Semiconductor Products Sector
Application Note

AN2154

Low-Cost, 3-Phase, AC Motor Control System with Power Factor Correction Based on MC68HC908MR32

By: Petr Stekl and Zdenek Kubiczek
Motorola Czech System Application Laboratory
Roznov pod Radhostem, Czech Republic

Introduction

This application note describes the design of a 3-phase, ac induction motor drive with digital power factor correction (PFC) on a single low-cost microcontroller, the MC68HC908MR32 (MR32).

Most usual variable speed ac induction motor drive designs contain a full-bridge rectifier and a large dc-bus capacitor at the input. Such a circuit draws a peak current from the wall socket, which provokes a high content of harmonics. The low-power factor (PF), as described in the present circuit, reduces the necessary power from the mains and increases the efficiency of the mains supply network. The international standard IEC1000-3-2 defines the limits of the harmonic content of the input current for mains supplied equipment. To meet the norms, the design requires a power factor correction at the input.

The present design allows simultaneous driving of the power factor correction algorithm and the ac induction motor. This way, significant cost reductions of the design can be achieved. Moreover, a wide range of input voltages from 90 volts to 265 volts ac and regulation of dc-bus voltage can be achieved.

Another benefit of the design is the possibility of easy modification of the system by software updates.

System Concept

The design is based on the conjunction of the ac induction motor control system with the digital power factor correction application.

Detailed ideas of the ac induction control design are described in the application note, *Low-Cost, 3-Phase, AC Motor Control System Based on MC68HC*, Motorola order number AN1664/D.

This application note presents only basic facts concerning particular designs and it is focused primarily on the conjunction of both algorithms.

The system is driven by the Motorola 8-bit microcontroller (MCU) MC68HC908MR32. This MCU simultaneously performs PFC control and motor control. The block diagram of the whole motor control system can be seen in **Figure 1**.

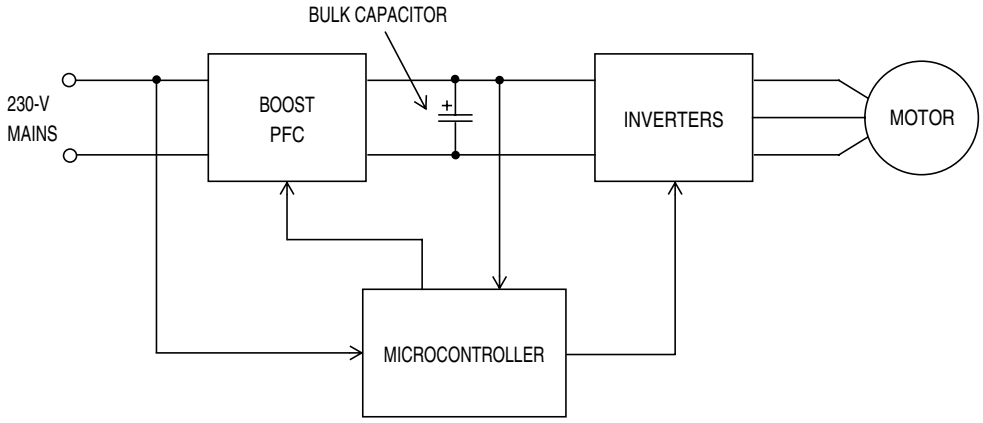

Figure 1. Block Diagram of the Motor Control System with PFC

The MCU drives the boost PFC and, through inverters, it also drives a motor. The microcontroller has built-in interfacing peripherals. Through input interfaces (like analog-to-digital converters or port pins), the MCU collects signals from sensors and converts these analog values into digital forms that are further processed by a control program. As a result of the program, the control parameters are converted through output interfaces (like PWM channels, digital-to-analog converters, or port pins) into control signals, which drive the PFC and the ac induction motor.

The present design can help a user start the development of his own system according to his particular requirements. It can save much engineering time and speed up time to market.

System Hardware

System Specifications

The system hardware consists of the following equipment:

- MR32 control board
- Optoisolation board
- 3-phase, high-voltage ac BLDC (brushless dc) power stage board 180 W
- ac induction motor with speed sensor
- 12- to 15-V dc power supply
- 110-V/60-Hz ac or 230-V/50-Hz ac power supply/wall socket

The printed circuit boards (PCB) and the user's manuals are available from Motorola.

Table 1. System Specifications

Nominal input voltage	90–265 V rms
Input voltage frequency	50/60 Hz
Maximum input current	2.3 A rms
Maximum electrical output power	550 W
Maximum dc-bus voltage	400 V
Output phase voltage	0–230 V rms
Output voltage frequency	0–100 Hz

MR32 Control Board

For a more detailed description of the MR32 control board, refer to *Motorola Embedded Motion Control MC68HC908MR32 Control Board User's Manual,* Motorola document order number MEMCMR32CBUM/D, that comes with the board (kit number ECCTR908MR32).

The control board is designed to be used as an aid for hardware and software design for single, 3-phase, permanent magnet, brush, or brushless dc motor drive applications. It does not contain the MC68HC908MR32 microcontroller. The control board is designed to be directly connected to an MR32 EVM board, which is a part of an MMDS/MMEVS emulation system, connected via an impedance matched ribbon cable. A daughter board is designed to house the MR32 microcontroller and will plug into the control board in place of the emulator cable. With the daughter board plugged into the control board, standalone operation of the system is possible.

The features of the control board include:

- Six motor control PWM (pulse-width modulation) outputs with LED (light-emitting diode) indicators
- Speed control potentiometer
- Optoisolated half-duplex RS232 interface
- START/STOP and FORWARD/REVERSE switches
- Hall effect inputs (for brushless dc motor control)
- Back EMF (electromotive force) inputs (for brushless dc motor control)
- Configuration jumpers
- 2-position DIP (dual in-line package) switches for user option control
- Emulator/daughter board connectors
- Processor reset switch
- Two system fault inputs
- Nine analog inputs
- Three software controlled LEDs
- On-board regulated power supply
- Motor I/O (input/output) interface via a ribbon cable connector

MR32 Control Board Jumper Settings

The 3-phase ac induction drive control system with PFC requires the following jumper settings of the MR32 control board:

- JP1 closed — Speed sensor (tacho)
- JP4 closed — PFC zero crossing signal
- JP5 closed — PFC PWM signal
- JP2 and JP3 open

The software is independent of the DIP switch settings.

Optoisolation Board

The function of the optoisolation board is to provide a galvanic isolation barrier between the control board's input/outputs (I/O), both analog and digital parts, and the high-voltage system power board's I/Os. These isolated signals, to and from the optoisolation board, are connected via two 40-pin ribbon cables. The pin assignment for both connectors is the same. The signal flow through the optoisolation board, in both directions, is a one-to-one relation of its source. For a more detailed description of the optoisolation board, refer to *Motorola Embedded Motion Control Optoisolation Board User's Manual,* Motorola document order number MEMCOBUM/D.

The power requirement for the control board's circuitry is satisfied with a single external 12- to 15-volt dc power supply. It can be connected to the optoisolation board either via connector JP1, labeled "Ext. Power 12V DC," or power jack J3. Either one, but not both, may be used. This power supply is fed to the control board through the 40-pin ribbon cable connector. The excitation for the power stage side circuitry is supplied to the power stage through the 40-pin output connector located on the optoisolation board.

In addition to usual motor control signals, an MC68HC705JJ7 microcontroller serves as a serial link. It allows the controller board's software to identify the configuration of the optoisolation board and the power stage board. It passes configuration information to the control board for processing and checking the system configuration.

Power Stage

For a more detailed description of the 3-phase, ac, BLDC, high-voltage power stage and for board setup, refer to *Motorola Embedded Motion Control 3-Phase AC BLDC High-Voltage Power Stage User's Manual,* Motorola document order number MEMC3PBLDCPSUM/D.

The power stage provides a high-power drive circuitry for various types of motors. It is suitable for driving ac induction, permanent magnet, brush and brushless dc motors. The power stage consists of a set of two printed circuit boards. One of the PC boards is a power module containing the power IGBTs (isolating gate bipolar transistor), a brake IGBT, a power factor corrector field effect resistor (FET), and temperature sensing diodes. The second PC board contains IGBT drive circuits, analog signal conditioning, low-voltage power supplies, power factor control circuitry, and an MC68HC705JJ7 microcontroller, used for board configuration and identification.

The features of the power stage include:

- 1-phase bridge rectifier
- Power factor switch and diode
- Power factor correction coil
- dc-bus brake IGBT and brake current limiting resistors
- 3-phase bridge inverter (six IGBTs)
- Individual phase and dc-bus current sensing shunt resistors with Kelvin connections
- Power stage temperature sensing diodes
- IGBT gate drivers
- Current and temperature signal conditioning
- 3-phase back-EMF voltage sensing and ZC (zero crossing) detection circuitry
- Board identification processor (MC68HC705JJ7)
- Low-voltage on-board power supplies
- Cooling fans

Power Stage Jumper Settings

To operate properly, the 3-phase ac induction drive control system with PFC requires the JP201 jumper to be in the PFC position.

Software

The system software includes three algorithms:

- 3-phase V/Hz ac induction motor control algorithm
- Digital power factor correction algorithm
- PC-Master communication routines

The system software can be executed in the MMDS05/08 emulator system or in a programmed MC68HC908MR32, resident on a daughter board, plugged into the control board.

The algorithms can be judged as independent and their software design and basic features can be described separately.

Basic features of the V/Hz control software include:

- Controlled acceleration and deceleration
- Speed in the range of 0 to 2400 rpm for 4-pole motor
- The drive can run clockwise or counterclockwise.
- Speed is sensed by a tachometer/generator.
- PWM frequency is 16 kHz.
- Operation in PC-Master or manual operating mode
- Overvoltage and overcurrent protection

Basic features of the PC-Master communication software include:

- Ability to read/write any RAM variable
- Ability to read any ROM variable
- Execution of PC-Master commands

Basic features of the power factor correction software include:

- Automatic input voltage detection 110 V/60 Hz and 230 V/50 Hz
- dc-bus voltage regulation
- Overvoltage and overcurrent protection

Software Design

In this section, the design of particular applications is described first, separately, and then the overall functionality is described.

Initilization

The main routine provides initialization of the microcontroller:

- Clears RAM
- Initializes PLL clock
- Initializes PWM module:
  - Center-aligned complementary PWM mode, positive polarity (MOR register)
  - COP and LVI enable (MOR register)
  - PWM modulus defines the PWM frequency (PMOD register)
  - 2-µs dead time (DEADTM register)
  - PWM interrupt reload every fourth PWM cycle (PCTL2 register)
  - FAULT2 (overcurrent fault) in manual mode, interrupt enabled (FCR register)
- Sets up I/O ports
- Initializes timer A for input capture (IC), output compare (OC), and for software timer reference
- Initializes timer B for PWM generation of the PFC
- Initializes the A/D (analog-to-digital) converter
- Detects connected boards
- Detects input line voltage limits
- Detects input line frequency
- Calibrates the PFC feedback offset `FB_offset`
- If any error occurs, the fault LED is turned on, the failure register is set, and the software waits for reset.
- Enables interrupts

When the MR32 is reset, the software configures the various system I/O, the FORWARD/REVERSE switch, and the START/STOP switch. The dc-bus voltages are checked and the speed potentiometer's value is

input. The yellow LED of the control board is illuminated when the system is ready. Afterward, the identification of the connected boards (for example, optoisolation and power boards) is checked. The default operation mode is set to manual. The PC-Master operation mode can be set by the PC-Master command. The PFC algorithm is initialized. When it is passed, the fault flag (`Failure`) is tested for any system fault. Anytime a fault is detected in the system, the red LED of the control board is turned on. The status of the LEDs is described later in this application note.

Power Factor Correction

The PFC algorithm drives the analog part of the PFC. The MR32 features are assigned to the algorithm as follows:

- Timer A: input capture (TCH0A pin) — ZC detection
- Timer B: channels 0 and 1 (TCH0B pin) — Buffered PWM signal generation
- Analog-to-digital converter, channel 2 (ATD2 pin) — dc-bus voltage sensing
- Input/output port A (PTA0 pin) — PFC inhibit signal

The control algorithm performs these tasks:

- Converts a sensed output voltage into a digital value
- Calculates software regulator for feedback loop
- Programs its PWM channel to create the pattern of the input current
- Synchronizes (using mains zero crossing detector) operation to the mains frequency

The interrupt handlers used for PFC operation are:

- The input capture interrupt handler on channel 0 timer A performs the synchronization for the PFC.
- The output compare interrupt handler generates the waveform of the input current.
- The A/D interrupt handler performs the control of the PFC output voltage.

The flow diagram of the control PFC software is depicted in **Figure 2**.

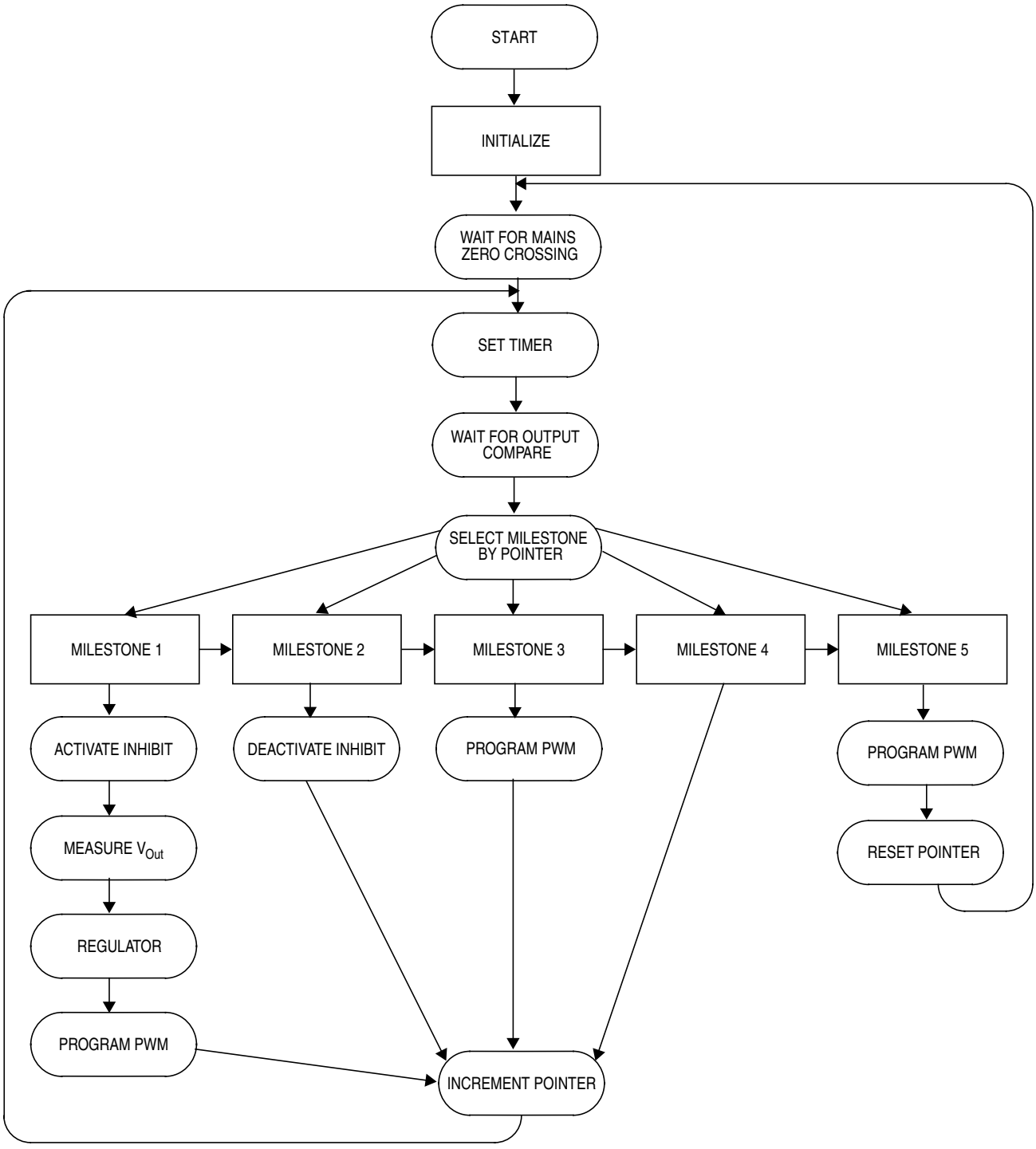

Figure 2. PFC Control Software Flow Diagram

The desired shape of the input current is a non-sine wave with stairs. The number and size of the stairs is optimized to fulfill these requirements:

- Least complex to avoid overloading of the microcontroller
- Harmonic currents content complies with standard IEC1000-3-2.

The waveform of the input current consists of two current levels and five time intervals per half-period time range. The generated current waveform is shown in **Figure 3**.

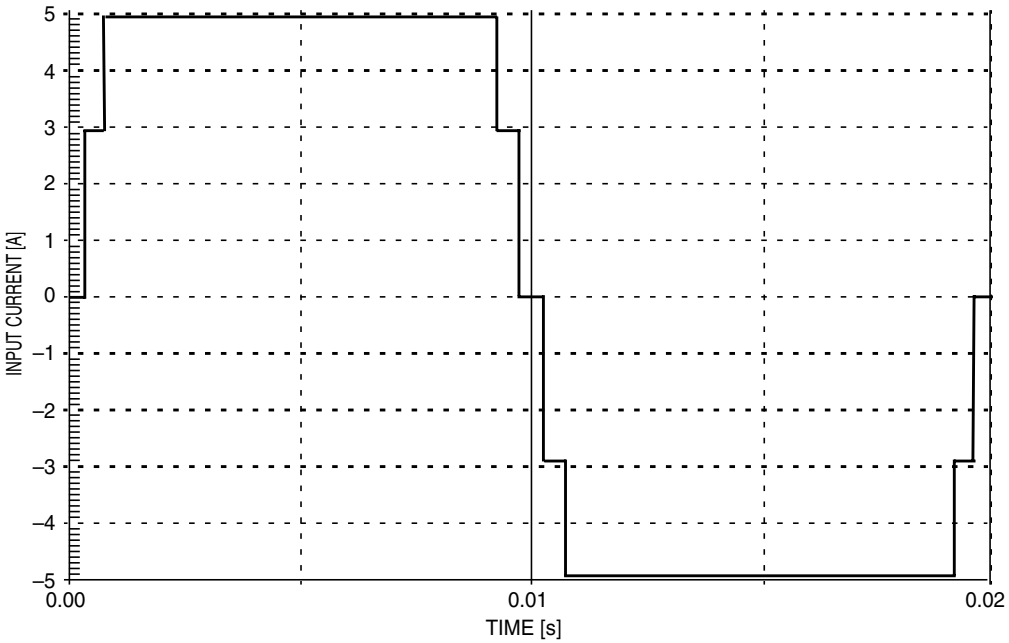

Figure 3. Input Current for 1-kW Output Power

Input Capture Interrupt Handler on TIMA, Channel 0

After initialization, the PFC software waits for the mains zero-crossing (ZC) signal which is used for synchronization of the PFC algorithm with the line voltage. The `PFCzc( )` function is called to service the interrupt. The output compare channel 1 is programmed for the first input current waveform milestone. The programmed value corresponds to 848-µs time interval. Output compare interrupts on channel 1 are enabled. The input capture detection (ZC signal) is disabled for better noise immunity. If the PFC disable flag in `PFC_status` status word is set (for example, overvoltage was detected), the PFC output is disabled for the next half period. The COP is serviced here as well.

Output Compare Interrupt Handler on TIMA, Channel 1

The `PFCwave()` function is called to service the interrupt. The main task of this function is to define the input current waveform milestones (time intervals) and hence form the input current waveform. The variable `Point` is used to determine the next step. During each step, the next output compare on channel 1 is programmed and a proper input current level is set by programming the duty cycle of timer B PWM duty cycle.

When step 2 (`Point=2`) is reached, the ADC conversion on channel 1 is started. It senses the dc-bus voltage and the ADC conversion complete interrupt is enabled. When the last step (`Point=5`) is reached, the output compare on TIMA channel 1 is disabled and the input capture on channel 0 is enabled (ZC synchronization). The position of the milestones on the inductor current can be seen from **Figure 4**.

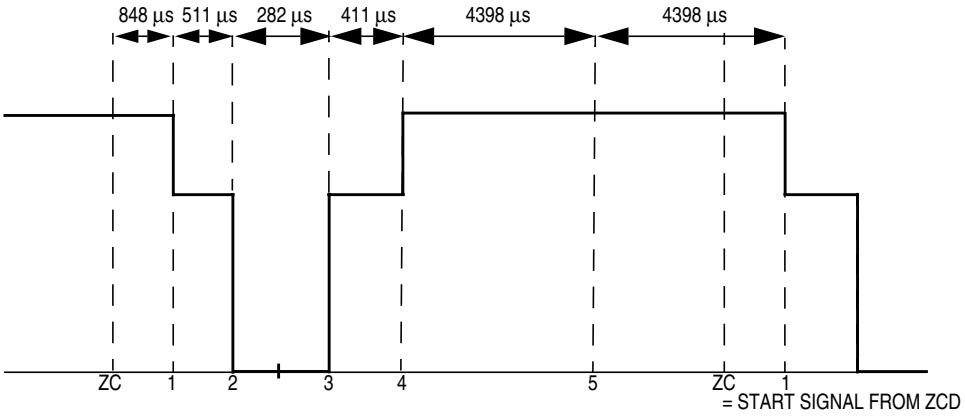

Figure 4. Milestones of Induction Current

ADC Conversion Complete Interrupt Handler, Channel 1

The `PFCcntrl()` function is called to service the interrupt. The sensed dc-bus voltage is stored into variable `Out_volt_new`. ADC interrupts are disabled and ADC conversion on channel 0 is started for the required speed sensing. During initialization, the algorithm jumps to the feedback loop offset calibration routine. When the offset is found, the `CALIB_OFFSET` flag in the `PFC_status` status word is cleared and the `LOAD_ON_FLG` flag in the `Mcs_status` status word is set. Once the `CALIB_OFFSET` flag is cleared, the calibration routine is skipped in the next algorithm execution. The sensed voltage is fed into the regulator. The nominal dc-bus output voltage is set to 375 Vdc. The required output voltage is determined by the `Out_Volt_Max` variable. The output of the

voltage regulator is the amplitude of the input inductor current. The calculated input current is limited to the maximum input value and zeroed if negative. The new value of the variable `Curr_level1` variable is derived, representing the higher level of the input current, and the new value of the variable `Curr_level2` variable is calculated for lower level of the input current.

PC-Master Communication

The PC-Master Communication is a program resident in the MR32 that communicates via the SCI (serial communications interface) with the PC-Master software to parse commands, return status information to the PC, and process control information from the PC.

The MR32 features are assigned to the algorithm as follows.

The PC-Master Communication software is intended to be used as an aid in developing the motor control software. When using the PC-Master software, all of the required actions of the motor control software are manipulated by the operator. The PC-Master software executes on a PC that is connected to the isolated RS232 serial port on the control board. The PC-Master software user interface displays various sets of views (tabs) programmed in HTML code with embedded objects and scripts. They are used to control the board's application, present its block diagram, show graphically the status of variables, and many other features that can be extended by the user. A small program is resident in the MR32 that communicates with the PC-Master software to parse commands, return status information to the PC, and process control information from the PC.

The actions controlled by the PC-Master are:

- Start/Stop control
- Motor speed setpoint
- Reset the drive system
- Motor rotation direction control clockwise/counter clockwise (CW/CCW)

The following are default variables read by the PC-Master software and displayed to the user:

- Required speed
- Actual motor speed
- dc-bus voltage
- Power module temperature
- Display system status and error flags

The PC-Master commands are described in **Table 2**.

Table 2. PC-Master Commands

Command	Command Code	Demo Suitcase Action
Set manual mode	00	Setting of manual mode
Set PC-master mode	01	Setting of PC-master mode

The status and error flags are read by the PC-Master in 1-s periods. The flags are used to determine the status of the operation of the software by the PC-Master application.

Descriptions of the control, status, and failure flags are found in **Table 3**.

Table 3. PC-Master Status and Error Flags

Register	Bit No.	Description
Motor_Ctrl	0	0 — Stop the motor 1 — Start the motor
	1	0 — Forward direction 1 — Reward direction
	2–6	Reserved
	7	0 — Clear the bit = clear the error
Motor_Status	0	0 — START/STOP switch set to stop flag 1 — START/STOP switch set to start flag
	1	0 — Motor is stopped 1 — Motor is running
	2	Reserved
	3	0 — 230-V line detected 1 — 120-V line detected
	4–7	Reserved
Failure	0	1 — Overcurrent failure
	1	1 — Overheating failure
	2	1 — Overvoltage failure
	3—7	Reserved

Drive Operating Modes

The motor drive is able to operate in the following two operating modes (OM). The MANUAL/PC-Master OM option is set by the PC-Master command when the START/STOP switch is in the STOP position. (The motor stops and the green LED is off.) It is confirmed by hardware after the START/STOP switch has been set to the START position.

- MANUAL Operating Mode

  Default mode after reset. In the MANUAL OM, the required action is set by switches mounted on the control board (START/STOP, FORWARD/REVERSE) and a potentiometer (Speed). The status of the drive is displayed by LEDs.

- PC-Master Operating Mode

  Required software actions on motor drive are controlled by the PC-Master commands running on the PC connected to the control board. The motor can be stopped in case of emergency by setting the START/STOP switch to the STOP position. The actions, controlled by the PC-Master control page, are:
  - Start/stop the motor
  - Control the speed
  - Reset the drive
  - Drive direction

  The variables read by the PC-Master control page as default will be:
  - Required speed
  - Actual motor speed
  - dc-bus voltage
  - Power module temperature
  - Read the status and error flags of the software

  Any other variable can be read/written.

LEDs Status

The software uses active-high indicators connected to the MCU pins and configured as outputs according to the following description:

- Motor running LED (green) — PTC6
  - MCS status is Run; otherwise, the LED is off.
- Status LED (yellow) — PTC5
  - MCS status is Stand_By or Fault. LED is turned off.
  - MCS status is Run or Stop. LED is turned on.
- Fault LED (red) — PTC4
  - MCS status is Stand_By, Run, or Stop. LED is turned off.
  - MCS status is Initialization Error. LED is turned on.
  - MCS status is Fault (overvoltage, overcurrent). LED is flashing with a frequency of 0.5 Hz.

Otherwise, the LED 1, 2, and 3 are turned off.

Brief V/Hz Algorithm Description

The motor control software monitors the state of the sensors as they are periodically scanned in the software timer loop. The speed of the motor is calculated utilizing the input capture interrupt. The green LED of the control board will illuminate whenever the motor is running. According to the operational mode, setup, and state of the control signals (START/STOP switch, FORWARD/REVERSE switch, speed potentiometer), the speed command is calculated using an acceleration/deceleration ramp. The comparison between the actual speed command and the tachometer speed generates a speed error. The speed error is passed to the speed PI controller generating a new corrected motor frequency. The corresponding voltage is calculated using a V/Hz ramp. The PWM generation process calculates a system of 3-phase voltages representing the required amplitude and frequency, including dead times. The 3-phase PWM motor control signals are then output to the power stage.

The dc-bus voltage and the dc-bus current are measured during the control process. They are used for overvoltage and overcurrent protection of the drive. The overvoltage protection is performed by the software while the overcurrent fault signal utilizes a fault input of the microcontroller.

If any of the mentioned faults occur, the motor control PWM outputs are disabled for drive protection and the system fault state is displayed. These faults, depending on the operating mode of the system, are output to the LEDs on the controller and/or the PC terminal that is connected to the control board.

The control algorithm of the close loop ac drive is described in **Figure 5**. It consists of processes that are described briefly in the subsections that follow. For details, refer to the application note, *A 3-Phase AC Induction Motor Control System Based on the MC68HC908MR32*, Motorola document order number AN1857/D. It can be found on Motorola's Web pages.

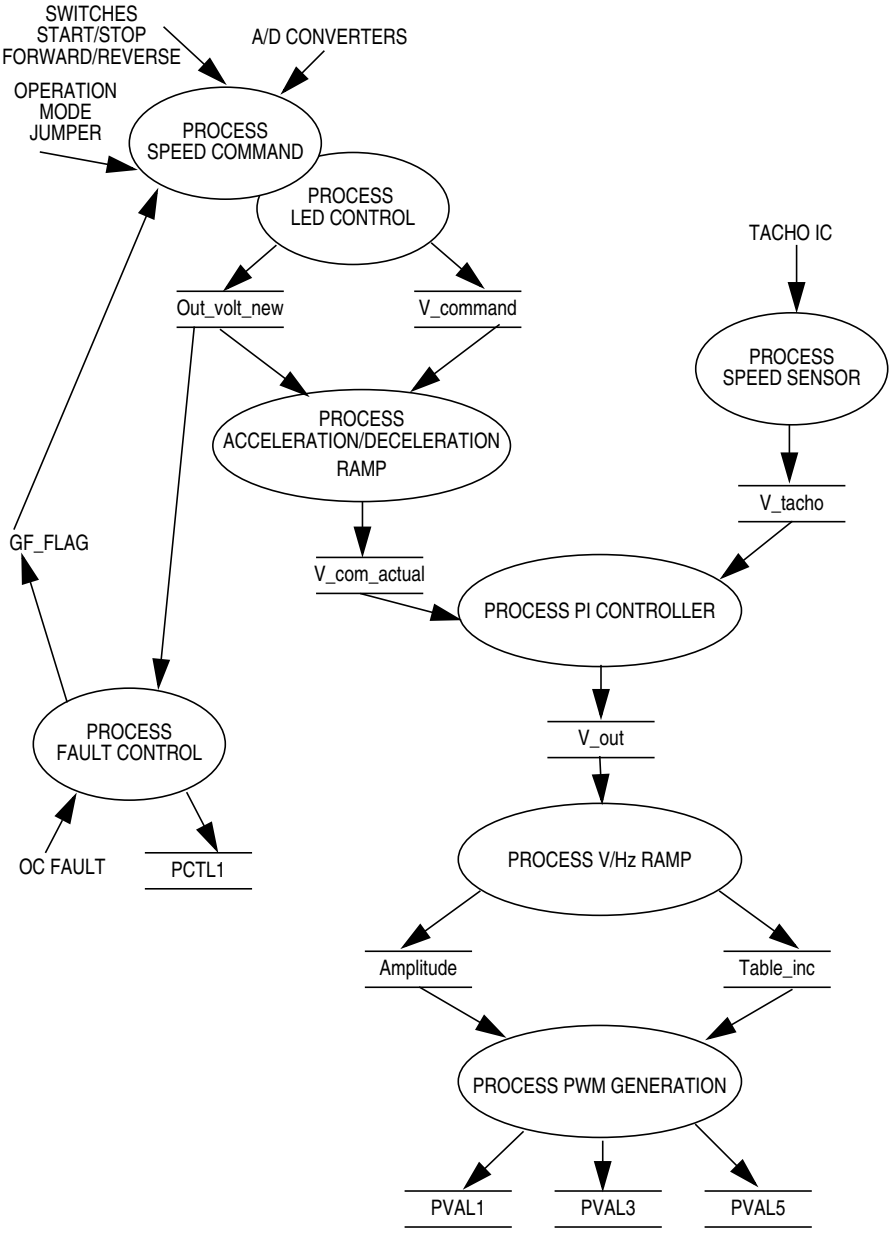

Figure 5. Data Flow Diagram

The measured dc-bus voltage is represented by the variable `Out_volt_new`. The input value from the speed potentiometer is labeled `Pot_voltage`. The input parameters of the process are evaluated and the speed command `V_command` is calculated accordingly. The general fault `GF_FLAG` is analyzed and the state of the drive is set. The drive state diagram is shown in **Figure 6**. The state of the LEDs is controlled according to the system status.

The calculated speed command `V_command` is a 2-byte variable where the first byte is integer and the second byte is remainder (for instance, 1.0 Hz = 0x0100). The upper byte represents the integer portion and the lower byte represents the fractional portion of the value. This format is kept throughout the program for all speed variables.

The system software calculates a new speed based on the requested speed, according to the acceleration / deceleration ramp.

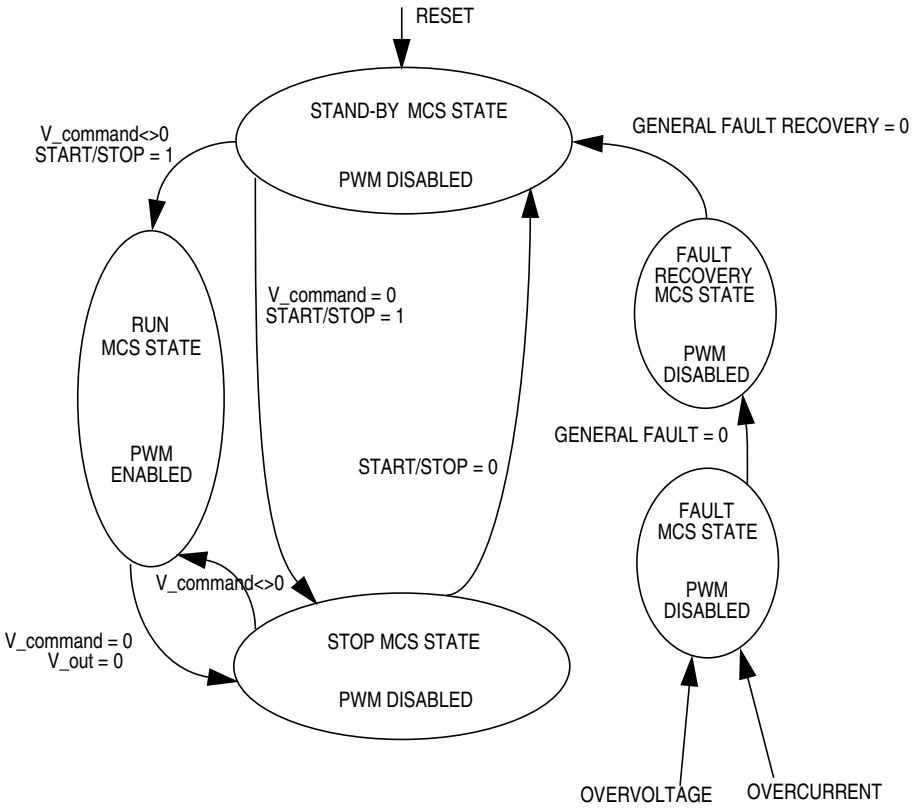

Figure 6. Drive State Diagram

Drive Deceleration

During deceleration the motor can work as a generator. In the generator state, the dc-bus capacitor is charged and its voltage could easily exceed its maximal voltage. Therefore, the dc-bus voltage is measured and compared with a limit. In case of a deceleration overvoltage, the deceleration is interrupted and the motor runs with a constant speed to discharge the capacitor down to an acceptable value. Deceleration can then continue. During deceleration, depending on the input line voltage, the dc-bus voltage, controlled by the PFC, is maintained below 340 volts.

Speed Control Loop

The speed sensing process utilizes the MR32's input capture interrupt function. The input capture interrupt reads the time between the rising edges of the speed sensor's output and calculates the actual motor speed, `V_tacho`. A software filter for the speed measurement can be incorporated in the process for better noise immunity. In this case, the actual motor speed is calculated as an average value of several measurements.

The general principle behind a PI control loop is shown in **Figure 7**. The speed closed-loop control is characterized by the measurement of the actual motor speed. This information is compared with the reference setting point and an error signal is generated. The magnitude and polarity of the error signal corresponds to the difference between the actual and the required speed. Based on the speed error, the PI controller generates the corrected motor frequency for the error compensation and reaches the required motor speed.

This PI process takes these two input parameters: actual speed command `V_com_actual` and actual motor speed measured by a tachogenerator `V_tacho`. Then it calculates the speed error and performs the speed PI control algorithm.

The output of the PI controller is a new frequency of the fundamental sine wave that is to be generated by the inverter, `V_out`.

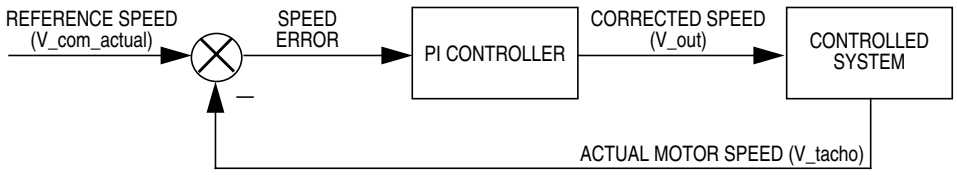

Figure 7. Closed Loop Control

Volt-per-Hertz Ramp Implementation

The drive is designed as a constant Volt-per-Hertz drive. This means that the control algorithm keeps the magnetizing current (flux) of the motor constant by varying the stator voltage with frequency. The ratio of voltage divided by the frequency is constant during the linear portion of the profile. The commonly used volt-per-Hertz ramp profile of a 3-phase ac induction motor is illustrated in **Figure 8**.

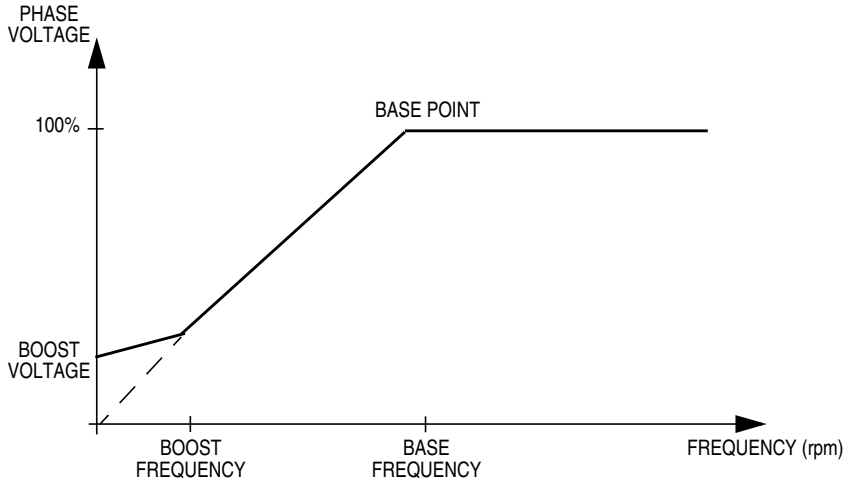

Figure 8. Volt-per-Hertz Ramp

The volt-per-Hertz ramp is defined with these parameters:

- Base point — Defined by the base frequency (usually 50 Hz or 60 Hz)
- Boost — Defined by the boost voltage and the boost frequency

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

The software function RAMP.C provides the voltage calculation according to a V/Hz ramp. The input of this software function is the generated inverter frequency, `V_out`. The output of the function is the amplitude of the generated voltage, `Amplitude`.

PWM Generation

Parameters, required by the PWM generation process, are the output of this software function:

- Table increment, `Table_inc`, that corresponds to the frequency `V_out`, is used to roll through the wave table for generation of the output inverter frequency.
- `Amplitude` of the generated inverter voltage

The process of sine wave generation provides 3-phase sine waves, each shifted by 120 degrees relative to each other. The sine waves can be pure sine waves or they can have a third harmonic component.

The calculation is based on the wave table stored in the MCU's ROM. The table describes either a pure sine wave or a sine wave with an additional third harmonic. The second case is often preferred because it allows the generation of the first harmonic sine voltage equal to the input ac line voltage. Because of quarter wave symmetry, only the first quadrant of the wave period is stored in the table. The wave values for other quadrants are calculated from the first one. The data format of the stored wave table is from `0x00` (for `0` voltage) up to PWM modulus/2 (for 100 percent voltage). Thus, the proper data scaling is secured.

For a detailed process description of PWM, refer to the application note, AN1857, *A 3-Phase, AC, Induction Motor Control System Based on the MC68HC908MR32*, that can be found on the Motorola Web pages.

The input parameters of the process are:

- The table increment `Table_inc` updates the wave pointer.
- `Amplitude` of the generated inverter voltage

The output parameters of the process are:

- PWM value for phase A: PVAL1 register
- PWM value for phase B: PVAL3 register
- PWM value for phase C: PVAL5 register

The process of setting these registers' values is accessed regularly in the rate given by the set PWM frequency and the selected PWM interrupt prescaler (register named PCTL2). This process has to be repeated often enough, compared to the wave frequency, to generate the correct wave shape. Therefore, for a 16-kHz PWM frequency, it is called every fourth PWM pulse. Thus the PWM registers are updated at a 4-kHz rate (for example, every 250 μs).

General Application Overview

The processes described previously are implemented in a single state machine and are illustrated in **Figure 9**, **Figure 10**, and **Figure 11**.

The general state diagram incorporates the main routine, entered from reset, and eight interrupt states. The main routine includes the initialization of the microcontroller and a software timer for the control algorithm timebase. The interrupt states provide calculation of actual speed of the motor, overcurrent fault handler, PWM generation process, zero crossing interrupt for the PFC, output compare to generate the input current waveform, A/D interrupt to control PFC output voltage, SCI read, and transmit routines.

A software timer routine provides the timing sequence for required subroutines. The software timer is performed instead of an output compare interrupt handler. The main program has several time-demanding interrupt routines, and more interrupt requirements can cause a software fault.

The software timer routine has two timed outputs:

- `READ_CONST` is a routine that scans inputs, calculates the speed command, handles fault routines, and the LED driver.
- `PI_CONST` is a routine that provides overvoltage protection during deceleration, speed ramp (acceleration/deceleration), PI controller, and V/Hz ramp, and provides parameters for PWM generation.

The interrupt handlers have following functions:

- The input capture (IC) interrupt handler reads the time between two subsequent IC edges (basic part of the process speed sensor).
- The fault interrupt handler takes care of overcurrent fault interrupt (overcurrent part of the process fault control).
- The PWM interrupt handler generates a system of 3-phase voltages for the motor (process PWM generation).
- The input capture interrupt handler on channel 0 of timer A makes the synchronization for the PFC.
- The output compare (OC) interrupt handler generates the waveform of the input current.
- The A/D interrupt handler performs the control of the PFC output voltage.
- The SCI read interrupt handler services receive interrupts for the PC-Master communication routines.
- The SCI transmit interrupt handler services transmit interrupts for the PC-Master communication routines.

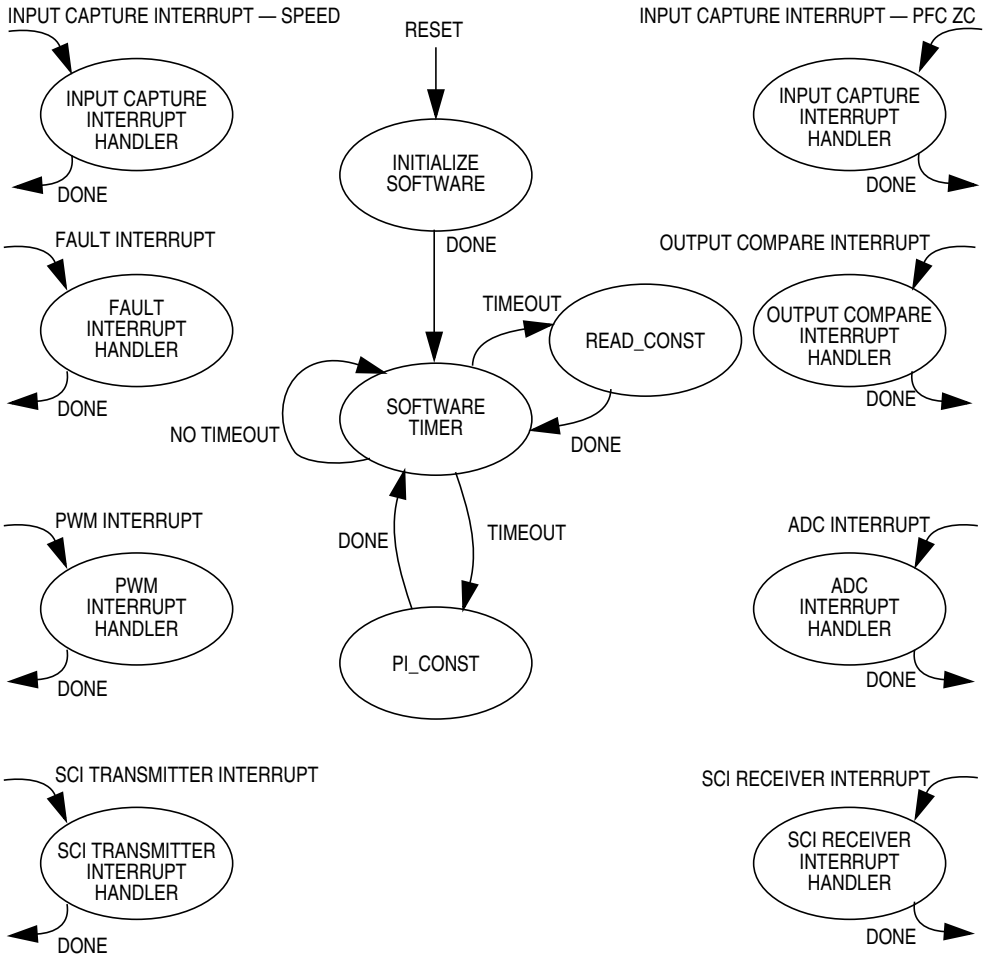

Figure 9. State Diagram — General Overview

READ_CONST is accessed from the main software timer in READ_CONST rates. The following sequence is performed. See **Figure 10**.

- All inputs are scanned (speed pot, START/STOP switch, FORWARD/REVERSE switch, temperature, dc-bus current). The dc-bus voltage is measured in the A/D interrupt routine.
- The speed command is calculated according to the operational mode.
- The dc-bus voltage is compared with the overvoltage limit and the overcurrent flag is checked
- In case of a fault condition, the fault recovery routine is entered and until the recovery time expires, the drive remains disabled.

- Finally, the LED driver controls individual LEDs according to the status of the drive.
- The PFC hardware is enabled/disabled according to the drive status.
- PC-Master commands are serviced.

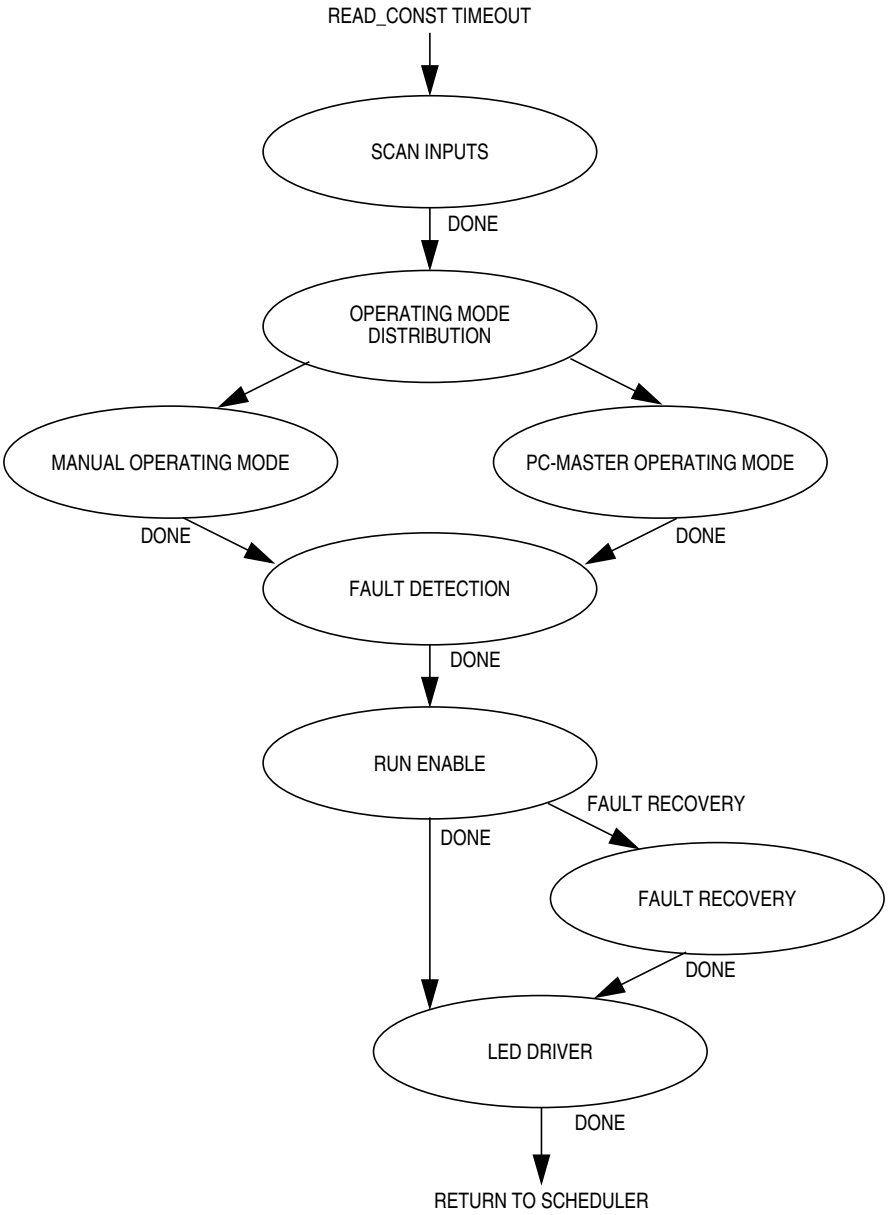

Figure 10. State Diagram — READ_CONST Timeout

PI_CONST is accessed from the main software timer in PI_CONST rates. The rates define the time constant of the PI controller.

The following sequence is performed (see **Figure 11**):

- During deceleration, the dc-bus voltage is checked and, in case of overvoltage, the deceleration is interrupted until the capacitor is discharged.
- When no deceleration overvoltage is measured, the acceleration/deceleration speed profile is calculated.
- The actual motor speed is calculated. It is based on the time measurement between two subsequent rising edges of the input capture.
- The PI speed controller is performed and the corrected motor frequency is calculated.

The corresponding voltage amplitude is calculated according to the Volt-per-Hertz profile. Thus, both parameters for PWM generation are available (Table_inc, Amplitude).

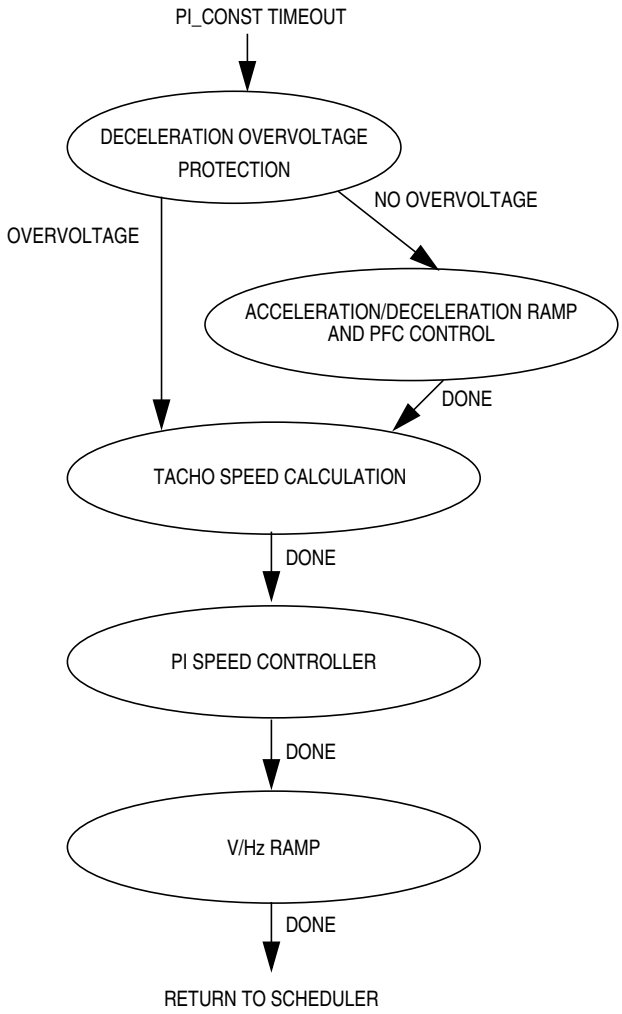

Figure 11. State Diagram — `PI_CONST` Timeout

Fault Handling

It is important to note that, in the event of a system fault, the software services the event in a timely manner. It accommodates two fault inputs, overcurrent and overvoltage.

- **Overcurrent:** The external hardware provides a rising edge on the fault input of the microcontroller's FAULT2 input through the fault generation circuit. This signal disables all motor control PWM's outputs (PWM1–PWM6) and sets the general fault flag, `GF_FLAG`**.** Also the `OVER_CURRENT` flag is set in the `Failure` register, used by the PC-Master control interface.

- **Overvoltage:** The sensed dc-bus voltage is compared with a limit within the software. In case of overvoltage, all motor control PWM outputs are disabled by the software, which sets bits in the MCU's PWM control register (PCTL1) and sets a bit in the general fault flag, `GF_FLAG`. Also, the `OVER_VOLTAGE` flag is set in the `Failure` register, used by the PC-Master control interface. The overvoltage fault is set only if the motor is braking (in generator mode). In motor mode (the motor supplies power to the load), if an overvoltage occurs, the PFC is disabled; the overvoltage failure is not detected and the motor is not blocked. It is important to note that the PFC output voltage operates very close to 400 volts, which is the overvoltage limit. The regulation overshoot would cause an overvoltage failure when PFC is running. Therefore, the overvoltage is blocked when the PFC is running.

If any of these faults occur, the fault LED will flash. The system remains disabled until the fault is cleared by switching the START/STOP switch to the STOP and then to the START position or cleared by the PC-Master by setting the bit `ERROR_CLEAR_PMFLG` in control register `Motor_Ctrl`. As soon as the START/STOP switch is set to START, the motor restarts.

Software Files

The software consists of the following parts:

Code_ISR.C — Contains the interrupt and reset vector addresses for the system's MR32 software

DigitPFC.C — Contains the software used to drive the power factor correction hardware resident on the power board. The software sets the duty cycle of timer B channel 0 that pulse-width modulates the power board's PFC input hardware at 125 kHz.

FAULT.C — Contains the fault interrupt service routine

MAIN.C — This is the entry point following a reset and after initial startup code. It contains the initialization software state code, the main state machine with the software timer state code.

MR_IDENT.C — Contains code that communicates with the MC68HC705JJ7 microcontrollers, resident on the optoisolation and power boards. The resulting information from this routine is used for configuration checking and input to system run time parameters.

PCMASTER.C — Contains the PC-Master SCI communication routines

PI.C — Contains the `PI_CONST` timeout code (deceleration overvoltage protection state, tacho speed calculation state, PI speed controller state, and calls acceleration/deceleration ramp state and V/Hz ramp state appropriately)

PWMCALC.C — Contains code to service the PWM interrupts which ultimately generate the sine outputs to the power stage

RAM.C — Contains the global RAM variable definitions for MR32 system software

RAMP.C — Contains code for acceleration and deceleration ramp state, V/Hz ramp state

SPEED.C — Contains the `READ_CONST` code (scan inputs state; operational mode distribution state; speed calculation, manual operational mode state, speed calculation; PC-Master operation mode state; fault detection state; run enable state; fault recovery state; LED driver state; PFC enable/disable logic)

TACHO.C — Contains timer A channel 0 interrupt code that calculates the time between tachometer interrupts

3RDHQUAD.H — Header file that contains a 256-word 1-quadrant sine table with third harmonic injection

CODE_FUN.H — Header file containing the prototypes of the external functions for V/Hz control

CONST.H — Header file that contains the global system constants definitions for system's MR32 software

MR_IDENT.H — Contains PCB board type identification constants

MR24IO.H — Provides the MR24/MR32 registers map

PCMASTER.H — Contains the PC-Master SCI communication constant and variables declarations

RAM.H — Header file that contains global RAM variable declarations for MR32 system software

MR24_VHz_PFC.PRM — Parameter file containing the interrupt and reset vector addresses for the system's MR32 software

References

For information regarding the PC-Master software, application note software, and microcontroller documentation, refer to www.motorola.com/mcu, Motorola's Web site for microcontrollers.

Conclusion

In this application note, the design of a 3-phase, ac, speed-controlled, induction motor software is described with a special attention to power factor correction and optional PC-Master communication. Any hardware part of this system can be purchased from Motorola. The toolset enables the MR32 to be evaluated in a system without the necessity of building prototype hardware.

The design, described in this application note, illustrates the efficiency and simplicity of using the MC68HC908MR32 microcontroller as the processing heart of a robust motor control system for low-cost industrial and consumer motor control applications.

In addition, Motorola application notes *Low-Cost 3-Phase AC Motor Control System Based on MC68HC908MR24*, document order number AN1664/D, and *A 3-Phase AC Induction Motor Control System Based on the MC68HC908MR32*, document order number AN1857/D, give the reader a full view on the presented application software and hardware, which can be found on Motorola Web pages.

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Freescale Semiconductor, Inc.