VF Control of 3-Phase Induction Motor Using Space Vector Modulation

Author: Rakesh Parekh
Microchip Technology Inc.

INTRODUCTION

VF control using the Sine PWM algorithm is a popular algorithm for AC induction motor control; however, this algorithm has certain drawbacks which affect the overall system efficiency. A more advanced switching algorithm, like Space Vector Modulation (SVM), overcomes the drawbacks of the Sine PWM algorithm and increases the overall system efficiency.

This application note includes the description of the SVM theory and its advantages over the Sine PWM. It also discusses the SVM digital implementation for VF control using Microchip's PIC18FXX31 8-bit microcontrollers. See the **"References"** section for more information on AC induction motors and their control.

SINE PWM

Traditionally, VF control using the Sine PWM algorithm is implemented using a Voltage Source Inverter (VSI) controlled by a programmable device (microcontroller or DSP). Its popularity is mainly due to its easy implementation and minimum online computational requirement. However, this algorithm has the following drawbacks:

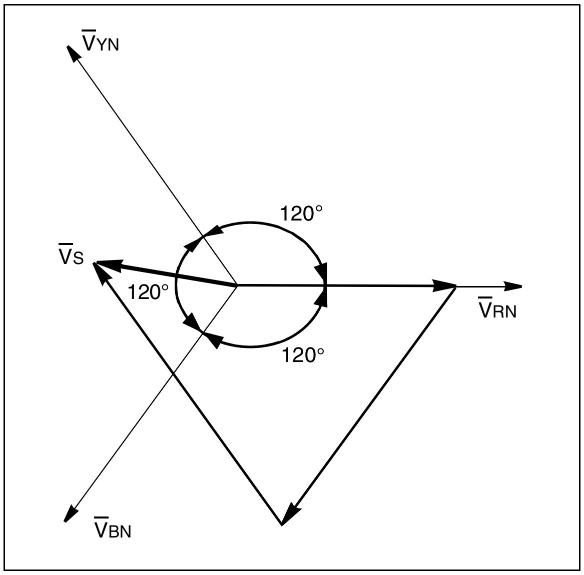
- The Sine PWM algorithm is unable to fully utilize the available DC bus supply voltage (V_{DC}) to the VSI. The generated line-to-line voltage is less than 90% of V_{DC} in the linear operating region. See **Appendix B: "Sine PWM"** for more information.
- This algorithm gives more Total Harmonic Distortion (THD).
- Often, to reduce run-time processing load for slow controllers, three 120° phase-shifted sine tables are created in the controller memory. This is an inefficient usage of the controller memory.
- There is no degree of freedom in implementation.
- This algorithm does not facilitate more advanced vector control implementation.

SPACE VECTOR MODULATION (SVM)

The SVM is a sophisticated, averaging algorithm which gives 15% more voltage output compared to the Sine PWM algorithm, thereby increasing the V_{DC} utilization. It also minimizes the THD as well as switching loss. Like Sine PWM, the SVM is also a scalar control. The direct controlled variables are the motor voltage and the motor frequency.

The 3-phase line-to-neutral sine waves required for driving the 3-phase AC induction motor can be represented as 120° phase-shifted vectors ($\overline{V}_{RN}$, $\overline{V}_{YN}$, and $\overline{V}_{BN}$) in space, as shown in Figure 1.

FIGURE 1: 3-PHASE VOLTAGE VECTORS AND THE RESULTANT SPACE REFERENCE VECTOR



For a balanced 3-phase system, these vectors sum to zero. Therefore, they can be expressed as a single space reference vector ($\overline{V}_S$). By controlling the amplitude and the frequency of $\overline{V}_S$, the motor voltage and the motor frequency can be controlled. Hence, this algorithm is known as the SVM.

A typical block diagram of a VSI controlled by the PIC18FXX31, which implements SVM, is shown in Figure 2. Point 0 is the midpoint of VDC (sometimes called the Virtual Neutral Point). For safe operation of the VSI, whenever one switch of a half bridge (Q1) is on, the other switch of the same half bridge (Q0) should be off and vice versa. This gives rise to eight distinct switching states of the VSI. Table 1 lists all the possible VSI switching states and respective line-to-neutral voltages.

States 1 through 6 are called the active states, as the energy is supplied from the supply to the motor during these states. States 0 and 7 are called the inactive states, as no energy is supplied from the supply to the motor during these states. Each state can be represented as a voltage vector in space. Figure 3 shows the space vector representation of all the possible switching states.

FIGURE 2: BLOCK DIAGRAM OF PICmicro® MCU-CONTROLLED VSI

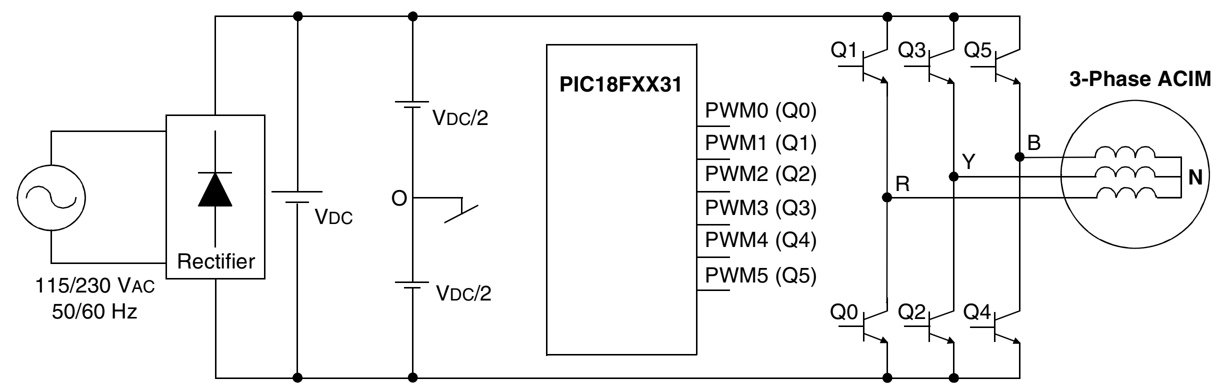


TABLE 1: VSI SWITCHING STATES AND RESPECTIVE LINE TO NEUTRAL VOLTAGES

Switching State	On Switches	V_{RN}	V_{YN}	V_{BN}	Space Voltage Vector
0	Q0, Q2, Q4	0	0	0	$\overline{V}_0$
1	Q1, Q2, Q4	2/3 V_{DC}	-1/3 V_{DC}	-1/3 V_{DC}	$\overline{V}_1$
2	Q1, Q3, Q4	1/3 V_{DC}	1/3 V_{DC}	-2/3 V_{DC}	$\overline{V}_2$
3	Q0, Q3, Q4	-1/3 V_{DC}	2/3 V_{DC}	-1/3 V_{DC}	$\overline{V}_3$
4	Q0, Q3, Q5	-2/3 V_{DC}	1/3 V_{DC}	1/3 V_{DC}	$\overline{V}_4$
5	Q0, Q2, Q5	-1/3 V_{DC}	-1/3 V_{DC}	2/3 V_{DC}	$\overline{V}_5$
6	Q1, Q2, Q5	1/3 V_{DC}	-2/3 V_{DC}	1/3 V_{DC}	$\overline{V}_6$
7	Q1, Q3, Q5	0	0	0	$\overline{V}_7$

As seen in Figure 3, the entire space is distinctively divided into six equal sized sectors of 60°. Each sector is bounded by two active vectors. $\overline{V}_0$ and $\overline{V}_7$ are the voltage vectors with zero amplitude and are located at the hexagon origin.

$\overline{V}_S$ is the resultant output due to the switching states of the VSI. For digital implementation of SVM, the VSI is switched at a very high frequency (F_{PWM}). This frequency is high enough (>20 kHz) so as not to generate audible noise due to switching. F_{PWM} decides the sample time T_S for $\overline{V}_S$, where $T_S = 1/F_{PWM}$. There are various switching ways to generate $\overline{V}_S$ from $\overline{V}_0$, $\overline{V}_1 \ldots \overline{V}_7$. Mathematically, it can be represented as shown in Equation 1. Variables T_0, $T_1 \ldots T_7$ in Equation 1 are on time for the corresponding VSI states and T_S is the sample time.

When the VSI follows the switching state pattern, 1-2-3-4-5-6-1-2..., it is called the Six-Step PWM algorithm. This algorithm is easier to implement compared to all the other control algorithms. It can generate the line-to-line fundamental voltage more than the V_{DC}. But this algorithm generates maximum THD. Also, the line-to-line and the line-to-neutral waveforms are not sine waves.

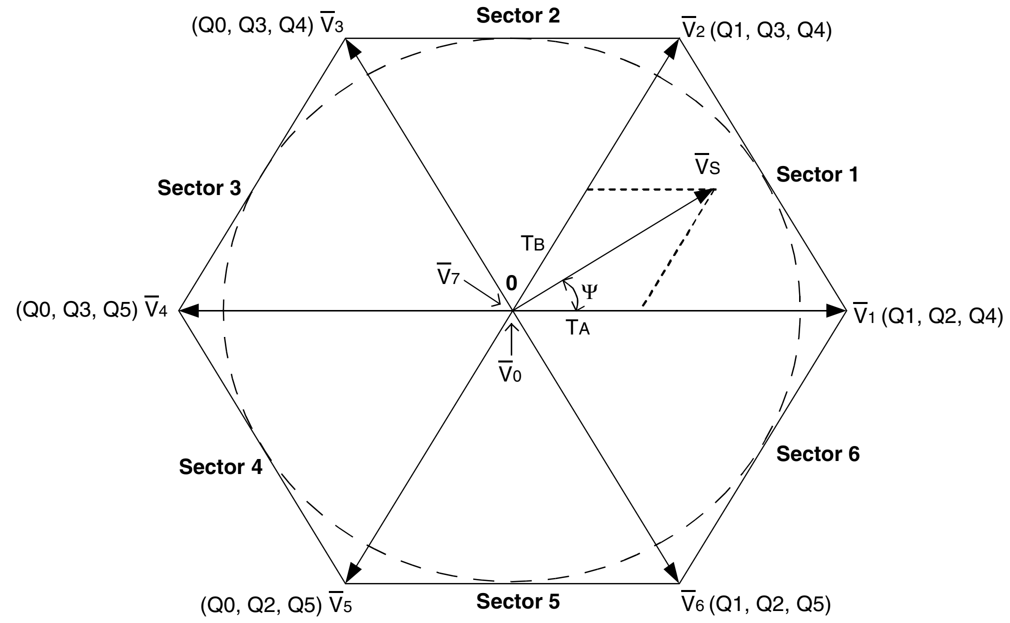
SVM Switching Rules

To implement the SVM algorithm, the following switching rules are implemented:

- The trajectory of $\overline{V}_S$ should be a circle.
- Only one switching per state transition.
- Not more than three switchings in one T_S.
- The final state of one sample must be the initial state of the next sample.

These rules help in limiting the number of switching actions and hence, there is a reduction in the switching losses. Also, they maintain symmetry in switching waveforms at the VSI output to achieve the lower THD. The SVM algorithm implementation, using these switching rules, is called Conventional SVM.

FIGURE 3: SPACE VECTOR HEXAGON



EQUATION 1:

$$\overline{V}_S = \left(\frac{T_0}{T_S} \times \overline{V}_0\right) + \left(\frac{T_1}{T_S} \times \overline{V}_1\right) + \left(\frac{T_2}{T_S} \times \overline{V}_2\right) + \left(\frac{T_3}{T_S} \times \overline{V}_3\right) + \left(\frac{T_4}{T_S} \times \overline{V}_4\right) + \left(\frac{T_5}{T_S} \times \overline{V}_5\right) + \left(\frac{T_6}{T_S} \times \overline{V}_6\right) + \left(\frac{T_7}{T_S} \times \overline{V}_7\right)$$

$$T_S = T_0 + T_1 + T_2 + T_3 + T_4 + T_5 + T_6 + T_7$$

Different SVM Algorithms

There are various ways to implement the SVM, such as:

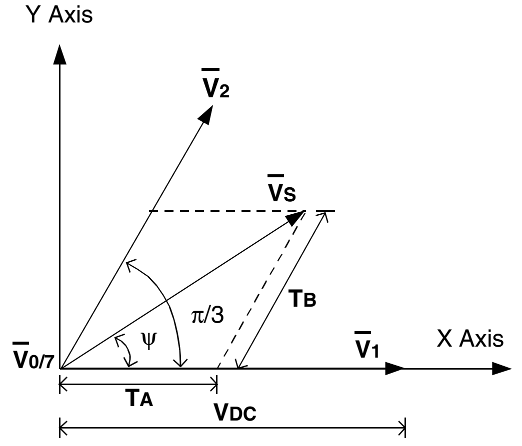
- Conventional SVM
- Basic Bus Clamping SVM
- Boundary Sampling SVM
- Asymmetric Zero-Changing SVM, etc.

All SVM algorithms have the same on time for active as well as inactive vectors. They differ mainly in the implementation of the inactive vectors, such as T_0 and/or T_7 distribution within T_S. Discussion of the various other SVM algorithms is beyond the scope of this application note.

Time Calculation to Generate $\overline{V}_S$

Let us take an example where $\overline{V}_S$ is in Sector 1 at a vector angle (Ψ), as shown in Figure 4.

FIGURE 4: VECTOR $\overline{V}_S$ IN SECTOR 1



It is assumed that during time T_S, $\overline{V}_S$ remains steady. For implementing the conventional SVM using SVM switching rules, $\overline{V}_S$ is split as shown in Equation 2.

EQUATION 2:

$$\overline{V}_S = \left(\frac{T_A}{T_S}\times\overline{V}_1\right) + \left(\frac{T_B}{T_S}\times\overline{V}_2\right) + \left(\frac{T_{0/7}}{T_S}\times\overline{V}_{0/7}\right)$$

Equation 2 means that the VSI is in active state 1 for T_A time and it is in active state 2 for T_B time. For the remaining time of T_S, no voltage is applied. This can be achieved by applying inactive state 0 (or 7) for the remaining time T_0 (or T_7).

EQUATION 3:

$$T_S = T_A + T_B + T_{0/7}$$

The time intervals, T_A, T_B and $T_{0/7}$, have to be calculated such that the average volt seconds produced by the vectors, $\overline{V}_1$, $\overline{V}_2$ and $\overline{V}_{0/7}$ along the X and Y axes, are the same as those produced by the desired reference space vector $\overline{V}_S$.

The modulation index or amplitude ratio is defined as:

$$m = \frac{|\overline{V}_S|}{V_{DC}}$$

where $|\overline{V}_S|$ is the amplitude or the length of $\overline{V}_S$.

Resolving $\overline{V}_S$ along the X and Y axes, we get:

EQUATION 4:

$$(V_{DC}\times T_A) + (V_{DC}\times\cos\pi/3\times T_B) = |\overline{V}_S|\times\cos\psi\times T_S$$

and

$$V_{DC}\times\sin\pi/3\times T_B = |\overline{V}_S|\times\sin\psi\times T_S$$

Solving for T_A and T_B, we get:

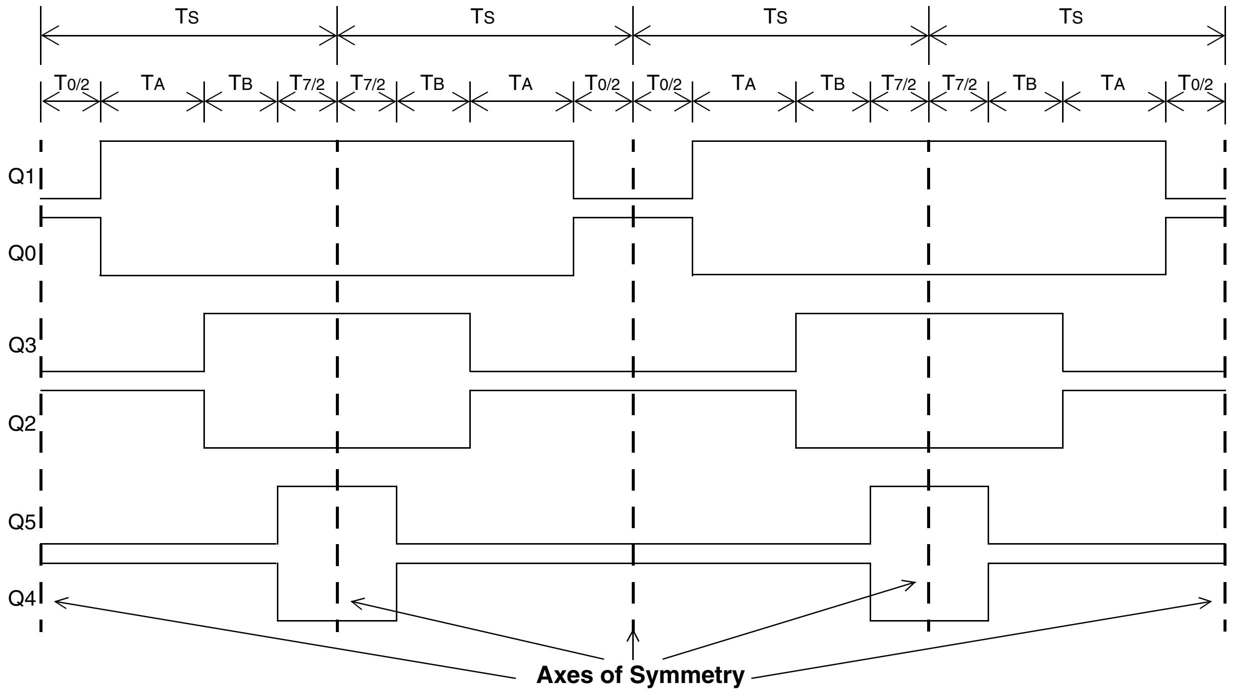
EQUATION 5:

$$\frac{T_A}{T_S} = \frac{2}{\sqrt{3}}\times m\times\sin\left(\frac{\pi}{3}-\psi\right)$$

$$\frac{T_B}{T_S} = \frac{2}{\sqrt{3}}\times m\times\sin\psi$$

$T_{0/7}$ can be found from Equation 3. For better THD, T_0 (or T_7) is split into two and then applied at the beginning and at the end of the T_S. The typical VSI switching waveforms in Sector 1, as defined by Equation 2, Equation 3 and the switching rules for the conventional SVM using center aligned PWM, are as given in Figure 5.

FIGURE 5: TYPICAL VSI SWITCHING WAVEFORMS IN SECTOR 1



We can observe the different axes of symmetry in all the waveforms as shown in Figure 5. These symmetries are mainly responsible for having lower THD in SVM compared to Sine PWM in the linear operating region.

From Figure 3, it is clear that in the linear operating region, the maximum line-to-line voltage amplitude can be achieved when $\overline{V_S}$ is rotated along the largest inscribed circle in the space vector hexagon. In mathematical terms, this is equivalent to:

EQUATION 6:

$$m_{max} = \frac{\text{Radius of Largest Inscribed Circle}}{V_{DC}}$$

From Figure 4 and Equation 6, it is also clear that:

EQUATION 7:

$$m_{max} = \frac{V_{DC} \times \cos\pi/6}{V_{DC}} = \cos\pi/6 = \frac{\sqrt{3}}{2}$$

By solving Equation 2, Equation 5 and Equation 7, we get:

EQUATION 8:

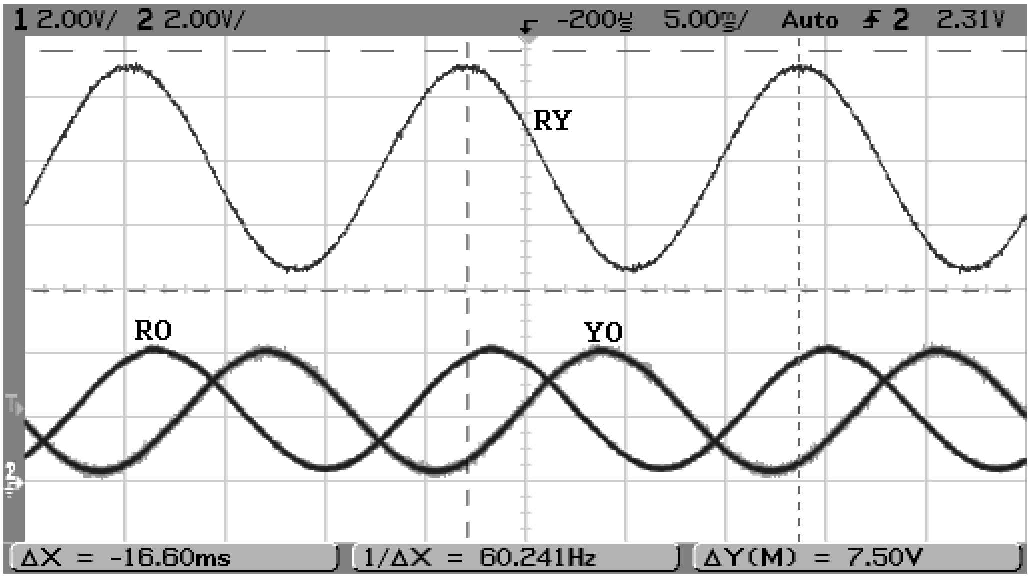
$$\begin{aligned}&\text{Maximum Line-to-Line Voltage}\\ &= \frac{2}{\sqrt{3}} \times m_{max} \times V_{DC}\\ &= \frac{2}{\sqrt{3}} \times \frac{\sqrt{3}}{2} \times V_{DC} = V_{DC}\end{aligned}$$

Equation 8 shows that it is possible to get line-to-line voltage amplitude as high as V_{DC} using the SVM algorithm in the linear operating range. This is the main advantage of the SVM algorithm when compared to the Sine PWM algorithm. Due to higher line-to-line voltage amplitude, the torque generated by the motor is higher. This results in better dynamic response of the motor.

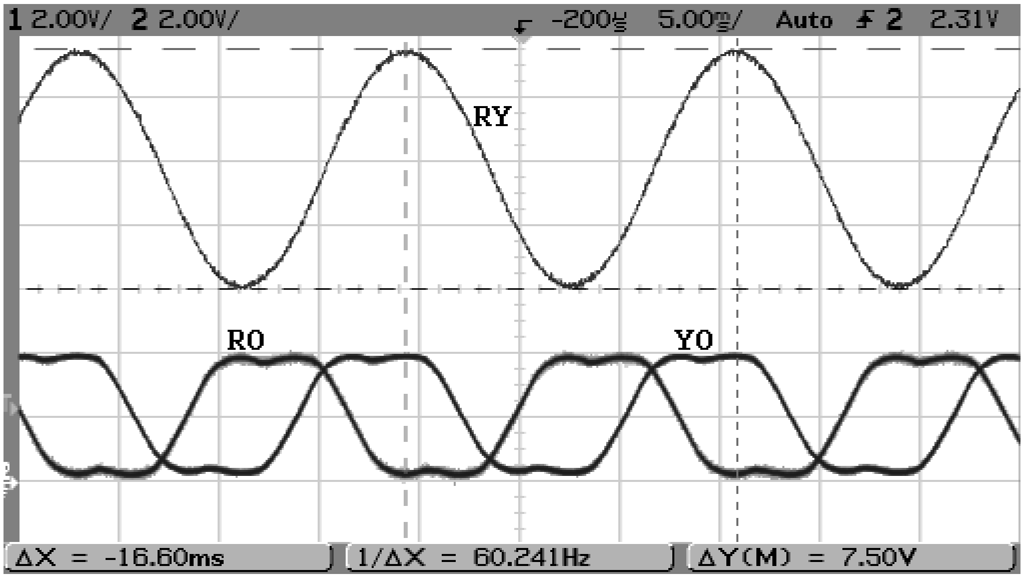
The reason for the higher line-to-line voltage in SVM can be explained with the help of Figure 6. It shows the phase voltage (line-to-virtual neutral point) generated by Sine PWM and SVM. For clarity, only two phase voltages (RO and YO) and their resultant line-to-line voltage (RY) are shown in each figure.

The Sine PWM generated phase voltages are sine waves. With 120° phase shift between them, the resultant line-to-line voltage is approximately 86.6% of VDC. But, the SVM generated phase voltages have a third harmonic component superimposed on the fundamental component. The addition of this harmonic component is due to the effective usage of inactive states which is not possible in the Sine PWM. With 120° phase shift between them, the third harmonic component is cancelled out in the resultant line-to-line voltage in such a way that the resultant line-to-line voltage is boosted to VDC (100%). Thus, SVM generates line-to-line voltage with higher amplitude (about 15% more) compared to Sine PWM.

FIGURE 6: GENERATED PHASE VOLTAGES AND CORRESPONDING LINE-TO-LINE VOLTAGE IN (A) SINE PWM AND (B) SVM



A) Sine PWM Generated Waveforms



B) SVM Generated Waveforms

Advantages of SVM

The advantages of SVM vis-a-vis Sine PWM are as follows:

- Line-to-line voltage amplitude can be as high as V_{DC}. Thus, 100% V_{DC} utilization is possible in the linear operating region.
- In the linear operating range, modulation index range is 0.0 to 1.0 in the Sine PWM; whereas in the SVM, it is 0 to 0.866. Line-to-line voltage amplitude is 15% more in the SVM with the modulation index = 0.866, compared to the Sine PWM with the modulation index = 1. Hence, it has the better usage of the modulation index depth.
- With the increased output voltage, the user can design the motor control system with reduced current rating, keeping the horsepower rating the same. The reduced current helps to reduce inherent conduction loss of the VSI.
- Only one reference space vector is controlled to generate 3-phase sine waves.
- Implementation of the switching rules gives less THD and less switching loss.
- Flexibility to select inactive states and their distribution in switching time periods gives two degrees of freedom.
- As the reference space vector is a two-dimensional quantity, it is feasible to implement more advanced vector control using SVM.

HARDWARE USED FOR SVM IMPLEMENTATION

A PICDEM™ MC Development Board is used to develop and test the SVM control firmware. The PICDEM MC has a single-phase diode bridge rectifier, converting AC input to DC and a power capacitor bank that keeps the DC bus stable. A 3-phase IGBT-based inverter bridge is used to control the output voltage from the DC bus. See **Appendix A: "PICDEM™ MC Board Technical Information"** for schematics of the PICDEM™ MC Development Board.

The control circuit and power circuits are optically isolated with respect to each other. An on-board flyback power supply generates +5 VD, with respect to the digital ground used for powering up the control circuit, including the PICmicro® device. The +5 VA and +15 VA are generated with respect to the power ground (negative of DC bus). The feedback interface circuit is powered by the +5 VA, while the +15 VA supplies power to the IGBT drivers located inside the Integrated Power Module (IPM).

With optical isolation between the power and the control circuits, the programming and debugging tools can be plugged into the development board when main power is connected to the board. The board communicates with a host PC over a serial port configured with an on-chip Enhanced USART (EUSART). The on-board user interface has two toggle switches, a potentiometer and four LEDs for indication.

In this application note, switch SW1 is used to toggle between the motor Run and Stop and SW2 is used to toggle the motor rotation direction. A potentiometer is used to set the speed reference as well as the modulation index. The LEDs are used for indication of different states of control.

DIGITAL IMPLEMENTATION

To implement the SVM in the digital domain, the power control PWM module of the PIC18FXX31 is utilized. The module provides up to 8 PWM output channels with the dedicated PWM timer as its time base. The module has the capability to generate center aligned PWM with 14-bit resolution. This is the most important feature required for the SVM implementation.

$\overline{V}_S$ needs to be created and rotated in space for SVM implementation. To approximate the position of $\overline{V}_S$, Equation 3, Equation 5 and the previously mentioned switching rules are utilized. Looking at Equation 5, one will notice that in the same sector, T_A and T_B are inverted with respect to each other. Hence, only one look-up table with time entries is needed. A look-up table with T_B entries (`TABLE_TB_COUNT<MSB:LSB>`) is created. The size of the look-up table is decided by the angle resolution used. The total sector angle is 60° (π/3 radians). To get a good resolution with an 8-bit microcontroller like the PIC18FXX31, the entire sector is divided into 256 points, giving an angle resolution of 0.234°. The center aligned PWM is used for better THD ($F_{PWM} = 1/2 * T_S$).

The required motor speed in Hz is decided by the rate at which the $\overline{V}_S$ is rotated. For this purpose, it is necessary to find both the vector angle and the vector update step size. To speed up the online calculations, the constant, `DEGREE_CONSTANT`, is defined; this is then used to calculate the vector update step size and the vector angle as shown in Equation 9.

Looking at the definition of `DEGREE_CONSTANT`, one will notice that its value, without any multiplication factor, will result in a fractional number less than unity for F_{PWM} > 1.536 kHz. Almost all motor control applications have F_{PWM} much higher than 1.536 kHz. Handling a fractional number with any 8-bit microcontroller will require more CPU processing time. This requirement is difficult to meet in the SVM implementation, where $\overline{V}_S$ is updated at every T_{PWM} (= $1/F_{PWM}$) time interval. At the same time, the multiplication factor value needs to be such that its post-calculation adjustment requires the least possible microcontroller processing time.

It is proposed that the multiplication factor be 256. This will result in a 16-bit value for the vector update step size and hence, 16-bit vector angle pointer (`VECTOR_ANGLE<MSB:LSB>`). As an adjustment for the multiplication factor, `VECTOR_ANGLE_LSB` is discarded. `VECTOR_ANGLE_MSB` is used as the table pointer for reading the value of T_A and T_B from the look-up table. Whenever a carry is generated due to the Equation 9 addition, it physically means that the $\overline{V}_S$ has advanced to the next sector and hence, the sector count (`SECTOR_NO`) is incremented by one. The motor voltage is decided by the amplitude of $\overline{V}_S$ (modulation index m). To implement the same digitally, values of T_A and T_B are multiplied by m. Based on T_S, T_A and T_B, the duty cycle values for all 3 phases (R, Y and B) are calculated as shown in Table 2.

Equations shown in Table 2 for Sector 1 are evident in Figure 5. Similarly, equations for other sectors are derived with the switching rule constraints.

EQUATION 9:

$$\text{DEGREE\_CONSTANT} = \frac{360 \times 256 \times \text{Multiplication Factor}}{60 \times F_{PWM}}$$

$$\text{Vector Update Step Size} = \text{DEGREE\_CONSTANT} \times \text{Required Motor Speed (Hz)}$$

$$\text{Vector Angle} = \text{Vector Angle} + \text{Vector Update Step Size}$$

TABLE 2: DUTY CYCLE VALUES FOR THE THREE MOTOR PHASES BASED ON $\overline{V}_S$ LOCATION

Sector No.	Phase R Duty Cycle	Phase Y Duty Cycle	Phase B Duty Cycle
1	$T_{0/2}$	$T_{0/2} + T_A$	$T_S - T_{0/2}$
2	$T_{0/2} + T_B$	$T_{0/2}$	$T_S - T_{0/2}$
3	$T_S - T_{0/2}$	$T_{0/2}$	$T_{0/2} + T_A$
4	$T_S - T_{0/2}$	$T_{0/2} + T_B$	$T_{0/2}$
5	$T_{0/2} + T_A$	$T_S - T_{0/2}$	$T_{0/2}$
6	$T_{0/2}$	$T_S - T_{0/2}$	$T_{0/2} + T_B$

SVM CONTROL FIRMWARE

The firmware is developed using PIC18F4431 and it implements the VF control using the SVM algorithm. Apart from the basic SVM control, the firmware incorporates various control and protection routines, such as overcurrent protection, overvoltage protection, overtemperature protection, acceleration and deceleration routine and rotation direction reversal.

To implement the VF control, an analog potentiometer (R44) connected to RA1/AN1 is read. Using the `CONVERT_MANUAL_COUNT_TO_HZ` routine, the potentiometer setting is converted to the required motor speed in Hz. The potentiometer setting is also interpreted as the modulation index, m. To get the required motor speed, the vector update step size is calculated by calling the `CALCULATE_UPDATE_STEP_SIZE` routine. As described in the previous section, T_A and T_B values are scaled with m.

All protection routines (overcurrent, overvoltage and overtemperature) are checked at a fixed time interval (presently, 5 ms set by overflow rate of the Timer1). The acceleration and deceleration routine is called at one-second intervals.

Overcurrent Protection

A shunt resistor (R110) in the negative DC bus gives a voltage corresponding to the current flowing into the motor winding. This voltage is amplified and compared with a reference. The current comparison setting allows a current up to 6.3A. If the current exceeds 6.3A, the FAULTA pin goes low, indicating the overcurrent Fault. The firmware is configured in the Cycle-by-Cycle Fault mode. When the Fault persists for more than a preprogrammed count (`MAX_FLTA_COUNT`) in the fixed time interval, then the motor is stopped and the Fault is indicated by the blinking LED1.

Overvoltage Protection

The DC bus voltage is attenuated using potential dividers and compared with a fixed reference. If the jumper JP5 is open, the reference is set for 200V on the DC bus. If jumper JP5 is shorted, the reference is set for 400V. The FAULTB pin is used to monitor the overvoltage Fault. If the overvoltage Fault persists for more than a preprogrammed count (`MAX_FLTB_COUNT`) in the fixed time interval, then the motor is stopped and the Fault is indicated by the blinking LED2.

Overtemperature Protection

The power module has an NTC thermal sensor, outputting 3.3V at 110°C on the junction of IGBTs. The NTC output is connected to AN8 through an optocoupler. The temperature is continuously measured and if it exceeds 110°C (`MAX_JUNCTION_TEMP)` for more than a preprogrammed count (`MAX_TEMP_FILT_COUNT`), then the motor is stopped and the overtemperature Fault is indicated by the blinking LED3.

Whenever the motor is stopped due to any of the above mentioned Faults, it can be restarted by removing the Fault condition and then by pressing either SW1 or SW2.

Acceleration and Deceleration

The `RAMP_SPEED` routine is called every 1 second to implement the acceleration and deceleration feature. Both the acceleration and deceleration rates are user-selectable and are given in the form of Hz/s.The acceleration and deceleration features are active only when the motor is in run condition.

SOURCE CODE FILES

The entire source code can be downloaded from Microchip's web site, **www.microchip.com**. It includes the following files:

- `...\SVM\ACSVM_OL.asm`

  This file is located in the source code folder (Main Routine section).
- `...\SVM\ACSVM_OL_routines.asm`

  This file is located in the source code folder (Control Routine section).
- `...\SVM\TIME_TABLE.asm`

  This file is located in the source code folder (Look-up Table for T_B).
- `...\SVM\InterfaceACSVMConstant.inc`

  This file is located in the source code folder (System Parameters and User-Defined Constants section).
- `...\SVM\InterfaceACSVMVar.inc`

  This file is located in the source code folder (User-Defined Variable section).

An Excel file (`PARAMETERS.xls`) is included in the source code folder which has two worksheets. The `TB_TABLE` worksheet calculates T_B entries based on the main oscillator frequency (F_{OSC}), F_{PWM}, the required dead time, etc. The `LIST` worksheet creates a table to be stored in the data EEPROM of the PIC18FXX31 device. This table contains all user-selectable compile-time parameters, such as the motor rated speed in Hz, F_{PWM} in kHz, the required dead time in µs, etc. The user should make sure that the table is entered in the `ACSVM_OL.asm` file as shown in the worksheet.

An overview of the firmware's logic flow is provided in Figure 7 and Figure 8. A complete list of system parameters and user-defined functions is provided in Table 3 through Table 5.

FIGURE 7: SVM IMPLEMENTATION FLOWCHART (MAIN ROUTINE)

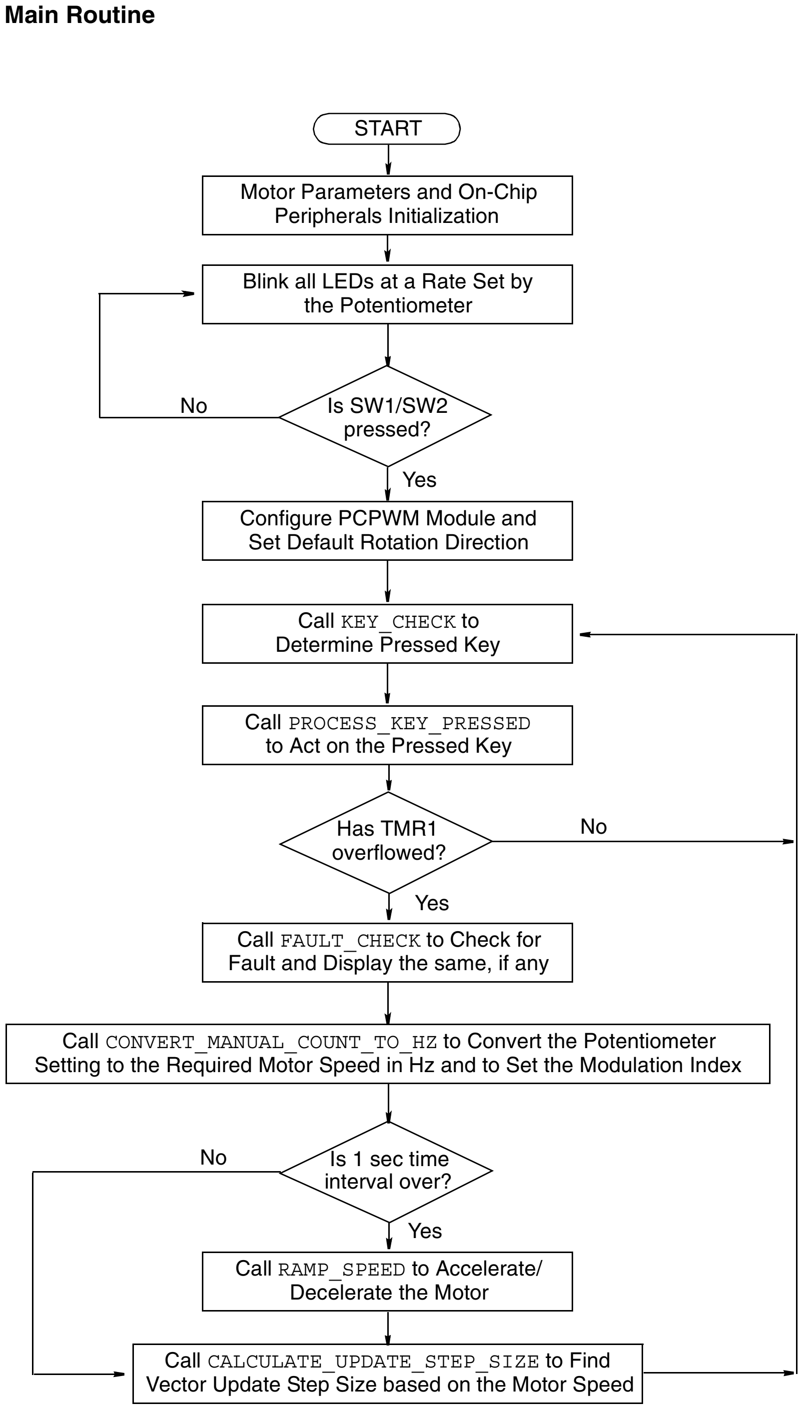
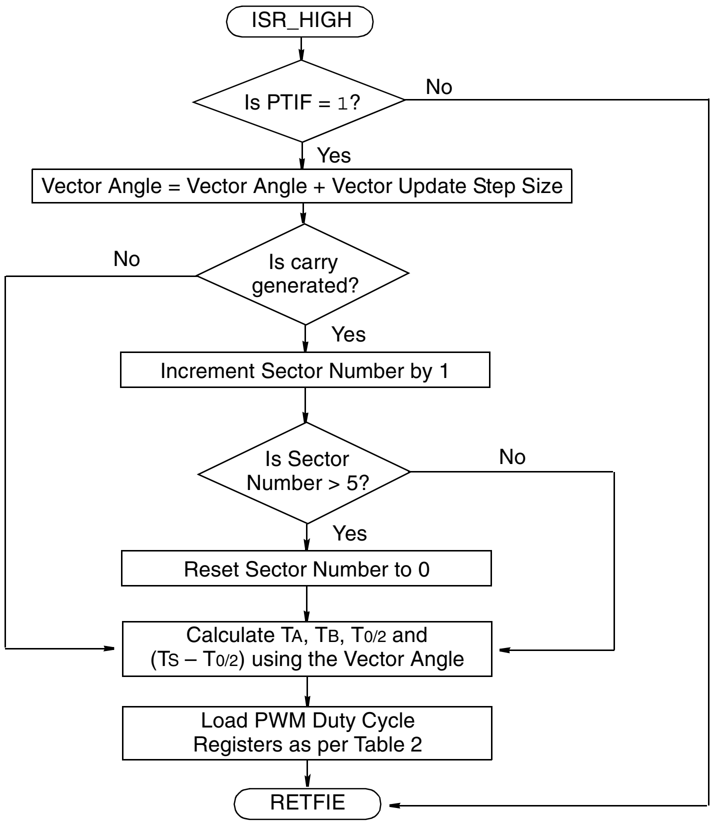


FIGURE 8: SVM IMPLEMENTATION FLOWCHART (INTERRUPT SERVICE ROUTINES)

High Priority Interrupt Service Routine



Low Priority Interrupt Service Routine

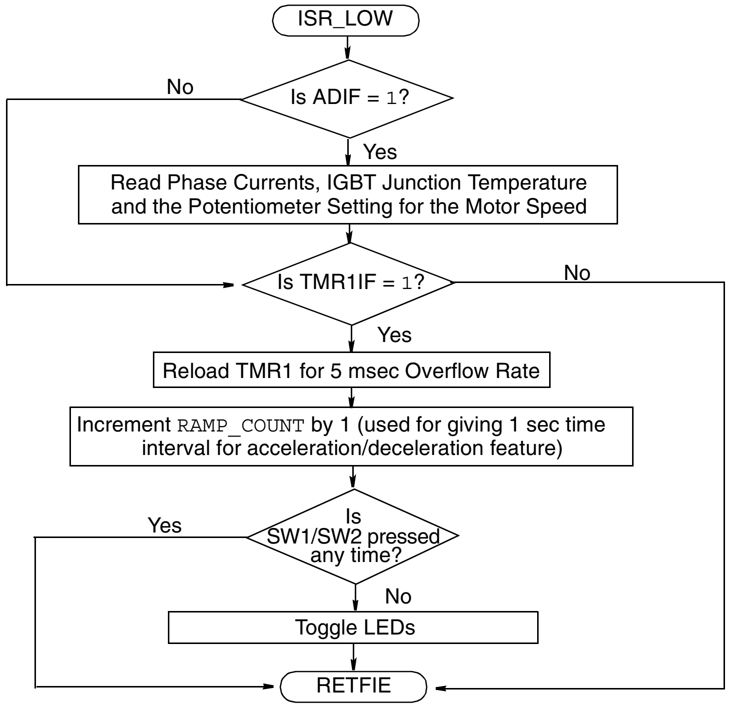


TABLE 3: USER-DEFINED PARAMETERS IN SVM FIRMWARE

Name	Description
`CHANGE_OVER_DELAY`	Defines rotation direction changeover for the motor. Its value equals the required changeover time in ms per 39 ms.
`DIRECTION_AT_POR`	Defines default rotation direction at start (`1` = Forward, `0` = Reverse).
`CYCLE_COUNT_MAX`	Defines blinking rate for Fault indication. Its value equals the required blink time interval in ms per 5 ms.
`MAX_FAULT_CHECK_COUNT`	Defines time window for Fault recognition. If any Fault occurs for more than predefined count in this time window, then the Fault is recognized and appropriate action is taken.
`MAX_FLTA_COUNT`	Defines count for overcurrent Fault recognition.
`MAX_FLTB_COUNT`	Defines count for overvoltage Fault recognition.
`MAX_JUNCTION_TEMP`	Defines limit for overtemperature Fault.
`MAX_TEMP_FILT_COUNT`	Defines count for overtemperature Fault recognition.
`TMR1L_COUNT` and `TMR1H_COUNT`	Defines overflow rate for Timer1. In the present application, it is set for 5 ms.
`PARAMETER_BUFFER_SIZE`	Defines array size for storing various compile-time parameters as well as run-time parameters. It is set at 0x35 (do not change this value).

TABLE 4: VARIABLES IN SVM FIRMWARE

Name	Description
`REQD_SPEED_REF`	Stores required motor speed set by the potentiometer.
`SET_SPEED_HZ`	Stores actual motor speed (Hz).
`RAMP_COUNT`	Stores count for Timer1 overflow. It is used for generating 1s time interval as required for acceleration/deceleration feature.
`CURRENT_R`	Stores either total DC bus current (when used with DC shunt current measurement) or R phase current (when used with phase current sensor).
`CURRENT_Y`	Stores Y phase current (when used with phase current sensor).
`CURRENT_B`	Stores B phase current (when used with phase current sensor).
`JUNCTION_TEMPH`	Stores junction temperature of the VSI switch.
`SECTOR_NO`	Stores present sector location of $\overline{V}_S$.
`MODULATION_INDEX`	Stores required amplitude of $\overline{V}_S$.
`VECTOR_ANGLE<MSB:LSB>`	Stores present vector angle of $\overline{V}_S$ in a sector.
`TB_COUNT<MSB:LSB>`	Stores T_B count as pointed by the vector angle.
`TA_COUNT<MSB:LSB>`	Stores T_A count as pointed by the vector angle.
`HALF_T0_COUNT<MSB:LSB>`	Stores $T_{0/2}$ count.
`TS_MINUS_HALF_T0<MSB:LSB>`	Stores $T_S - T_{0/2}$ count.
`TABLE_TB_COUNT<MSB:LSB>`	Stores array base address for T_B count.
`PARAMETER_BUFFER`	Stores array base address for compile-time and run-time parameters.

TABLE 5: FUNCTIONS IN SVM FIRMWARE

Name	Description
`KEY_CHECK`	Checks the status of Run/Stop (SW1) and Fwd/Rev (SW2) keys.
`PROCESS_KEY_PRESSED`	Acts on command issued by the last pressed key.
`FAULT_CHECK`	Checks for various Faults (overcurrent, overvoltage, overtemperature) and acts if any Fault is recognized.
`CONVERT_MANUAL_COUNT_TO_HZ`	Converts the potentiometer setting into the required motor speed (Hz) and sets the modulation index (m).
`RAMP_SPEED`	Implements the acceleration/deceleration feature.
`CALCULATE_UPDATE_STEP_SIZE`	Calculates new vector update step size depending on the motor speed.
`ISR_PWM`	Responds to setting of PTIF. This routine rotates by vector angle update step size and calculates new duty cycle values for all phases (R, Y and B).
`ISR_ADC`	Responds to setting of ADIF. This routine reads the potentiometer setting, phase currents and junction temperature of the VSI switch and stores them at appropriate locations.
`ISR_TMR1`	Responds to setting of TMR1IF. This routine reloads Timer1 for 5 ms overflow rate and increments `RAMP_COUNT` by 1 for generating a one-second interval (required for the acceleration/deceleration feature). This routine also blinks all LEDs at start when no key is pressed.

RESOURCE USAGE

The SVM control application consumes CPU resources, as shown in Table 6. Substantial CPU resources, especially memory and processing time, are still available to users for the development of their own applications.

TABLE 6: RESOURCES USED IN THE MOTOR CONTROL DEMO BOARD (USING PIC18F4431)

Resource Type	Used	Available to User when PIC18F4431 is Used
Program Memory	1942 bytes	14442 bytes
Data Memory	93 bytes	675 bytes
EEPROM	44 bytes	212 bytes
PWM Channels	6	2
CCP/Fault Input Channels	2	0
ADC Channels	5	4
EUSART	0	1
QEI Module	0	1
Timers	1	3
External Interrupts	0	3
I/O Lines	20	16
CPU Processing Time (F_{PWM} = 20 kHz, F_{OSC} = 40 MHz)	~27%	~63%

Resource utilization, as mentioned in Table 6, is for a general purpose, relocatable code implementing the VF control using the SVM algorithm on PIC18F4431 with 14-bit PWM resolution. A customized solution with only 8-bit PWM resolution can conceivably result in an additional 10% savings in CPU processing time.

CONCLUSION

VF control using the SVM in the open loop is more energy efficient compared to the Sine PWM. With an on-chip dedicated motor control peripheral like the power control PWM module and the rich instruction set, the PIC18FXX31 is well suited to give a low-cost solution, implementing the VF control using the SVM algorithm for the 3-phase AC induction motor control. In addition, the on-chip resources, such as the ADC and the multiple timers, allow users to implement other control (acceleration and deceleration) and protection (overcurrent, overvoltage, overtemperature) features.

REFERENCES

R. Parekh, *AN887, "AC Induction Motor Fundamentals"* (DS00887). Microchip Technology Inc., 2003.

P. Yedamale, *AN843, "Speed Control of 3-Phase Induction Motor Using PIC18 Microcontrollers"* (DS00843). Microchip Technology Inc., 2002.

R. Parekh, *AN889, "VF Control of 3-Phase Induction Motors Using PIC16F7X7 Microcontrollers"* (DS00889). Microchip Technology Inc., 2004.

"PICDEM™ MC Development Board for PIC18FXX31 User's Guide" (DS51453). Microchip Technology Inc., 2004.

APPENDIX A: PICDEM™ MC BOARD TECHNICAL INFORMATION

FIGURE A-1: PICDEM™ MC DEVELOPMENT BOARD FUNCTIONAL BLOCK DIAGRAM

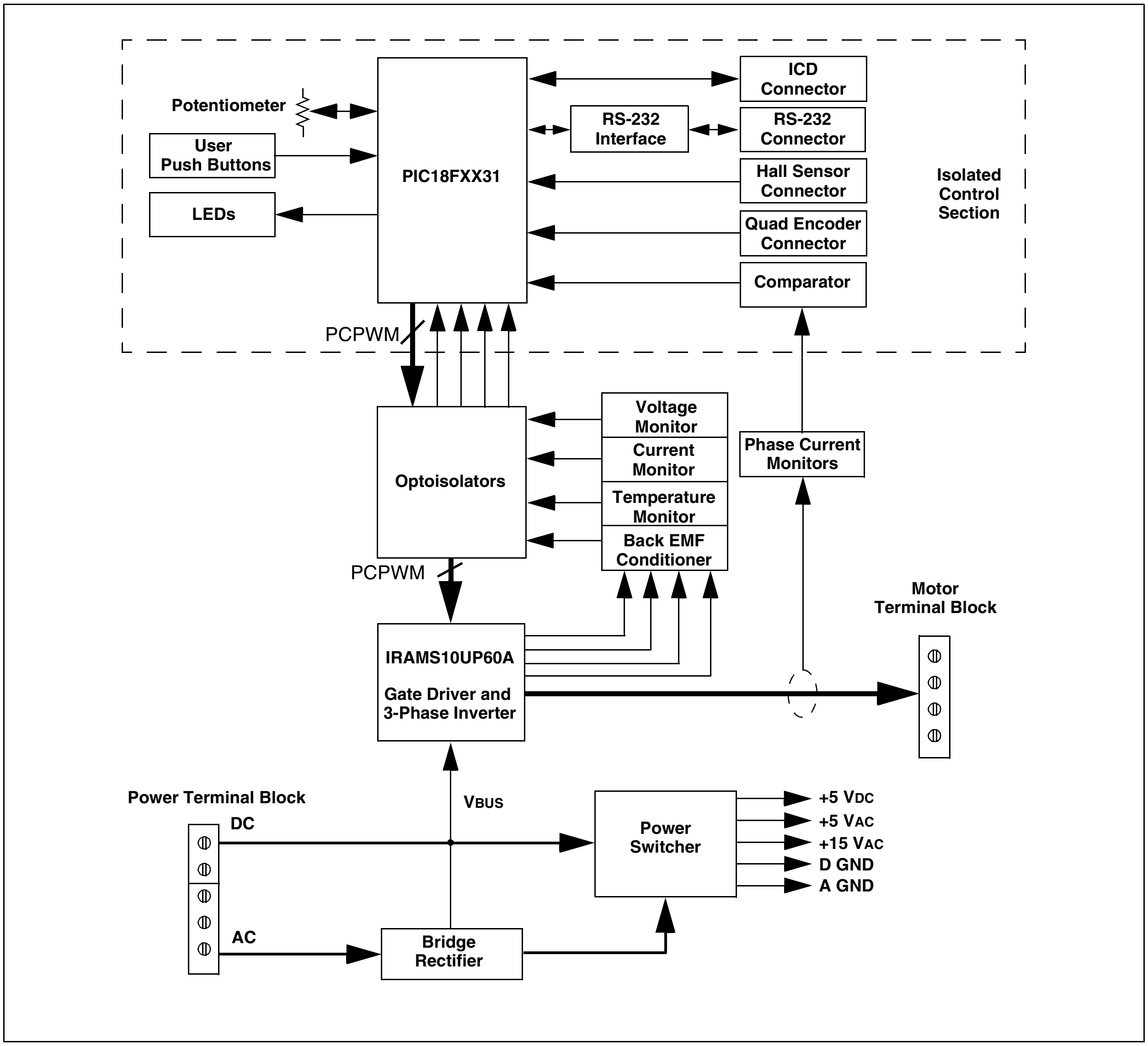


FIGURE A-2: BOARD SCHEMATIC, PART 1 (PIC18F4X31 MICROCONTROLLER, PCPWM ISOLATORS, CURRENT COMPARATOR AND ASSOCIATED PARTS)

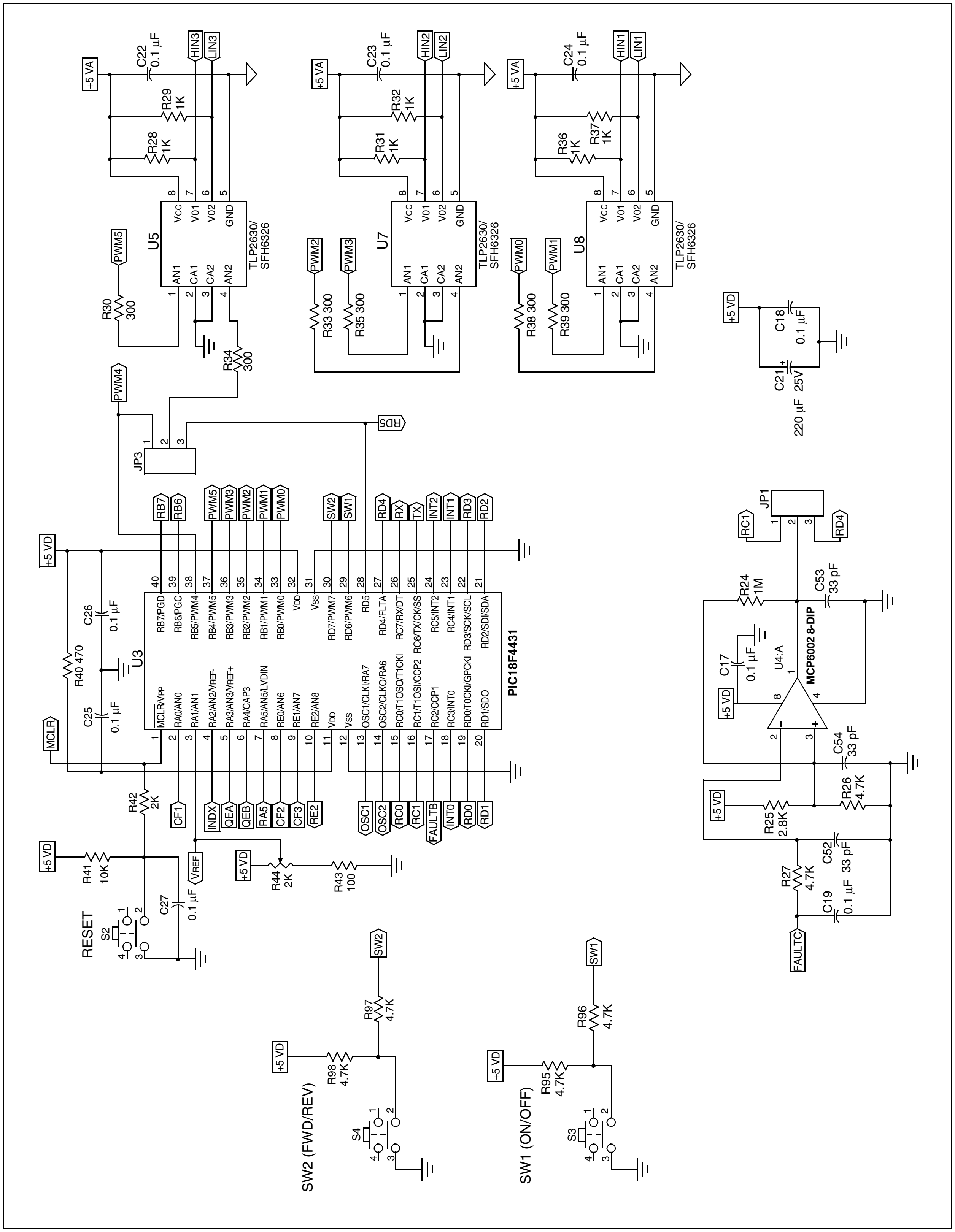


FIGURE A-3: BOARD SCHEMATIC, PART 2 (PIC18F2X31 MICROCONTROLLER SOCKET, USART, CLOCK OSCILLATOR NETWORK AND OPTIONAL LIN INTERFACE)

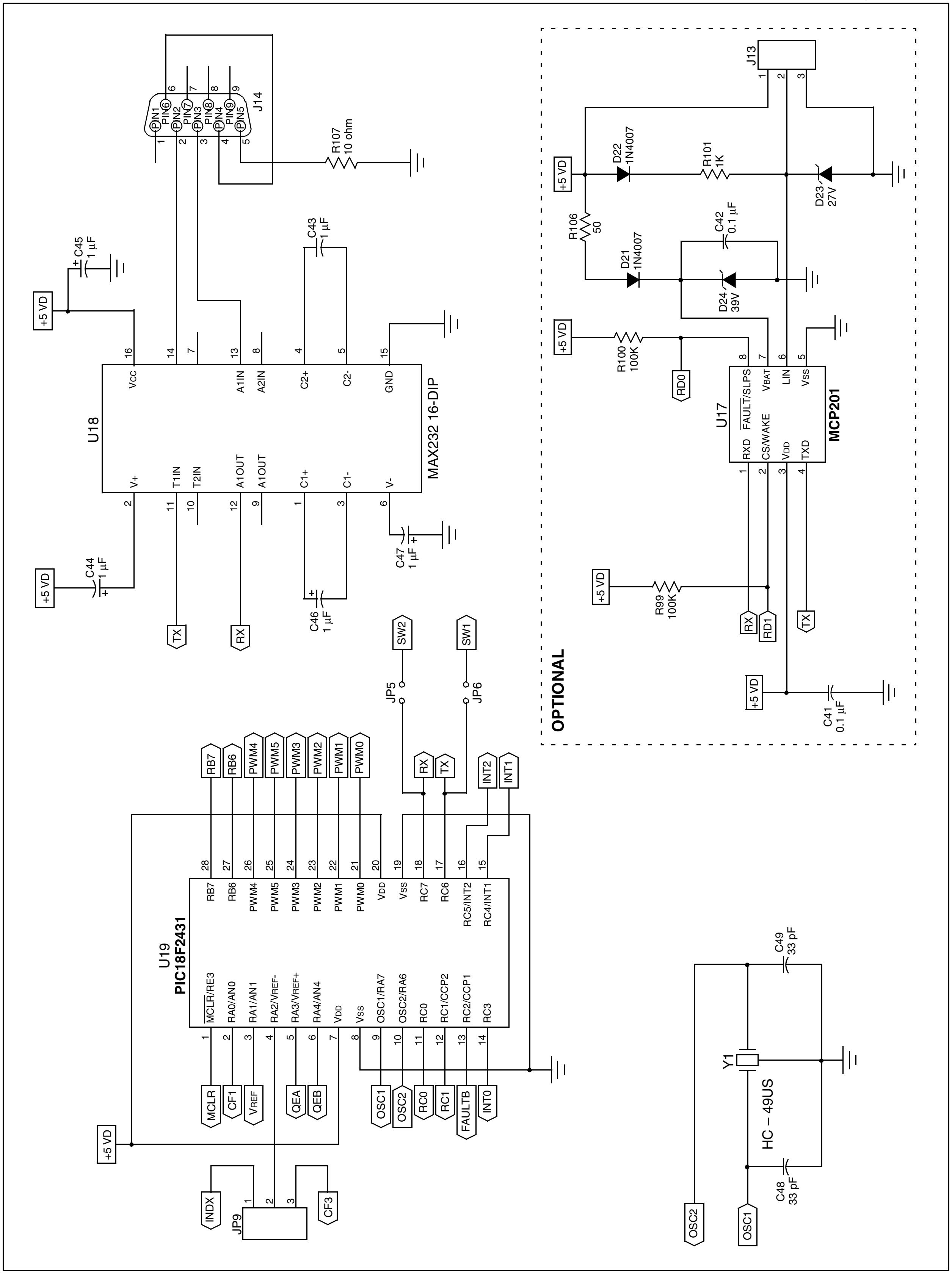


FIGURE A-4: BOARD SCHEMATIC, PART 3 (SENSOR AND MICROCONTROLLER HEADER CONNECTORS, MONITOR LEDS)

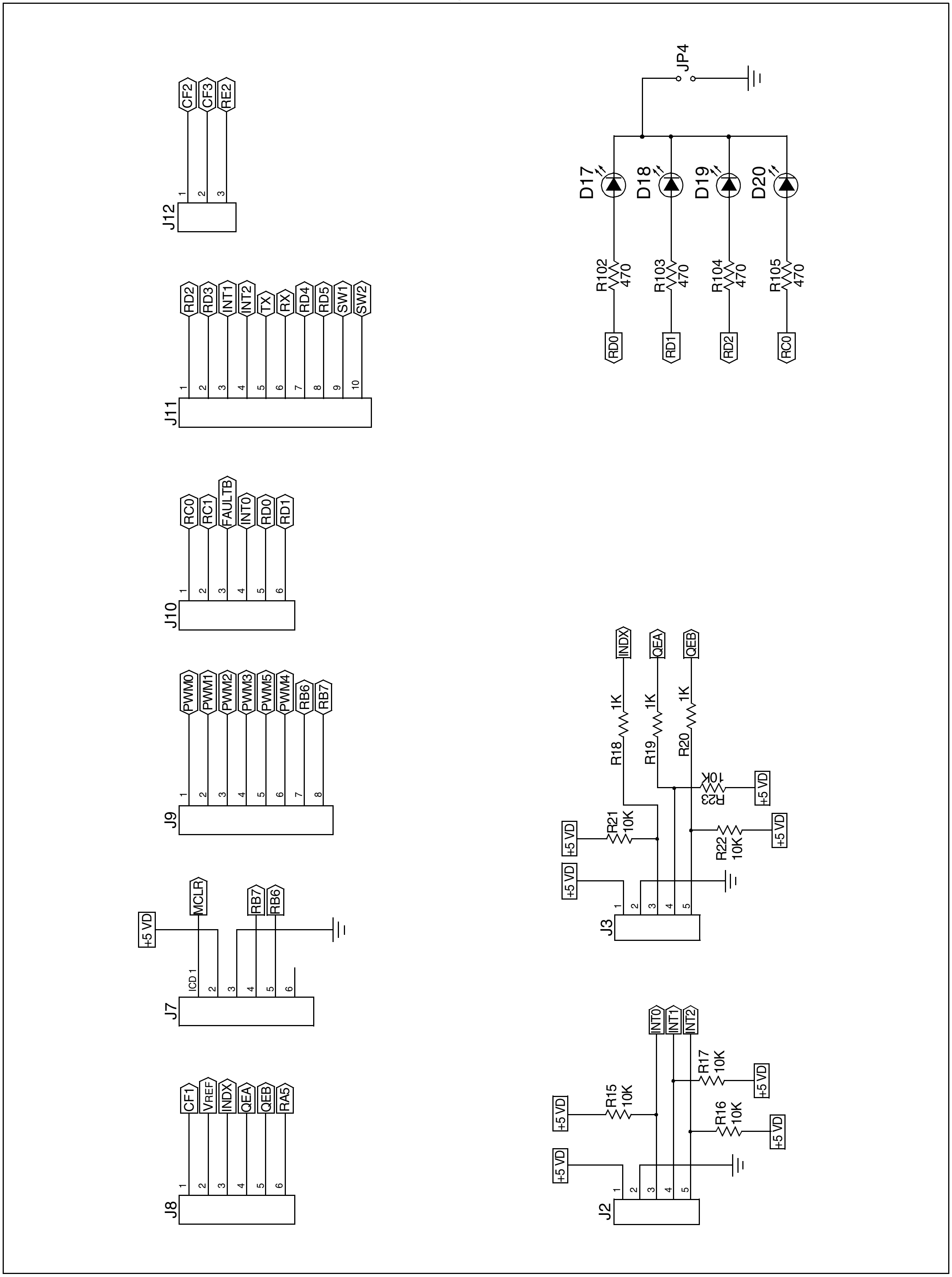


FIGURE A-5: BOARD SCHEMATIC, PART 4 (SIGNAL CONDITIONER FOR SENSORLESS BLDC OPERATION)

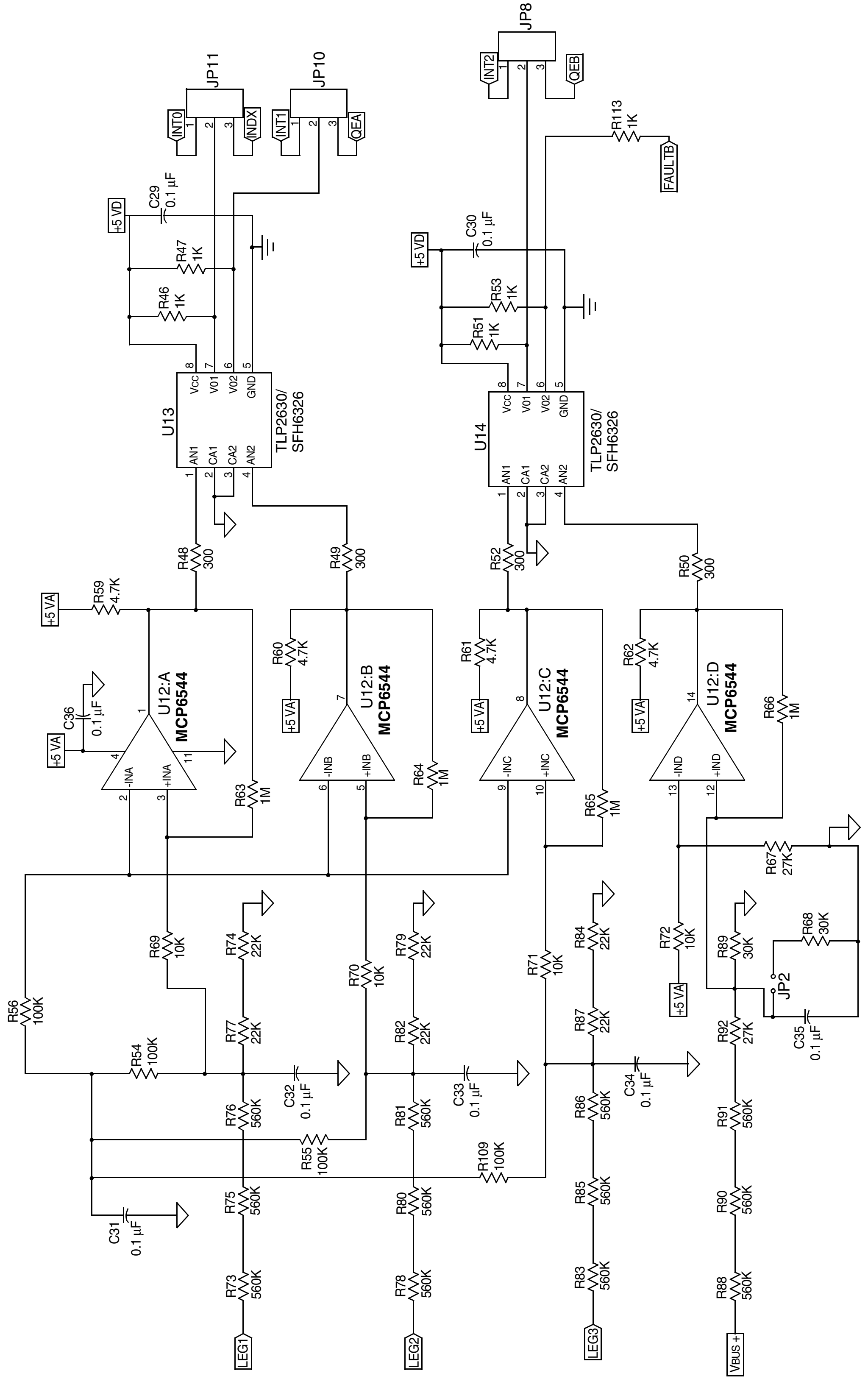


FIGURE A-6: BOARD SCHEMATIC, PART 5 (3-PHASE INVERTER POWER MODULE AND SHUNT CURRENT MEASUREMENT)

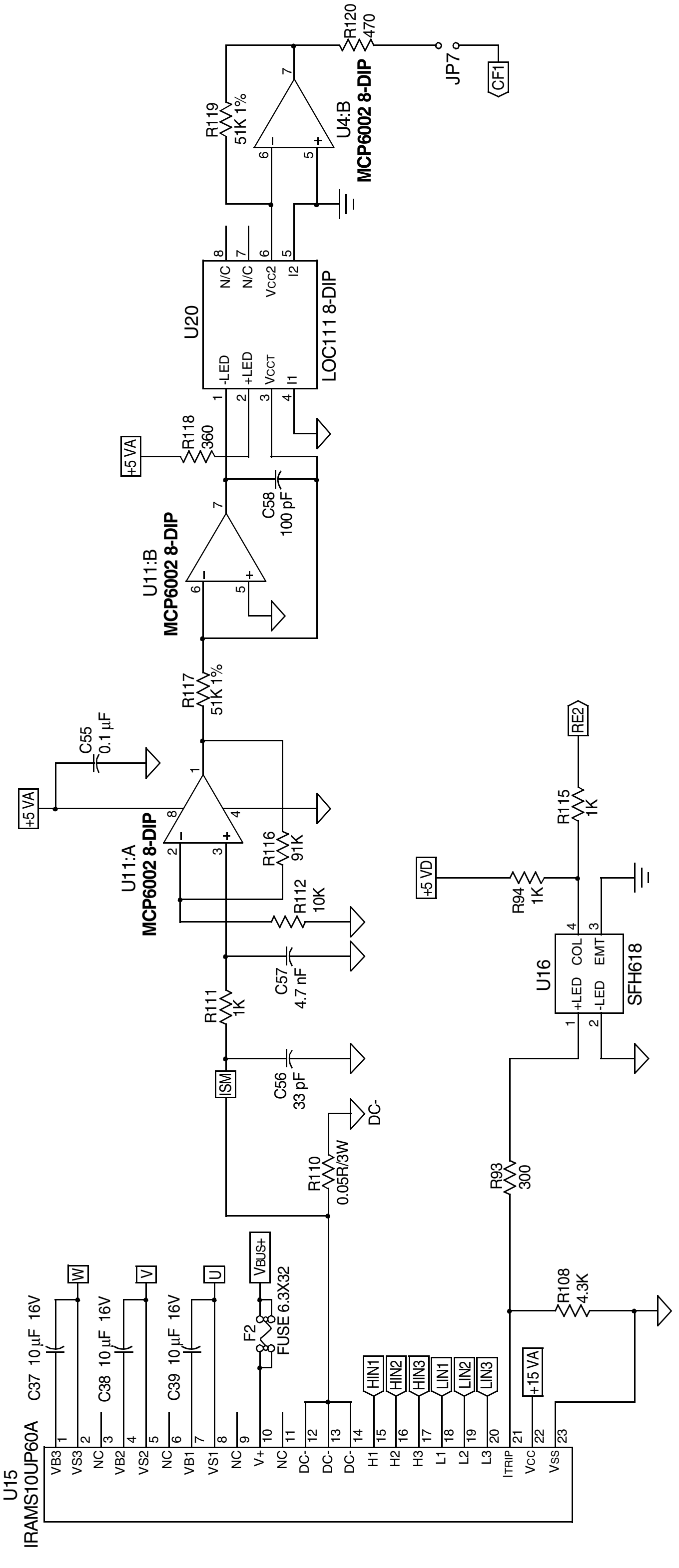


FIGURE A-7: BOARD SCHEMATIC, PART 6 (MOTOR TERMINAL BLOCK AND OPTIONAL CURRENT TRANSDUCER CIRCUITRY)

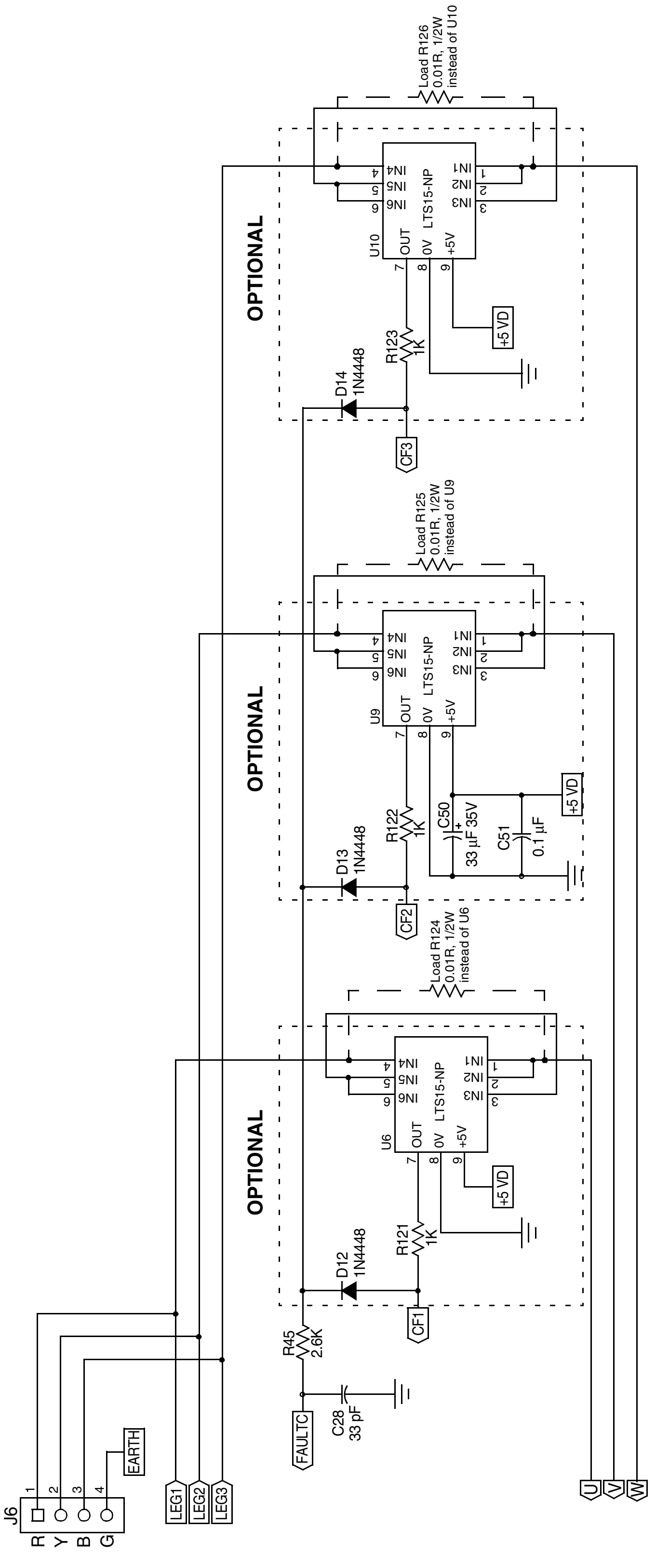


FIGURE A-8: BOARD SCHEMATIC, PART 7 (POWER SUPPLY)

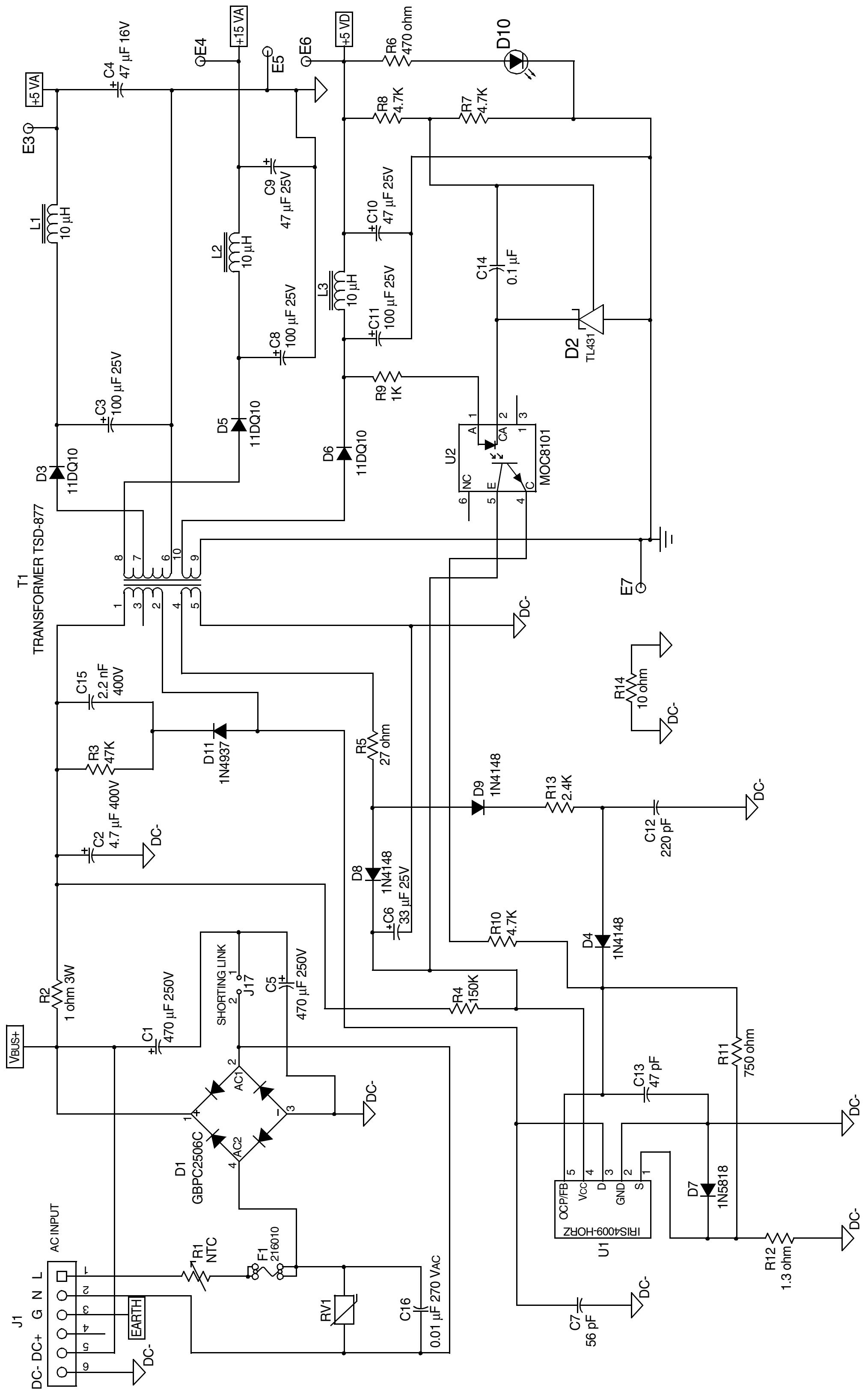


TABLE A-1: SIGNALS USED IN THE PICDEM™ MC SCHEMATIC

Signal Name	Function
+15 VA	Non-isolated DC supply voltage for power components.
+5 VD	Isolated supply voltage for digital components.
CF1, CF2 or CF3	Current feedback signal from designated motor phase winding. CF can also represent total motor current when current transducer measurement is used.
DC-	DC bus return path.
FAULTB	PCPWM Fault signal input (overvoltage).
FAULTC	Fault signal input from comparator (overcurrent).
HIN1, HIN2 or HIN3	Upper leg input for designated phase to 3-phase inverter (isolated signal).
INDX	Index position signal to QEI inputs on microcontroller.
INT0, INT1 or INT2	Hall effect sensor signal to interrupt-on-change inputs on microcontroller.
LEG1, LEG2 or LEG3	Current transducer signal for designated motor winding phase.
LIN1, LIN2 or LIN3	Lower leg input for designated phase to 3-phase inverter (isolated signal).
$\overline{\text{MCLR}}$	Microcontroller hardware Reset.
PWM0 through PWM5	PCPWM waveform outputs from microcontroller.
QEA, QEB	Quadrature encoder sensor signals to QEI inputs on microcontroller.
RAn, RBn, RCn, RDn or REn	Bit n of the designated port of the microcontroller.
RX and TX	RS-232 serial receive and transmit.
SW1, SW2	Push button input from designated switch to microcontroller.
U, V, W	Drive level output from inverter power module to motor.
VBUS+	DC high voltage to inverter power module.
VREF	External reference voltage for overcurrent detect.

APPENDIX B: SINE PWM

The Sine PWM is implemented using a VSI as shown in Figure 2.

At any instant, either the top or the bottom switch of a half bridge is on. Hence, the resultant phase-to-virtual neutral point ‘O’ (V_{RO}, V_{YO} and V_{BO}) can be represented as:

EQUATION B-1:

$$V_{io} = \frac{V_{DC}}{2} \times V_{if} \text{ (where } _i = R, Y, B)$$

V_{if} represents the 3-phase waveforms in space with 120° ($2\pi/3$) phase shift between them. Each phase waveform can be represented as shown in Equation B-2:

EQUATION B-2:

$$\begin{aligned} V_{Rf} &= m \times \sin\theta \\ V_{Yf} &= m \times \sin(\theta + 2\pi/3) \\ V_{Bf} &= m \times \sin(\theta + 4\pi/3) \end{aligned}$$

Substituting Equation B-2 into Equation B-1, we get:

EQUATION B-3:

$$\begin{aligned} V_{RO} &= \frac{V_{DC}}{2}(m \times \sin\theta) \\ V_{YO} &= \frac{V_{DC}}{2}(m \times \sin(\theta + 2\pi/3)) \\ V_{BO} &= \frac{V_{DC}}{2}(m \times \sin(\theta + 4\pi/3)) \end{aligned}$$

The resultant line-to-line output voltage is given as:

EQUATION B-4:

$$\begin{aligned} V_{RY} = V_{RO} - V_{YO} &= \frac{\sqrt{3} \times V_{DC}}{2} \times m \times \sin(\theta + \pi/6) \\ V_{YB} &= \frac{\sqrt{3} \times V_{DC}}{2} \times m \times \sin(\theta + (5\pi)/6) \\ V_{RB} &= \frac{\sqrt{3} \times V_{DC}}{2} \times m \times \sin(\theta + 3\pi/2) \end{aligned}$$

From Equation B-4, it is clear that the maximum line-to-line voltage in the linear operating range is achieved when m = 1.

EQUATION B-5:

$$\text{Maximum line-to-line voltage} = \frac{\sqrt{3} \times V_{DC}}{2}$$

This clearly shows that in Sine PWM, the V_{DC} utilization is less than 90% (~86.6%) in the linear operating range.

APPENDIX C: MOTOR CONTROL MADE EASY

To assist motor control developers, Microchip has developed the PICDEM™ MC Development Board based on the PIC18FXX31. This demo board has all the necessary hardware for a range of motor control, for example, AC Induction motor, BLDC motor and Stepper motor. Various control algorithms have been developed using the demo board to assist users in developing motor control application. Also, a PC-based GUI has been developed for helping users in configuring different motor control parameters and giving real-time capability to monitor the motor speed, the 3-phase currents and temperature.

All source code and the motor control GUI are free to use and can be downloaded from the Microchip web site at:

www.microchip.com.

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights.

Trademarks

The Microchip name and logo, the Microchip logo, Accuron, dsPIC, KEELOQ, microID, MPLAB, PIC, PICmicro, PICSTART, PRO MATE, PowerSmart, rfPIC, and SmartShunt are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

AmpLab, FilterLab, Migratable Memory, MXDEV, MXLAB, PICMASTER, SEEVAL, SmartSensor and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, Application Maestro, dsPICDEM, dsPICDEM.net, dsPICworks, ECAN, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, MPASM, MPLIB, MPLINK, MPSIM, PICkit, PICDEM, PICDEM.net, PICLAB, PICtail, PowerCal, PowerInfo, PowerMate, PowerTool, rfLAB, rfPICDEM, Select Mode, Smart Serial, SmartTel, Total Endurance and WiperLock are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2005, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

Printed on recycled paper.

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2002

Microchip received ISO/TS-16949:2002 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona and Mountain View, California in October 2003. The Company's quality system processes and procedures are for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

MICROCHIP

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://support.microchip.com
Web Address:
www.microchip.com

Atlanta
Alpharetta, GA
Tel: 770-640-0034
Fax: 770-640-0307

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
Kokomo, IN
Tel: 765-864-8360
Fax: 765-864-8387

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

San Jose
Mountain View, CA
Tel: 650-215-1444
Fax: 650-961-0286

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8676-6200
Fax: 86-28-8676-6599

China - Fuzhou
Tel: 86-591-8750-3506
Fax: 86-591-8750-3521

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

China - Shunde
Tel: 86-757-2839-5507
Fax: 86-757-2839-5571

China - Qingdao
Tel: 86-532-502-7355
Fax: 86-532-502-7205

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-2229-0061
Fax: 91-80-2229-0062

India - New Delhi
Tel: 91-11-5160-8631
Fax: 91-11-5160-8632

Japan - Kanagawa
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Kaohsiung
Tel: 886-7-536-4818
Fax: 886-7-536-4803

Taiwan - Taipei
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

Taiwan - Hsinchu
Tel: 886-3-572-9526
Fax: 886-3-572-6459

EUROPE

Austria - Weis
Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

Denmark - Ballerup
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Massy
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Ismaning
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

England - Berkshire
Tel: 44-118-921-5869
Fax: 44-118-921-5820

03/01/05