# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUMMARY</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>CONTINUOUS SPACE VECTOR MODULATION (SVM)</td>
<td>3</td>
</tr>
<tr>
<td>1.1</td>
<td>Space Vector Modulation - What is it?</td>
<td>3</td>
</tr>
<tr>
<td>1.2</td>
<td>Generation of the PWM switching signals</td>
<td>4</td>
</tr>
<tr>
<td>1.3</td>
<td>Inverter capability and reference voltage definition</td>
<td>8</td>
</tr>
<tr>
<td>1.4</td>
<td>Limiting the applied voltage vector</td>
<td>10</td>
</tr>
<tr>
<td>1.5</td>
<td>Determination of the sector</td>
<td>11</td>
</tr>
<tr>
<td>2</td>
<td>THE SVPWM LIBRARY ROUTINES</td>
<td>12</td>
</tr>
<tr>
<td>2.1</td>
<td>Using the SVPWM routines</td>
<td>12</td>
</tr>
<tr>
<td>2.2</td>
<td>Formats of inputs and outputs</td>
<td>13</td>
</tr>
<tr>
<td>2.3</td>
<td>Usage of the DSP registers</td>
<td>13</td>
</tr>
<tr>
<td>2.4</td>
<td>The program code</td>
<td>14</td>
</tr>
<tr>
<td>2.5</td>
<td>Access to the library: the header file</td>
<td>18</td>
</tr>
<tr>
<td>3</td>
<td>SOFTWARE EXAMPLE: TESTING THE CONVERSION ROUTINES</td>
<td>18</td>
</tr>
<tr>
<td>3.1</td>
<td>The main program: main.dsp</td>
<td>18</td>
</tr>
<tr>
<td>3.2</td>
<td>The main include file: main.h</td>
<td>21</td>
</tr>
<tr>
<td>3.3</td>
<td>Example output</td>
<td>21</td>
</tr>
</tbody>
</table>
Summary

The introduction of space vectors, originally only for the purpose of analysis of three-phase machines, has led to the development of an inherently digital modulation method, in contrast to some mere digital approximations to traditional analogue techniques. This technique is nowadays commonly known as space vector modulation (SVM). This application note gives an extensive introduction to its theory and provides routines that allow for easy implementation of SVM into the user’s control algorithm. They are implemented in a library-like module for immediate and intuitive application.

1 Continuous Space Vector Modulation (SVM)

1.1 Space Vector Modulation - What is it?

An inverter is nowadays commonly used in variable speed AC motor drives to produce a variable, three-phase, AC output voltage from a constant DC voltage. Since AC voltage is defined by two characteristics, amplitude and frequency, it is essential to work out a strategy that permits control over both these quantities.

Pulse width modulation (PWM) controls the average output voltage over a sufficiently small period, called switching period, by producing pulses of variable duty-cycle. Here, sufficiently small means that the switching period is small compared to the period of the desired output voltage so that the output voltage may be considered equal to the desired. A classical example is the so-called sine-triangular PWM. A high frequency triangular wave, called the carrier wave, is compared to a sinusoidal signal representing the desired output, called the reference wave. Usually, ordinary signal generators produce these signals. Whenever the carrier wave is less than the reference, a comparator produces a high output signal, which turns the upper transistor in one leg of the inverter on and the lower switch off. In the other case the comparator sets the firing signal low, which turns the lower switch on and the upper switch off. The typical waveforms are shown in Figure 1.

![Figure 1: Sine-triangular pulse width modulation](image)
It may be shown that the magnitude of the fundamental component varies linearly with the fraction

\[
m_{\text{triangular}} = \frac{\text{magnitude of fundamental}}{\text{magnitude of carrier}},
\]

called the modulation index.

A sinusoidal output voltage may therefore be produced, which is proportional to the desired value and whose magnitude is varied by varying the reference's magnitude. The resultant output voltage, of course, contains all the high frequency switching components. It is also an easy task to vary its frequency by changing the frequency of the reference signal.

However, the maximum output voltage amplitude obtainable with this approach is \( \frac{1}{2}V_d \) as \( m_{\text{triangular}} \) may not exceed unity. By applying a rectangular square wave on one inverter leg the fundamental is \( \frac{\sqrt{2}}{2} \cdot V_d = \frac{\sqrt{3}}{2} \cdot V_d \). This mode is known as six-step and entails increased harmonic distortion. It follows that only 78.5% of the inverter's capacity is used with sinusoidal modulation. In addition, a separate modulator has to be used for each of the inverter legs, generating three reference signals forming a balanced three-phase system.

The approach to pulse width modulation that is described in this application note is based on the space vector representation of the voltage in the stationary reference frame, which was defined in a previous note. Given a set of inverter pole voltages \( (V_{A0}, V_{B0}, V_{C0}) \), the vector components \( (V_\alpha, V_\beta) \) in this frame are found by the Forward Clarke transform:

\[
\vec{V} = V_\alpha + jV_\beta = \frac{2}{3} (V_{A0} \tilde{a}^0 + V_{B0} \tilde{a}^1 + V_{C0} \tilde{a}^2), \quad \tilde{a} = e^{j\frac{2\pi}{3}}
\]

It is known that a balanced three-phase set of voltages is represented in the stationary reference frame by a space vector of constant magnitude, equal to the amplitude of the voltages, and rotating with angular speed \( \omega = 2\pi \cdot f_{\text{ref}} \). As will be seen in the next section, the eight possible states of an inverter are represented as two null-vectors and six active-state vectors forming a hexagon. SVM now approximates the rotating reference vector in each switching cycle by switching between the two nearest active-state vectors and the null-vectors. In order to maintain the effective switching frequency of the power devices at a minimum, the sequence of toggling between these vectors is organised such that only one leg is affected in every step.

It may be anticipated that the maximum obtainable output voltage is increased with ordinary SVM up to 90.6% of the inverter capability. It is also a relatively easy task to improve this technique in order to reach full inverter capability.

### 1.2 Generation of the PWM switching signals

With a three-phase voltage source inverter there are eight possible operating states. For instance, in Figure 2a the upper switch of the inverter's pole A is on, whereas the other legs both have the lower switch turned on. Hence the pole voltages are \((\pm V_d, -\pm V_d, -\pm V_d)\) for poles A, B, C respectively. In the following this state is denoted as \((1,0,0)\) and, according to the definition of \((1)\), may be depicted as space vector \(\vec{V}_1 = \frac{\sqrt{3}}{2} V_d e^{j0}\). Repeating these considerations for the other states one finds two null-vectors \(\vec{V}_0\) for state \((0,0,0)\) and \(\vec{V}_7\) for state \((1,1,1)\) and 6 non-null-vectors \(\vec{V}_1\) to \(\vec{V}_6\) shown in Figure 2b.
It is easily shown that the six non-null states, in the following called active states, are represented by space vectors

\[ \vec{V}_k = \frac{2}{3} V_d e^{j(k-1)\frac{\pi}{3}} \]

with \( k=1,...,6 \) (2)

forming a regular hexagon and dividing it into six equal sectors denoted as I, II, III, IV, V, VI in Figure 2b.

A mean space vector \( \vec{V}_{ref} \) over a switching period \( T_S \) can be defined. Assuming that \( T_S \) is sufficiently small, \( \vec{V}_{ref} \) can be considered approximately constant during this interval, and it is this vector which generates the fundamental behaviour of the machine.

The Continuous Space Vector Modulation technique is based on the fact that every vector \( \vec{V}_{ref} \) inside the dashed hexagon can be expressed as a weighted average combination of the two adjacent active space vectors and the null-state vectors 0 and 7. Therefore, in each cycle imposing the desired reference vector may be achieved by switching between these four inverter states. Looking at Figure 2b one finds that, assuming \( \vec{V}_{ref} \) to be laying in sector \( k \), the adjacent active vectors are \( \vec{V}_k \) and \( \vec{V}_{k+1} \), where \( k+1 \) is set to 1 for \( k=6 \). In order to obtain optimum harmonic performance and the minimum switching frequency for each of the power devices, the state sequence is arranged such that the transition from one state to the next is performed by switching only one inverter leg. This condition is met if the sequence begins with one zero-state and the inverter poles are toggled until the other null-state is reached. To complete the cycle the sequence is reversed, ending with the first zero-state. If, for instance, the reference vector sits in sector 1, the state sequence has to be ...0127210... whereas in sector 4 it is ...0547450... The central part of the space vector modulation strategy is the computation of both the active and zero-state times for each modulation cycle. These may be calculated by equating the applied average voltage to the desired value.

Figure 2: a) Configuration of the switches in the state \( V1=(1,0,0) \), b) Representation of the inverter states in the stationary reference frame
In the following, $T_k$ denotes half the on-time of vector $\vec{V}_k$. $T_0$ is half the null-state time. Hence, the on-times are evaluated by the following equations:

$$\int_0^{\frac{r_s}{T}} \vec{V}_{ref} \, dt = \int_0^{\frac{r_n}{T}} \vec{V}_0 \, dt + \int_{\frac{r_n}{T}}^{\frac{r_n+T_s}{T}} \vec{V}_k \, dt + \int_{\frac{r_n+T_s}{T}}^{\frac{r_n+T_s+T_k}{T}} \vec{V}_7 \, dt$$

$$T_0 + T_k + T_{k+1} = \frac{T_s}{2}$$  \hspace{1cm} (4)

Taking into account that $\vec{V}_0 = \vec{V}_7 \equiv \vec{0}$ and that $\vec{V}_{ref}$ is assumed constant and the fact that $\vec{V}_k$, $\vec{V}_{k+1}$ are constant vectors, equation (3) reduces to

$$\vec{V}_{ref} \cdot \frac{T_s}{2} = \vec{V}_k \cdot T_k + \vec{V}_{k+1} \cdot T_{k+1}$$  \hspace{1cm} (5)

Splitting this vectorial equation into its real and imaginary components, from (2) follows that:

$$\left( \begin{array}{c} V_x \\ V_y \end{array} \right) \cdot \frac{T_s}{2} = 2 \left( \begin{array}{c} V_x \\ V_y \end{array} \right) \left( \begin{array}{c} \cos \frac{(k-1)\pi}{3} \\ \sin \frac{(k-1)\pi}{3} \end{array} \right) + T_{k+1} \left( \begin{array}{c} \cos \frac{k\pi}{3} \\ \sin \frac{k\pi}{3} \end{array} \right) = 2 \left( \begin{array}{c} V_x \\ V_y \end{array} \right) \left( \begin{array}{c} \cos \frac{(k-1)\pi}{3} \\ \sin \frac{(k-1)\pi}{3} \end{array} \right)$$

where $k$ is to be determined from the argument of the reference vector

$$\alpha = \arg \left( \begin{array}{c} V_x \\ V_y \end{array} \right)$$  \hspace{1cm} (7)

such that

$$\frac{(k-1)\pi}{3} \leq \arg \left( \begin{array}{c} V_x \\ V_y \end{array} \right) \leq \frac{k\pi}{3}.$$  \hspace{1cm} (8)

The request of minimal number of commutations per cycle is met if in every odd sector the sequence of applied vectors is $\vec{V}_0 \vec{V}_k \vec{V}_{k+1} \vec{V}_7 \vec{V}_{k+1} \vec{V}_k \vec{V}_0$, whereas in even sector the active vectors are applied in the reversed order, hence $\vec{V}_0 \vec{V}_{k+1} \vec{V}_k \vec{V}_7 \vec{V}_k \vec{V}_{k+1} \vec{V}_0$.

Solving system (6), one finds:

$$\left( \begin{array}{c} T_k \\ T_{k+1} \end{array} \right) = \frac{\sqrt{3}}{2 \left( \begin{array}{c} V_x \\ V_y \end{array} \right)} \left( \begin{array}{c} \sin \frac{k\pi}{3} \\ -\sin \frac{(k-1)\pi}{3} \end{array} \right) \left( \begin{array}{c} \cos \frac{k\pi}{3} \\ -\cos \frac{(k-1)\pi}{3} \end{array} \right) \cdot \left( \begin{array}{c} V_x \\ V_y \end{array} \right)$$  \hspace{1cm} (9)

The total null-state time $T_0$ may be divided in an arbitrary fashion between the two zero states. A common solution is to divide $T_0$ equally between the two null-state vectors $\vec{V}_0$ and $\vec{V}_7$. From (4), $T_0$ results as

$$T_0 = \frac{T_s}{2} - (T_k + T_{k+1})$$  \hspace{1cm} (10)

As an example for the switching scheme, in sector 1 one finds:
Assuming that it is desired to produce a balanced system of sinusoidal phase voltages, it is known that the corresponding space vector locus is circular. Imposing 
\[ \bar{V}_{\text{Re}f} = |\bar{V}_{\text{Re}f}| e^{j\omega t} = |\bar{V}_{\text{Re}f}| (\cos(\omega t) + j \sin(\omega t)), \]
where \(|\bar{V}_{\text{Re}f}|\) is the magnitude and \(\omega\) is the angular frequency of the desired phase voltages, it follows from (9) that

\[ \begin{pmatrix} T_k \\ T_{k+1} \end{pmatrix} = \frac{\sqrt{3}}{2} \frac{|\bar{V}_{\text{Re}f}|}{V_d} T_s \begin{bmatrix} \sin \frac{k \pi}{3} & -\cos \frac{k \pi}{3} \\ -\sin \frac{(k-1) \pi}{3} & \cos \frac{(k-1) \pi}{3} \end{bmatrix} \begin{bmatrix} \cos \omega t \\ \sin \omega t \end{bmatrix} \] (11)

While \(0 \leq \omega t \leq \frac{\pi}{3}\) the reference vector lies in sectors 1 and equation (11) reduces to

\[ \begin{pmatrix} T_1 \\ T_2 \end{pmatrix} = \frac{\sqrt{3}}{2} \frac{|\bar{V}_{\text{Re}f}|}{V_d} T_s \begin{bmatrix} \sin \left(\frac{\pi}{3} - \omega t\right) \\ \sin \omega t \end{bmatrix} \] (12)

The mean values of the inverter pole voltages averaged over one switching cycle are thus

\[ V_{a0}(\omega t) = \frac{\sqrt{3}}{2 T_s} \left(-\frac{\pi}{2} + T_1 + T_2 + T_0 + T_2 + T_0 - \frac{\pi}{2}\right) = \frac{\sqrt{3}}{2} \frac{|\bar{V}_{\text{Re}f}|}{V_d} V_d \cos(\omega t - \frac{\pi}{6}) \]

\[ V_{b0}(\omega t) = \frac{\sqrt{3}}{2 T_s} \left(-\frac{\pi}{2} - T_1 + T_2 + T_0 + T_2 - T_1 - \frac{\pi}{2}\right) = \frac{\sqrt{3}}{2} \frac{|\bar{V}_{\text{Re}f}|}{V_d} V_d \sin(\omega t - \frac{\pi}{6}) \] (13)

\[ V_{c0}(\omega t) = -V_{a0}(\omega t) \]

Analogous, solving equation (11) for the other sector one finds
\[
V_{A0}(\omega t) = \frac{\sqrt{3}}{2} |\vec{V}_{Re, f}| \cos(\omega t - \frac{\pi}{6}) \quad 0 \leq \omega t \leq \frac{\pi}{6}
\]
\[
= \frac{3}{2} |\vec{V}_{Re, f}| \cos \omega t \quad \frac{\pi}{6} \leq \omega t \leq \frac{2\pi}{3}
\]
\[
= \frac{\sqrt{3}}{2} |\vec{V}_{Re, f}| \cos(\omega t - \frac{\pi}{3}) \quad \frac{2\pi}{3} \leq \omega t \leq \pi
\]
\[
= \frac{3}{2} |\vec{V}_{Re, f}| \cos \omega t \quad \pi \leq \omega t \leq \frac{4\pi}{3}
\]
\[
= \frac{\sqrt{3}}{2} |\vec{V}_{Re, f}| \cos(\omega t + \frac{\pi}{6}) \quad \frac{4\pi}{3} \leq \omega t \leq \frac{5\pi}{3}
\]
\[
= \frac{3}{2} |\vec{V}_{Re, f}| \cos \omega t \quad \frac{5\pi}{3} \leq \omega t \leq 2\pi
\]

\begin{align*}
V_{B0}(\omega t) &= V_{A0}(\omega t - \frac{2\pi}{3}) \\
V_{C0}(\omega t) &= V_{A0}(\omega t - \frac{4\pi}{3})
\end{align*}

\[V_{A0}(\omega t)\] is shown in Figure 4.

\[\begin{array}{c}
\text{Figure 4: Pole voltage for the ideal SV-modulation}
\end{array}\]

Evaluating the line-to-line voltages from (14) leads to
\[
V_{AB}(\omega t) = V_{A0}(\omega t) - V_{B0}(\omega t) = \sqrt{3}|\vec{V}_{Re, f}| \sin(\omega t + \frac{\pi}{4})
\]
\[
V_{BC}(\omega t) = V_{A0}(\omega t - \frac{2\pi}{3}) \quad \text{for } 0 \leq \omega t \leq 2\pi
\]
\[
V_{CA}(\omega t) = V_{AB}(\omega t - \frac{4\pi}{3})
\]

Therefore it turns out that they are sinusoidal, as expected.

### 1.3 Inverter capability and reference voltage definition

The hexagon of Figure 2b represents the range of realisable voltage space vectors. Using the space vector modulation process it is possible to realise any arbitrary voltage space vector that lies within this hexagon. The maximum fundamental phase voltage that may be produced by the inverter for a given dc link voltage occurs under six-step operation. The resultant phase voltage developed by the inverter is shown
in Figure 5, where the six different voltage levels, corresponding to operation at each of the active inverter states, are clearly seen.

![Diagram of six-step voltage waveform](image)

**Figure 5: Resultant inverter phase voltage and corresponding fundamental component for six-step operation.**

The fundamental component of the six-step voltage waveform is also illustrated in Figure 5. From the Fourier analysis, the fundamental voltage magnitude is given by:

\[ V_{\text{max sixstep}} = \frac{4 V_d}{\pi 2}. \]  

(16)

This voltage level is achieved only at the expense of significant low frequency distortion. It was already mentioned that for conventional sinusoidal modulation the maximum achievable fundamental voltage is:

\[ V_{\text{max sin pwm}} = \frac{V_d}{2}. \]  

(17)

so that only 78.5% of the available inverter capacity is used.

If the space vector modulator is required to produce a balanced three-phase system of voltages of magnitude \( V_{\text{ref}} \) and frequency \( \omega \), given by:

\[ V_{AN} = V_{\text{ref}} \cos(\omega t) \]
\[ V_{BN} = V_{\text{ref}} \cos(\omega t - \frac{2\pi}{3}) \]
\[ V_{CN} = V_{\text{ref}} \cos(\omega t - \frac{4\pi}{3}) \]  

(18)

the corresponding reference voltage space vector is given by:

\[ \vec{V}_{\text{ref}} = V_{\text{ref}} [\cos(\omega t) + j \sin(\omega t)] = V_{\text{ref}} e^{j\omega t} \]  

(19)

It is found useful to define the modulation index as the ratio of the desired peak fundamental magnitude to the maximal fundamental output in six-step mode:

\[ m = \frac{|\vec{V}_{\text{ref}}|}{V_{\text{max sixstep}}} = \frac{\pi}{2} \frac{V_{\text{ref}}}{V_d} \]  

(20)
Therefore, the reference space vector describes a circular trajectory of radius $V_{\text{ref}}$ at an angular velocity $\omega$ in the complex plane. Clearly, the largest possible voltage magnitude that may be achieved using the space vector modulation strategy corresponds to the radius of the largest circle that can be inscribed within the hexagon of Figure 2b. This circle is tangential to the midpoints of the lines connecting the ends of the active state vectors. The maximum fundamental phase voltage that may be achieved is:

$$\left| V_{\text{Ref}} \right|_{\text{max}} = \frac{\sqrt{3}}{2} V_d \sqrt{3} = \frac{1}{\sqrt{3}} V_d$$

(21)

Following the definition of modulation index introduced in above, the corresponding maximum modulation index is given by

$$m_{\text{max,cont}} = \frac{\left| V_{\text{Ref}} \right|_{\text{max}}}{V_{\text{max,sixstep}}} = \frac{\sqrt{3}/2 \cdot V_d}{2\sqrt{3}/2} = \frac{\pi}{2\sqrt{3}} = 0.906$$

(22)

As seen, the maximum peak fundamental magnitude that may be obtained with the SVM technique is about 90.6 % of the inverter capacity. This represents a 15% increase in the maximum voltage compared with conventional sinusoidal modulation.

The maximum output line voltage magnitude is, from equation (15):

$$\hat{V}_{AB} = \sqrt{3} \left| V_{\text{Ref}} \right|_{\text{max}} = V_d$$

(23)

With the definition of modulation index the computation of the inverter switching times does not require the knowledge of the adopted DC-link voltage but depends only on the desired modulation index. This may be seen observing that $\left| V_{\text{Ref}} \right| = m \cdot V_{\text{max,sixstep}} = m \cdot \sqrt{3}/2 \cdot V_d$ and substituting this equation in (41), leading to:

$$\begin{bmatrix} T_k \\ T_{k+1} \end{bmatrix} = m \cdot \frac{\sqrt{3}}{\pi} T_s \cdot \begin{bmatrix} \sin k\pi/3 & -\cos k\pi/3 \\ -\sin (k-1)\pi/3 & \cos (k-1)\pi/3 \end{bmatrix} \begin{bmatrix} \cos \omega T_s \\ \sin \omega T_s \end{bmatrix}$$

(24)

It is possible to increase this modulation index up to operation in six-step mode ($m=1$) by operating in discontinuous SVM introducing additional distortion of the line currents. This is achieved by directly modifying the magnitude and/or angle of the reference vector. This issue is not treated in this application note. However, the adopted scaling choice of $m$ leaves the options to the user of easily extending the inverter's range of operation to full modulation.

### 1.4 Limiting the applied voltage vector

In the previous section it was found that the maximum modulation index obtainable with continuous space vector modulation is $m_{\text{max,cont}} = 0.906$. At this point the reference vector describes the maximum circular trajectory which may be inscribed within the hexagon. The circle and the hexagon are tangential at exactly one position for each sector, at $\alpha = \frac{\pi}{6} + \frac{(k-1)}{3} \frac{\pi}{6}$, where $k$ is the sector's number. However, even at this point relatively large areas within the hexagon remain unused. These areas are largest in the region of the active-state vectors, and represent unused inverter capacity. As the reference vector's magnitude increases further, the locus passes outside the hexagon in the neighbourhood of these points, whereas it still lies inside it in the zones near the active switching states. Thus, the possibility of a higher modulation index is still available in these regions, i.e. during periodic time intervals of limited duration.
When the desired trajectory passes outside the hexagon, the time average equations give negative (and therefore meaningless) values for the on-duration $T_0$ of the zero-state vectors. It is possible to overcome this problem by simply 'rescaling' the active state times $T_k$ and $T_{k+1}$ to:

$$
\begin{align*}
T'_k &= \frac{T_S}{2} \frac{T_k}{T_k + T_{k+1}} \\
T'_{k+1} &= \frac{T_S}{2} \frac{T_{k+1}}{T_k + T_{k+1}}
\end{align*}
$$

so that $T'_k + T'_{k+1} = \frac{1}{2} T_S$ and $T_0 = 0$. It is clear from equation (5) that the magnitude of the produced vector is reduced by the same factor.

$$
\vec{V}'_{\text{ref}} = \frac{2}{T_S} \left( \vec{V}_k \cdot T_k + \vec{V}_{k+1} \cdot T'_{k+1} \right) = \frac{2}{T_S} \left( \vec{V}_k \cdot T_k + \vec{V}_{k+1} \cdot T_{k+1} \right) \left( \frac{T_S}{2(T_k + T_{k+1})} \right) = \frac{T_S}{2(T_k + T_{k+1})} \cdot \vec{V}_\text{ref}
$$

The effective locus now follows the hexagon line causing a reduction in the output fundamental voltage.

This may be shown, for instance, in sector I by observing that:

$$
\vec{V} = \frac{2}{T_S} \cdot \frac{T_S}{2} \left( \vec{V}_1 \cdot T_1 + \vec{V}_2 \cdot T'_{2} \right)
$$

Hence, from equation (12):

$$
\vec{V} = \frac{2}{3} V_d \cdot \frac{1}{\sin(\frac{\pi}{3} + \alpha)} \left( \begin{array}{c}
\sin(\frac{\pi}{3} - \alpha) \\
\sin(\frac{\pi}{3} + \alpha)
\end{array} \right) = \frac{2}{3} V_d \cdot \frac{1}{\sin(\frac{\pi}{3} + \alpha)} \left( \begin{array}{c}
\sqrt{3} / 2 \sin \alpha \\
\sqrt{3} / 2 \cos \alpha
\end{array} \right) = \frac{V_d}{\sqrt{3}} \cdot \frac{1}{\sin(\frac{\pi}{3} + \alpha)} \left( \begin{array}{c}
\cos \alpha \\
\sin \alpha
\end{array} \right)
$$

This is the Cartesian representation of a vector with magnitude

$$
|\vec{V}| = \frac{V_d}{\sqrt{3}} \cdot \frac{1}{\sin(\frac{\pi}{3} + \alpha)}
$$

and argument $\alpha$ and it is easy to show that it describes the first segment of the hexagon for $\alpha$ varying between 0 and $\frac{\pi}{3}$.

Although the rescaling of the on-state times solves the problem of the obtained meaningless switching instants, it also causes a reduction of the output voltage with respect to the desired value.

### 1.5 Determination of the sector

Given an arbitrary reference vector $V_\alpha + jV_\beta$, the phase angle can be evaluated by

$$
\Theta = \arctan \left( \frac{V_\beta}{V_\alpha} \right) \quad \Theta \in [0, 2\pi]
$$

© Analog Devices Inc., January 2000 Page 11 of 22
However, this approach involves two time-intensive operations arctan(x) and the division and, in addition, imposes the condition on $V_\alpha$ to be not equal to zero. These difficulties may be overcome by observing that each quadrant is shared by two sectors. The quadrant in which the reference vector sits is easily determined by examining the sign of its real and imaginary components. Having determined the quadrant, for instance 1, the vector lies in sector I if

$$\arctan \left( \frac{V_\beta}{V_\alpha} \right) \leq \frac{\pi}{3}$$

hence

$$V_\beta \leq V_\alpha \tan \left( \frac{\pi}{3} \right) = \sqrt{3} \cdot V_\alpha,$$  \hspace{1cm} (32)

otherwise the vector lies in sector II. In our case this condition reduces to $\sin \omega t \leq \sqrt{3} \cdot \cos \omega t$. This method needs only a multiplication and a few comparisons.

Analogous, in quadrant 2 (with $V_\alpha < 0, V_\beta > 0$) one finds:

sector III if $$\arctan \left( \frac{V_\beta}{-V_\alpha} \right) \leq \frac{\pi}{3}$$

or $$V_\beta \leq \sqrt{3} \cdot (-V_\alpha);$$ sector II otherwise.

Similar considerations apply to quadrant 3 and 4.

2 The SVPWM Library Routines

2.1 Using the SVPWM routines

The routines are developed as an easy-to-use library, which has to be linked to the user’s application. The library consists of two files. The file “svpwm.dsp” contains the assembly code for the subroutines. This package has to be compiled and can then be linked to an application. The user simply has to include the header file “svpwm.h”, which provides function-like calls to the routines. The example file in Section 3 will demonstrate the usage of all the conversion routines.

The following table summarises the set of macros defined in this library.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calculate On-times</td>
<td>SVM_Calc_Ontimes ($V_{\alpha \beta}$, OnTime_vector);</td>
</tr>
<tr>
<td>Calculate Duty-cycles</td>
<td>SVM_Calc_Dutycycles (OnTime_vector, Dutycycles_vector);</td>
</tr>
<tr>
<td>Update PWM Block</td>
<td>SVM_Update_Dutycycles (Dutycycles_vector);</td>
</tr>
</tbody>
</table>

Table 1 Implemented routines

The input vector $V_{\alpha \beta}$ consists of two elements holding the components of the stator quantity vector. In the case of a necessity to re-scale the on-times, this vector will become also an output, since the routine will reflect the reduction of it’s magnitude. The OnTime_vector contains the current sector and the on-times of both the active vectors and zero-state vectors after a call to SVM_Calc_Ontimes. Finally, the Dutycycles_vector holds the duty-cycles for each phase after a call to SVM_Calc_Dutycycles. They may
be loaded into the PWM block by a call to SVM_Update_Dutycycles. The format of inputs and outputs are explained in more detail in the next section.

The reason for splitting the SV modulation process into three separate subroutines is to maintain the maximum of flexibility. Having access to the on-times and duty-cycles leaves the options of compensating for switching dead-times, inverter on-state voltages and/or DC-link voltage ripple in an optimal fashion.

The routines do not require any configuration constants from the main include-file “main.h” that comes with every application note. For more information about the general structure of the application notes and including libraries into user applications refer to the Library Documentation File. Section 5 shows an example of usage of this library. In the following sections each routine is explained in detail with the relevant segments of code which is found in either “svpwm.h” or “svpwmdsp”. For more information see the comments in those files.

2.2 Formats of inputs and outputs

As already mentioned, the SVM module expects the input to be a vector in the stationary reference frame $\vec{V}_{\text{ref}} = V_\alpha + jV_\beta$. Since all quantities in the DSP core are scaled in the 1.15 fixed-point format, it becomes natural to normalise the reference vector to be represented in this format. If $V_{\text{ref}} = \left| \vec{V}_{\text{ref}} \right| = \sqrt{V_\alpha^2 + V_\beta^2}$ denotes the magnitude of the reference vector, it is clear that scaling the reference as in (20) leads to a reference vector with magnitude equal to the modulation index and comprised in the range from zero to one (90.6% for continuous SVM).

In conclusion, if the input vector is of constant magnitude $m$ and rotates at constant angular speed, the amplitude $\hat{V}_{xN}$ of the produced three-phase system is related to $m$ such that

$$\hat{V}_{AN,BN,CN} = \frac{2}{\pi} V_d \cdot m$$

(33)

where $m$ is in the range of 0 (0x0000) to 0.906 (0x7415) for continuous SVM (reference vector inside the hexagon at all times) and 0.906 (0x7416) to 1 (0x7FFF) when trajectories that are partially outside the hexagon are demanded. In the latter case, $V_\alpha$ and $V_\beta$ will be replaced by the scaled values after executing SVM_Calc_OnTimes.

The output OnTime_vector stores information about the sector (1 to 6) that the reference vector lays in, the on-time of the first vector to be applied ($V_k$ for odd sectors, $V_{k+1}$ for even sectors), the on-time of the second vector to be applied ($V_{k+1}$ for odd sectors, $V_k$ for even sectors) and finally the total zero state time $T_0$.

The output DutyCycles_vector holds the duty cycles value for channel A, B and C respectively.

On-times and duty-cycles are expressed as increments of the PWM timer, therefore in the range of 0 to the content of the PWM period register PWMTM.

2.3 Usage of the DSP registers

The macros listed in Table 1 are based on two subroutines, namely SVM_OnTimes_ and SVM_DutyCycles_. They are described in detail in the next section. The following table gives an overview of the usage of the core DSP registers, when the macros are called.
2.4 The program code

The following code contained in the file “svpwm.dsp” defines the two routines mentioned in the previous section.

A vector of sine and cosine values at multiples of 60° is defined which serves for the computation of equation (24). The values actually include also the common factor in that equation of \( \frac{\sqrt{3}}{\pi} \) in order to save one multiplication. As they are stored in program memory, the format is the usual 1.15 format extended to the right with zeros in order to obtain a 24-bit value.

Table 2 Usage of DSP core registers for the macros

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Input and modified DAG registers</th>
<th>Output</th>
<th>Modified core registers</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM_Calc_Ontimes</td>
<td>I1 = ^Reference</td>
<td>N/A</td>
<td>ALL except se</td>
</tr>
<tr>
<td></td>
<td>I0 = ^Ontimes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I6, I7</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>M0, M1, M6, M6 = 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>L0, L1, L6, L7 = 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVM_Calc_DutyCycles</td>
<td>I1 = ^Ontimes</td>
<td>N/A</td>
<td>ax0, ay0, ay1, ar, af,</td>
</tr>
<tr>
<td></td>
<td>I2 = ^Dutycycles</td>
<td></td>
<td>mr1, mr2,</td>
</tr>
<tr>
<td></td>
<td>I4</td>
<td></td>
<td>sr0, sr1</td>
</tr>
<tr>
<td></td>
<td>M1, M2 = 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>L1, L2 = 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVM_Update_DutyCycles</td>
<td>I1 = ^Dutycycles</td>
<td>PWMCHA,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>M1 = 1</td>
<td>PWMCHB,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>L1 = 0</td>
<td>PWMCHC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ar</td>
<td></td>
</tr>
</tbody>
</table>

2 ^vector stands for ‘address of vector’

3 N/A: The output values are stored in the output vector in Data Memory. No DSP core register is used.
The first task of the SVM routine is to determine the current sector, which the reference vector lays in. The following piece of code achieves this through the algorithm presented in section 1.5. After executing this piece, ax0 contains the number of the sector (0x0001 to 0x0006).

```
SVPWM_Ontimes_:
    ax1= DM(I0,M0);
    ay1= DM(I0,M0);
    my0=One_Over_Sqrt3;

    ar = pass ay1;
    if lt jump Sector456;
       af=pass ax1, ar = ax1;
       if lt jump Sector32;
          mx0=ay1;
          mr=mx0*my0 (rnd);
          none=mr1 -af;
          if gt jump Sector2;
             Sector1: ax0= 1;
             jump Solve_equation;

           Sector32:
              af=abs ar;
              if AV af=af-1;
              mx0=ay1;
              mr=mx0*my0 (rnd);
              none=mr1 - af;
              if gt jump Sector2;
                 Sector3: ax0= 3;
                 jump Solve_equation;
                 Sector2: ax0= 2;
                 jump Solve_equation;

           Sector456:
              af=pass ax1, ar = ax1;
              if lt jump Sector45;
                 ar = pass ay1;
                 ar=abs ar;
                 if AV ar=ar-1;
                 mx=mr*my0 (rnd);
                 none=mr1 - af;
                 if gt jump Sector5;
                    Sector6: ax0= 6;
                    jump Solve_equation;

           Sector45:
              ar=abs ar;
              if AV ar=ar-1;
              af=pass ar;

                 ar = pass ay1;
                 ar=abs ar;
                 if AV ar=ar-1;

                 mx=ar*my0 (rnd);
                 none=mr1 - af;
                 if gt jump Sector5;
                    Sector4: ax0= 4;
                    jump Solve_equation;

           Sector5: ax0= 5;
                    jump Solve_equation;
```

Next, equation [24] is effectively computed. The core is a sequence of MAC operations, which implement the matrix multiplication. The comments show what values are calculated. Then the values are multiplied by the PWM period. However, since the PWMTM register is represents half of the PWM period, the obtained value has to be multiplied by two (shifted to the left). Note that it is not possible to incorporate the factor 2 into the vectors of constant since some of them would be greater than one. Also, depending on whether the sector is
even or odd, the values for $Ta$ and $Tb$ are organised into $sr$ such that the lower word contains the on-time for the first vector to be applied and the higher word the on-time for the second vector.

```
Solve_equation:
  DM(I1,M1)=ax0;  { store no. of sector }
  ay0= ~SineVector - 1;
  ar = ax0 + ay0;
  I6 = ar;
  M6 = 1;
  L6 = ~SineVector;
  ay0= ~CosineVector - 1;
  ar = ax0 + ay0;
  I7 = ar;
  M7 = 1;
  L7 = ~CosineVector;
  mx0 = ax1;
  mx1 = ay1;
  my1 = DM(PWMTM);  {Ts/2 }
    my0 = PM(I7,M7);  {cos k-1}
  mr = mx1 * my0 (SS), my0 = PM(I6,M6);  {sin k-1}
  mr = mr1 * my1 (SS), my0 = PM(I6,M6);  {sin k }
  si = mr1;
  mr = mx0 * my0 (SS), my0 = PM(I7,M7);  {cos k }
  mr = mr - mx1*my0 (rnd);
  mr = mr1 * my1 (SS);  { Ta/2 for odd sectors }
  astat = ax0;
  if EQ jump Sector_odd;

Sector_even:
  sr = ASHIFT mr1 by 1 (HI);  { Tb for even sectors }
  sr = sr or ASHIFT si BY 1 (LO);  { Ta for even sectors }
  jump SVM_saturate;

Sector_odd:
  sr = ASHIFT si by 1 (HI);  { Tb for odd sectors }
  sr = sr or ASHIFT mr1 BY 1 (LO);  { Ta for odd sectors }

At last, the computation of the zero-state time is executed. Clearly, if this value is negative, the demanded vector lays outside the hexagon and needs to be re-scaled, as explained in section 1.4. In that case, a division is necessary (code labelled by saturation). Since the on-times are positive numbers and $Ta+Tb$ is greater than $Ts/2$, a very simple division routine is implemented. The last instructions just copies the obtained values into the output vector.

```
SVM_saturate:
  ay0 = sr0;
  ar = srl + ay0;  { Ta + Tb }
  ay0= ar;
  ar = DM(PWMTM);  {Ts/2 }
  af = pass ar;  {store Ts/2 into af for eventual division }
  ar = ar - ay0;  { T0 }
  if GE jump NO_saturation;

saturation: ax0 = ay0;  { do division with MSW of num already in af }
  ay0 = 0;
  ASTAT = 0;  { Ts/2 is always < (Ta+Tb) and both are positive}
  divq ax0;  {unsigned division }
  divq ax0; divq ax0; divq ax0;
  divq ax0; divq ax0; divq ax0;
  divq ax0; divq ax0; divq ax0;
  divq ax0; divq ax0; divq ax0;
  my0 = ay0;  {Ts/2/(Ta+Tb) }
  mr = sr0 * my0 (SS);  { Ta scaled }
```
sr0 = mr1;
  mr = sr1 * my0 (SS);  { Tbc scaled }
  sr1 = mr1;
  ar = pass 0;       { TO }

  mr = mx0 * my0 (SS);  { Valpha scaled }
  DM(I0,M0) = mr1;
  mr = mx1 * my0 (SS);  { Vbeta scaled }
  DM(I0,M0) = mr1;

NO_saturation:
  DM(I1,M1) = sr0;       { first ontime }
  DM(I1,M1) = sr1;       { second ontime }
  DM(I1,M1) = ar;        { TO }
  rts;

The routines for calculating the duty-cycles is really only a jump-table which assigns the right values to
channels A, B and C, depending on the current sector. The duty-cycles are calculated from the on-times and
denoted as small, middle and big duty-cycle. Then, depending on the sector, these values are related to the
correct channel and copied into the output vector.

Jump_table:
Sec1:  DM(I2,M2) = mr1;
     DM(I2,M2) = sr1;
     DM(I2,M2) = sr0;
     rts;
Sec2:  DM(I2,M2) = sr1;
     DM(I2,M2) = mr1;
     DM(I2,M2) = sr0;
     rts;
Sec3:  DM(I2,M2) = sr0;
     DM(I2,M2) = mr1;
     DM(I2,M2) = sr1;
     rts;
Sec4:  DM(I2,M2) = sr0;
     DM(I2,M2) = sr1;
     DM(I2,M2) = mr1;
     rts;
Sec5:  DM(I2,M2) = sr1;
     DM(I2,M2) = sr0;
     DM(I2,M2) = mr1;
     rts;
Sec6:  DM(I2,M2) = mr1;
     DM(I2,M2) = sr0;
     DM(I2,M2) = sr1;
     rts;
2.5 Access to the library: the header file

The library may be accessed by including the header file “svpwm.h” in the application code, as was already explained in the section 2.1.

The header file is intended to provide function-like calls to the routines presented in the previous section. It defines the calls shown in Table 1. The file is mostly self-explaining. The only comment is that, since the input and output vectors are pointed to by index registers of the DM-DAG, they must be defined in data memory.

3 Software Example: Testing the Conversion Routines

3.1 The main program: main.dsp

The example demonstrates how to use the routines. Similarly to a previous note; it generates a balanced three-phase system. The magnitude is stored in a data memory location AD_IN. The (rotating) reference vector is created by applying the forward Park transformation to the (constant) vector $V_{dq}=(AD_IN, 0)$. The angle theta is calculated at each step in a similar fashion as in the mentioned application note. Then the three routines listed in Table 1 are called in sequence. The different outputs are converted and may be displayed on a scope by means of the digital to analog converter in the usual manner. This section will only explain the few and intuitive modifications to those applications.

The file “main.dsp” contains the initialisation and PWM Sync and Trip interrupt service routines. To activate, build the executable file using the attached build.bat either within your DOS prompt or clicking

---

4 AN331-03: Generation of Three-Phase Sine-Wave PWM
5 AN331-11: Reference Frame Conversions with the ADMC331
on it from Windows Explorer. This will create the object files and the main.exe example file. This file may be run on the Motion Control Debugger.

In the following, a brief description of the additional code (put in evidence by bold characters) is given.

---

**Start of code – declaring start location in program memory**

```
.MODULE/RAM/SEG=USER_PM1/ABS=0x30     Main_Program;
```

**Next, the general systems constants and PWM configuration constants (main.h – see the next section) are included. Also included are the PWM library, the DAC interface, the reference conversion library and the space vector modulation module definitions**

```
#include <main.h>;
#include <pwm331.h>;
#include <dac331.h>;
#include <trigono.h>;
#include <refframe.h>;
#include <svpwm.h>;
```

**Variables, Labels and Scope definitions. Here is where all the vectors for the SVM routines are declared.**

```
VAR/DM/RAM/SEG=USER_DM AD_IN;   { Volts/Hertz Command (0-1)  }
.INIT AD_IN : 0x7415;  { Corresponds to 0.906      }

VAR/DM/RAM/SEG=USER_DM Theta;  { Current angle      }
.INIT Theta : 0x0000;

VAR/DM/RAM/SEG=USER_DM Vdq_ref[2];   { rotor ref.frame     }
VAR/DM/RAM/CIRC/SEG=USER_DM Valphabeta_ref[2]; { alphabeta frame     }

VAR/RAM/DM/SEG=USER_DM OnTime_struct[1*4];
.INIT OnTime_struct: 0x0000, 0x0000, 0x0000, 0x0000;

VAR/RAM/DM/SEG=USER_DM Dutycycles_struct[1*3];
.INIT Dutycycles_struct: 0x0000, 0x0000, 0x0000;

VAR/DM/RAM/SEG=USER_DM VrefA;   { Voltage demands            }
VAR/DM/RAM/SEG=USER_DM VrefB;
VAR/DM/RAM/SEG=USER_DM VrefC;
.INIT VrefA : 0x0000;
.INIT VrefB : 0x0000;
.INIT VrefC : 0x0000;
```

**First, the PWM block is set up to generate interrupts every 100µs (see “main.h” in the next Section). The main loop just waits for interrupts.**

```
Start of program code

{******************************************************************************
* Start of program code           
{******************************************************************************}

Startup:
```

```
```
The interrupt service routine simply shows how to generate a balanced three-phase system similar to the already cited previous application note. Since the conversion routines make use of the pointer register I1, the DAC operations are to be stopped during their execution. Obviously, this may be removed when removing the DAC from the final application (refer to the application note on the DAC for a description of its commands and to section 2.3 for details of register usage of the conversion routines). What follows (in bold characters) is the actual generation. First, the current angle \( \Theta \) is computed by incrementing it by an amount which is dependent on the demanded frequency \( AD_{IN} \). This is almost identical as in the application note AN331-03, except for the fact that this time only one angle is required. This angle is then used by the forward Park transformation to convert a constant vector in the rotor reference frame of components \( AD_{IN} \) and 0 into a uniformly rotating vector in the stationary reference frame. Note that the default value given in this example is set to the limit of continuous SVM mentioned in section 1.3. This reference vector is the one used in the actual sequence of calls, which calculates on-times and duty-cycles and updates the PWM block. The following lines just update the DAC buffer for the digital-to-analog conversion. The default is the conversion of the angle \( \Theta \), the demanded duty-cycles for channels A, B and C (multiplied by 0x8 for visualisation purposes), the current sector (multiplied by 0x800 for visualisation purposes) and the reference vector \( V_{\alpha\beta} \). Section 3.3 will show the outputs that are produced by this example.

```
{********************************************************************************************}
{ PWM Interrupt Service Routine            }
{********************************************************************************************}
PWMSYNC_ISR:
DAC_Pause; { Required only when I1, M1 or L1 is used   }
ad = DM(AD_IN);
mar = 0;                           {Clear mr                     }
mr1 = dm(Theta);                   {Preload Theta                 }
my0 = Delta;                       {Compute new angle & store    }
dm(Theta) = mr1;
DM(Vdq_ref) = ar;                   {Set constant Vdq reference (AD_IN,0) }
ar = pass 0;                        {required                       }
DM(Vdq_ref+1) = ar;                 {required                       }
refframe_Set_DAC_registers_for_transformations;
refframe_Forward_Park_angle(Vdq_ref, Valphabeta_ref , mr1);
{ generate Vreference in alpha-beta frame }
SVPWM_Calc_Ontimes(Valphabeta_ref, OnTime_struct); { use SVPWM routines  }
SVPWM_Calc_Dutycycles(OnTime_struct, Dutycycles_struct);
SVPWM_Update_DutyCycles(Dutycycles_struct);
Dac_Update;
my0 = DM(Theta); DAC_Put(1, my0);    { output on DACs, amplified by multiplication }
mx0 = 0x8;
my0 = DM(Dutycycles_struct );       {required                       }
my0 = DM(Dutycycles_struct+1);      {required                       }
my0 = DM(Dutycycles_struct+2);      {required                       }
mx0 = 0x800;
my0 = DM(OnTime_struct );           {required                       }
my0 = DM(Valphabeta_ref );          {required                       }
my0 = DM(Valphabeta_ref+1);         {required                       }
DAC_Update;
RTI;
```
3.2 The main include file: main.h

This file contains the definitions of ADMC331 constants, general-purpose macros and the configuration parameters of the system and library routines. It should be included in every application. For more information refer to the Library Documentation File.

This file is mostly self-explaining. As already mentioned, the SVPWM library does not require any configuration parameters. The following defines the parameters for the PWM ISR used in this example.

```
{********************************************************************************************
{ Library: PWM block
{ file : PWM331.dsp
{ Application Note: Usage of the ADMC331 Pulse Width Modulation Block

.CONST  PWM_freq         = 10000; {Desired PWM switching frequency [Hz] }
.CONST  PWM_deadtime     = 1000; {Desired deadtime [nsec]           }
.CONST  PWM_minpulse     = 1000; {Desired minimal pulse time [nsec]           }
.CONST  PWM_syncpulse    = 1540; {Desired sync pulse time [nsec]              }

{********************************************************************************************
```

3.3 Example output

The signals that are generated by this demonstration program are shown in the following figures. Assume the desired modulation index be equal to the upper limit of continuous SVM, i.e. m=0.906 (0x7415). Therefore the reference vector in the rotor frame is:

\[
\begin{bmatrix}
V_d \\
V_q
\end{bmatrix} = m \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix}
\]  

(34)

Applying the forward Park transformation to (34), leads to the following input to the SVM routines, expressed in the stationary reference frame:

\[
\begin{bmatrix}
V_\alpha \\
V_\beta
\end{bmatrix} = V \cdot \begin{bmatrix} \cos(\omega t) \\ \sin(\omega t) \end{bmatrix}
\]  

(35)

The left side of Figure 6 shows the produced reference versus the angle (-\(\pi\) to +\(\pi\)). On the right side, the typical pole voltages of the inverter are represented.

![Figure 6: Reference vector (left) and inverter pole voltages (right) in continuous SVM](image_url)
The effects of the scaling in order to limit the reference vector within the hexagon are shown in the next figure. This time, the modulation index is set to full modulation, i.e. m=+1 (0x7FF). Therefore, the (scaled) reference vector entirely follows the hexagon.

![Image of reference vector and inverter pole voltages in full modulation]

*Figure 7: Reference vector (left) and inverter pole voltages (right) in full modulation*