SS-PWM - Multilevel PWM with Sort-&-Select balancing

SD014 | Posted on October 11, 2021 | Updated on May 27, 2025

Jessy ANÇAY
Sales & Project Engineer
imperix • in

Table of Contents

The SS-PWM peripheral provides a specialized Pulse Width Modulation scheme (PWM) for multilevel converters, which directly integrates means for balancing series-connected submodules. Such an approach has received quick and widespread adoption, notably for Modular Multilevel Converters, Cascaded H-Bridges, and similar converter topologies.

The SS-PWM block generates gate driving signals, which are dynamically assigned to suitable submodules following a “Sort and select” algorithm. The corresponding operating principle is relatively simple:

1. At first, submodules are **sorted** according to their capacitor voltage.
2. Then, the submodules **selected** to contribute to the arm voltage are those with the highest charge level when the power is flowing out of the arm, respectively with the least charge level when the power is flowing into the arm. This way, some balancing is achieved over time so that all submodules converge toward the same charge level.

Multilevel PWM operation

The FPGA-based implementation of the SS-PWM is detailed in TN160. It is inspired by the original patent by R. Marquardt. The implemented multilevel modulator features the following key characteristics:

- All submodules are continuously sorted. In other words, **all ranks are identified**, as opposed to techniques that only identify the most and least charged submodules.
- By design, it is guaranteed that **only one commutation can occur at once**. This ensures that no ineffective modulation actions are made and that switching losses are minimized.
- Both **half-bridge** and **full-bridge** submodules are supported.
- **Up to N=8 submodules** per arm are supported.
- Both **PWM** and **staircase modulation** approaches are supported.

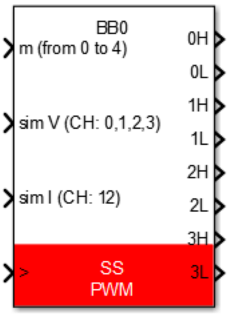
Similarly to other PWM modules, the following parameters of the multilevel PWM module can be configured:

- The sampling frequency is freely configurable (identical to the control interrupt frequency).
- The switching frequency is freely configurable.
- PWM update is possible once or twice per period (when the carrier reaches its minimum and/or maximum).
- Complimentary High and Low signals offer a configurable dead-time generation.
- Multilevel PWM outputs can be activated or deactivated during the operation.

Simulink SS-PWM block

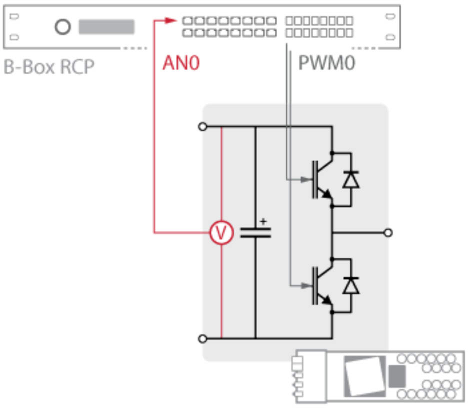
Signal specification

- The input `m` is the modulation signal. Its range is [0; N] in `half-bridge` mode and [-N; N] in `full-bridge` mode. N is the number of submodules per arm.
- The input signal `sim V` is a vector containing the submodules voltages. This input is only effective during simulation. Measurements are directly taken from the corresponding analog input channels during real-time operation.
- The input signal `sim I` corresponds to the arm current and is only used for simulation.
- The input signal > is the clock input and must be connected to the CONFIG block or to an independent CLK block. When connected to a variable-frequency clock, the switching frequency can be tuned during runtime (see PN121).
- The outputs are the generated PWM signals, according to the block configuration. The outputs are only used for simulation.

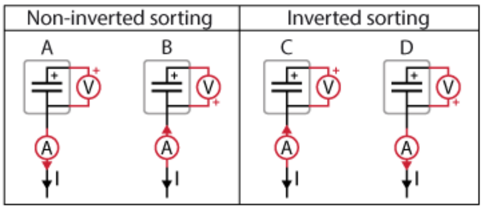


Parameters

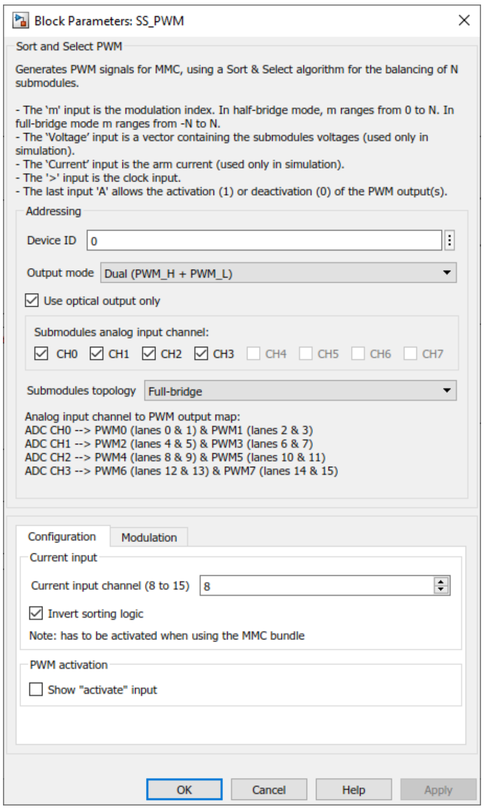
- `Device ID` selects which B-Box/B-Board to address when used in a multi-device configuration.
- `Output mode` selects between complementary signals with a deadtime, a single PWM with a global active signal or a single PWM.
- `Use optical output only` selects if the PWM signals can be addressed only to the optical outputs, or if the electrical outputs (lanes 16 to 31) can also be used.
- `Submodules analog input channel` configures the number of modules per modulator, with a maximum of 8 modules. The checkboxes define the analog input channel connected to each module. The PWM outputs are automatically assigned to each analog input. The following illustration shows the scenario where CH0 is enabled, meaning that a module is connected to analog input 0 and to PWM output 0.
- `Submodules topology` defines if the modules are half-bridge or full-bridge modules.
- `Current input channel` selects the analog input channel, ranging from 8 to 15, of the arm current measurement.
- `Invert sorting logic` inverts the sorting logic of the SS-PWM sorting algorithm. As described by the schematic below, if a positive/negative measured current results in an increase/decrease of measured voltage, respectively, then the logic is non-inverted. On the other hand, the sorting logic must be inverted if a positive/negative measured current results in a decrease/increase in measured voltage, respectively. Also, this parameter being purely defined by the hardware, it is only used for code generation and has no effect in simulation mode.
- `Show "activate" input` makes the `A` (active) signal input visible. If not checked, the SS-PWM block is always active by default (once the controller outputs are enabled).
- Carrier type sets the carrier to TRIANGLE, INVTRIANGLE, *or No Carrier (Staircase modulation).* This option can be used to either generate N+1 or 2N+1 level output waveforms.
- `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. (for *TRIANGLE* and *INVTRIANGLE* carriers only)
- `Deadtime duration`: configures the dead-time duration.



Analog input to PWM output mapping



Sorting logic configuration



Sort and Select PWM

Generates PWM signals for MMC, using a Sort & Select algorithm for the balancing of N submodules.

- The 'm' input is the modulation index. In half-bridge mode, m ranges from 0 to N. In full-bridge mode m ranges from -N to N.
- The 'Voltage' input is a vector containing the submodules voltages (used only in simulation).
- The 'Current' input is the arm current (used only in simulation).
- The '>' input is the clock input.
- The last input 'A' allows the activation (1) or deactivation (0) of the PWM output(s).

Addressing

Device ID: 0

Output mode: Dual (PWM_H + PWM_L)

☑ Use optical output only

Submodules analog input channel:
☑ CH0 ☑ CH1 ☑ CH2 ☑ CH3 □ CH4 □ CH5 □ CH6 □ CH7

Submodules topology: Full-bridge

Analog input channel to PWM output map:
ADC CH0 --> PWM0 (lanes 0 & 1) & PWM1 (lanes 2 & 3)
ADC CH1 --> PWM2 (lanes 4 & 5) & PWM3 (lanes 6 & 7)
ADC CH2 --> PWM4 (lanes 8 & 9) & PWM5 (lanes 10 & 11)
ADC CH3 --> PWM6 (lanes 12 & 13) & PWM7 (lanes 14 & 15)

Current input

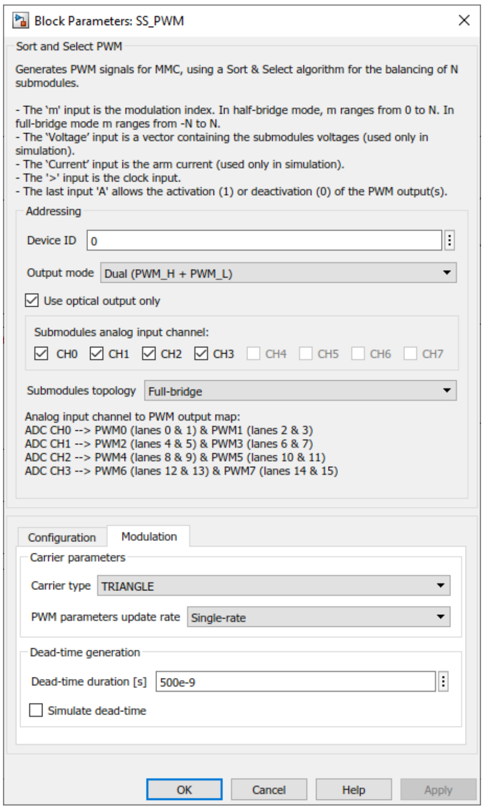
Current input channel (8 to 15): 8

☑ Invert sorting logic

Note: has to be activated when using the MMC bundle

PWM activation

□ Show "activate" input



Sort and Select PWM

Generates PWM signals for MMC, using a Sort & Select algorithm for the balancing of N submodules.

- The 'm' input is the modulation index. In half-bridge mode, m ranges from 0 to N. In full-bridge mode m ranges from -N to N.
- The 'Voltage' input is a vector containing the submodules voltages (used only in simulation).
- The 'Current' input is the arm current (used only in simulation).
- The '>' input is the clock input.
- The last input 'A' allows the activation (1) or deactivation (0) of the PWM output(s).

Addressing

Device ID: 0

Output mode: Dual (PWM_H + PWM_L)

☑ Use optical output only

Submodules analog input channel:
☑ CH0 ☑ CH1 ☑ CH2 ☑ CH3 □ CH4 □ CH5 □ CH6 □ CH7

Submodules topology: Full-bridge

Analog input channel to PWM output map:
ADC CH0 --> PWM0 (lanes 0 & 1) & PWM1 (lanes 2 & 3)
ADC CH1 --> PWM2 (lanes 4 & 5) & PWM3 (lanes 6 & 7)
ADC CH2 --> PWM4 (lanes 8 & 9) & PWM5 (lanes 10 & 11)
ADC CH3 --> PWM6 (lanes 12 & 13) & PWM7 (lanes 14 & 15)

Carrier parameters

Carrier type: TRIANGLE

PWM parameters update rate: Single-rate

Dead-time generation

Dead-time duration [s]: 500e-9

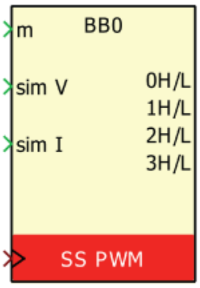
□ Simulate dead-time

The parameters `dead-time` and `show ”activate” input` are common to all multilevel PWM blocks and are further documented on the PWM page.

PLECS SS-PWM block

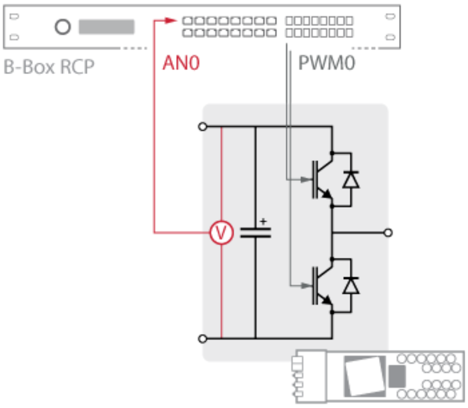
Signal specification

- The input `m` is the modulation signal. Its range is [0; N] in `half-bridge` mode and [-N; N] in `full-bridge` mode. N is the number of submodules per arm.
- The input signal `sim V` is a vector containing the submodules voltages. This input is only effective during simulation. Measurements are directly taken from the corresponding analog input channels during real-time operation.
- The input signal `sim I` corresponds to the arm current and is only used for simulation.
- The input signal `>` is the clock input and must be connected to the CONFIG block or to an independent CLK block. When connected to a variable-frequency clock, the switching frequency can be tuned during runtime (see PN121).
- The outputs are the generated PWM signals, according to the block configuration. The outputs are only used for simulation.

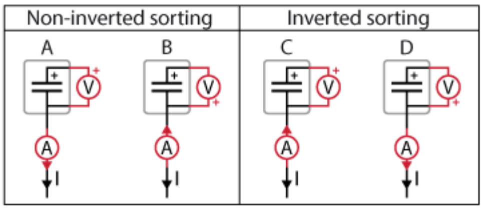


Parameters

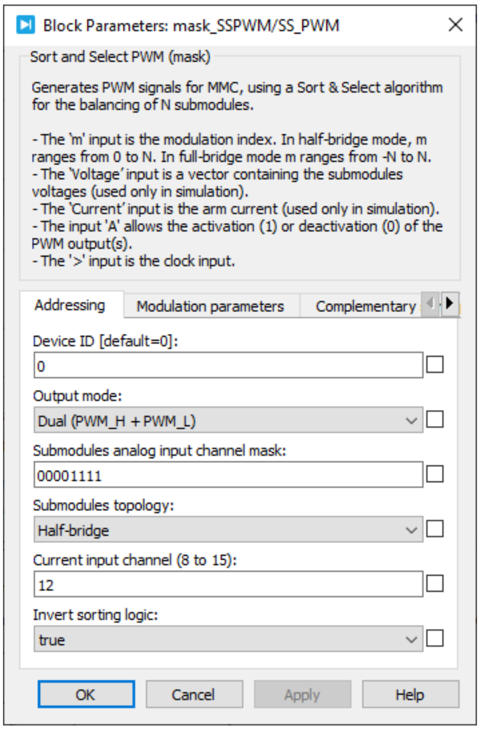
- `Device ID` selects which B-Box/B-Board to address when used in a multi-device configuration.
- `Output mode` selects between complementary signals with a deadtime, a single PWM with a global active signal or a single PWM.
- `Submodules analog input channel mask` configures the number of modules per modulator, with a maximum of 8 modules. The mask must contain 8 bits that define to which analog input a submodule is connected. The mask's LSB corresponds to analog input 0 and the MSB to analog input 7. The PWM outputs are automatically assigned to each analog input. The following illustration shows the scenario where CH0 is enabled, using the following mask: “00000001”, meaning that a module is connected to analog input 0 and to PWM CH 0.
- Submodules topology defines if the modules are half-bridge or full-bridge modules.
- Current input channel selects the analog input channel, ranging from 8 to 15, of the arm current measurement.
- Invert sorting logic inverts the sorting logic of the SS-PWM sorting algorithm. As described by the schematic below, if a positive/negative measured current results in an increase/decrease of measured voltage, respectively, then the logic is non-inverted. On the other hand, the sorting logic must be inverted if a positive/negative measured current results in a decrease/increase in measured voltage, respectively. Also, this parameter being purely defined by the hardware, it is only used for code generation and has no effect in simulation mode.
- Carrier type sets the carrier to TRIANGLE, INVTRIANGLE or No Carrier (Staircase modulation). This option can be used to either generate N+1 or 2N+1 level output waveforms.
- PWM activation makes the A (active) signal input visible. By default, the SS-PWM block is always active (once the controller's outputs are enabled).
- `PWM parameters update rate` selects when the duty-cycle and phase parameters are applied.
  - *Once per PWM period*: they are applied at the end of the carrier period.
  - *Twice per PWM period*: 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)
- `Deadtime duration`: configures the dead-time duration.



Analog input to PWM output mapping



Sorting logic configuration



Block Parameters: mask_SSPWM/SS_PWM

Sort and Select PWM (mask)

Generates PWM signals for MMC, using a Sort & Select algorithm for the balancing of N submodules.

- The 'm' input is the modulation index. In half-bridge mode, m ranges from 0 to N. In full-bridge mode m ranges from -N to N.
- The 'Voltage' input is a vector containing the submodules voltages (used only in simulation).
- The 'Current' input is the arm current (used only in simulation).
- The input 'A' allows the activation (1) or deactivation (0) of the PWM output(s).
- The '>' input is the clock input.

Device ID [default=0]: 0 □

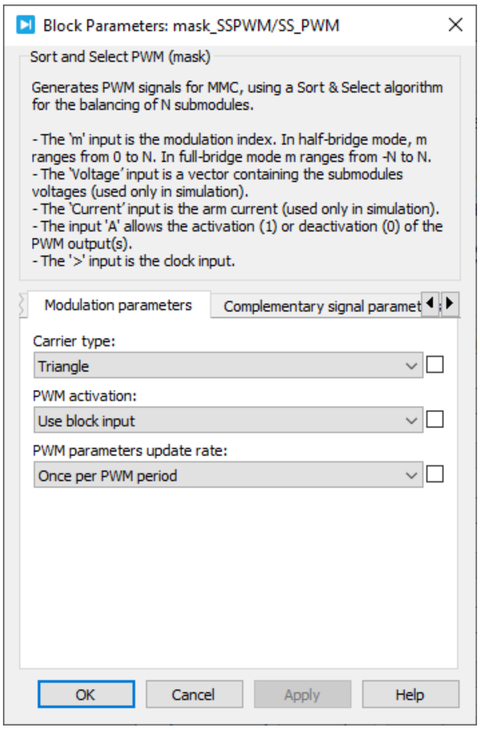
Output mode: Dual (PWM_H + PWM_L) □

Submodules analog input channel mask: 00001111 □

Submodules topology: Half-bridge □

Current input channel (8 to 15): 12 □

Invert sorting logic: true □


Block Parameters: mask_SSPWM/SS_PWM
Modulation parameters
Complementary signal paramet
OK
Cancel
Apply
Help

Sort and Select PWM (mask)

Generates PWM signals for MMC, using a Sort & Select algorithm for the balancing of N submodules.

- The 'm' input is the modulation index. In half-bridge mode, m ranges from 0 to N. In full-bridge mode m ranges from -N to N.
- The 'Voltage' input is a vector containing the submodules voltages (used only in simulation).
- The 'Current' input is the arm current (used only in simulation).
- The input 'A' allows the activation (1) or deactivation (0) of the PWM output(s).
- The '>' input is the clock input.

Carrier type: Triangle □

PWM activation: Use block input □

PWM parameters update rate: Once per PWM period □

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 SS-PWM

`SsPwm_ConfigureOutputMode` **— Configure the PWM output** mode

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

Selects the PWM outputs mode.

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

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

Parameters

- `modulator`: the SS-PWM modulator id (*SS_MODULATOR_0* or *SS_MODULATOR_1*)
- `outMode`: the output mode to use (*COMPLEMENTARY* or *INDEPENDENT*)
- `device`: the B-Box/B-Board to address when used in a multi-device configuration

`SsPwm_ConfigureSubmodules` **— Select the submodules configuration**

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

Selects the modules to connect to the modulator.

The LSB of the mask corresponds to analog input 0 and a maximum of eight modules can be configured.

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

Parameters

- `modulator`: the SS-PWM modulator id (*SS_MODULATOR_0* or *SS_MODULATOR_1*)
- `sm_mask`: 8 bit mask that defines to which analog input a module is connected
- `device`: the B-Box/B-Board to address when used in a multi-device configuration

`SsPwm_ConfigureSubmodulesTopology` — Selects the submodules topology

```
void SsPwm_ConfigureSubmodulesTopology(unsigned int modulator,
tSmTopology topology, unsigned int device=0);Code language: C++ (cpp)
```

Sets the submodules topology to half-bridge or full-bridge.

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

- `modulator`: the SS-PWM modulator id (*SS_MODULATOR_0* or *SS_MODULATOR*_1)
- `topology`: the submodule's topology (HALF_BRIDGE or FULL-BRIDGE)
- `device`: the B-Box/B-Board to address when used in a multi-device configuration

`SsPwm_ConfigureCurrentInput` — Set the current analog input channel

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

Configures the analog input channel to use for the current measurement

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

Parameters

- `modulator`: the SS-PWM modulator id (*SS_MODULATOR_0* or *SS_MODULATOR*_1)
- `input`: the analog input channel number for the current measurement
- `device`: the B-Box/B-Board to address when used in a multi-device configuration

`SsPwm_ConfigureInvertedSorting` — **Invert the sorting logic**

```
void SsPwm_ConfigureInvertedSorting(unsigned int modulator,
bool invert_sorting, unsigned int device=0);Code language: C++ (cpp)
```

Inverts the sorting logic. This parameter is defined by the direction of the voltage and current measurement. Further details are given above.

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

- `modulator`: the SS-PWM modulator id (*SS_MODULATOR_0* or *SS_MODULATOR*_1)
- `invert_sorting`: inverts the sorting logic
- `device`: the B-Box/B-Board to address when used in a multi-device configuration

`SsPwm_ConfigureCarrier` — **Select the carrier waveform**

```
void SsPwm_ConfigureCarrier(unsigned int modulator, tPwmCarrier carrier,
bool enabled, unsigned int device=0);Code language: C++ (cpp)
```

Select the carrier shape of the modulator.

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

- `modulator`: the SS-PWM modulator id (*SS_MODULATOR_0* or *SS_MODULATOR*_1)
- `carrier`: the carrier to use if enabled (*TRIANGLE* or *INVTRIANGLE*)
- `enabled`: enable (use the selected carrier) or disable the carrier (staircase modulation)
- `device`: the B-Box/B-Board to address when used in a multi-device configuration

`SsPwm_ConfigureClock` — **Select a CLOCK**

```
void SsPwm_ConfigureClock(unsigned int modulator, tClock clock,
unsigned int device=0);Code language: C++ (cpp)
```

Connects a clock generator to the multilevel PWM modulator.

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

See: CLK – Clock generator

Parameters

- `modulator`: the SS-PWM modulator id (*SS_MODULATOR_0* or *SS_MODULATOR*_1)
- `clock`: the clock to use (*CLOCK_0*, *CLOCK_1*, *CLOCK_2* or *CLOCK_3*)
- `device`: the B-Box/B-Board to address when used in a multi-device configuration

`SsPwm_ConfigureUpdateRate` — Select an update rate

```
void SsPwm_ConfigureUpdateRate(unsigned int modulator, tPwmRate rate,
unsigned int device=0);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 SS-PWM modulator id (*SS_MODULATOR_0* or *SS_MODULATOR*_1)
- `rate`: the update rate to use (*SINGLE_RATE* or *DOUBLE_RATE*)
- `device`: the B-Box/B-Board to address when used in a multi-device configuration

`SsPwm_ConfigureActivateAsRealTime` — Configures the real-time activate

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

Makes the activate signals real-time tunable.

It enables the use of SsPwm_Activate and SsPwm_Deactivate in the control interrupt.

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

- `modulator`: the SS-PWM modulator id (*SS_MODULATOR_0* or *SS_MODULATOR*_1)
- `device`: the B-Box/B-Board to address when used in a multi-device configuration

`SsPwm_ConfigureActivateOutput` — Select the global active output lane

```
void SsPwm_ConfigureActivateOutput(unsigned int modulator, tPwmOutput output, unsigned int device=0);
```
Code language:

Selects the output lane to use for the global active signal. This function is only used when the output mode is set to *INDEPENDENT*.

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

Parameters

- `modulator`: the SS-PWM modulator id (*SS_MODULATOR_0* or *SS_MODULATOR_1*)
- `output`: the lane number of the global active signal.
- `device`: the B-Box/B-Board to address when used in a multi-device configuration

`SsPwm_SetModulationIndex` — Sets the modulation index

```
void SsPwm_SetModulationIndex(unsigned int modulator, float m,
unsigned int device=0);
```
Code language: C++ (cpp)

Sets the SS-PWM modulation index. Its range is [0; N] in `half-bridge` mode and [-N; N] in `full-bridge` mode, where N is the number of submodules connected to the modulator.

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

- `modulator`: the SS-PWM modulator id (*SS_MODULATOR_0* or *SS_MODULATOR*_1)
- `m`: the modulation index
- `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.

`SsPwm_ConfigureDeadTime` — Configure the dead time

```
void SsPwm_ConfigureDeadTime(tSsModulator 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 SS-PWM modulator id (*SS_MODULATOR_0* or *SS_MODULATOR*_1)
- `deadTime`: the dead-time duration in seconds
- `device`: the B-Box/B-Board to address when used in a multi-device configuration

`SsPwm_Activate` — Activate the PWM outputs

```
void SsPwm_Activate(tSsModulator 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 SS-PWM modulator id (*SS_MODULATOR_0* or *SS_MODULATOR_*1)
- `device`: the B-Box/B-Board to address when used in a multi-device configuration

`SsPwm_Deactivate` — Deactivate the PWM outputs

```cpp
void SsPwm_Deactivate(tSsModulator 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 SS-PWM modulator id (*SS_MODULATOR_0* or *SS_MODULATOR_*1)
- `device`: the B-Box/B-Board to address when used in a multi-device configuration