SV-PWM - Space vector PWM

SD013 | Posted on April 2, 2021 | Updated on May 27, 2025

Benoît STEINMANN
Software Team Leader
imperix

Table of Contents

The SV-PWM block generates PWM signals based on the Space Vector Modulation (SVM) algorithm. This algorithm determines the three vectors that are the closest to the reference vector and computes the dwell times for each one. Based on those times, a duty cycle is computed and a triangular carrier is used to generate the PWM signal for each phase. The SVM method is explained in the Space Vector Modulation (TN145) and SVPWM vs SPWM modulation techniques (TN146) notes.

When using the **single-rate update** configuration, the computed duty cycles are synchronously applied at the end of the PWM period. With the **double-rate update**, the duty-cycle is updated twice per period: in the middle and at the end (in other words when the carrier reaches its maximum and when it reaches its minimum).

The **frequency** of the carriers is configured by connecting the SV-PWM block to a CLK – Clock generator. The frequency can even be tuned during the control execution as explain in Variable frequency operation with the B-Box/B-Board (PN121).

Like the other PWM blocks, the Space Vector Modulation block supports **dead-time generation** and can be **activated or deactivated**. More information is available on the PWM page.

Simulink SV-PWM block

Signal specification

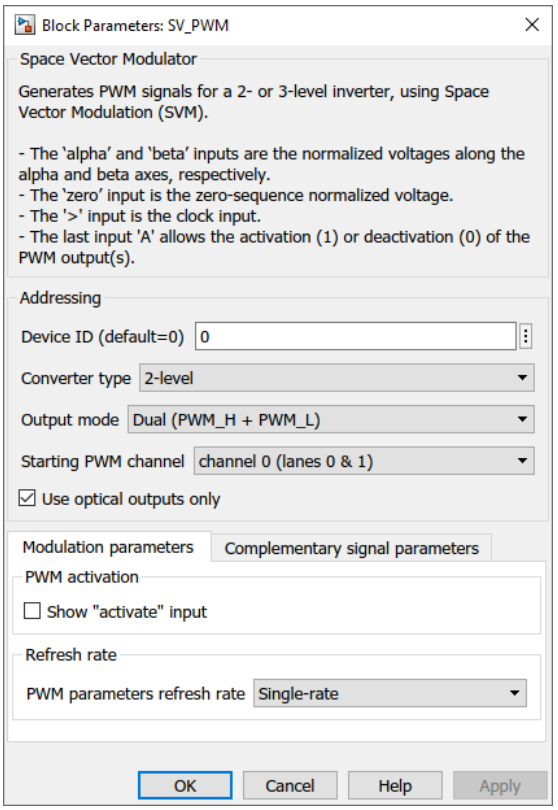
- The input αβ0 is the normalized reference vector in the stationary reference frame (-1.15 to 1.15).
- The input signal > is the clock input and must be connected to the CONFIG block or to an independent CLK.
- The outputs are the generated PWM signals, according to the selected `output mode` and the `converter type`. The outputs are only used in the simulation.

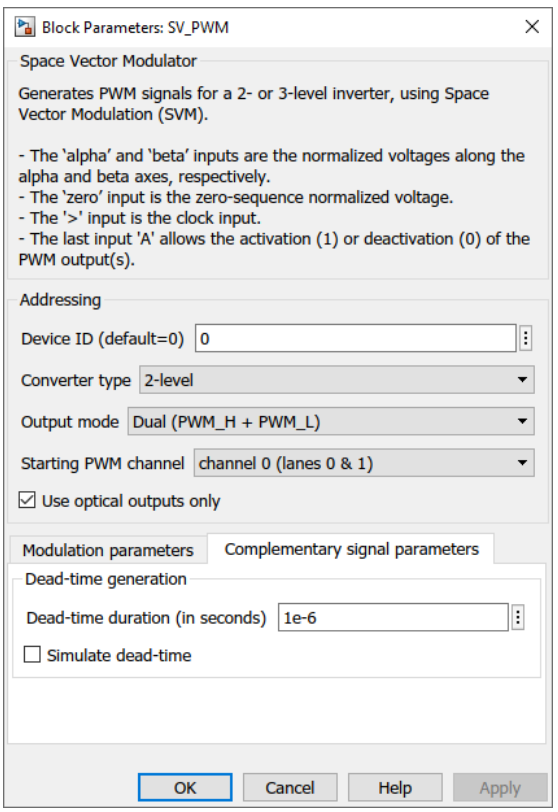
The parameters `output mode`, `addressed PWM`, `dead-time` and `show "activate" input` are common to all PWM blocks and are further documented on the PWM page.
While each component of the normalized reference vector can be in the range [-1.15; 1.15], the norm of that vector must be in the range [0; 1.15]. The limitation of the norm is further explained in SVPWM vs SPWM modulation techniques (TN146). In addition, the page Space Vector Modulation (TN145) presents a way to take this limitation into account in a closed-loop control algorithm.

Parameters

- `Device ID` selects which B-Box/B-Board to address when used in a multi-device configuration.

- `Converter type` configures the SV-PWM algorithm for a 2- or 3-level inverter.
- `Output mode` selects between a single PWM signal or complementary signals with a dead-time.
- `Starting PWM channel`: selects the first PWM output to be used for the set of PWM signals of the SV-PWM algorithm.
- `Use optical outputs only` selects if the PWM signals can be addressed only to the optical outputs, or if the electrical outputs (lanes 16 to 31) can also be used.
- `Show "activate" input` makes the `A` signal input visible. If not checked, the SV-PWM block is active by default.
- `PWM parameters update rate` selects when the duty-cycle and phase parameters are applied.
  - *Single-rate*: they are applied at the end of the carrier period.
  - *Double-rate*: they are applied twice per carrier period: when the carrier reaches its lowest point and when it reaches its highest point.
- `Dead-time duration`: configures the dead-time duration if the `Output mode` is set at *Dual (PWM_H + PWM_L)*.

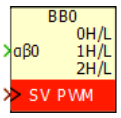




PLECS SV-PWM block

Signal specification

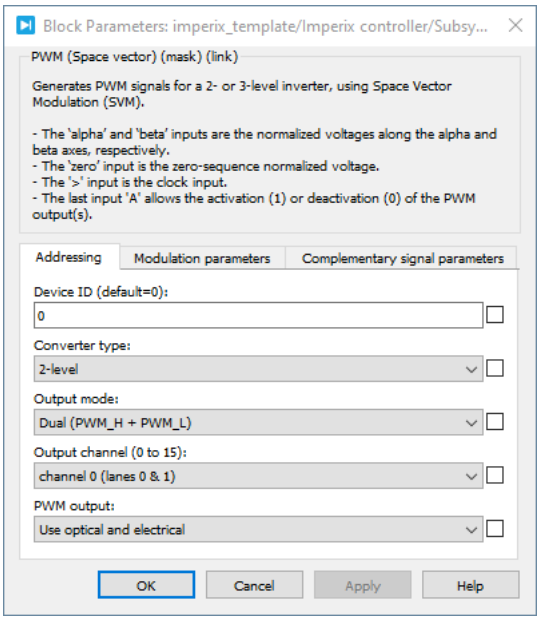
- The input αβ0 is the normalized reference vector in the stationary reference frame (-1.15 to 1.15).
- The input A allows the activation (1) or deactivation (0) of the PWM output(s).
- The input signal > is the clock input and must be connected to the CONFIG block or to an independent CLK.
- The target outport (only visible at the atomic subsystem level) are the generated PWM signals, according to the selected `Output mode` (see PWM page for information). These outputs are only used in the simulation.

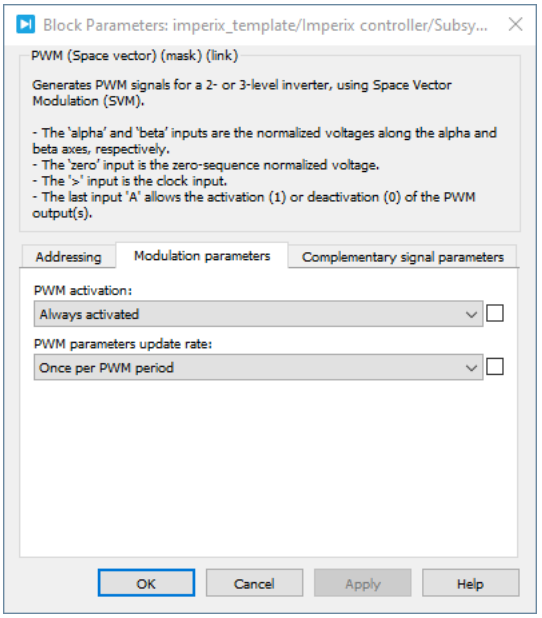


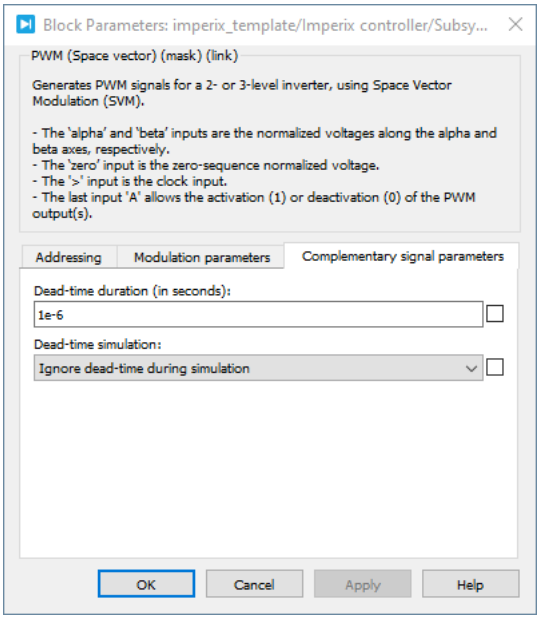
The parameters `output mode`, `addressed PWM`, `dead-time` and `show "activate" input` are common to all PWM blocks and are further documented on the PWM page.
While each component of the normalized reference vector can be in the range [-1.15; 1.15], the norm of that vector must be in the range [0; 1.15]. The limitation of the norm is further explained in SVPWM vs SPWM modulation techniques (TN146). In addition, the page Space Vector Modulation (TN145) presents a way to take this limitation into account in a closed-loop control algorithm.

Parameters

- `Device ID` selects which B-Box/B-Board to address when used in a multi-device configuration.
- `Converter type` configures the SV-PWM algorithm for a 2- or 3-level inverter.
- `Output mode` selects between a single PWM signal or complementary signals with a dead-time.
- `Starting PWM channel`: selects the first PWM output to be used for the set of PWM signals of the SV-PWM algorithm.
- `Use optical outputs only` selects if the PWM signals can be addressed only to the optical outputs, or if the electrical outputs (lanes 16 to 31) can also be used.
- `Show "activate" input` makes the A signal input visible. If not checked, the SV-PWM block is active by default.
- `PWM parameters update rate` selects when the duty-cycle and phase parameters are applied.
  - *Single-rate*: they are applied at the end of the carrier period.
  - *Double-rate*: they are applied twice per carrier period: when the carrier reaches its lowest point and when it reaches its highest point.
- `Dead-time duration`: configures the dead-time duration if the `Output mode` is set at *Dual (PWM_H + PWM_L)*







C++ functions

Specific to SV-PWM

`SvPwm_ConfigureOutputs` **— Configure the PWM outputs**

```
void SvPwm_ConfigureOutputs(tSvModulator modulator, tPwmOutput startingOutput, tPwmOutputType outputType, unsigned
```

Configures the PWM outputs of a Space Vector modulator.

It has to be called in `UserInit()`.

Parameters

- `modulator`: the SV-PWM modulator id (*SV_MODULATOR_0* to *SV_MODULATOR_4*)
- `startingOutput`: the first PWM channel or lane to address
- `outputType`: the nature of the PWM outputs (*OPTICAL_PWM_OUTPUT* or *ELECTRICAL_PWM_OUTPUT*)
- `startingDevice:` the first B-Box/B-Board to addressed when used in a multi-device configuration.

`SvPwm_ConfigureLevel` **— Select between 2-level and 3-level**

```
void SvPwm_ConfigureLevel(tSvModulator modulator, unsigned int level);
```
Code language: C++ (cpp)

Configures the number of levels of the power converter.

It has to be called in `UserInit()`.

Parameters

- `modulator`: the SV-PWM modulator id (*SV_MODULATOR_0* to *SV_MODULATOR_4*)
- `level`: the number of levels of the power converter (2 or 3)

`SvPwm_ConfigurePhase` — Set the carrier phase shift

```
void SvPwm_ConfigurePhase(tSvModulator modulator, float phase);
Code language: C++ (cpp)
```

Configures the carrier phase-shift relative to the CLOCK.

It can be called in `UserInit()` or in the interrupt routine.

Parameters

- `modulator`: the SV-PWM modulator id (*SV_MODULATOR_0* to *SV_MODULATOR_4*)
- `phase`: the carrier phase-shift relative to the CLK (0.0 to 1.0)

`SvPwm_ConfigureClock` — Select a CLOCK

```
void SvPwm_ConfigureClock(tSvModulator modulator, tClock clock);Code language: C++ (cpp)
```

Connects a clock generator to the modulator.

It has to be called in `UserInit()`.

See: CLK – Clock generator

Parameters

- `modulator`: the SV-PWM modulator id (*SV_MODULATOR_0* to *SV_MODULATOR_4*)
- `clock`: the clock to use (*CLOCK_0*, *CLOCK_1*, *CLOCK_2* or *CLOCK_3*)

`SvPwm_RunCartesian` — Run the algorithm in cartesian coordinates

```
void SvPwm_RunCartesian(tSvModulator modulator, float dAlpha, float dBeta, float dZero);Code language: C++ (cpp)
```

Runs the SV-PWM algorithm in cartesian coordinates, and applies the corresponding duty-cycles to the configured outputs.

It can be called in the interrupt routine.

Parameters

- `modulator`: the SV-PWM modulator id (*SV_MODULATOR_0* to *SV_MODULATOR_4*)
- `dAlpha`: the component of the Space Vector to apply along alpha axis (-1.15 to 1.15)
- `dBeta:` the component of the Space Vector to apply along beta axis (-1.15 to 1.15)
- `dZero`: the homopolar component of the Space Vector to apply (-1.15 to 1.15)

`SvPwm_RunPolar` — Run the algorithm in polar coordinates

```
void SvPwm_RunPolar(tSvModulator modulator, float m, float phi, float dZero);Code language: C++ (cpp)
```

Runs the SV-PWM algorithm in polar coordinates, and applies the corresponding duty-cycles to the configured outputs.

It can be called in the interrupt routine.

Parameters

- `modulator`: the SV-PWM modulator id (*SV_MODULATOR_0* to *SV_MODULATOR_4*)
- `m`: the norm of the Space Vector to apply (0.0 to 1.15)
- `phi:` the angle of the Space Vector to apply
- `dZero`: the homopolar component of the Space Vector to apply (-1.15 to 1.15)

`SvPwm_Run` — Run the algorithm in 60° coordinates

```
void SvPwm_Run(tSvModulator modulator, float Vg, float Vh, float dZero);Code language: C++ (cpp)
```

Runs the SV-PWM algorithm in the 60° coordinates, and applies the corresponding duty-cycles to the configured outputs.

It can be called in the interrupt routine.

Parameters

- `modulator`: the SV-PWM modulator id (*SV_MODULATOR_0* to *SV_MODULATOR_4*)

- `Vg`: the component of the Space Vector to apply along g axis (-1.15 to 1.15)
- `Vh`: the component of the Space Vector to apply along h axis (-1.15 to 1.15)
- `dZero`: the homopolar component of the Space Vector to apply (-1.15 to 1.15)

`SvPwm_ConfigureUpdateRate` **— Select an update rate**

```
void SvPwm_ConfigureUpdateRate(tSvModulator modulator, tPwmRate rate);Code language: C++ (cpp)
```

Select when the duty-cycle and phase parameters are applied.

- *Single-rate*: they are applied at the end of the carrier period.
- *Double-rate*: they are applied twice per carrier period: when the carrier reaches its lowest point and when it reaches its highest point. (for TRIANGLE and INVTRIANGLE carriers only)

It has to be called in `UserInit()`.

Parameters

- `modulator`: the SV-PWM modulator id (*SV_MODULATOR_0* to *SV_MODULATOR_4*)
- `rate`: the update rate to use (*SINGLE_RATE* or *DOUBLE_RATE*)

While each component of the normalized reference vector can be in the range [-1.15; 1.15], the norm of that vector must be in the range [0; 1.15]. The limitation of the norm is further explained in SVPWM vs SPWM modulation techniques (TN146). In addition, the page Space Vector Modulation (TN145) presents a way to take this limitation into account in a closed-loop control algorithm.

Functions common to all PWM drivers

These functions are common to all PWM blocks. Further documentation is available on the PWM page.

`SvPwm_ConfigureOutputMode` **— Select the PWM output mode**

```
void SvPwm_ConfigureOutputMode(tSvModulator modulator, tPwmOutMode outMode, unsigned int device=0);Code language: C++
```

Selects the PWM output mode.

If the output mode selected is *COMPLEMENTARY*, a dead-time must be configured using the `CbPwm_ConfigureDeadTime()` function.

It has to be called in `UserInit()`.

Parameters

- `modulator`: the SV-PWM modulator id (*SV_MODULATOR_0* to *SV_MODULATOR_4*)
- `outMode`: the output mode to use (*COMPLEMENTARY*, *INDEPENDENT* or *PWMH_ACTIVE*)
- `device`: the B-Box/B-Board to address when used in a multi-device configuration

`SvPwm_ConfigureDeadTime` **— Configure the dead time**

```
void SvPwm_ConfigureDeadTime(tSvModulator modulator, float deadTime, unsigned int device=0);Code language: C++ (cpp)
```

Configures the dead-time duration if the output mode is set as *COMPLEMENTARY*.

It has to be called in `UserInit()`.

Parameters

- `modulator`: the SV-PWM modulator id (*SV_MODULATOR_0* to *SV_MODULATOR_4*)
- `outMode`: the output mode to use (*COMPLEMENTARY*, *INDEPENDENT* or *PWMH_ACTIVE*)
- `device`: the B-Box/B-Board to address when used in a multi-device configuration

`SvPwm_Activate` **— Activate the PWM outputs**

```
void SvPwm_Activate(tSvModulator modulator, unsigned int device=0);
Code language: C++ (cpp)
```

Activates the addressed PWM output(s). If the addressed PWM output has been set as *COMPLEMENTARY* or *PWMH_ACTIVE* this function acts on both outputs.

It can be called in `UserInit()` or in the control interrupt routine.

Parameters

- `modulator`: the SV-PWM modulator id (*SV_MODULATOR_0* to *SV_MODULATOR_4*)
- `device`: the B-Box/B-Board to address when used in a multi-device configuration

`SvPwm_Deactivate` — Deactivate the PWM outputs

```
void SvPwm_Deactivate(tSvModulator modulator, unsigned int device=0);Code language: C++ (cpp)
```

Deactivates the addressed PWM output(s). If the addressed PWM output has been set as *COMPLEMENTARY* or *PWMH_ACTIVE* this function acts on both outputs.

It can be called in `UserInit()` or in the control interrupt routine.

Parameters

- `modulator`: the SV-PWM modulator id (*SV_MODULATOR_0* to *SV_MODULATOR_4*)
- `device`: the B-Box/B-Board to address when used in a multi-device configuration