HAL - Hall sensor input

SD102 | Posted on November 8, 2021 | Updated on May 27, 2025

Simon STROBL
Product Director
imperix

Table of Contents

The hall sensor interface (HAL) block provides access to the commutation signals from Hall effect sensors of a brushless DC motor.

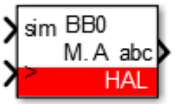
The B-Box RCP supports up to two machines with Hall sensors through the Motor Interface for B-Box RCP. Hall sensors are usually found in three-phase brushless DC machines and provide a 60° resolution on the rotor mechanical position. The hall sensor block provides the commutation signals for the A, B, and C phases. The control algorithm can use these signals to implement a six-step commutation method.

The HAL block is available starting from version 3.7.1.4 of the SDK. The Motor Interface for B-Box RCP is **required** to use this driver.

Simulink block

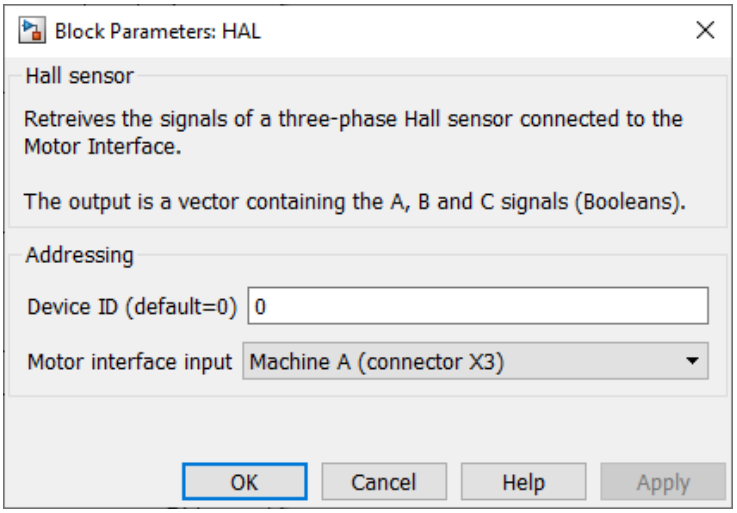
Signal specification

- The output signal is a vector containing the A, B, and C signals (Booleans)
- The `sim` input vector is used in simulation and represents the actual commutation signals, computed by the simulation plant model.
- The > input signal needs to be connected to the sampling clock generated by the CONFIG block to account for the exact sampling instant in simulation.



Parameters

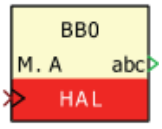
- `Device ID` selects which B-Box/B-Board to address when used in a multi-device configuration.
- `Motor Interface input` selects which connector of the Motor Interface is used as an input.



PLECS block

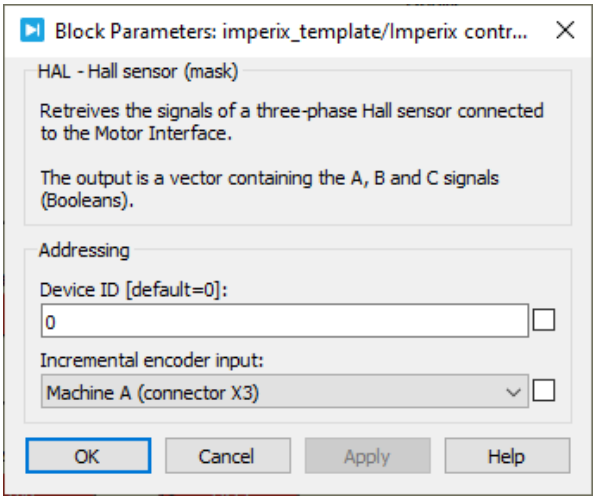
Signal specification

- The output signal is a vector containing the A, B, and C signals (Booleans)
- The `sim` input vector is used in simulation and represents the actual commutation signals, computed by the simulation plant model.
- The > input signal needs to be connected to the sampling clock generated by the CONFIG block to account for the exact sampling instant in simulation.



Parameters

- `Device ID` selects which B-Box/B-Board to address when used in a multi-device configuration.
- `Motor Interface input` selects which connector of the Motor Interface is used as an input.



C++ functions

`MotInt_EnableMotorInterface` **— Enable the drivers of the Motor Interface**

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

Enables the drivers of the Motor Interface.

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

Parameters

- `device`: the id of the addressed device (optional, used in multi-device configuration only).

`Hall_GetPhaseA` **— Read the commutation signal from phase A**

```
bool Hall_GetPhaseA(tMotIntMachine machine, unsigned int device=0);Code language: C++ (cpp)
```

Reads the commutation signal from the Hall sensor of phase A.

It has to be called during the control interrupt.

Parameters

- `machine`: the machine to configure (*MACHINE_A* or *MACHINE_B*).
- `device`: the id of the addressed device (optional, used in multi-device configuration only).

`Hall_GetPhaseB` **— Read the commutation signal from phase** B

```
bool Hall_GetPhaseB(tMotIntMachine machine, unsigned int device=0);Code language: C++ (cpp)
```

Reads the commutation signal from the Hall sensor of phase B.

It has to be called during the control interrupt.

Parameters

- `machine`: the machine to configure (*MACHINE_A* or *MACHINE_B*).
- `device`: the id of the addressed device (optional, used in multi-device configuration only).

`Hall_GetPhaseC` **— Read the commutation signal from phase** C

```
bool Hall_GetPhaseC(tMotIntMachine machine, unsigned int device=0);Code language: C++ (cpp)
```

Reads the commutation signal from the Hall sensor of phase C.

It has to be called during the control interrupt.

Parameters

- `machine`: the machine to configure (*MACHINE_A* or *MACHINE_B*).
- `device`: the id of the addressed device (optional, used in multi-device configuration only).