

# 3-Phase AC Motor Control with V/Hz Speed Closed Loop Using the DSP56F80X

Design of motor control application based on Motorola Software Development Kit

*Petr Uhlir  
Zdenek Kubiczek*

## 1. Introduction

This application note describes the design of a 3-phase AC induction motor drive with volt per hertz control in closed loop (hereinafter called V/Hz OL). It is based on Motorola's 56F80X digital signal processor (DSP), which is dedicated for motor control applications. The system is designed as a motor control system for driving medium power, 3-phase AC induction motors. The part is targeted toward applications in both industrial and home appliance industries, such as washing machines, compressors, air conditioning units, pumps, or simple industrial drives. The software design takes advantage of SDK (Software Development Kit) developed by Motorola.

The drive introduced here is intended as an example of a 3-phase AC induction motor drive. The drive serves as an example of AC V/Hz motor control system design using Motorola DSP with SDK support. It also illustrates the usage of dedicated motor control libraries included in the SDK.

This document includes the basic motor theory, system design concept, hardware implementation, and software design, including the PC Master visualization tool inclusion.

## 2. Motorola DSP Advantages and Features

The Motorola DSP56F80x family is well suited for digital motor control, combining the DSP's calculation capability with MCUs controller features on a single chip. These DSPs

## Contents

1. Introduction .....	1
2. Motorola DSP Advantages and Features .....	1
3. Target Motor Theory .....	3
3.1 3-phase AC Induction Motor Drives .....	3
3.2 Volts per Hertz Control .....	5
3.3 Speed Close Loop System .....	6
4. System Design Concept .....	7
5. Hardware .....	9
5.1 System Outline .....	9
5.2 High Voltage Hardware Set.....	9
6. Software Design .....	11
6.1 Data Flow .....	11
6.1.1 Acceleration/Deceleration Ramp .....	12
6.1.2 Speed Measurement .....	12
6.1.3 PI Controller.....	12
6.1.4 V/Hz Ramp.....	12
6.1.5 DC-Bus Voltage Ripple Elimination .....	13
6.1.6 PWM Generation .....	14
6.1.7 Fault Control .....	17
6.2 State Diagram .....	18
6.2.1 Initialization .....	18
6.2.2 Application State Machine .....	20
6.2.3 Check Run/Stop Switch .....	20
6.2.4 PWM Reload A ISR .....	20
6.2.5 PWM Fault A ISR .....	21
6.2.6 ADC Conversion Complete ISR .....	21
6.2.7 ADC High Limit ISR .....	21
6.2.8 ADC Low Limit ISR .....	21
6.2.9 Timer OC LED ISR .....	21
6.2.10 Timer OC Ramp ISR.....	21
7. SDK Implementation .....	22
7.1 Drivers and Library Function .....	22
7.2 Appconfig.h File.....	22
7.3 Drivers Initialization.....	22
7.4 Interrupts.....	23
7.5 PC Master .....	23
8. DSP Usage .....	23
9. References .....	24



offer a rich dedicated peripherals set, such as pulse width modulation (PWM) modules, analog-to-digital converter (ADC), timers, communication peripherals (SCI, SPI, CAN), on-board flash and RAM. Several parts comprise the family: DSP56F801/803/805/807, with different peripherals and on-board memory configurations. Generally, all are well suited for motor control.

The typical member of the family, the DSP56F805, provides the following peripheral blocks:

- Two pulse width modulator modules (PWMA & PWMB), each with six PWM outputs, three current status inputs, and four fault inputs, fault tolerant design with deadtime insertion, supports both center- and edge- aligned modes
- Two 12-bit, analog-to-digital convertors (ADCs), supporting two simultaneous conversions with dual 4-pin multiplexed inputs, ADC and can be synchronized by PWM modules synchronized
- Two quadrature decoders (Quad Dec0 & Quad Dec1), each with four inputs, or two additional quad timers A & B
- Two dedicated general purpose quad timers totalling 6 pins: Timer C with 2 pins and Timer D with 4 pins
- CAN 2.0 A/B module with 2-pin ports used to transmit and receive
- Two serial communication interfaces (SCI0 & SCI1), each with two pins, or four additional MPIO lines
- Serial peripheral interface (SPI), with configurable 4-pin port, or four additional MPIO lines
- Computer operating properly (COP) timer
- Two dedicated external interrupt pins
- Fourteen dedicated multiple purpose I/O (MPIO) pins and 18 multiplexed MPIO pins
- External reset pin for hardware reset
- JTAG/on-chip emulation (OnCE™)
- Software-programmable, phase lock loop-based frequency synthesizer for the DSP core clock
- Memory configuration
  - 32252 × 16-bit words of program flash
  - 512 × 16-bit words of program RAM
  - 2K × 16-bit words of data RAM
  - 4K × 16-bit words of data flash
  - 2K × 16-bit words of boot flash

The pulse-width-modulation (PWM) block offers high freedom in its configuration enabling to control the AC induction motor in efficient way.

The PWM block has the following features:

- Three complementary PWM signal pairs, or six independent PWM signals
- Features of complementary channel operation
- Deadtime insertion
- Separate top and bottom pulse width correction via current status inputs or software
- Separate top and bottom polarity control
- Edge-aligned or center-aligned PWM reference signals
- 15-bits of resolution

- Half-cycle reload capability
- Integral reload rates from one to 16
- Individual software-controlled PWM output
- Programmable fault protection
- Polarity control
- 20-mA current sink capability on PWM pins
- Write-protectable registers

The PWM outputs are configured in the complementary mode in this application.

## 3. Target Motor Theory

### 3.1 3-phase AC Induction Motor Drives

The AC induction motor is a workhorse with adjustable speed drive systems. The most popular type is the 3-phase, squirrel-cage AC induction motor. It is maintenance-free, lower noise and efficient motor. The stator is supplied by a balanced 3-phase AC power source.

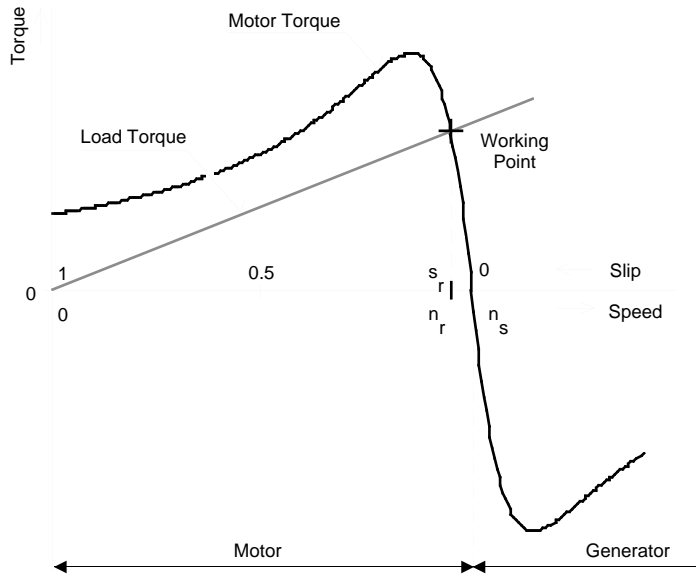
The synchronous speed  $n_s$  of the motor is given by

$$n_s = \frac{120 \times f_s}{p} \quad [rpm] \quad (\text{EQ 3-1.})$$

where  $f_s$  is the synchronous stator frequency in Hz, and  $p$  is the number of stator poles. The load torque is produced by slip frequency. The motor speed is characterized by a slip  $s_r$ :

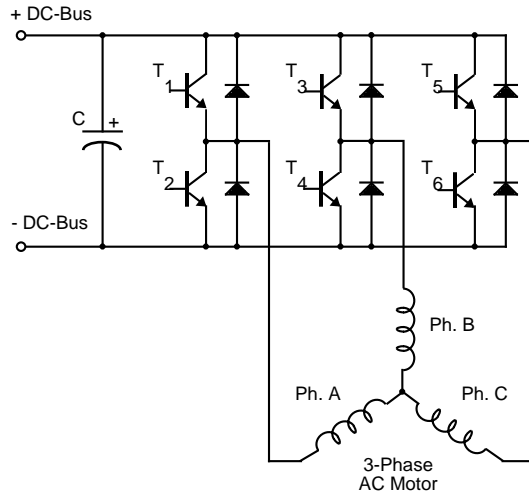
$$s_r = \frac{(n_s - n_r)}{n_s} = \frac{n_{sl}}{n_s} \quad [-] \quad (\text{EQ 3-2.})$$

where  $n_r$  is the rotor mechanical speed and  $n_{sl}$  is the slip speed, both in rpm. **Figure 3-1** illustrates the torque characteristics and corresponding slip. As can be seen from **EQ 3-1** and **EQ 3-2** the motor speed is controlled by variation of a stator frequency with influence of the load torque.



**Figure 3-1. Torque-Speed Characteristic at Constant Voltage and Frequency**

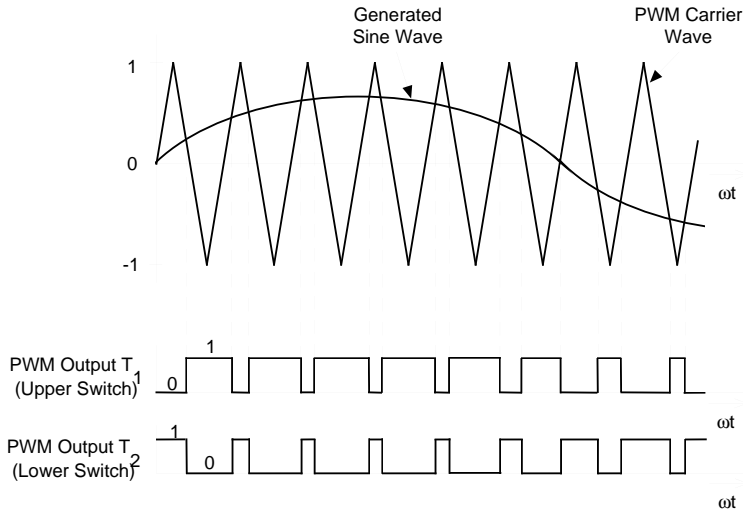
In adjustable speed applications the AC motors are powered by inverters. The inverter converts DC power to AC power at required frequency and amplitude. The typical 3-phase inverter is illustrated in [Figure 3-2](#).



**Figure 3-2. 3- Phase Inverter**

The inverter consists of three half-bridge units where the upper and lower switch is controlled complementarily - meaning when the upper one is turned-on, the lower one must be turned-off and vice versa. As the power device's turn-off time is longer than its turn-on time, some dead-time must be inserted between the turn-off of one transistor of the half-bridge and turn-on of it's complementary device. The output voltage is mostly created by a pulse width modulation (PWM) technique where an isosceles triangle carrier wave is compared with a fundamental-frequency sine modulating wave, and

the natural points of intersection determine the switching points of the power devices of a half bridge inverter. This technique is shown in **Figure 3-3**. The 3-phase voltage waves are shifted 120° to each other and thus a 3-phase motor can be supplied.



**Figure 3-3. Pulse Width Modulation**

The most popular power devices for motor control applications are Power MOSFETs and IGBTs.

A Power MOSFET is a voltage controlled transistor. It is designed for high frequency operation and it has a low voltage drop, thus it has low power losses. However, the saturation temperature sensitivity limits the MOSFET application in high power applications.

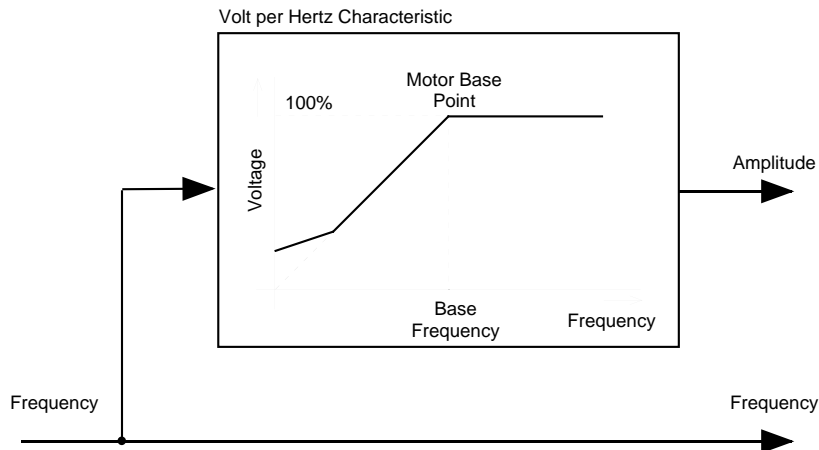
An insulated gate bipolar transistor (IGBT) is a bipolar transistor controlled by a MOSFET on its base. The IGBT requires low drive current, has fast switching time, and is suitable for high switching frequencies. The disadvantage is its higher voltage drop of the bipolar transistor, causing higher conduction losses.

### 3.2 Volts per Hertz Control

Volt per Hertz control methods is the most popular method of Scalar Control, controls the magnitude of the variable like frequency, voltage or current. The command and feedback signals are DC quantities, and are proportional to the respective variables.

The purpose of the volt per hertz control scheme is to maintain the air-gap flux of AC Induction motor in constant in order to achieve higher run-time efficiency. In steady state operation the machine air-gap flux is approximately related to the ratio  $V_s/f_s$ , where  $V_s$  is the amplitude of motor phase voltage and  $f_s$  is the synchronous electrical frequency applied to the motor. The control system is illustrated in **Figure 3-4**. The characteristic is defined by the base point of the motor. Below the base point the motor operates at optimum excitation because of the constant  $V_s/f_s$  ratio. Above this point the motor operates under-excited because of the DC-Bus voltage limit.

A simple close-loop volts/hertz speed control for an induction motor is the control technique targeted for low performance drives. This basic scheme is unsatisfactory for more demanding applications where speed precision is required.

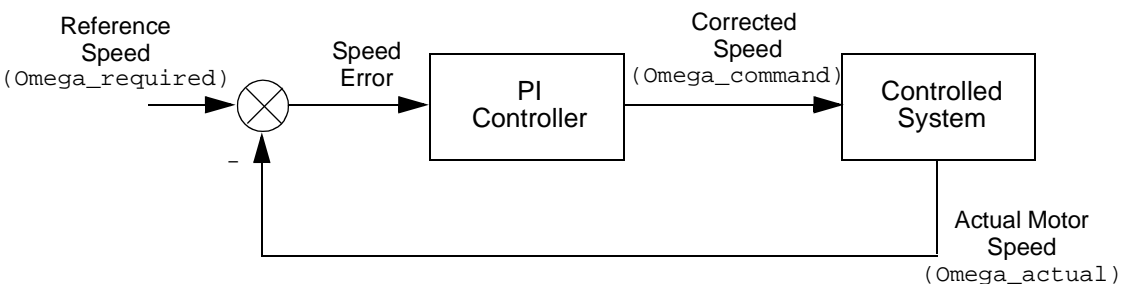


**Figure 3-4. Volts per Hertz Control Method**

### 3.3 Speed Close Loop System

To improve the system performance, a closed-loop volts per hertz control was introduced. In this method a speed sensor measures the actual motor speed and the system takes this input into consideration. A number of applications use the closed-loop volts per hertz method because of its simple and relatively good speed accuracy, but it is not suitable for systems requiring servo performance or excellent response to highly dynamic torque/speed variations.

**Figure 3-5** illustrates the general principle of the speed PI control loop.



**Figure 3-5. Closed Loop Control System**

The speed closed loop control is characterized by the measurement of the actual motor speed. This information is compared with the reference speed while the error signal is generated. The magnitude and polarity of the error signal correspond to the difference between the actual and required speed. Based on the speed error the PI controller generates the corrected motor stator frequency in order to compensate for the error.

In a case of AC V/Hz closed loop application, the feedback speed signal is derived from incremental encoder using the quadrature decoder. The speed controller constants have been tuned experimentally according to the actual load.

## 4. System Design Concept

The system is designed to drive a 3-phase AC induction motor. The application meets the following performance specifications:

- Targeted for DSP56F80XEVM platforms
- Running on 3-phase ACIM motor control development platform at variable line voltage 115 - 230V AC
- Control technique incorporates
  - motoring and generating mode
  - bi-directional rotation
  - V/Hz speed close loop
- Manual Interface (Start/Stop switch, Up/Down push button speed control, LED indication)
- PC Master Interface (motor start/stop, speed set-up)
- Power stage identification
- Overvoltage, undervoltage, overcurrent, and overheating fault protection

The introduced AC drive is designed as a DSP system that meets the following general performance requirements:

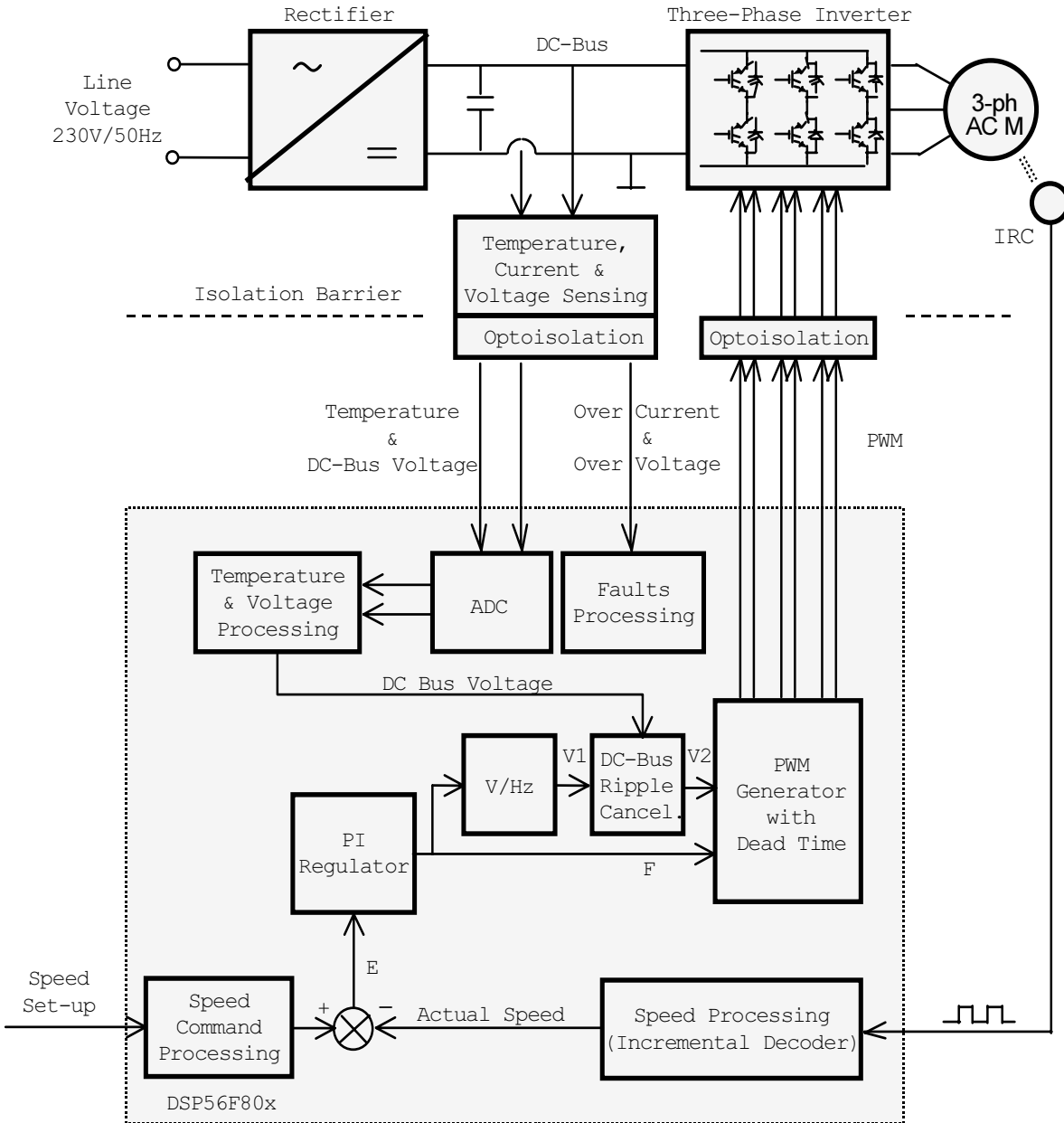
**Table 4-1. Motor / Drive Specification**

<b>Motor Characteristics:</b>	Motor Type	4 poles, three phase, star connected, squirrel cage AC motor (standard industrial motor)
	Speed Range:	< 5000 rpm
	Base Electrical Frequency:	50 Hz
	Max. Electrical Power:	180 W
	Delta Voltage (rms):	200V (Star)
<b>Drive Characteristics:</b>	Transducers:	IRC -1024 pulses per rev.
	Speed Range	<2250 rpm @ 230 V <1200 rpm @ 115 V
	Line Input:	230V / 50Hz AC 115V / 60Hz AC
	Max. DC Bus Voltage	400 V
	Control Algorithm	Close Loop Control
	Optoisolation	Required
<b>Load Characteristic:</b>	Type	Varying

The DSP runs the main control algorithm. According to the user interface input and feedback signals, it generates 3-phase PWM output signals for the motor inverter.

A standard system concept is chosen for the drive, and illustrated in [Figure 4-1](#). The system incorporates the following hardware boards:

- Power supply rectifier
- 3-phase inverter
- Feedback sensors: speed, DC-bus voltage, DC-bus current, temperature
- Optoisolation
- Evaluation board DSP56F80X



**Figure 4-1. System Concept**



**The Control Process:**

When the start command is accepted, using the Start/Stop switch, the state of the inputs is periodically scanned. According to the state of the control signals (Start/Stop switch, speed up/down buttons or PC Master set speed) the speed command is calculated using an acceleration/deceleration ramp.

The comparison between the actual speed command and the measured speed generates a speed error E. The speed error is brought to the speed PI controller that generates a new corrected motor stator frequency. With the use of the V/Hz ramp the corresponding voltage is calculated and then DC-bus ripple cancellation function eliminates the influence of the DC-bus voltage ripples to the generated phase voltage amplitude. The PWM generation process calculates a 3-phase voltage system at the required amplitude and frequency, includes dead time. Finally the 3-phase PWM motor control signals are generated.

The DC-bus voltage and power stage temperature are measured during the control process. They are overvoltage, undervoltage, and overheating protection of the drive. Both undervoltage protection and overheating are performed by ADC and software while the DC-bus overcurrent and overvoltage fault signals are connected to PWM fault inputs.

If any of the above mentioned faults occurs, the motor control PWM outputs are disabled in order to protect the drive and the fault state of the system is displayed in PC Master control page.

## 5. Hardware

### 5.1 System Outline

The motor control system is designed to drive the 3-phase AC motor in a speed close loop.

There are more SW versions targeted for a real DSP and evaluation module (DSP/EVM):

- DSP56F80X

The HW setup for a real DSP/EVM differs only by evaluation module (EVM) module used.

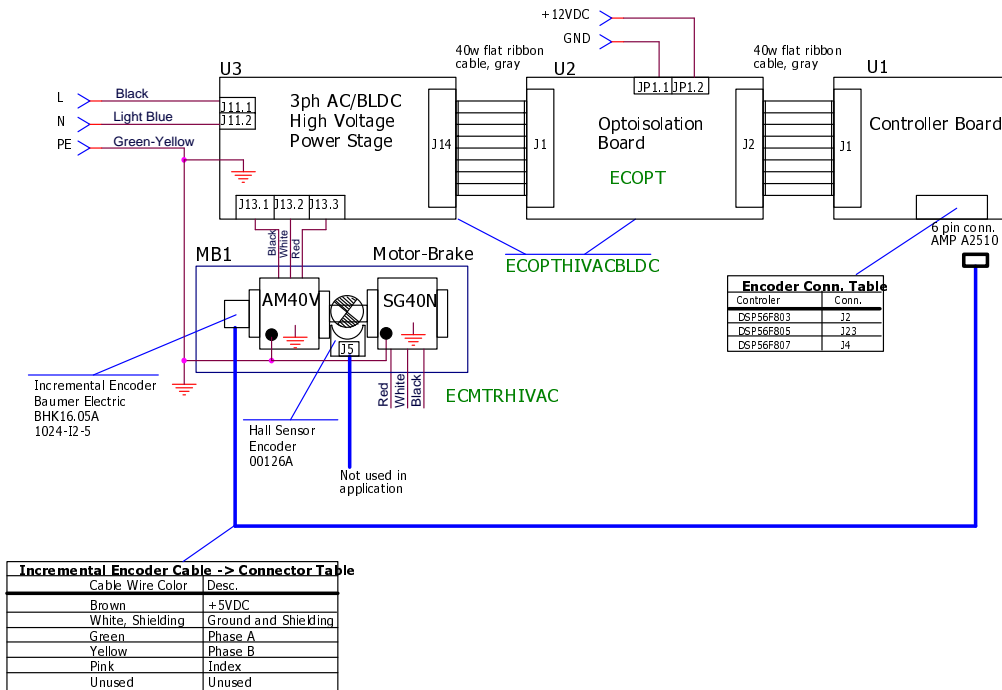
The designed software is capable to run only on high voltage HW set described below.

Other power module boards will be denied due to board identification build in SW. This feature protects misuse of HW module.

The HW setup is shown in [Figure 4-1](#), but it can also be found in the documents *Targetting\_DSP5680X\_Platform*, according to targeted DSP/EVM. That documents also describes EVM jumper settings.

### 5.2 High Voltage Hardware Set

The system configuration is shown in [Figure 5-1](#).



**Figure 5-1. High Voltage HW System Configuration**

All the system parts are supplied and documented according the following references:

- U1 - Controller board for DSP56F80X:
  - supplied as: DSP5680XEVM
  - described in: *DSP56F80XEVMUM/D DSP Evaluation Module Hardware User's Manual*
- U2 - 3-ph AC/BLDC high voltage power stage
  - supplied in kit with optoisolation board as: ECOPTHIVACBLDC
  - described in: *MEMC3BLDCPSUM/D - 3 Phase Brushless DC High Voltage Power Stage*
- U3 - Optoisolation board
  - supplied with 3-ph AC/BLDC high voltage power stage as: ECOPTHIVACBLDC
  - or supplied alone as: ECOPT - optoisolation board
  - described in: *MEMCOBUM/D Optoisolation board User's Manual*
- MB1 motor-brake AM40V + SG40N
  - supplied as: ECMTRHIVAC

**Warning:** It is strongly recommended to use opto-isolation (optocouplers and optoisolation amplifiers) during the development time to avoid any damage to the development equipment.

**Note:** The detailed description of individual boards can be found in comprehensive users' manuals belonging to each board. The user manual incorporates the schematic of the board, description of individual function blocks and bill of materials. Individual boards can be ordered from Motorola as a standard product from <http://mot-sps.com/motor/devtools/index.html>.

This section describes the design of the software blocks of the drive. The software will be described in terms of data flow and state diagrams.

## 6. Software Design

### 6.1 Data Flow

The requirements of the drive dictates the software gather some values from the user interface and sensors, process them and generate 3-phase PWM signals for inverter.

The control algorithm of close loop AC drive is described in **Figure 6-1**. The control algorithm contains the processes described in the following subsections. The detailed description is given to the subroutines 3-phase PWM calculation and volt per hertz control algorithm.

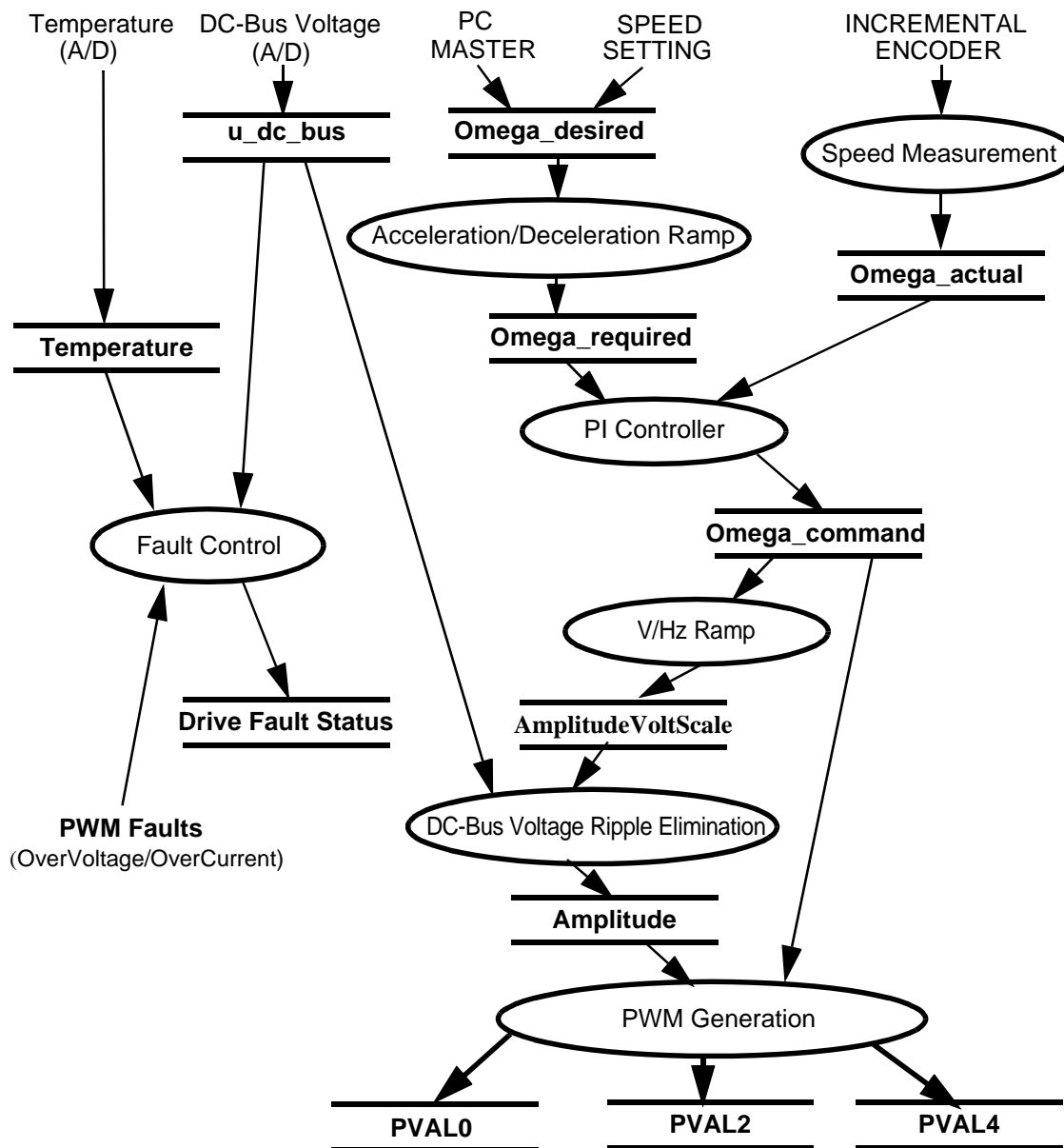


Figure 6-1. Data Flow

### 6.1.1 Acceleration/Deceleration Ramp

The process calculates the new actual speed command based on the required speed according to the acceleration/deceleration ramp. The desired speed is determined either by push buttons or by the PC Master.

During deceleration the motor can work as a generator. In the generator state the DC-bus capacitor is charged and its voltage can easily exceed its maximal voltage. Therefore, the voltage level in the DC-bus link is controlled by a resistive brake, operating in case of overvoltage.

The process input parameter is *Omega\_desired*, the desired speed.

The process output parameter is *Omega\_required*, used as an input parameter of the PWM generation process.

### 6.1.2 Speed Measurement

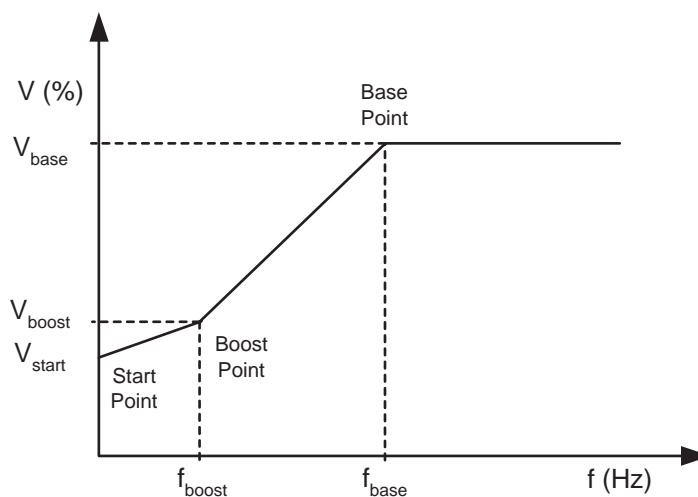
The speed measurement process uses the on-chip quadrature decoder. The process output is *MeasuredSpeed*, and is only used as an information value in PC Master.

### 6.1.3 PI Controller

The PI controller process takes the input parameters, actual speed command *Omega\_required*, and actual motor speed, measured by an incremental encoder *Omega\_actual*. The PI controller calculates a speed error and performs the speed PI control algorithm. The output of the PI controller is a frequency of the first harmonic sine wave to be generated by the inverter: *Omega\_command*.

### 6.1.4 V/Hz Ramp

The drive is designed as a volt per hertz drive. It means, the control algorithm keeps the constant motor's magnetizing current (flux) by varying the stator voltage with frequency. The commonly used volt per hertz ramp of a 3-phase AC induction motor is illustrated in **Figure 6-2**.



**Figure 6-2. Volt per Hertz Ramp**

The volt per hertz ramp is defined by following parameters:

- Base point - defined by  $f_{base}$  (usually 50Hz or 60Hz)
- Boost point- defined by  $V_{boost}$  and  $f_{boost}$
- Start point - defined by  $V_{start}$  at zero frequency

The ramp profile fits to the specific motor and can be easily changed to accommodate different ones.

**Process Description**

This process provides voltage calculation according to V/Hz ramp.

The input of this process is generated by desired inverter frequency: *Omega\_required*.

The output of this process is *AmplitudeVoltScale*, that is, a parameter required by DC-bus voltage ripple elimination process.

### 6.1.5 DC-Bus Voltage Ripple Elimination

**Process Description**

The voltage ripple elimination process eliminates the influence of the DC-bus voltage ripples to the generated phase voltage sinewaves. In fact, it lowers the 50 or 60Hz acoustic noise of the motor. Another positive aspect due to this function is the generated phase voltage. It is independent of the level of DC-bus voltage. So, the application is well adaptable in worldwide power supply system.

The process is performed by the *mcgenDCBVoltRippleElim* function, converting the phase voltage amplitude (*AmplitudeVoltScale*) to the sine wave amplitude (*Amplitude*) based on the actual value of the DC-bus voltage (*u\_dc\_bus*) and inverse value of the modulation index (*ModulationIndexInverse*).

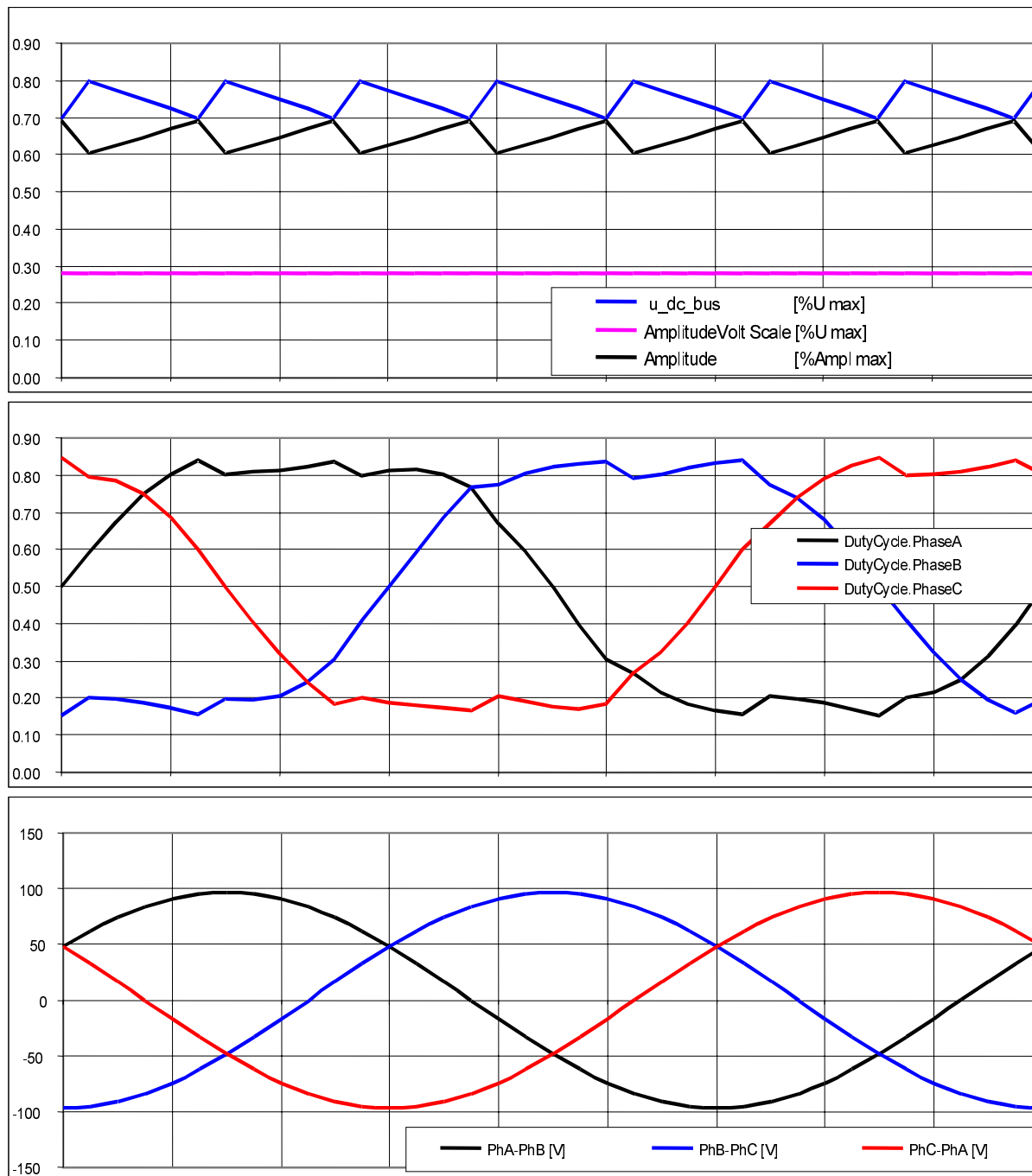
The *modulation index* is the ratio between the maximum amplitude of the first harmonic of the phase voltage (in voltage scale) and half of DC bus voltage (in voltage scale) which is defined by the following formula:

$$m_i = \frac{U_{phasemax}^{(1)}}{\frac{1}{2} \cdot u_{DCBus}} = \frac{2}{\sqrt{3}} \tag{EQ 6-1.}$$

The modulation index is specific to a given 3-phase generation algorithm and in the case of the application, it is 1.27.

**Note:** The result of the modulation index is based on the third harmonic injection PWM technique.

The first chart in **Figure 6-3** demonstrates how the *Amplitude* (in scale of generated sine wave amplitude) is counter-modulated in order to eliminate the DC-bus ripples. The second chart delineates the duty cycles generated by one of the 3-ph wave generation functions. The third chart contains symmetrical sine-waves of the phase-to-phase voltages actually applied to the 3-phase motor.



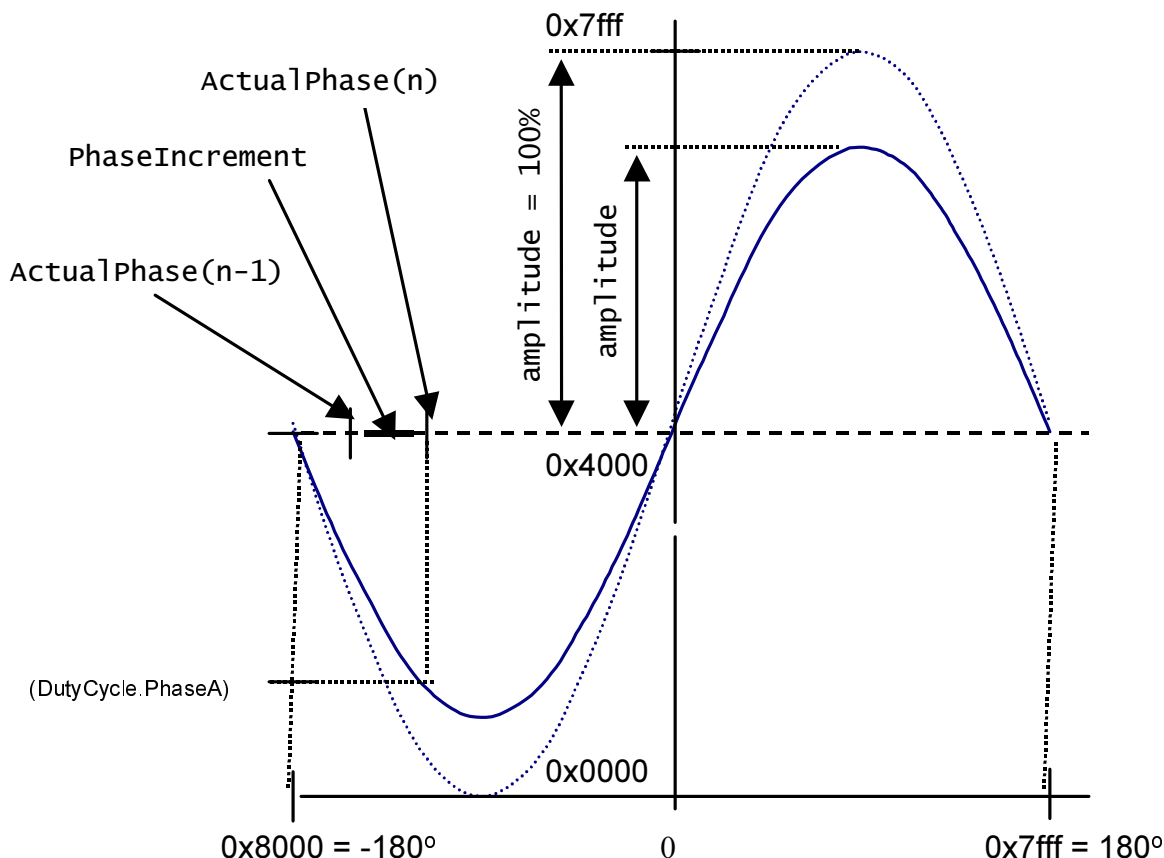
**Figure 6-3. 3-ph Waveforms with DC-Bus Voltage Ripple Elimination**

### 6.1.6 PWM Generation

#### Process Description

This process generates a system of 3-phase sinewaves with addition of third harmonic component shifted  $120^\circ$  to each other using `mcgen3PhWaveSine3rdHIntp` function from the motor control function library.

The function is based on a fix wave table describing the first quadrant of sine wave stored in data memory of the DSP. Due to symmetry of sine function, data in other quadrants are calculated using the data of first quadrant. It saves data memory. The sinewave generation for phase A, simplicity, is explained in **Figure 6-4**. Phases B and C are shifted 120° with respect to Phase A.



**Figure 6-4. Sinewave generation**

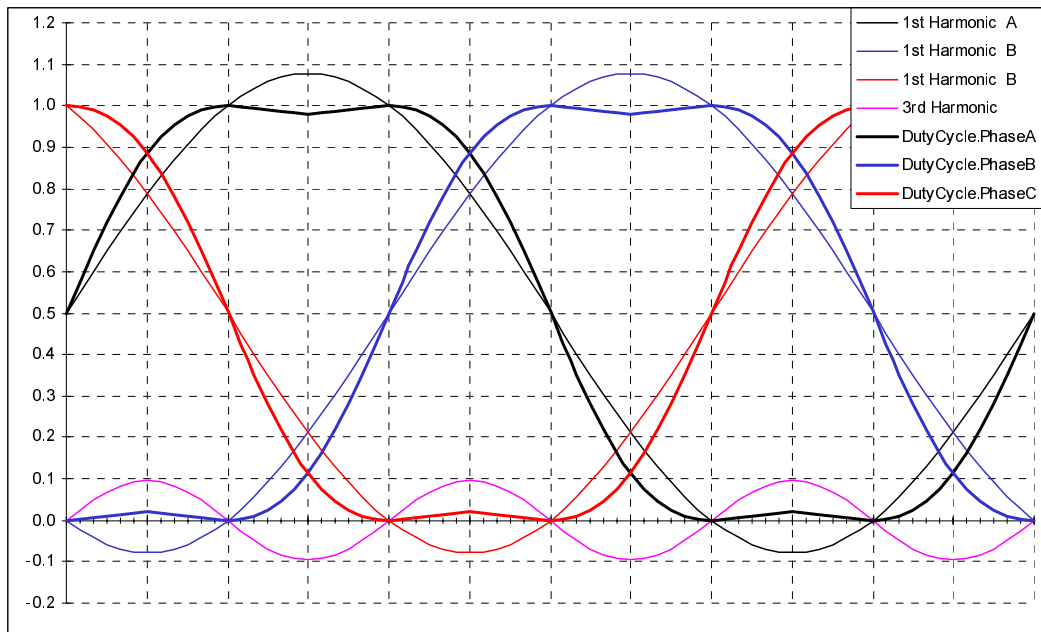
Each time the waveform generation function is called, ActualPhase from previous step is updated by PhaseIncrement, and according calculated phase the value of sine is fetched from the sine table (using function *tfr16SinPIxLUT* from DSP functional library). Then it's multiplied by amplitude and passed to the PWM. Explanation of the 3-phase waveform generation with 3rd harmonic addition, see the following formulas.

$$\begin{aligned}
 PWMA &= \frac{1}{\sqrt{3}} \cdot Amplitude \cdot \left( \sin \alpha + \frac{1}{6} \cdot \sin 3\alpha \right) + 0.5 \\
 PWMB &= \frac{1}{\sqrt{3}} \cdot Amplitude \cdot \left( \sin(\alpha - 120^\circ) + \frac{1}{6} \cdot \sin 3\alpha \right) + 0.5 \\
 PWMC &= \frac{1}{\sqrt{3}} \cdot Amplitude \cdot \left( \sin(\alpha - 240^\circ) + \frac{1}{6} \cdot \sin 3\alpha \right) + 0.5
 \end{aligned}
 \tag{EQ 6-1}$$

Where PWMA, PWMB and PWMC are calculated, dutycycles passed to PWM driver and amplitude determine the level of phase voltage amplitude.

The process that is performed in PWM reload callback function: *pwm\_Reload\_A\_ISR* is accessed regularly at the rate given by the set PWM reload frequency. This process is repeated often enough to compare it to the wave frequency. Wave length comparisons are made to generate the correct wave shape. Therefore, for 16kHz PWM frequency, it is called each 4th PWM pulse, thus the PWM registers are updated in 4kHz rate (each 250μsec).

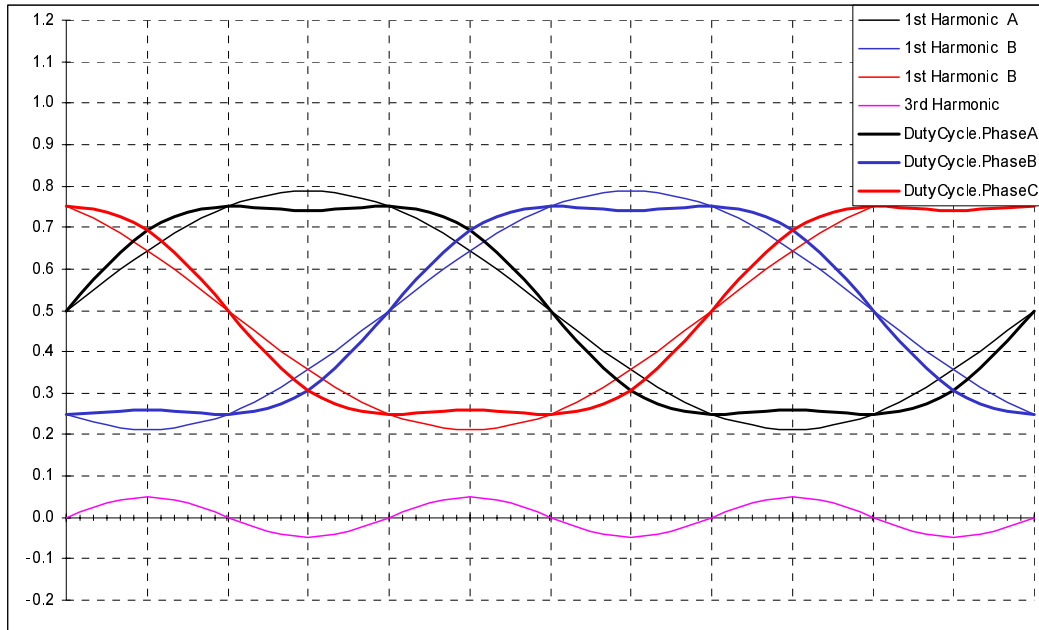
**Figure 6-5** shows the dutycycles generated by the *mcbgen3PhWaveSine3rdHIntp* function when *Amplitude* is 1 (100%).



**Figure 6-5. 3-ph Sine Waves with 3rd Harmonic Injection, *Amplitude* = 100%**

**Figure 6-6** defines the dutycycles generated by the *mcbgen3PhWaveSine3rdHIntp* function when *Amplitude* is 0.5 (50%).





**Figure 6-6. 3-ph Sine Waves with 3rd Harmonic Injection, Amplitude = 50%**

**Input process:**

- *Amplitude* - obtained from DC-bus ripple elimination process
- *Omega\_required* - obtained from acceleration/deceleration ramp process

**Output process:**

Results calculated by *mcgen3PhWaveSine3rdHIntp* function are directly passed to the PWM value registers using PWM driver.

### 6.1.7 Fault Control

This process is responsible for fault handling. The software accommodates five fault inputs: overcurrent, overvoltage, undervoltage, overheating and wrong identified hardware.

**Overcurrent:** In case of overcurrent in DC-Bus link, the external hardware provides a rising edge on the fault input pin FAULTA1 of the DSP. This signal immediately disables all motor control PWM's outputs (PWM1 - PWM6) and sets *DC\_Bus\_OverCurrent* bit of *DriveFaultStatus* variable.

**Overvoltage:** In case of overvoltage in DC-bus link, the external hardware provides a rising edge on the fault input pin FAULTA0 of the DSP. This signal immediately disables all motor control PWM's outputs (PWM1 - PWM6) and sets *DC\_Bus\_OverVoltage* bit of *DriveFaultStatus* variable.

**Undervoltage:** The DC-bus voltage sensed by ADC is compared with the limit within the software. In case of undervoltage after a period defined by *UNDERVOLTAGE\_COUNT* all motor control PWM outputs are disabled and *DriveFaultStatus* variable is set to *DC\_Bus\_UnderVoltage*.

**Overheating:** The temperature of power module sensed by ADC is compared with the limit within the software. In case of overheating after a period defined by *OVERHEATING\_COUNT* all motor control PWM outputs are disabled and *DriveFaultStatus* variable is set to *OverHeating*.

**Wrong Hardware:** In case wrong hardware is identified (different power module or missing optoisolation board) during initialization, *DriveFaultStatus* variable is set to *Wrong\_Hardware*.

If any of the above mentioned faults occurs, program run into infinite loop and waits for reset. Fault is signalled by user LEDs on controller board and on PC Master control screen.

## 6.2 State Diagram

The general state diagram incorporates the main routine entered from reset, and interrupt states. The main routine includes the initialization of the DSP and the main loop. The main loop incorporates initialization state, application state machine and check run/stop switch state.

The interrupt states provides calculation of the actual speed of motor, PWM reload interrupt, ADC service, Limit analog values handling, over current and over voltage PWM fault handler, and so on.

### 6.2.1 Initialization

The main routine provides initialization of the DSP:

- Initializes the PLL clock
- COP and LVI are disabled
- Identifies connected hardware
- Initializes analog-to-digital converter
- Initializes POSIX timers for speed ramp and LED handler
- Initializes PWM module:
  - Center aligned complementary PWM mode, positive polarity
  - Sets callback for PWM reload to (every 4th. PWM pulse)
  - Sets callback for PWM faults
  - Sets of PWM modulus - (defines the PWM frequency)
  - enables fault interrupts
- Sets-up I/O ports (push buttons, switch, brake)
- Initializes quadrature decoder for speed measurement
- Initializes algorithms (V/Hz look-up table, sinewave generator)
- Enables interrupts

The board identification routine identifies the connected power stage board by decoding the identified message send from the power stage. If the wrong power stage is identified, the program goes to the infinite loop, displaying the fault status on the LED. The state can be left only by the RESET.

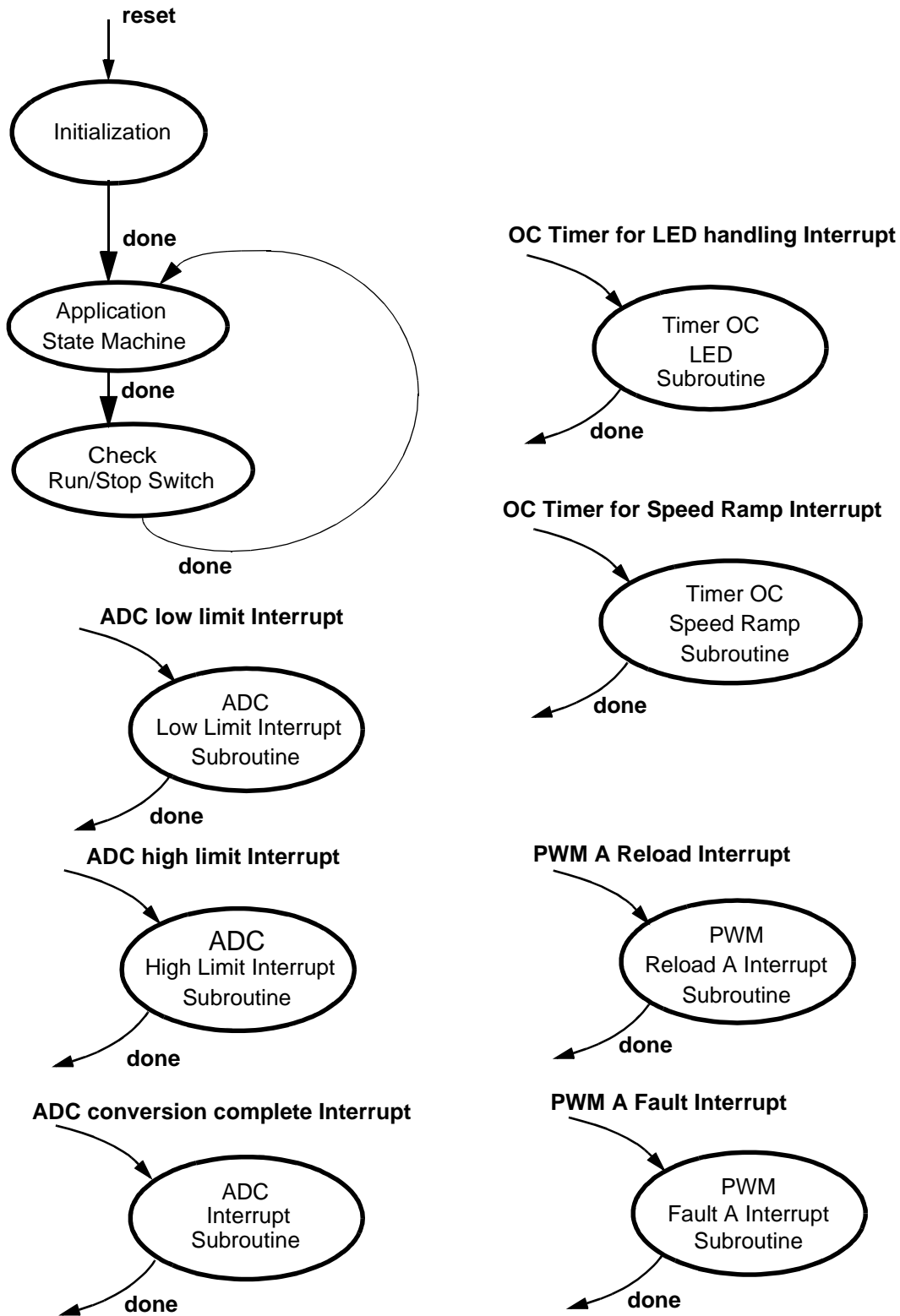
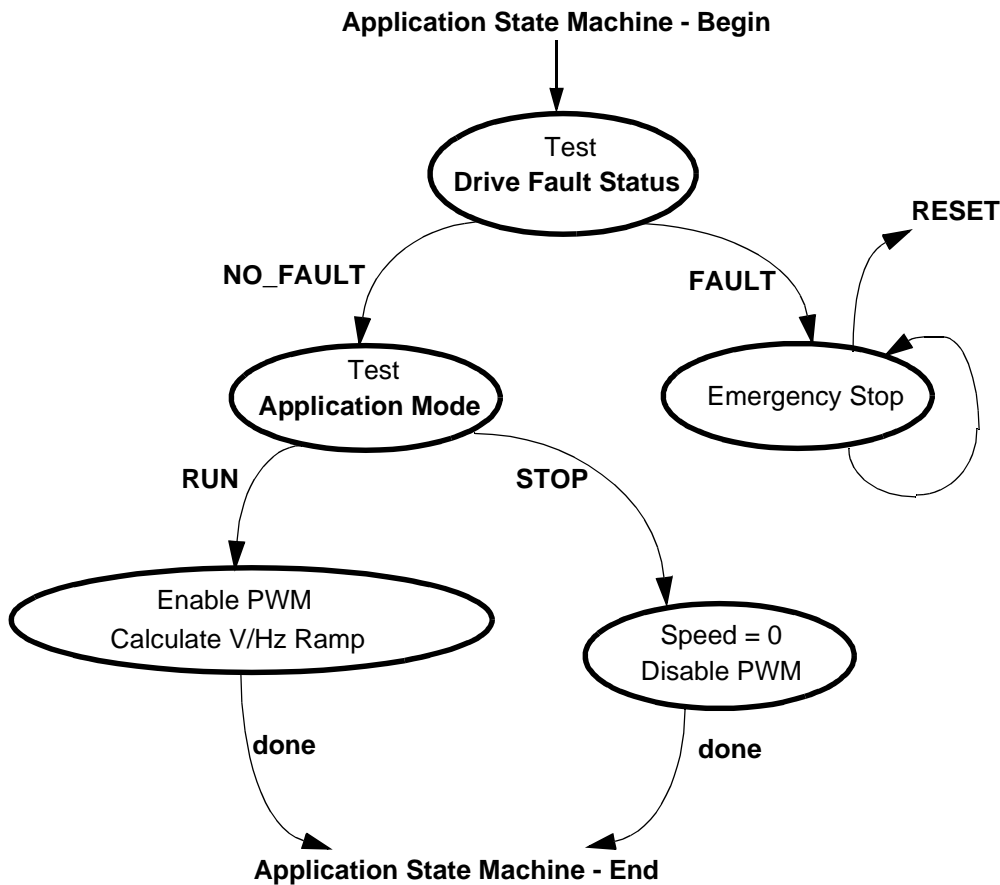


Figure 6-7. State Diagram - General Overview

## 6.2.2 Application State Machine

This state controls the main application functionalities, depicted in [Figure 6-8](#).



**Figure 6-8. State - Application State Machine**

## 6.2.3 Check Run/Stop Switch

In this state, the Run/Stop switch is checked according to the Application Mode setting; whether set to RUN or STOP.

## 6.2.4 PWM Reload A ISR

This subroutine is called at PWM A (or PWM in case of DSP56F803) reload interrupt. It provides:

- Measurement of actual speed (*MeasuredSpeed*)
- Elimination of DC-bus voltage ripples (*mcgenDCBVoltRippleElim* function)
- Calculation of waveform generator (*mcgen3PhWaveSine3rdHIntp* function)
- Updating of PWM value registers
- Start of ADC conversion

The name of callback function in code: *pwm\_Reload\_A\_ISR(void)*.

## 6.2.5 PWM Fault A ISR

This disable PWM module and sets *DriveFaultStatus*  $\neq$  *DC\_Bus\_OverVoltage* or *DC\_Bus\_OverCurrent* according to fault input pin level in case of over voltage or over current in DC-Bus line.

Name of callback function in code: *pwm\_Fault\_A\_ISR(void)*.

This subroutine is called at PWM A (or PWM in case of DSP56F803) Fault Interrupt.

## 6.2.6 ADC Conversion Complete ISR

The following analog inputs are read:

- DC-Bus Voltage
- DC-Bus Current
- Temperature of Power Stage Module

Also the detection of faults caused by Over heating and Under voltage is performed in this subroutine.

Name of callback function in code: *ADC\_Callback\_ISR*.

This subroutine is called at ADC conversion completion.

## 6.2.7 ADC High Limit ISR

This subroutine turns on the resistive brake in DC-Bus link when actual voltage in DC-Bus *u\_dc\_bus* is higher than *BRAKE\_HIGH\_LIMIT*.

Name of callback function in code: *ADC\_High\_Limit\_CallBack\_ISR*.

## 6.2.8 ADC Low Limit ISR

This subroutine turns off the resistive brake in DC-Bus link when actual voltage in DC-Bus *u\_dc\_bus* is lower than *BRAKE\_LOW\_LIMIT*.

Name of callback function in code: *ADC\_Low\_Limit\_CallBack\_ISR*.

## 6.2.9 Timer OC LED ISR

This subroutine takes care of LED handling.

Name of callback function in code: *LedISR(void)*.

Access frequency is defined by constant *TMR\_1\_PERIOD* in definition section of program.

## 6.2.10 Timer OC Ramp ISR

This subroutine takes care of Speed ramp calculation.

Name of callback function in code: *RampISR(void)*.

Access frequency is defined by constant *TMR\_2\_PERIOD* in definition section of program.

## 7. SDK Implementation

The Motorola Embedded SDK is a collection of APIs, libraries, services, rules and guidelines. This software infrastructure is designed to let DSP5680x software developers create high-level, efficient, portable code. This chapter describes how the AC V/Hz motor control application is written under SDK.

### 7.1 Drivers and Library Function

The AC V/Hz motor control application uses the following drivers:

- ADC driver
- Timer driver
- Quadrature timer driver
- Quadrature decoder driver
- PWM driver
- LED driver
- Switch driver
- Button driver
- Brake driver

All drivers except the timer driver are included in *bsp.lib* library. The timer driver is included in *sys.lib* library.

The AC V/Hz motor control application uses the following library functions:

- *boardId* (board identification, *bsp.lib* library)
- *mcgen3PhWaveSine3rdHIntp* (waveform generation function, *mcfunc.lib* library)
- *mcgenDCBVoltRippleElim* (DC-bus ripple cancellation function *mcfunc.lib* library)
- *lutGetValue* (look-up table - performing V/Hz ramp *mcfunc.lib* library)
- *rampGetValue* (speed ramp *mcfunc.lib*)

### 7.2 Appconfig.h File

The purpose of the *appconfig.h* file is to provide a mechanism for overwriting default configuration settings which are defined in the *config.h* file.

There are two *appconfig.h* files. The first *appconfig.h* file is dedicated for External RAM (*..\ConfigExtRam* directory) and second one is dedicated for FLASH memory (*..\ConfigFlash* directory). In case of AC V/Hz motor control application both files are identical.

The *appconfig.h* file is divided into two sections. The first section defines which components of SDK libraries are included to application, the second part overwrites standard setting of components during their initialization.

### 7.3 Drivers Initialization

Each peripheral on the DSP chip or on the EVM board is accessible through a driver. The driver initialization of all used peripheral is described in this chapter. For detailed description of drivers see document *Embedded SDK Targeting Motorola DSP5680x Platform*.

To use the driver, the following steps must be completed.

- Include the driver support to the *appconfig.h*
- Fill the configuration structure in the application code for specific drivers (depends on driver type)
- Initialize the configuration setting in *appconfig.h* for specific drivers (depends on driver type)
- Call the OPEN (create) function

The access to individual driver functions is provided by the *ioctl* function calls.

## 7.4 Interrupts

The SDK serves the interrupt routine calls and automatically clears the interrupt flags. The user defines the callback functions called during interrupts. The callback functions are assigned during the drivers initialization - `open()`. The callback function assignment is defined as one item of initialization structure and is used as a parameter of function - `open()`. Some drivers define the callback function in *appconfig.h* file.

## 7.5 PC Master

PC Master was designed to provide the debugging, diagnostic and demonstration tool for development of algorithms and applications. It consists of component running on a PC and a part running on the target development board.

The PC Master application is part of the Motorola Embedded SDK and may be selectively installed during SDK installation.

To enable the PC Master operation on the target board application, the following lines must be added to the *appconfig.h* file:

```
#define INCLUDE_SCI          /* SCI support */
#define INCLUDE_PCMASTER    /* PC Master support */
```

It automatically includes the SCI driver and installs all necessary services.

The baud rate of the SCI communication is 9600Bd. It is set automatically by PC Master driver.

The detailed PC Master description is provided by the *PC Master User Manual*.

# 8. DSP Usage

**Table 8-1** displays the required memory to run the 3-phase AC V/Hz close loop application. A part of the DSP memory is still available for other tasks.

**Table 8-1. RAM and FLASH Memory Usage for SDK2.3 and CW 4.0**

Memory (in 16-Bit Words)	Available in DSP56F803 DSP56F805	Used Application + Stack	Used Application Without PC Master, SCI
Program FLASH	31.5K	14638	7421
Data RAM	2K	855 + 352 stack	523 + 352 stack
Data FLASH	4K	436	436

## 9. References

DSP56F800 16-bit Digital Signal Processor, *Family Manual, DSP56F800FM/D*, Rev. 1, 01/2000, Motorola.

DSP56F80x 16-bit Digital Signal Processor, *User's Manual, DSP56F801-7UM/D*, Rev. 0, 04/2000, Motorola.


*Green Electronics/Green Bottom Line*. Lee H. Goldberg. (Chapter 2 "Energy Efficient 3-phase AC Motor Drives for Appliance and Industrial Applications." by Radim Visinka).

"Low Cost 3-phase AC Motor Control System Based On MC68HC908MR24." Motorola Semiconductor Application Note, AN1664, 1998.

Motorola Software Development Kit

web page: <http://e-www.motorola.com/motor/>.

OnCE™ is a registered trademark of Motorola, Inc.

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part. Motorola and  are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

#### How to reach us:

**USA/EUROPE/Locations Not Listed:** Motorola Literature Distribution: P.O. Box 5405, Denver, Colorado 80217.  
1-303-675-2140 or 1-800-441-2447

**JAPAN:** Motorola Japan Ltd.; SPS, Technical Information Center, 3-20-1 Minami-Azabu. Minato-ku, Tokyo 106-8573 Japan.  
81-3-3440-3569

**ASIA/PACIFIC:** Motorola Semiconductors H.K. Ltd.; Silicon Harbour Centre, 2 Dai King Street, Tai Po Industrial Estate, Tao Po, N.T., Hong Kong. 852-26668334

**Technical Information Center: 1-800-521-6274**

**HOME PAGE:** <http://motorola.com/semiconductors/dsp>

**MOTOROLA HOME PAGE:** <http://motorola.com/semiconductors/>



**MOTOROLA**

**For More Information On This Product,  
Go to: [www.freescale.com](http://www.freescale.com)**

AN1910/D