

# PP-PWM – Programmed Patterns PWM

SD015 | Posted on March 9, 2022 | Updated on May 27, 2025



Jessy ANÇAY

Sales & Project Engineer

imperix • in

---

## Table of Contents

- [Programmed Pattern PWM operation](#)
- [Simulink PP-PWM block](#)
  - [Signal specification](#)
  - [Parameters](#)
- [PLECS PP-PWM block](#)
  - [Signal specification](#)
  - [Parameters](#)
- [C++ functions](#)
  - [Specific to Programmed Pattern PWM](#)
  - [Functions common to all PWM drivers](#)

The FPGA-based PP-PWM (programmed pulse pattern) peripheral provides a specialized PWM scheme for two and three-level inverters, which relies on pre-computed pulse patterns. This type of modulation technique is often used to eliminate specific harmonics from the current spectrum. It is notably useful in applications that operate with a low pulse number, i.e. a low ratio between switching frequency and fundamental signal frequency.

The PP-PWM peripheral can be used for implementing all sorts of modulation techniques that rely on pre-computed pulse patterns, such as [Selective Harmonic Elimination](#) (SHE or SHEPWM) or other Optimized Pulse Patterns (OPPs).

The Programmed Pattern PWM module features the following characteristics:

- Three-phase output PWM for **2 and 3-level** inverters
- A maximum of **16 switching angles** per quarter period (64 angles per period)
- Angles update every **quarter period**

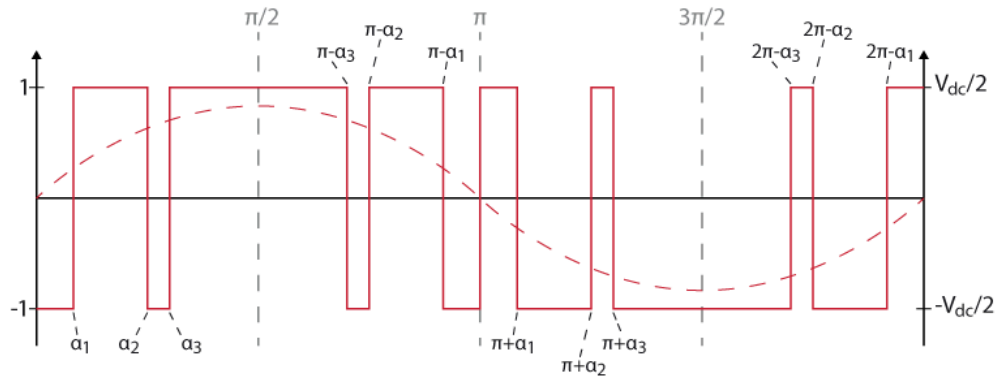
Similarly to other PWM modules, the following parameters of the PP PWM module can be configured:

- Complimentary High and Low signals offer a configurable dead-time generation
- Programmed Pattern PWM outputs can be activated or deactivated during operation

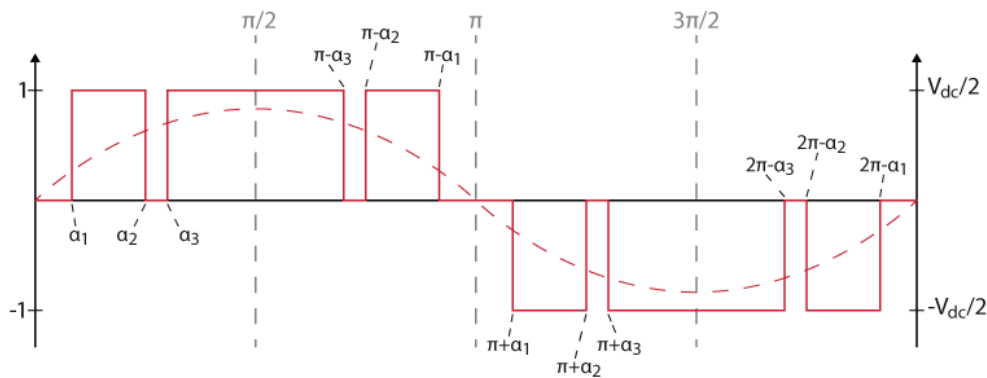
## Programmed Pattern PWM operation

To generate a PWM signal, the Programmed Pattern PWM module needs a set of switching angles between  $[0; \frac{\pi}{2}]$ . Thanks to the quarter-wave symmetry of the reference sinusoidal signal, only the angles for the first quarter of the sinewave are required. The angles for the three other quarters will be reconstructed by symmetry inside the FPGA peripheral.

Along with the angles, a vector of transitions with the same length as the angles vector is required. This vector of transition will specify for each angle, whether the switching signal will go “up or down”. A value of 1 encodes an upwards transition while a -1 encodes a downward transition. Two waveform examples for 2- and 3-level inverters are given below.



2-level Programmed Pulse PWM



3-level Programmed Pulse

When comparing both waveforms, it can be noted that the 2-level waveform contains an additional transition in the middle of the period (angle =  $\pi$ ) since a 2-level inverter cannot generate a zero voltage output.

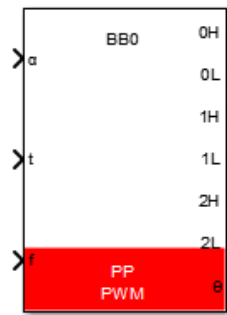
At its output, the Programmed Pattern PWM module provides the gating signals and also indicates the current phase angle, for instance allowing for synchronization with the grid.

## Simulink PP-PWM block

### Signal specification

- The input  $\alpha$  contains the vector of angles between  $[0; \frac{\pi}{2}]$  that define the programmed pattern.
- The input signal  $t$  is a vector containing the transitions for each angle.
- The input signal  $f$  defines the pulse frequency.

- The outputs are the generated PWM signals, according to the block configuration and the phase angle  $\theta$ . The PWM outputs are only used for simulation.
- The phase angle output  $\theta$  is the phase angle for grid synchronization.



## Parameters

- Device ID selects which B-Box/B-Board to address when used in a multi-device configuration.
- Converter type configures the programmed pattern PWM for a two or three-lvl inverter.
- Number of angles configures the number of angles per quarter period.
- Show "activate" input makes the A (active) signal input visible. If not checked, the PP-PWM block is always active by default (once the controller outputs are enabled).
- Deadtime duration: configures the dead-time duration.

Block Parameters: PP\_PWM

Pulse Pattern PWM (mask)

Generates PWM signals for a 2- or 3-level inverter, using Pulse Pattern Modulation (PP PWM).

- The 'alpha' input is the list of angles for the first quarter of a period, bounded between  $[0; \pi/2]$ .
- The 't' input is the list of transition corresponding to the angles.
- The 'f' input defines the pulse frequency.
- The last input 'A' allows the activation (1) or deactivation (0) of the PWM output(s).

Parameters

Device ID

Converter type 2-Level

Number of angles

Output mode: Dual (PWM\_H + PWM\_L)

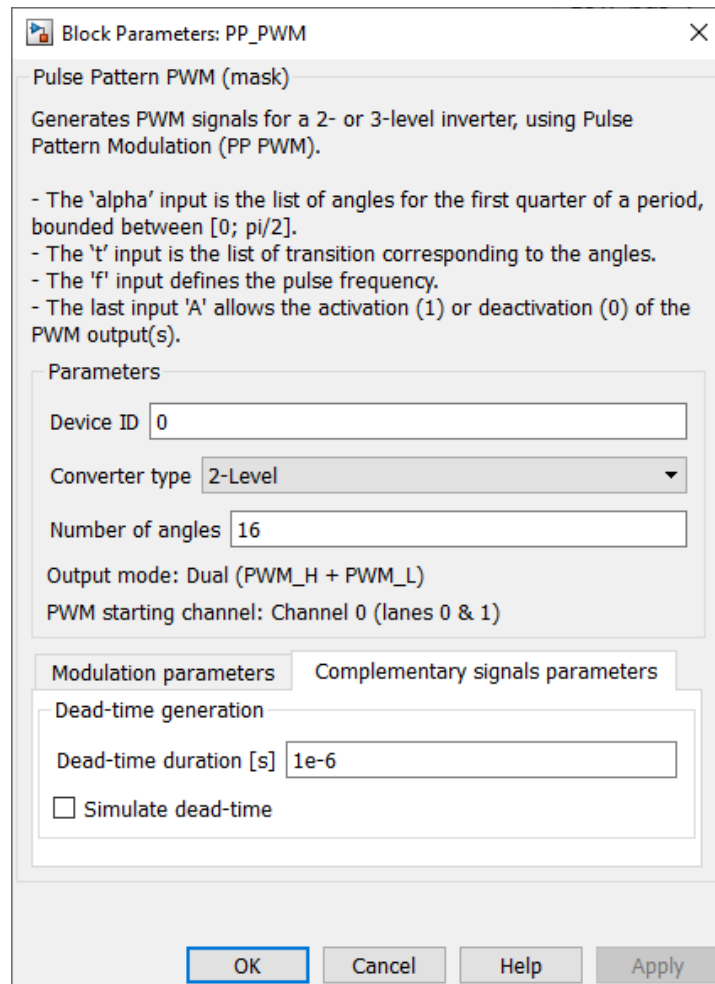
PWM starting channel: Channel 0 (lanes 0 & 1)

Modulation parameters    Complementary signals parameters

PWM activation

☐ Show "activate" input

OK    Cancel    Help    Apply

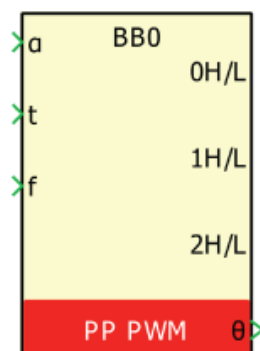


The parameters dead-time and show "activate" input are common to all multilevel PWM blocks and are further documented on the [PWM page](#).

## PLECS PP-PWM block

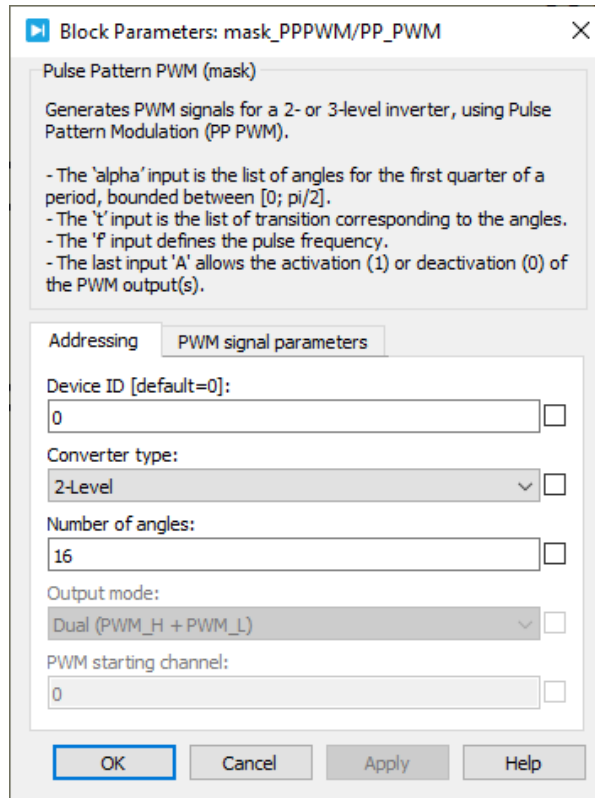
### Signal specification

- The input  $\alpha$  contains the vector of angles between  $]0; \frac{\pi}{2}[$  that define the programmed pattern.
- The input signal  $t$  is a vector containing the transitions for each angle.
- The input signal  $f$  defines the pulse frequency.
- The outputs are the generated PWM signals, according to the block configuration and the phase angle  $\theta$ . The PWM outputs are only used for simulation.
- The phase angle output  $\theta$  is the phase angle for grid synchronization.



# Parameters

- Device ID selects which B-Box/B-Board to address when used in a multi-device configuration.
- Converter type configures the programmed pattern PWM for a two or three-lvl inverter.
- Number of angles configures the number of angles per quarter period.
- PWM activation makes the A (active) signal input visible. If not checked, the PP-PWM block is always active by default (once the controller outputs are enabled).
- Deadtime duration: configures the dead-time duration.



**Block Parameters: mask\_PPPWM/PP\_PWM**

**Pulse Pattern PWM (mask)**  
Generates PWM signals for a 2- or 3-level inverter, using Pulse Pattern Modulation (PP PWM).

- The 'alpha' input is the list of angles for the first quarter of a period, bounded between  $[0; \pi/2]$ .
- The 't' input is the list of transition corresponding to the angles.
- The 'f' input defines the pulse frequency.
- The last input 'A' allows the activation (1) or deactivation (0) of the PWM output(s).

**Addressing** | PWM signal parameters

Device ID [default=0]:

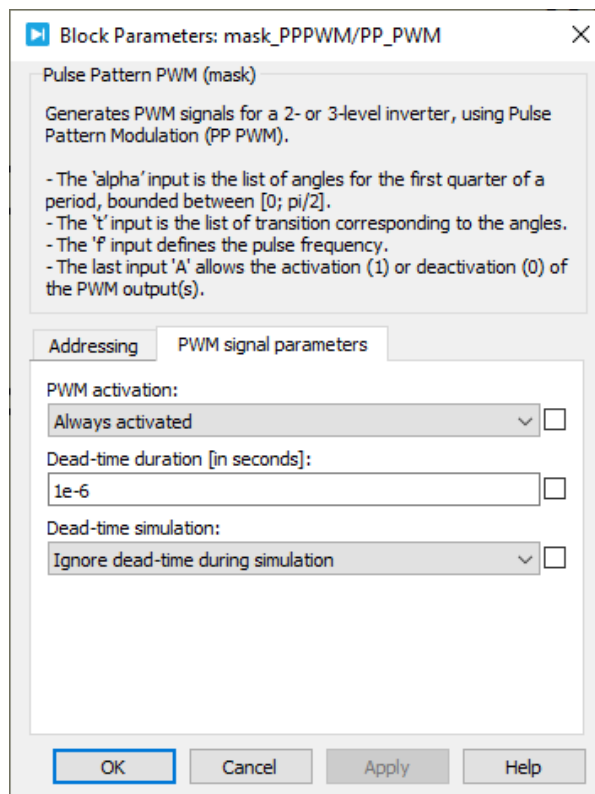
Converter type:

Number of angles:

Output mode:

PWM starting channel:

OK Cancel Apply Help



**Block Parameters: mask\_PPPWM/PP\_PWM**

**Pulse Pattern PWM (mask)**  
Generates PWM signals for a 2- or 3-level inverter, using Pulse Pattern Modulation (PP PWM).

- The 'alpha' input is the list of angles for the first quarter of a period, bounded between  $[0; \pi/2]$ .
- The 't' input is the list of transition corresponding to the angles.
- The 'f' input defines the pulse frequency.
- The last input 'A' allows the activation (1) or deactivation (0) of the PWM output(s).

**Addressing** | **PWM signal parameters**

PWM activation:

Dead-time duration [in seconds]:

Dead-time simulation:

OK Cancel Apply Help

The parameters dead-time and show "activate" input are common to all multilevel PWM blocks and are further documented on the [PWM page](#).

## C++ functions

### Specific to Programmed Pattern PWM

**PpPwm\_ConfigureLevel — Configures the number of levels of the power converter**

```
void PpPwm_ConfigureLevel(unsigned int level, unsigned int device=0);Code language: C++ (cpp)
```

Configures the number of levels of the power converter.

It has to be called in UserInit().

#### Parameters

- level: the number of levels of the power converter (2 or 3)
- device: the B-Box/B-Board to address when used in a multi-device configuration

**PpPwm\_ConfigureNumberOfAngles — Set the number of angles per quarter period**

```
void PpPwm_ConfigureNumberOfAngles(unsigned int numAngles,  
unsigned int device=0);Code language: C++ (cpp)
```

Configure the number of angles (maximum 16) per quarter period.

It has to be called in UserInit().

#### Parameters

- numAngles: number of angles per quarter period
- device: the B-Box/B-Board to address when used in a multi-device configuration

**PpPwm\_ConfigureActivateAsRealTime — Configure the real-time activate**

```
void PpPwm_ConfigureActivateAsRealTime(unsigned int device=0);Code language: C++ (cpp)
```

Makes the activate signals real-time tunable.

It enables the use of PpPwm\_Activate and PpPwm\_Deactivate in the control interrupt.

It has to be called in UserInit().

- device: the B-Box/B-Board to address when used in a multi-device configuration

**PpPwm\_GetCurrentAngle — Retrieves the phase angle**

```
float PpPwm_GetCurrentAngle(unsigned int device=0);Code language: C++ (cpp)
```

Retrieves the value of the phase angle in radian.

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

- device: the B-Box/B-Board to address when used in a multi-device configuration

**PpPwm\_SetAngles — Sets the list of switching angles**

```
bool PpPwm_SetAngles(float angles[], unsigned int variations,  
unsigned int device=0);
```

Code language: C++ (cpp)

Sets the list of switching angles and the corresponding variations

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

- angles: the table containing the list of switching angles
- variations: the variations/transitions corresponding to each angles
- device: the B-Box/B-Board to address when used in a multi-device configuration

#### **PpPwm\_SetFrequency — Set the pulse pattern frequency**

```
void PpPwm_SetFrequency(float frequency, unsigned int device=0);
```

Code language: C++ (cpp)

Sets the frequency of the pulse pattern

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

#### **Parameters**

- frequency: the pulse pattern frequency
- device: the B-Box/B-Board to address when used in a multi-device configuration

## **Functions common to all PWM drivers**

These functions are common to all PWM blocks. Further documentation is available on the [PWM page](#).

#### **PpPwm\_ConfigureDeadTime — Configure the dead time**

```
void PpPwm_ConfigureDeadTime(float deadTime, unsigned int device=0);
```

Code language: C++ (cpp)

Configures the dead-time duration.

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

#### **Parameters**

- deadTime: the dead-time duration in seconds
- device: the B-Box/B-Board to address when used in a multi-device configuration

#### **PpPwm\_Activate — Activate the PWM outputs**

```
void PpPwm_Activate(unsigned int device=0);
```

Code language: C++ (cpp)

Activates the addressed PWM output(s). This function acts on both PWM outputs (PWM high and PWM low).

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

#### **Parameters**

- device: the B-Box/B-Board to address when used in a multi-device configuration

#### **PpPwm\_Deactivate — Deactivate the PWM outputs**

```
void PpPwm_Deactivate(unsigned int device=0);
```

Code language: C++ (cpp)

Deactivates the addressed PWM output(s). This function acts on both PWM outputs (PWM high and PWM low).

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

**Parameters**

- `device`: the B-Box/B-Board to address when used in a multi-device configuration