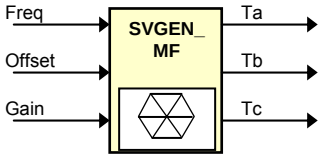
SVGEN_MF	*Space Vector Generator (magnitude/frequency)*

Description

This module calculates the appropriate duty ratios needed to generate a given stator reference voltage using space vector PWM technique. The stator reference voltage is described by it's magnitude and frequency.



Availability

This IQ module is available in one interface format:

1) The C interface version

Module Properties

Type: Target Independent, Application Independent

Target Devices: x281x or x280x

C Version File Names: svgen_mf.c, svgen_mf.h

IQmath library files for C: IQmathLib.h, IQmath.lib

Item	C version	Comments
Code Size□ (x281x/x280x)	331/331 words	
Data RAM	0 words•	
xDAIS ready	No	
XDAIS component	No	IALG layer not implemented
Multiple instances	Yes	
Reentrancy	Yes	

• Each pre-initialized "_iq" SVGENMF structure consumes 22 words in the data memory

□ Code size mentioned here is the size of the ***calc()*** function

C Interface

Object Definition

The structure of SVGENMF object is defined by following structure definition

```
typedef struct  { _iq  Gain;              // Input: reference gain voltage
                  _iq  Offset;            // Input: reference offset voltage
                  _iq  Freq;              // Input: reference frequency
                  _iq  FreqMax;           // Parameter: Maximum step angle
                  _iq  Alpha;             // History: Sector angle
                  _iq  NewEntry;          // History: Sine (angular) look-up pointer
                  Uint32  SectorPointer;  // History: Sector number (Q0)
                  _iq  Ta;                // Output: reference phase-a switching function
                  _iq  Tb;                // Output: reference phase-b switching function
                  _iq  Tc;                // Output: reference phase-c switching function
                  void   (*calc)();       // Pointer to calculation function
                } SVGENMF;

typedef SVGENMF * SVGENMF_handle;
```

Item	Name	Description	Format*	Range(Hex)
Inputs	Gain	reference gain voltage	GLOBAL_Q	80000000-7FFFFFFF
	Offset	reference offset voltage	GLOBAL_Q	80000000-7FFFFFFF
	Freq	reference frequency	GLOBAL_Q	80000000-7FFFFFFF
Outputs	Ta	Duty ratio of PWM1 (CMPR1 register value as a fraction of associated period register, TxPR, value).	GLOBAL_Q	80000000-7FFFFFFF
	Tb	Duty ratio of PWM3 (CMPR2 register value as a fraction of associated period register, TxPR, value).	GLOBAL_Q	80000000-7FFFFFFF
	Tc	Duty ratio of PWM5 (CMPR3 register value as a fraction of associated period register, TxPR, value).	GLOBAL_Q	80000000-7FFFFFFF
SVGENMF parameters	FreqMax	FreqMax = 6*fb*T	GLOBAL_Q	00000000-7FFFFFFF
Internal	Alpha	Sector angle	GLOBAL_Q	80000000-7FFFFFFF
	NewEntry	Sine (angular) look-up pointer	GLOBAL_Q	80000000-7FFFFFFF
	SectorPointer	Sector number	Q0	0-5

*GLOBAL_Q valued between 1 and 30 is defined in the IQmathLib.h header file.

Special Constants and Data types

SVGENMF

The module definition is created as a data type. This makes it convenient to instance an interface to space vector generator using magnitude and frequency. To create multiple instances of the module simply declare variables of type SVGENMF.

SVGENMF_handle

User defined Data type of pointer to SVGENMF module

SVGENMF_DEFAULTS
Structure symbolic constant to initialize SVGENMF module. This provides the initial values to the terminal variables as well as method pointers.

Methods

void svgenmf_calc(SVGENMF_handle);

This definition implements one method viz., the space vector generator computation function. The input argument to this function is the module handle.

Module Usage

Instantiation
The following example instances two SVGENMF objects
SVGENMF svgen_mf1, svgen_mf2;

Initialization
To Instance pre-initialized objects
SVGENMF svgen_mf1 = SVGENMF_DEFAULTS;
SVGENMF svgen_mf2 = SVGENMF_DEFAULTS;

Invoking the computation function
svgen_mf1.calc(&svgen_mf1);
svgen_mf2.calc(&svgen_mf2);

Example
The following pseudo code provides the information about the module usage.

```
main()
{

}

void interrupt periodic_interrupt_isr()
{
        svgen_mf1.Gain = gain1;              // Pass inputs to svgen_mf1
        svgen_mf1.Freq = offset1;            // Pass inputs to svgen_mf1

        svgen_mf2.Gain = gain2;              // Pass inputs to svgen_mf2
        svgen_mf2.Freq = offset2;            // Pass inputs to svgen_mf2

        svgen_mf1.calc(&svgen_mf1);          // Call compute function for svgen_mf1
        svgen_mf2.calc(&svgen_mf2);          // Call compute function for svgen_mf2

        Ta1 = svgen_mf1.Ta;                  // Access the outputs of svgen_mf1
        Tb1 = svgen_mf1.Tb;                  // Access the outputs of svgen_mf1
        Tc1 = svgen_mf1.Tc;                  // Access the outputs of svgen_mf1

        Ta2 = svgen_mf2.Ta;                  // Access the outputs of svgen_mf2
        Tb2 = svgen_mf2.Tb;                  // Access the outputs of svgen_mf2
        Tc2 = svgen_mf2.Tc;                  // Access the outputs of svgen_mf2
}
```

Technical Background

The Space Vector Pulse Width Modulation (SVPWM) refers to a special switching sequence of the upper three power devices of a three-phase voltage source inverters (VSI) used in application such as AC induction and permanent magnet synchronous motor drives. This special switching scheme for the power devices results in 3 pseudo-sinusoidal currents in the stator phases.

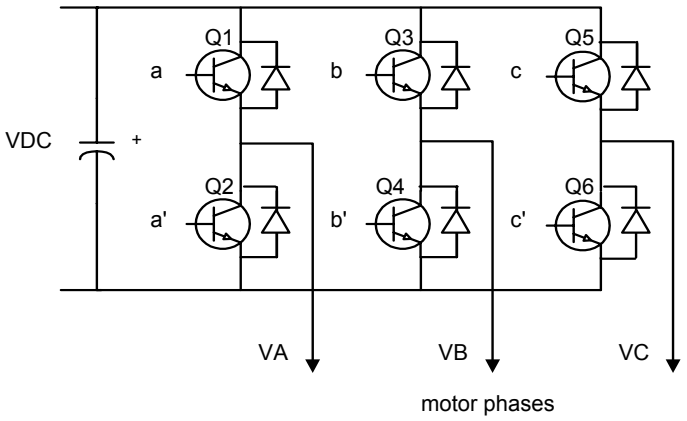


Figure 1 Power circuit topology for a three-phase VSI

It has been shown that SVPWM generates less harmonic distortion in the output voltages or currents in the windings of the motor load and provides more efficient use of DC supply voltage, in comparison to direct sinusoidal modulation technique.

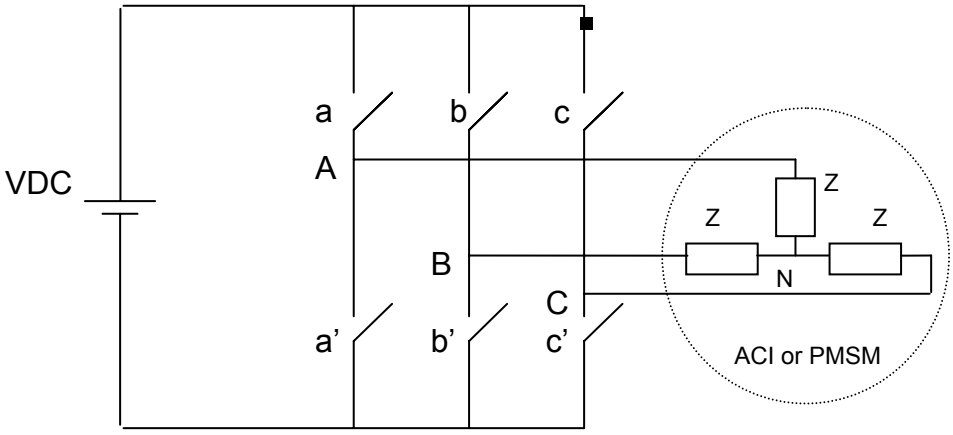


Figure 2: Power bridge for a three-phase VSI

For the three phase power inverter configurations shown in Figure 1 and Figure 2, there are eight possible combinations of on and off states of the upper power transistors.

These combinations and the resulting instantaneous output line-to-line and phase voltages, for a dc bus voltage of $V_{DC,}$ are shown in Table 1.

c	b	a	V_{AN}	V_{BN}	V_{CN}	V_{AB}	V_{BC}	V_{CA}
0	0	0	0	0	0	0	0	0
0	0	1	$2V_{DC}/3$	$-V_{DC}/3$	$-V_{DC}/3$	V_{DC}	0	$-V_{DC}$
0	1	0	$-V_{DC}/3$	$2V_{DC}/3$	$-V_{DC}/3$	$-V_{DC}$	V_{DC}	0
0	1	1	$V_{DC}/3$	$V_{DC}/3$	$-2V_{DC}/3$	0	V_{DC}	$-V_{DC}$
1	0	0	$-V_{DC}/3$	$-V_{DC}/3$	$2V_{DC}/3$	0	$-V_{DC}$	V_{DC}
1	0	1	$V_{DC}/3$	$-2V_{DC}/3$	$V_{DC}/3$	V_{DC}	$-V_{DC}$	0
1	1	0	$-2V_{DC}/3$	$V_{DC}/3$	$V_{DC}/3$	$-V_{DC}$	0	V_{DC}
1	1	1	0	0	0	0	0	0

Table 1. Device on/off patterns and resulting instantaneous voltages of a 3-phase power inverter

The quadrature quantities (in d-q frame) corresponding to these 3 phase voltages are given by the general Clarke transform equation:

$$V_{ds} = V_{AN}$$

$$V_{qs} = \frac{2V_{BN} + V_{AN}}{\sqrt{3}}$$

In matrix from the above equation is also expressed as,

$$\begin{bmatrix} V_{ds} \\ V_{qs} \end{bmatrix} = \frac{2}{3}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix}\begin{bmatrix} V_{AN} \\ V_{BN} \\ V_{CN} \end{bmatrix}$$

Due to the fact that only 8 combinations are possible for the power switches, V_{ds} and V_{qs} can also take only a finite number of values in the (d-q) frame according to the status of the transistor command signals (c,b,a). These values of V_{ds} and V_{qs} for the corresponding instantaneous values of the phase voltages (V_{AN}, $V_{BN,}$ V_{CN}) are listed in Table 2.

c	b	a	V_{ds}	V_{qs}	Vector
0	0	0	0	0	O_0
0	0	1	$2\ V_{DC}/3$	0	U_0
0	1	0	$-V_{DC}/3$	$V_{DC}/\sqrt{3}$	U_{120}
0	1	1	$V_{DC}/3$	$V_{DC}/\sqrt{3}$	U_{60}
1	0	0	$-V_{DC}/3$	$-V_{DC}/\sqrt{3}$	U_{240}
1	0	1	$V_{DC}/3$	$-V_{DC}/\sqrt{3}$	U_{300}
1	1	0	$-2\ V_{DC}/3$	0	U_{180}
1	1	1	0	0	O_{111}

Table 2: Switching patterns, corresponding space vectors and their (d-q) components

These values of V_{ds} and V_{qs}, listed in Table 2, are called the (d-q) components of the basic space vectors corresponding to the appropriate transistor command signal (c,b,a). The space vectors corresponding to the signal (c,b,a) are listed in the last column in Table 2. For example, (c,b,a)=001 indicates that the space vector is U_0.The eight basic space vectors defined by the combination of the switches are also shown in Figure 3.

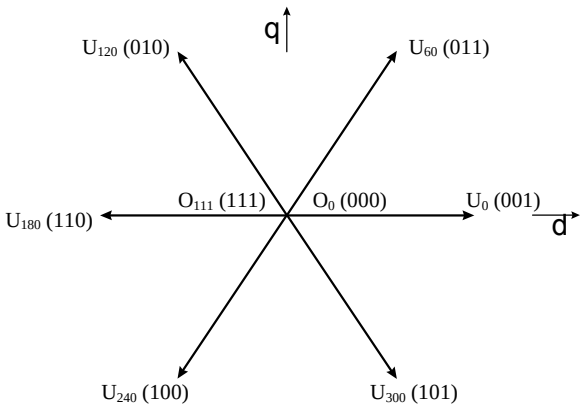


Figure 3: Basic space vectors

In Figure 3, vectors corresponding to states 0 (000) and 7 (111) of the switching variables are called the zero vectors.

Decomposing the reference voltage vector V*

The objective of Space Vector PWM technique is to approximate a given stator reference voltage vector V* by combination of the switching pattern corresponding to the basic space vectors. The reference voltage vector V* is obtained by mapping the desired three phase output voltages(line to neutral) in the (d-q) frame through the Clarke transform defined earlier. When the desired output phase voltages are balanced three phase sinusoidal voltages, V* becomes a vector rotating around the origin of the (d-q) plane with a frequency corresponding to that of the desired three phase voltages.

The magnitude of each basic space vector, as shown in Fig. 4, is normalized by the maximum value of the phase voltages. Therefore, when the maximum bus voltage is V_{DC}, the maximum line to line voltage is also V_{DC}, and so the maximum phase voltage(line to neutral) is $V_{DC}/\sqrt{3}$. From Table 2, the magnitude of the basic space vectors is $2V_{DC}/3$. When this is normalized by the maximum phase voltage($V_{DC}/\sqrt{3}$), the magnitude of the basic space vectors becomes $2/\sqrt{3}$. These magnitudes of the basic space vectors are indicated in Figure 4.

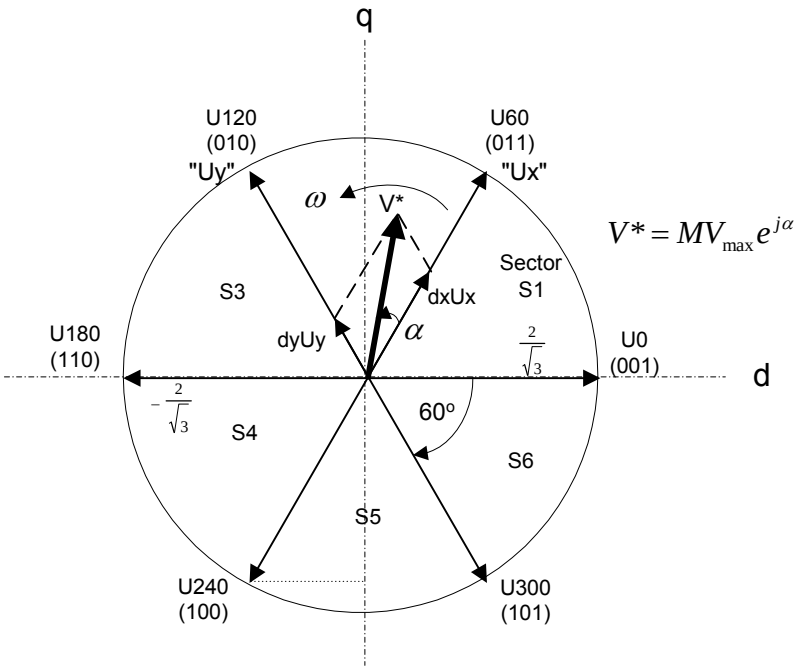


Figure 4: Projection of the reference voltage vector

Representing the reference vector V* with the basic space vectors requires precise control of both the vector magnitude M (also called the modulation index) and the angle α. The aim here is to rotate V* in the d-q plane at a given angular speed (frequency) ω. The vector magnitude M controls the resultant peak phase voltage generated by the inverter.

In order to generate the reference vector V*, a time average of the associated basic space vectors is required, i.e. the desired voltage vector V* located in a given sector, can be synthesized as a linear combination of the two adjacent space vectors, Ux and Uy which frame the sector, and either one of the two zero vectors. Therefore,

$$V^* = dxUx + dyUy + dzUz$$

where Uz is the zero vector, and dx, dy and dz are the duty ratios of the states X, Y and Z within the PWM switching interval. The duty ratios must add to 100% of the PWM period, i.e: dx + dy + dz = 1.

Vector V* in Fig. 4 can also be written as:

$$V^* = MV_{max}e^{j\alpha} = dxUx + dyUy + dzUz$$

where M is the modulation index and V_{max} is the maximum value of the desired phase voltage.

By projecting V* along the two adjacent space vectors Ux and Uy, we have,

$$\begin{cases} MV_{\max}\cos\alpha = dx|Ux| + dy|Uy|\cos 60^\circ \\ MV_{\max}\sin\alpha = dy|Uy|\sin 60^\circ \end{cases}$$

Since the voltages are normalized by the maximum phase voltage, V_{max}=1. Then by knowing |Ux| = |Uy| = 2/√3 (when normalized by maximum phase voltage), the duty ratios can be derived as,

$$dx = M\sin(60-\alpha)$$

$$dy = M\sin(\alpha)$$

These same equations apply to any sector, since the d-q reference frame, which has here no specific orientation in the physical space, can be aligned with any space vector.

As shown in Fig. 4, sine of α is needed to decompose the reference voltage vector onto the basic space vectors of the sector the voltage vector is in. Since this decomposition is identical among the six sectors, only a 60° sine lookup table is needed. In order to complete one revolution (360°) the sine table must be cycled through 6 times.

For a given step size the angular frequency (in cycles/sec) of V* is given by:

$$\omega = \frac{STEP \times fs}{6 \times 2^m}$$

Where f_s = sampling frequency (i.e. PWM frequency), STEP = angle stepping increment, and *m* = # bits in the integration register.

For example, if f_s = 24KHz, *m* = 16 bits & STEP ranges from 0→2048 then the resulting angular frequencies will be as shown in Table 3.

STEP	Freq(Hz)	STEP	Freq(Hz)	STEP	Freq(Hz)
1	0.061	600	36.62	1700	103.76
20	1.22	700	42.72	1800	109.86
40	2.44	800	48.83	1900	115.97
60	3.66	900	54.93	2000	122.07
80	4.88	1000	61.04	2100	128.17
100	6.10	1100	67.14	2200	134.28

Table 3 Frequency mapping

From the table it is clear that a STEP value of 1 gives a frequency of 0.061Hz, this defines the frequency setting resolution, i.e. the actual line voltage frequency delivered to the AC motor can be controlled to better than 0.1 Hz.

For a given f_s the frequency setting resolution is determined by *m* the number of bits in the integration register. Table 4 shows the theoretical resolution which results from various sizes of m.

m (# bits)	Freq res(Hz)	m (# bits)	Freq res(Hz)
8	15.6250	17	0.0305
12	0.9766	18	0.0153
14	0.2441	19	0.0076
16	0.0610	20	0.0038

Table 4 Resolution of frequency mapping

For IQmath implementation, the maximum step size in per-unit, *FreqMax*, for a given base frequency, fb and a defined GLOBAL_Q number is therefore computed as follows:

$$\text{FreqMax} = 6 \times f_b \times T_s \times 2^{\text{GLOBAL\_Q}}$$

Equivalently, by using _IQ() function for converting from a floating-point number to a _iq number, the *FreqMax* can also be computed as

$$\text{FreqMax} = \text{\_IQ}(6 \times f_b \times T_s)$$

where Ts is the sampling period (sec).

Realization of the PWM Switching Pattern

Once the PWM duty ratios dx, dy and dz are calculated, the appropriate compare values for the compare registers in 28xx can be determined. The switching pattern in Figure 5 is adopted here and is implemented with the Full Compare Units of 28xx. A set of 3 new compare values, Ta, Tb and Tc, need to be calculated every PWM period to generate this switching pattern.

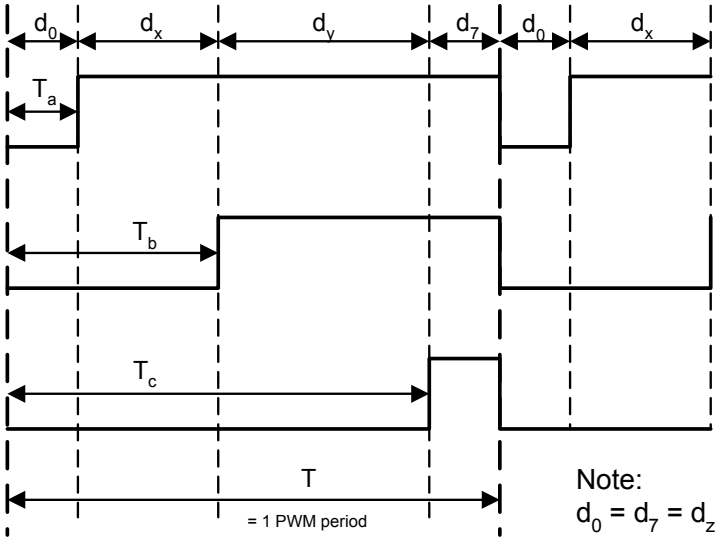


Figure 5. PWM output switching pattern

From Figure 5, it can be seen:

$$Ta = (T - dx - dy)/2$$

$$Tb = dx + Ta$$

$$Tc = T - Ta$$

If we define an intermediate variable T1 using the following equation:

$$T1 = \frac{T - dx - dy}{2}$$

Then for different sectors Ta, Tb and Tc can be expressed in terms of T1. Table 5 depicts this determination.

Sectors	U_0, U_{60}	U_{60}, U_{120}	U_{120}, U_{180}	U_{180}, U_{240}	U_{240}, U_{300}	U_{300}, U_0
Ta	T1	dy+Tb	T-Tb	T-Tc	dx+Tc	T1
Tb	dx+Ta	T1	T1	dy+Tc	T-Tc	T-Ta
Tc	T-Ta	T-Tb	dx+Tb	T1	T1	dy+Ta

Table 5: Calculation of duty cycle for different sectors

The switching pattern shown in Figure 5 is an asymmetric PWM implementation. However, 28xx devices can also generate symmetric PWM. Little change to the above implementation is needed to accommodate for this change. The choice between the symmetrical and asymmetrical case depends on the other care-about in the final implementation.