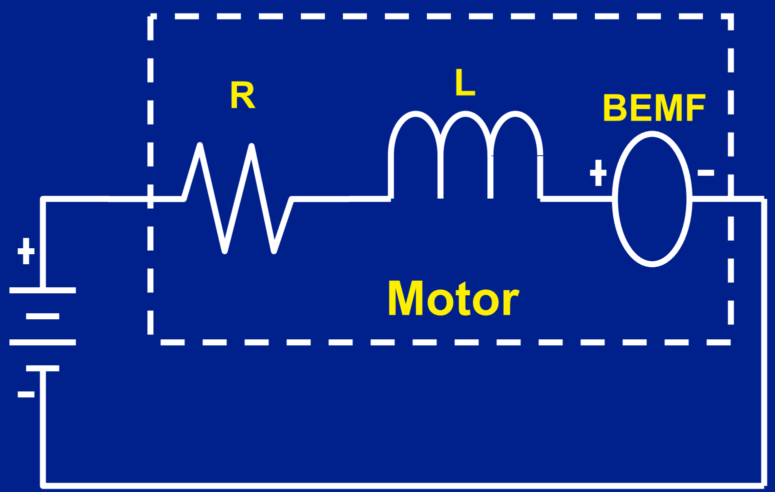
Why sensorless?

- Reliability – especially aerospace, military
- Physical space restrictions – axial length.
- Issues surrounding sealing of connections
- Applications where rotor runs “flooded”
- Manufacturability – alignment and duty cycle tolerance
- Cost – especially on low power systems
  - Even at high volumes, position sensing can add $3 to system cost.

BLDC Sensorless Techniques

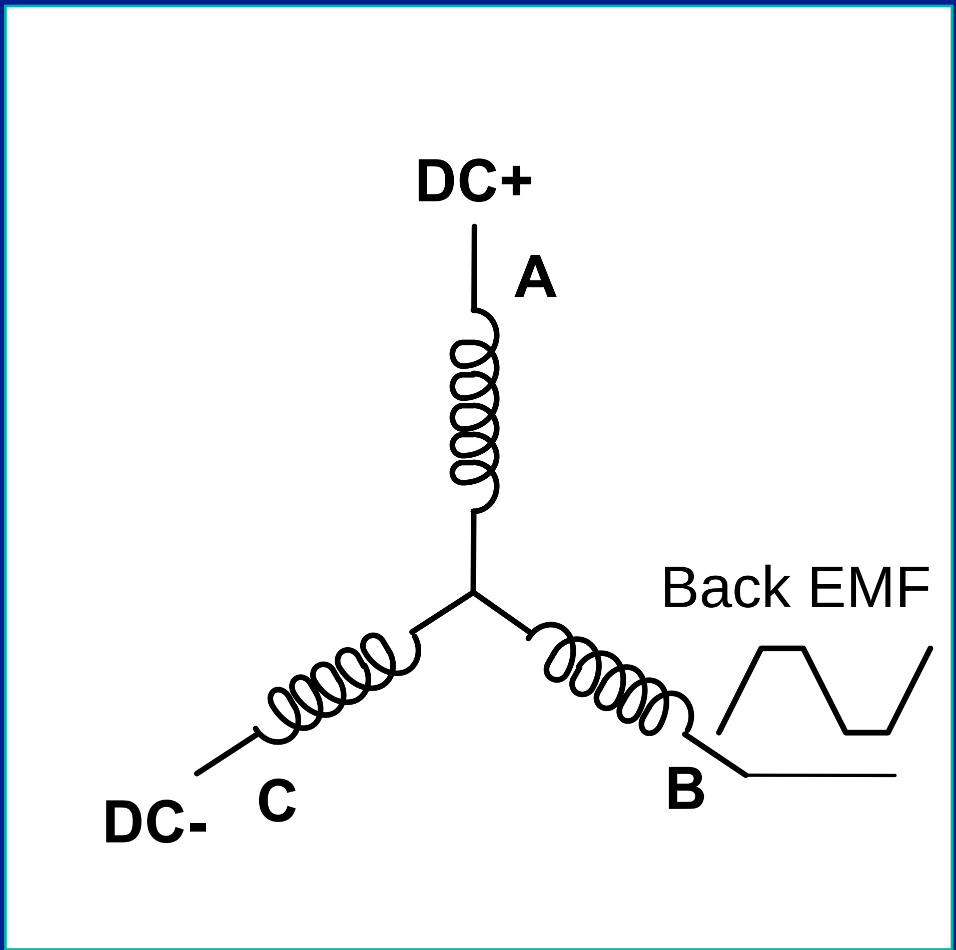
- AN901 uses Back EMF sensing
  - Reliable
  - Varies linearly with speed
  - Works over a wide range of BLDC Motors
  - Relatively easy to implement
  - Works well for applications like Fan or pump speed control
- Method used is called Back EMF “zero crossing” method
  - Consists of monitoring the voltage of the inactive winding for “zero crossing”

What is BEMF?



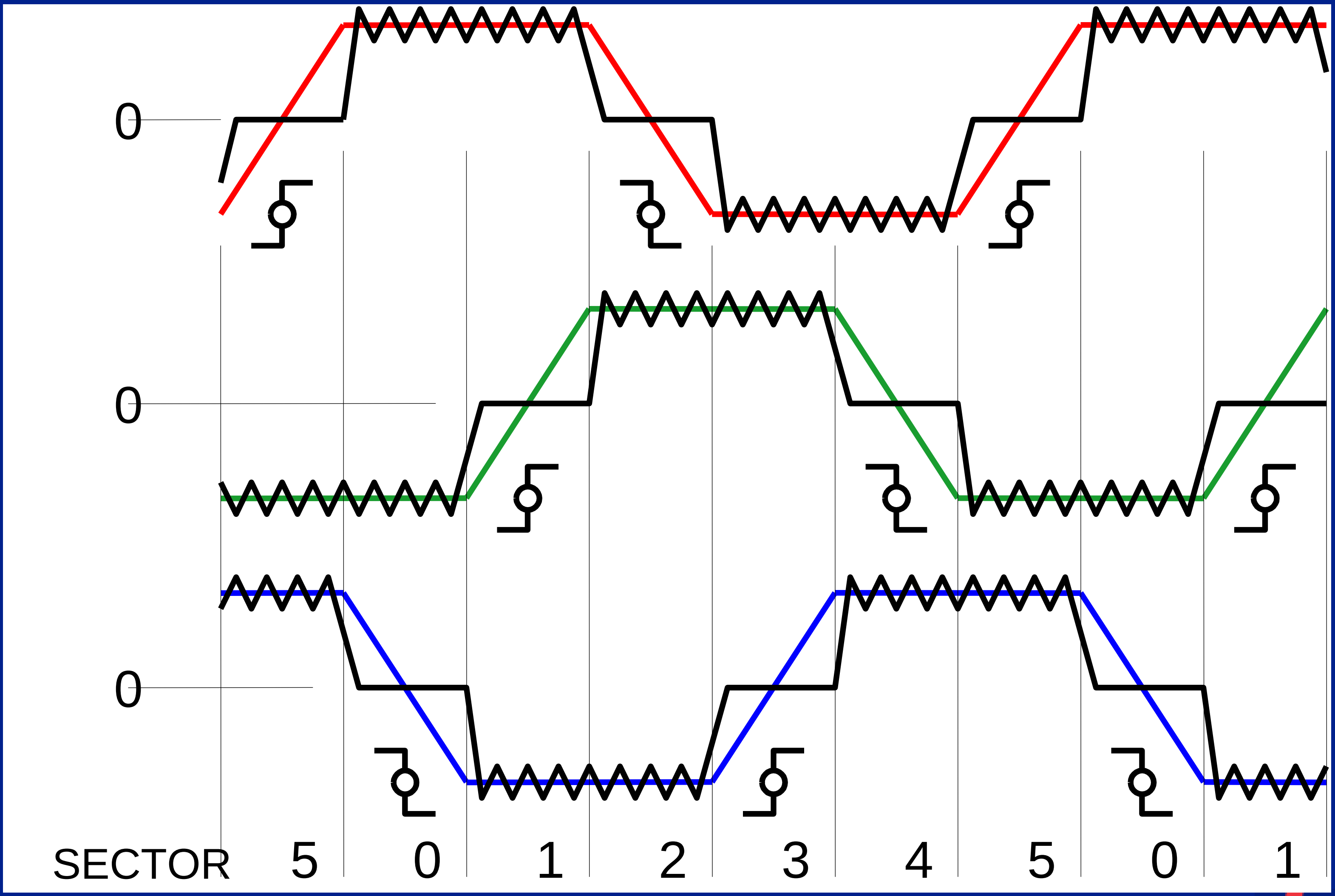
- When a DC motor spins, the PM rotor, moving past the stator coils induces a electrical potential in the coils called Back EMF.
- BEMF is directly proportional to speed
- BEMF = RPM/Kv
- In order to sense BEMF we have to spin the motor.

BLDC Motor Back EMF



- Phase A and C are energized
- Inactive Phase B has induced Back EMF
- Normally the phase which is not energized, is monitored for Back EMF
- **Important: Motor has to be spinning**

Back EMF Crossing Diagram



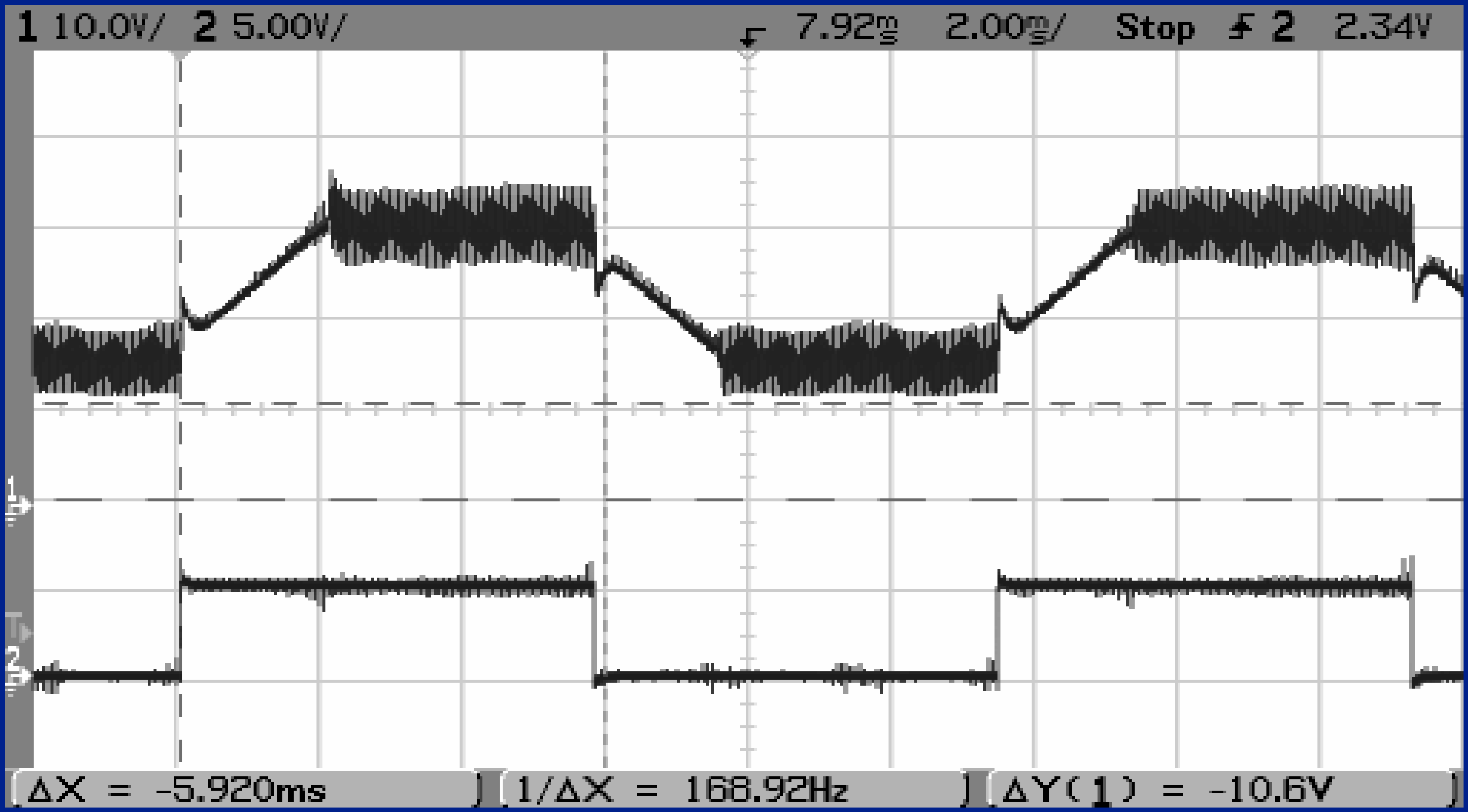
The Back EMF "zero crossing" method in detail

- In every electrical cycle, there are periods when each phase is not being driven.
- During these regions one end of the inactive phase is referenced to the star point and the other is monitored.
- The monitored voltage will cross the 1/2 VDD point at 30 electrical degrees.
- Knowing the last "zero crossing" time we know the 60 electrical degree time (T60)
- T60 divided by 2 = T30 is loaded in TMR2.
- The ISR of TMR2 then commutes the next pair of windings at T30 seconds later

BEMF v/s Hall sensors

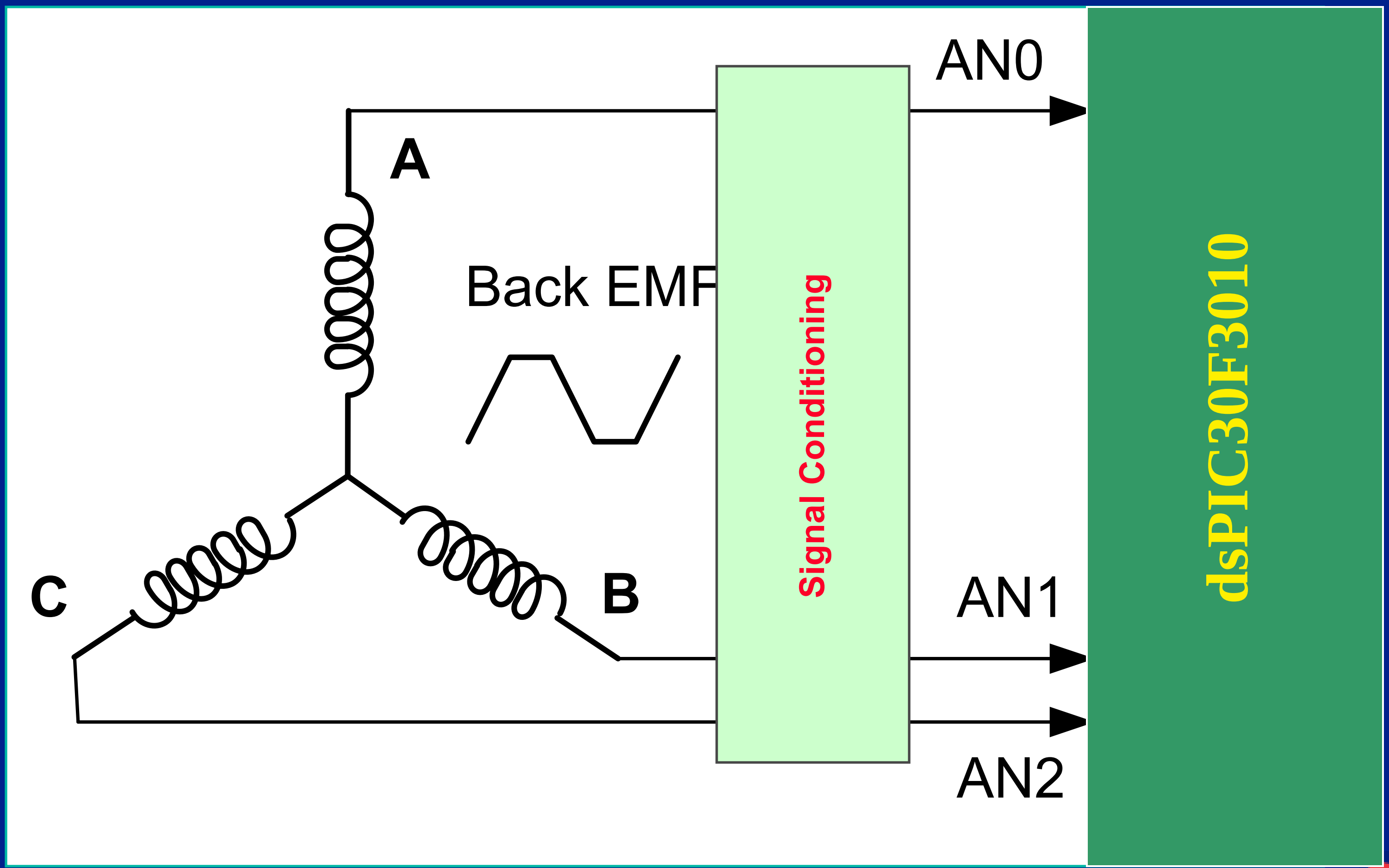
Back EMF

Hall sensor



AN901 method to Monitor Back EMF

- Back EMF signal read using A/D Channels



How to “Start Spinning”

- The motor is energized Open Loop (no feedback)
- The speed is ramped up to a programmable value
- At a given time two winding are energized. The third is monitored for Back EMF
- The unexcited windings are then monitored for two rising edges (120° information)
- From the time and sequencing of the edges we can determine the speed and rotation direction
- The BEMF sensing algorithm is now applied to rotate the motor

Starting Algorithm Parameters used in AN901

- Lock Position 1 Time
  - Before starting, motor is rotated to a known position
  - The amount of time that the rotor is held in that position is LP1T
- Lock Position 1 Demand
  - Speed at which the rotor moves to the lock position
  - If value is too high then rotor may overshoot the position

Starting Algorithm Parameters used in AN901

- Ramp Start/End Speed:
  - Open loop speed to get the rotor moving before back EMF is monitored
  - Too low a speed will not generate enough back EMF
  - Too high a speed may cause an over-current stall
  - Rotor is accelerated from Ramp Start speed to the Ramp End Speed in the Ramp Duration time – Acceleration Profile.

Starting Algorithm Parameters used in AN901

- Ramp Start/End Demand:
  - The amount of “torque” required to spin the motor without slipping
  - If the rotor appears to be spinning slowly as the ramp time proceeds then the ramp demand needs to be increased
  - If the whole motor vibrated when the ramp time increases then the demand is too high and most likely the over current will trip

Lab 8. Running Sensorless BLDC Motor

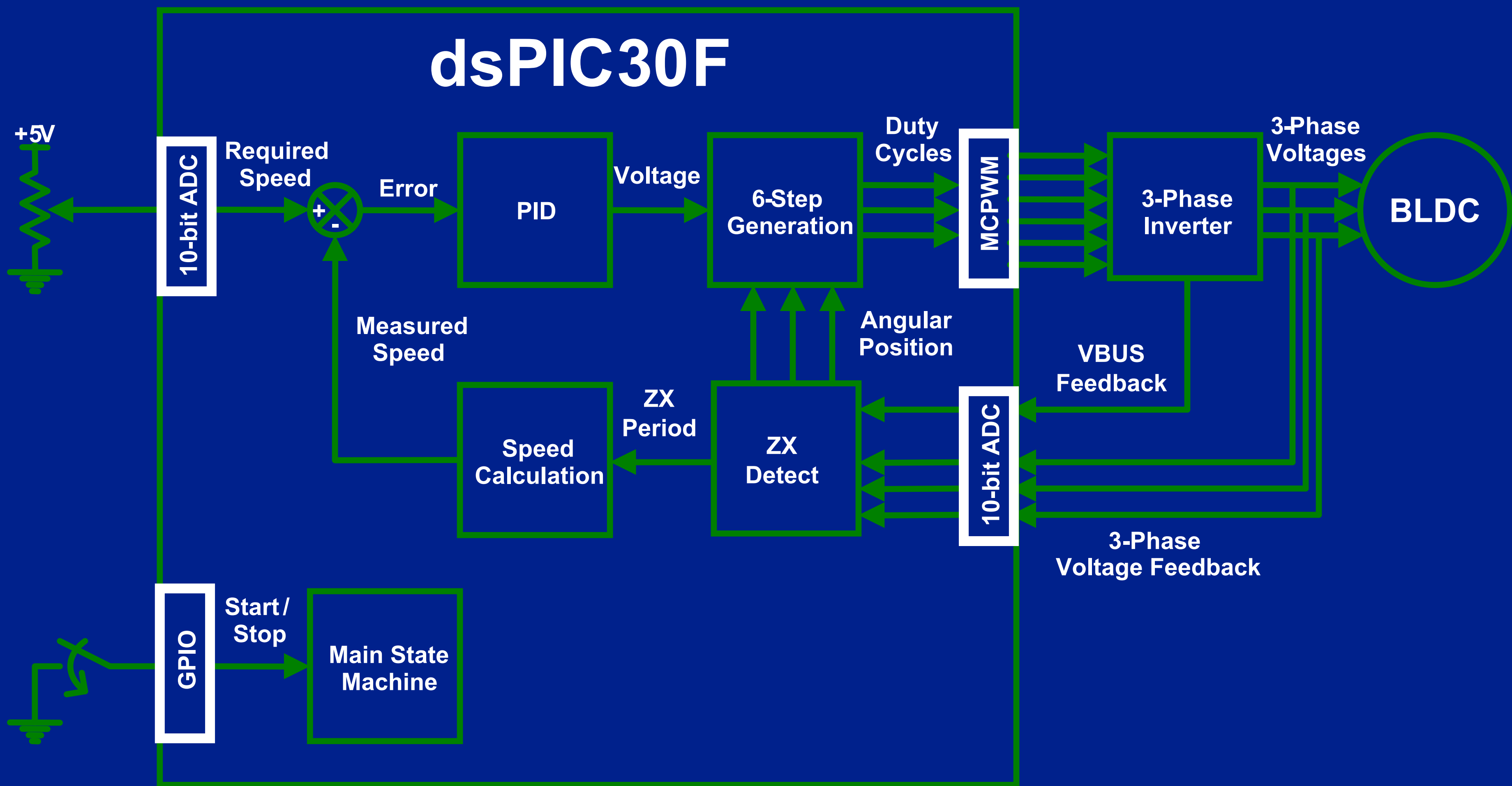
Lab8 Jumper settings

- Turn MCLV board over and refer to the jumper settings for “dsPIC Sensorless”
- Keep Potentiometer REF(R14) and R60 in center position
- Disconnect Hall Sensors from Motor (Black connector)

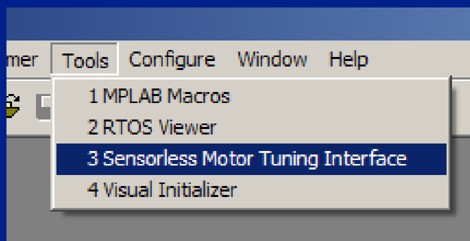
dsPIC® DSC Sensored Settings

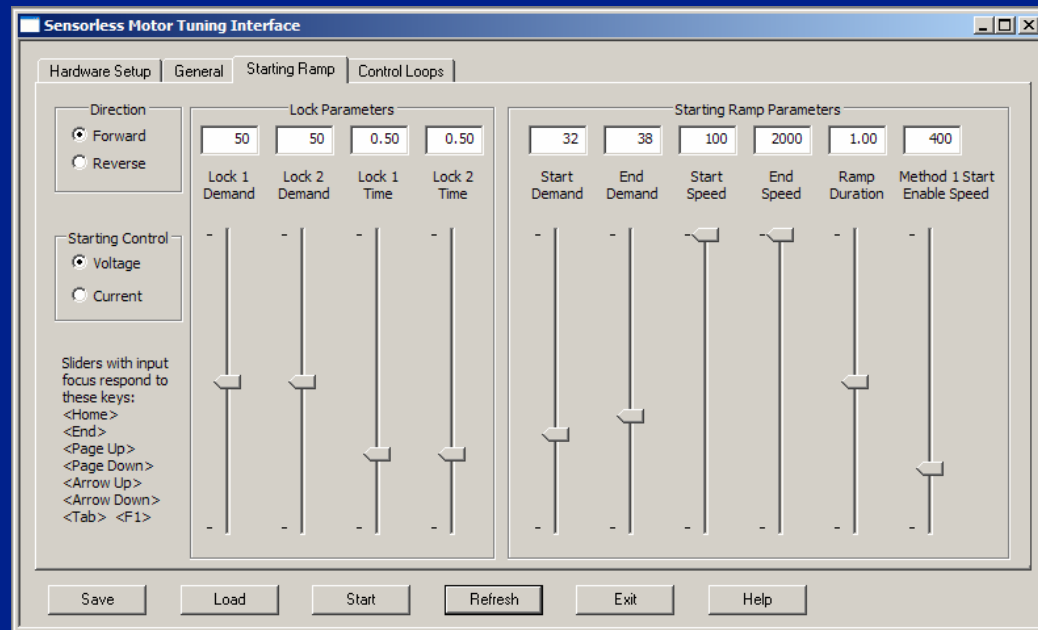
Jumper	Position
J7	2-3
J8	**NC**
J11	2-3
J12	**NC**
J13	2-3
J14	**NC**
J15	**NC**
J10	**NC**
J16	1-2
J17	1-2
J19	**NC**

Running Sensorless BLDC Motor



Sensorless Motor Tuning Interface (SMTI)





- **Visual tool for Tuning sensorless BLDC Applications**
- **Runs with MPLAB® ICD 2**
- **Can change any parameter specified in AN901**
- **See GS005 for Operation Details**

Lab8 – Running Sensorless BLDC Motor using dsPIC® DSC

- Instructions for Lab8:
  - Use workspace "C:\WIB\Lab8\Lab8.mcw"
  - Follow Lab 1 instructions to:
    - Compile code
    - Program dsPIC
    - Run code
  - Disconnect Black connector from Motor

Continued...

Lab8 – Running Sensorless BLDC Motor using dsPIC® DSC

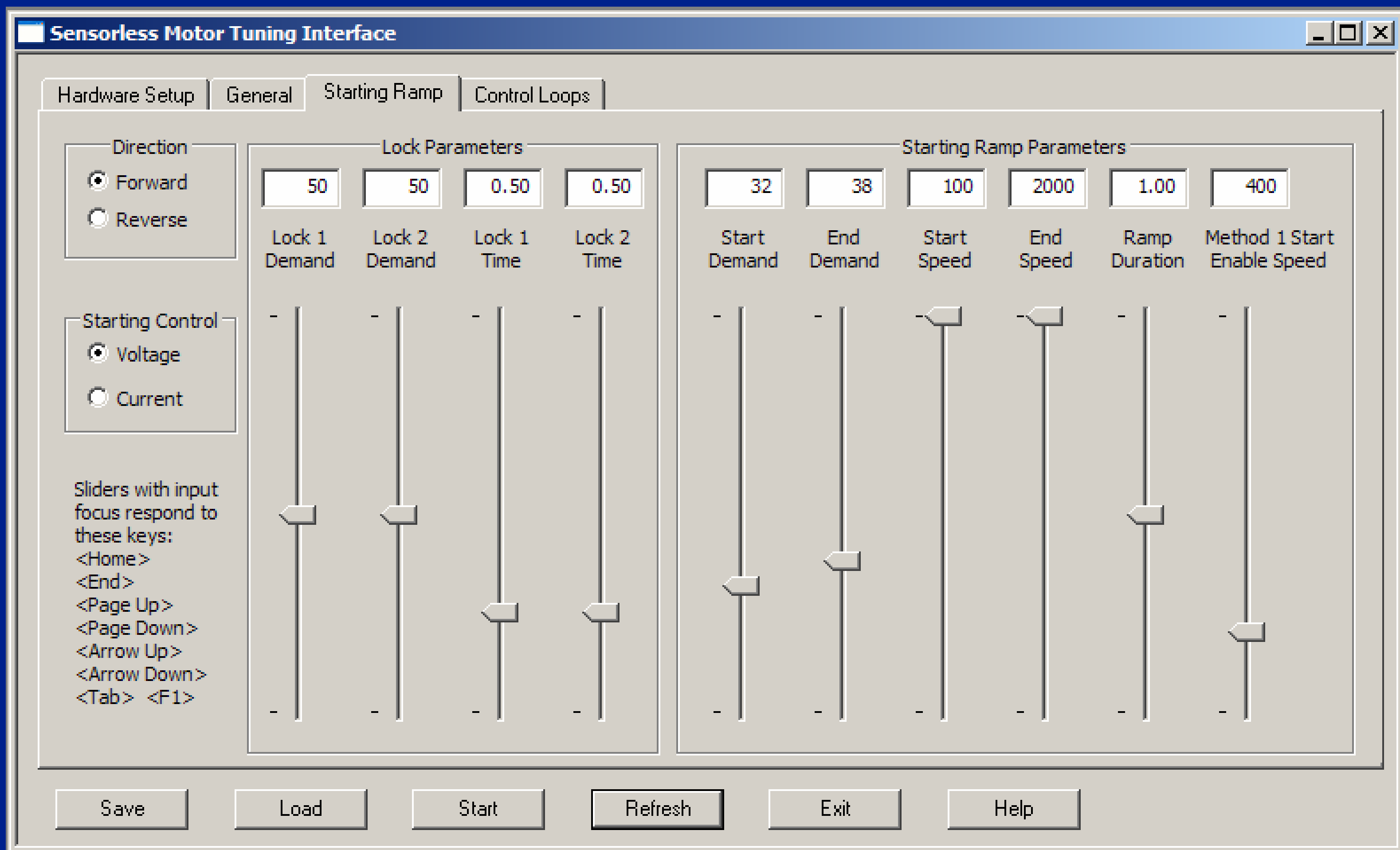
- Open Sensorless Motor Tuning Interface
- Click on Halt in the SMTI window
- Click on Refresh in the SMTI window

PART I

- Change parameters as per next slide

SMTI runs ONLY under Debugger, not Programmer

Lab8 – Running Sensorless BLDC Motor using dsPIC® DSC



Lab8 – Running Sensorless BLDC Motor using dsPIC® DSC

- Click on Start in the SMTI window
- Press S2 to start motor
- Motor appears to start but does not spin
- WHY?
  - Start demand is to low to keep the motor running while ramping up.
  - Hint. Increase the start/end speed ramp demand when motor slips

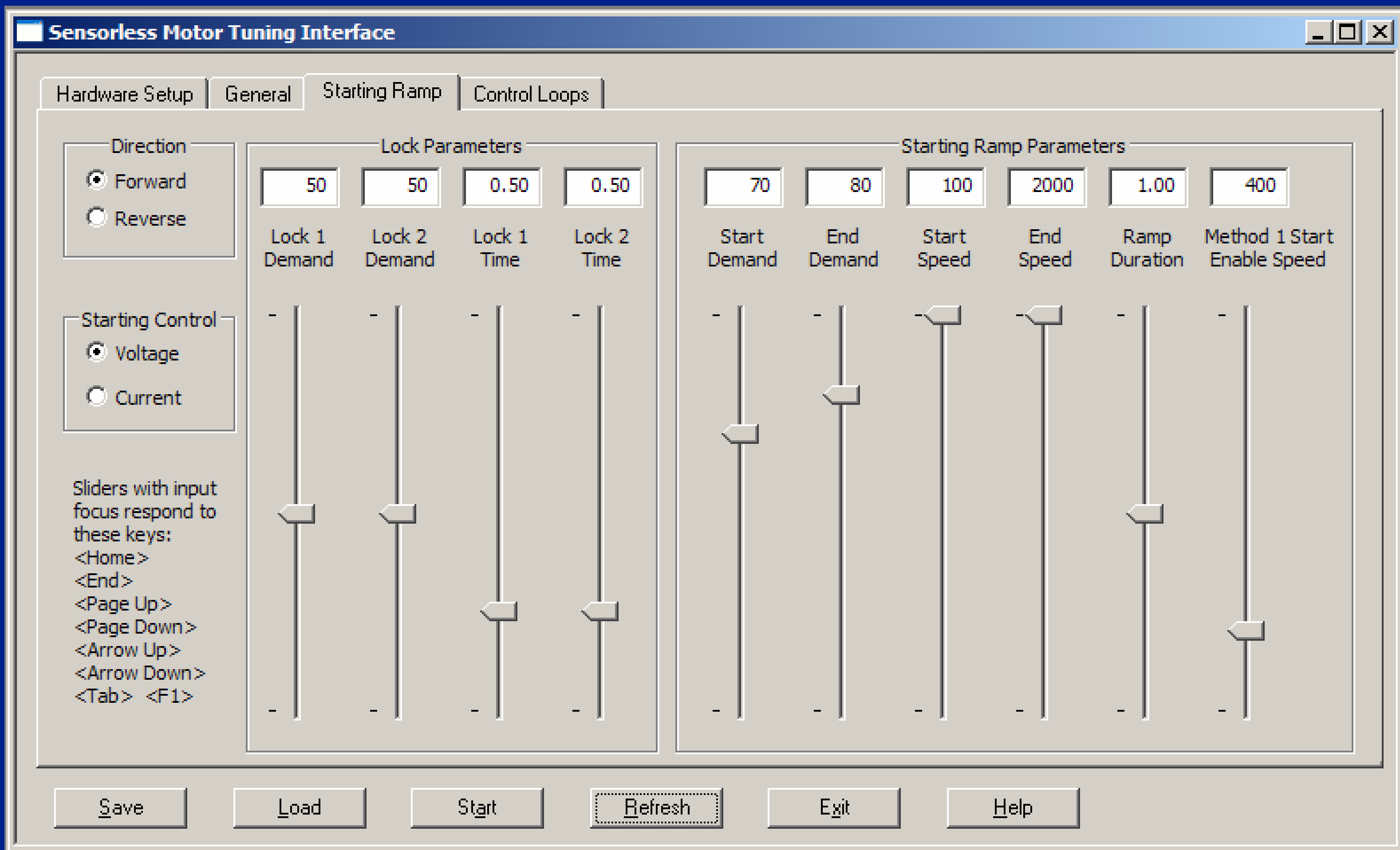
Lab8 – Running Sensorless BLDC Motor using dsPIC® DSC

- Press S2 to stop/reset the motor
- Click on Halt in the SMTI window

PART II

- Change parameters as per next slide

Lab8 – Running Sensorless BLDC Motor using dsPIC® DSC



Lab8 – Running Sensorless BLDC Motor using dsPIC® DSC

- Click on Start in the SMTI window
- Press S2 to start motor
- Motor appears to start but does not spin
- Observe the mechanics of the motor
- What is happening?
  - The motor is vibrating in each sector while ramping up, because the demand was to high
  - This generates a lot of start up current and bad back EMF feedback
  - Hint. Reduce the start/end demands when motor vibrates in every sector while ramping. If we reduce the demands more than we should, the motor will start slipping again

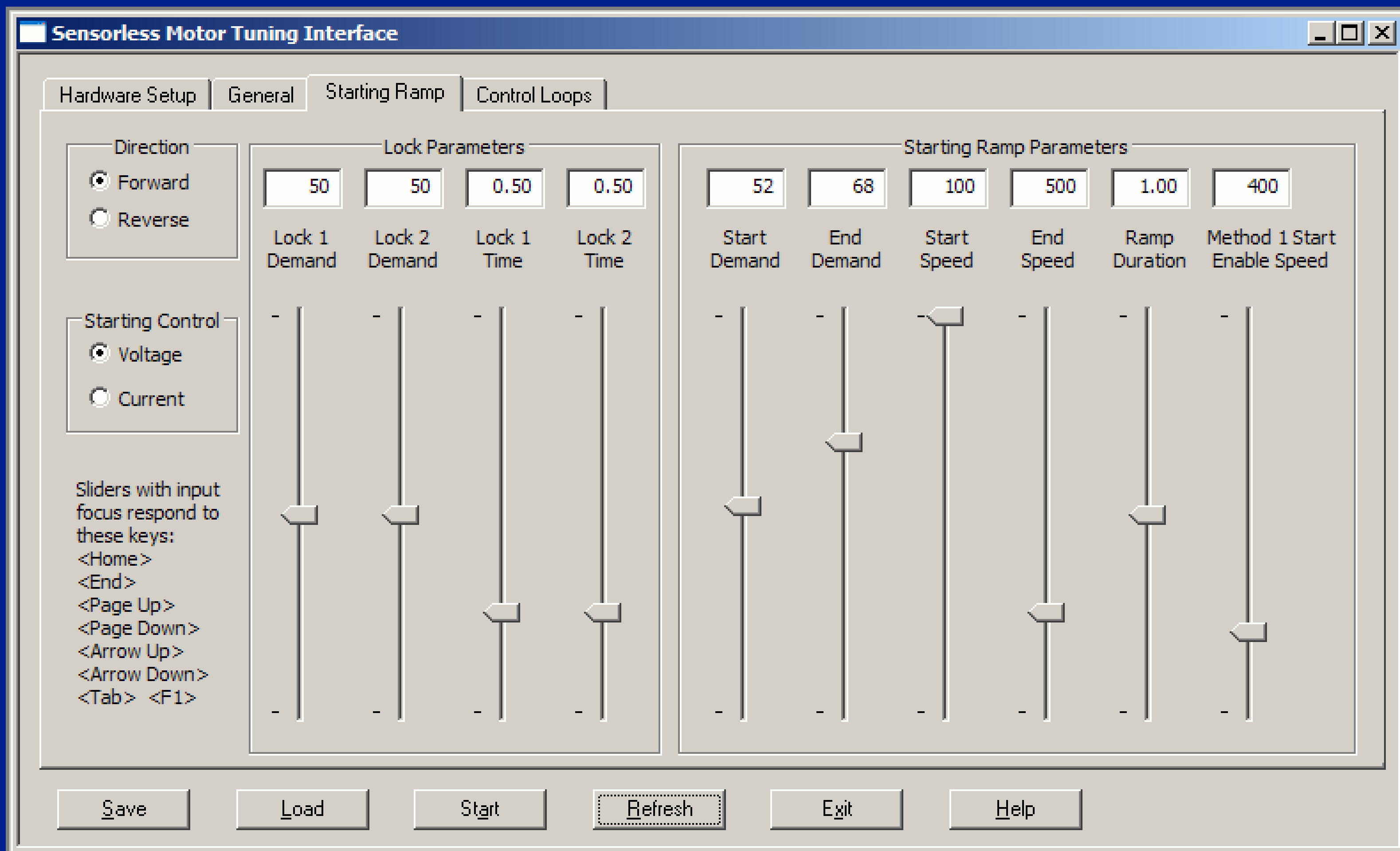
Lab8 – Running Sensorless BLDC Motor using dsPIC® DSC

- Press S2 to stop/reset the motor
- Click on Halt in the SMTI window

Part III

- Change parameters as per next slide

Lab8 – Running Sensorless BLDC Motor using dsPIC® DSC



Lab8 – Running Sensorless BLDC Motor using dsPIC® DSC

- Click on Start in the SMTI window
- Press S2 to start motor
- Motor appears to start but does not spin
- What is wrong?
  - The motor is ramping with good torque and does not slip, but the end speed is too low. Back EMF zero crossings are still not detectable by the controller.
  - Hint. When ramping up a motor, try to set a maximum speed of 50% of the motor rated speed, which is around 2000 RPM in this Motor

Digital Signal Controller

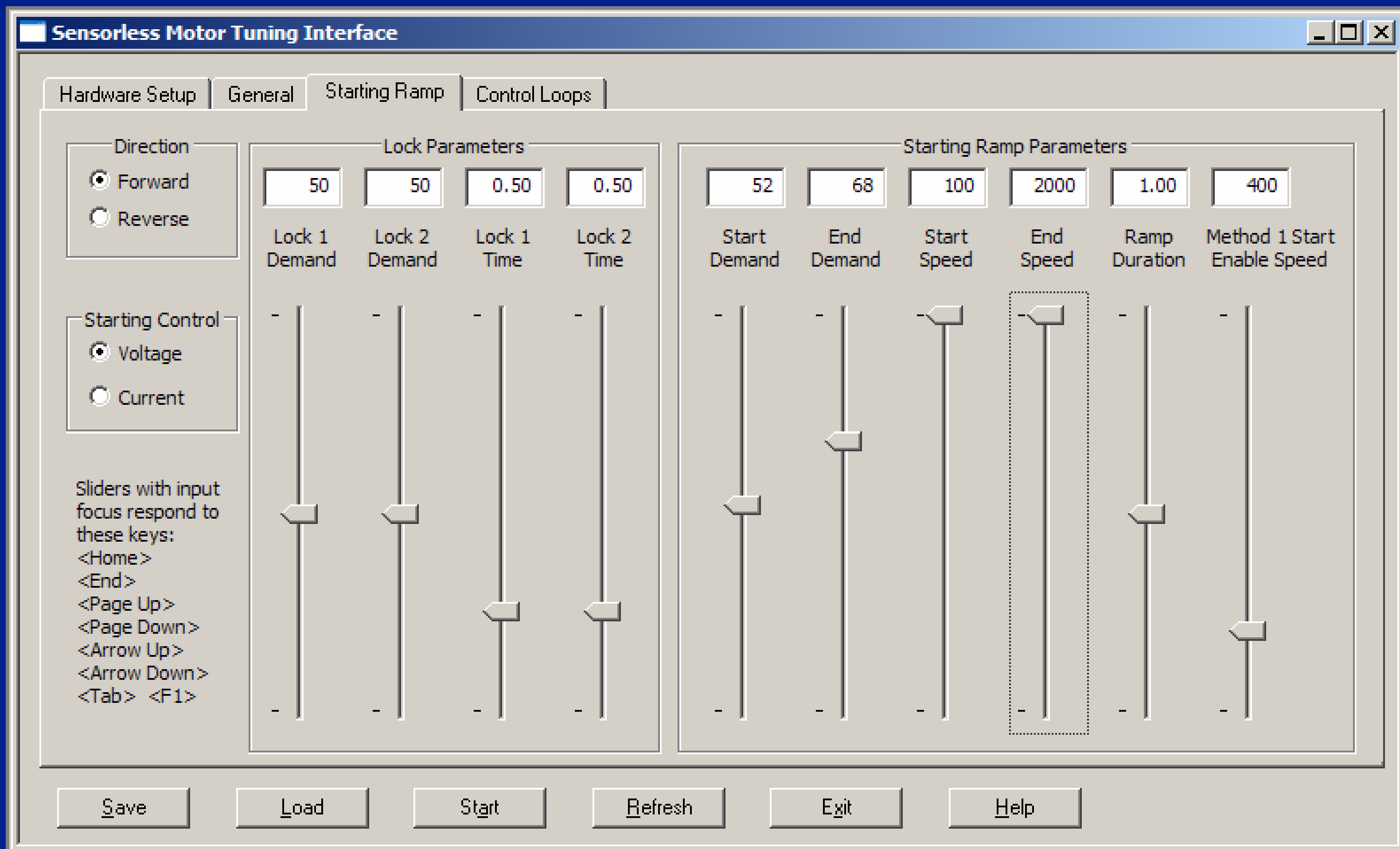
Lab8 – Running Sensorless BLDC Motor using dsPIC® DSC

- Press S2 to stop/reset the motor
- Click on Halt in the SMTI window

PART IV

- Change parameters as per next slide

Lab8 – Running Sensorless BLDC Motor using dsPIC® DSC



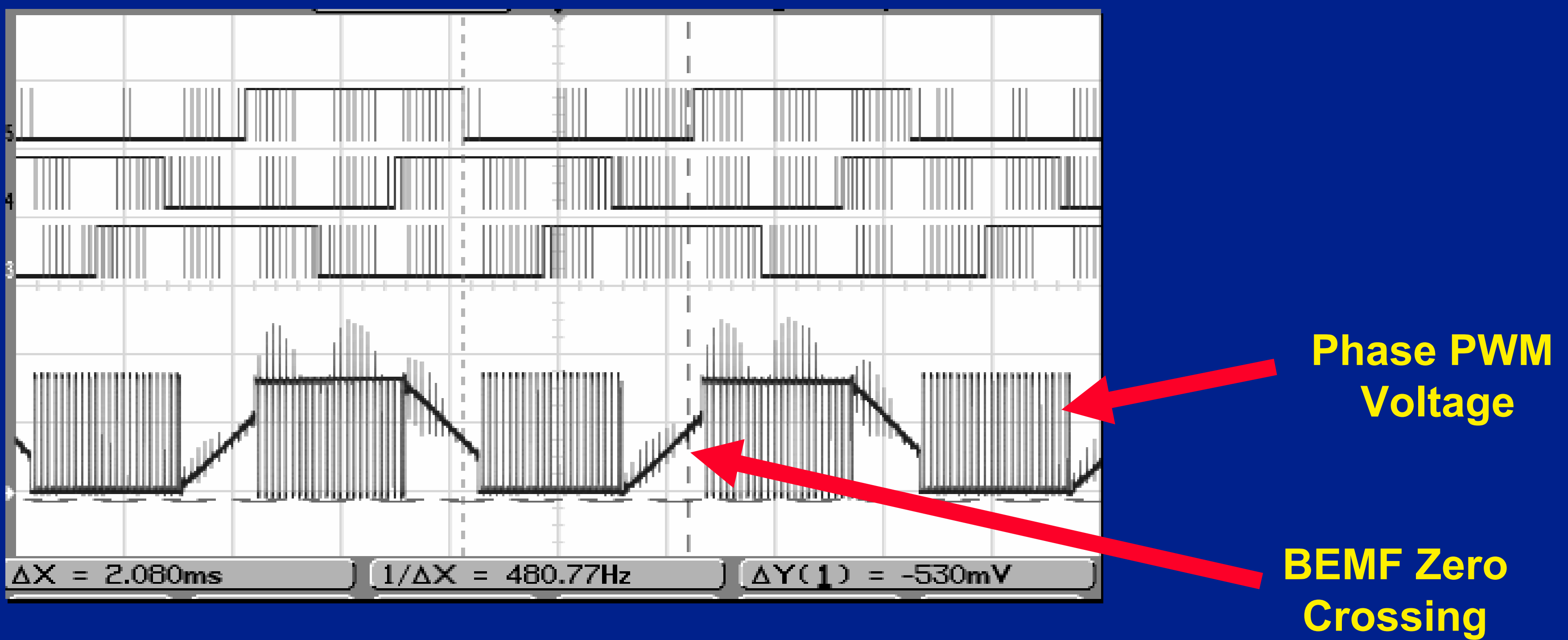
Lab8 – Running Sensorless BLDC Motor using dsPIC® DSC

- Keep Pot in center position
- Click on Start in the SMTI window
- Press S2 to start motor
- Does the motor spin?
- Press S2 to stop/reset the motor

Discuss why the motor spins

Use AN992: Sensorless Control of BLDC motor using dsPIC30F2010, for details

Lab8 – Running Sensorless BLDC Motor using dsPIC® DSC



Summary

- BLDC motor basics
- Blindly spin a BLDC motor
- Improve efficiency by using Hall sensors
- Used dsPIC peripherals to spin a sensored BLDC motor
- Controlling BLDC Speed with Digital PID
- Reducing audible noise of BLDC with Sinusoidal control
- Extending speed range with Phase Advance control
- Techniques for sensorless control
- Modified Parameters in AN901 to spin a sensorless BLDC motor

Reference Application Notes and Collateral

- GS001: Getting Started with BLDC Motors and dsPIC30F
- GS002: Measuring Speed and Position with the QEI Module
- GS004: Driving an ACIM with the dsPIC MWPWM Module
- GS005: Using the dsPIC30F Sensorless Motor Tuning Interface
- CE003: Driving a BLDC with Sinusoidal Voltages using dsPIC30F
- AN901: Sensorless Control of BLDC Motor using dsPIC30F
- AN907: Using the dsPIC30F for Vector Control of an ACIM
- AN957 : Sensored Control of BLDC Motor using dsPIC30F2010
- AN992: Sensorless Control of BLDC Motor using dsPIC30F2010

Thanks for Attending!

Good Luck in Your Development