EnDat - Digital encoder input

SD109 | Posted on January 30, 2026 | Updated on January 30, 2026

François LEDENT
Development Engineer
imperix

Table of Contents

The EnDat block instantiates an EnDat master to communicate with compatible digital encoders and similar digital sensors, typically in motor drive applications.

This block is only compatible with the B-Box4, which provides two RS-485-based serial ports (SERIAL A and SERIAL B). Both ports can interface with sensors using three widely adopted communication protocols: SSI, BiSS-C and EnDat2.2 (this page).

EnDat protocol in a nutshell

The EnDat protocol is a master-slave serial interface standard developed by HEIDENHAIN and widely used in industrial applications.

Similarly to SSI and BiSS-C, the interface is composed of two signals, CLOCK (driven by the master) and DATA (bidirectional), regardless of the resolution (i.e., number of bits per revolution) of the sensor. Differential signaling improves the noise immunity and allows for longer cable runs, making it suitable for industrial environments. Starting from version 2.2, EnDat supports propagation delay compensation and a clock frequency up to 16MHz, therefore faster than SSI or BiSS-C.

EnDat provides room for a control mechanism to access the slave's configuration memory and transmits special commands. However, this mechanism significantly increases complexity and is currently not supported in the imperix implementation.

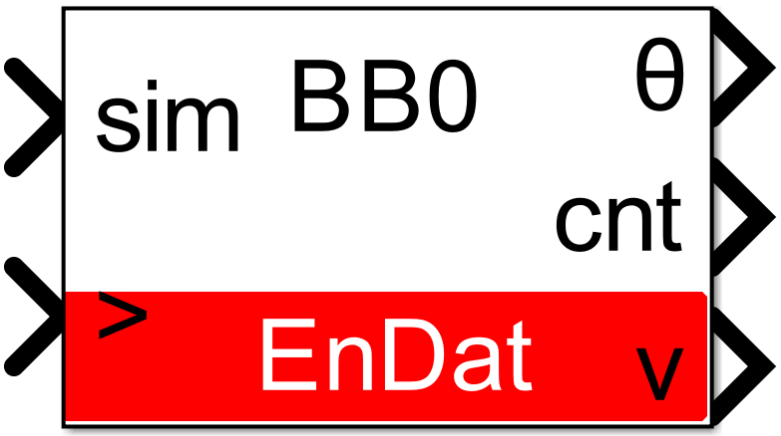
Strengths and limitations

Similarly to SSI and BiSS-C, EnDat can be easily integrated and performs reliably even in electrically noisy environments. The synchronous data transfer provides accurate, low-latency position feedback, making it well suited for real-time control. It supports higher clock frequencies (up to 16 MHz) and allows for the compensation of the propagation delay.

Simulink block

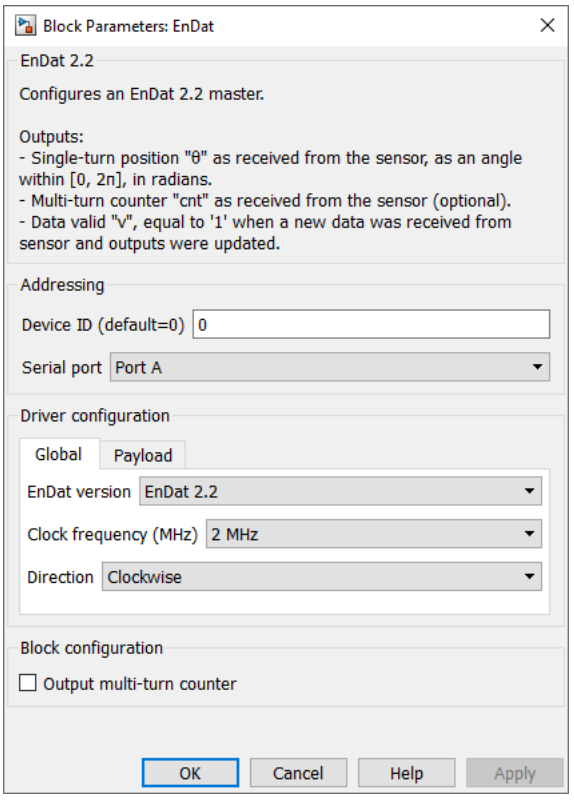
Signal specification

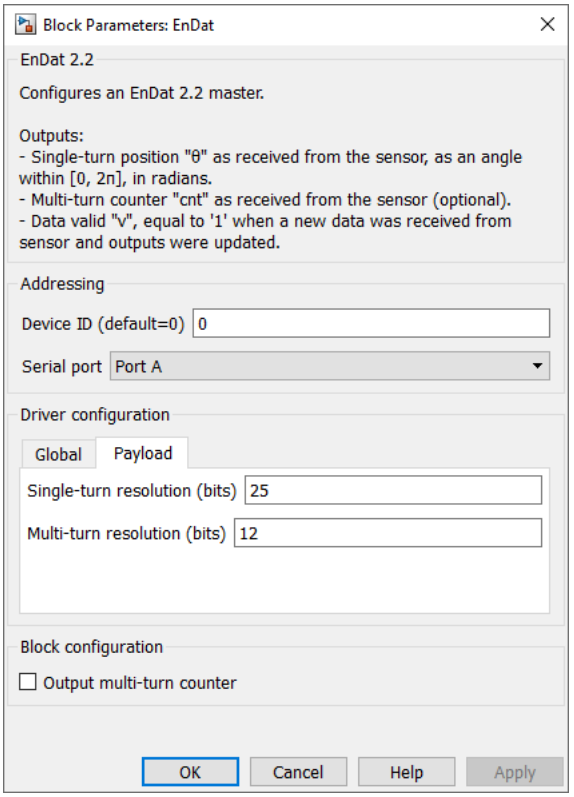
- The output θ is the single-turn position, in radians, in $[0, 2\pi]$.
- The output `cnt` is the multi-turn counter, as received from the sensor. This port is hidden by default but it can be shown using the `Output multi-turn counter` checkbox.
- The output v is the data valid signal, set to 1 each time a new data are available.
- The input `sim` is a 2-dimensional vector used in simulation and represents the angle value (in radians) and multi-turn counter (-) computed by the simulation plant model.
- The > input signal needs to be connected to the sampling clock generated by the CONFIG block.



Parameters

- `Device ID` selects which device to address when used in a multi-device configuration.
- `Serial port` selects the serial port.
- `EnDat version` specifies the EnDat version of the sensor, either EnDat 2.2 or EnDat 2.1.
- `Clock frequency` specifies the transmission clock frequency for the BiSS-C communication, in MHz. The clock frequency cannot exceed 10 MHz, as defined by the BiSS-C standard.
- `Direction` specifies the direction as clockwise or counterclockwise. When counterclockwise is selected, the output angle θ is 2π-θ', where θ' is the angle received from the sensor.
- `Single-turn resolution` specifies the resolution (i.e., number of bits) for the single-turn position. The single-turn resolution cannot exceed 32 bits.
- `Multi-turn resolution` specifies the resolution (i.e., number of bits) for the multi-turn counter. The multi-turn resolution cannot exceed 32 bits.
- `Output multi-turn counter` shows or hides the multi-turn counter output `cnt`.

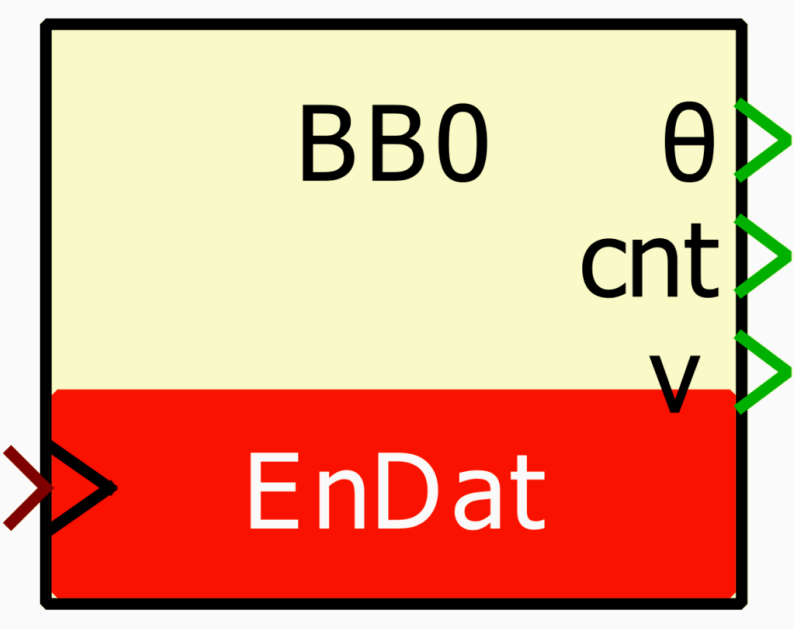




PLECS block

Signal specification

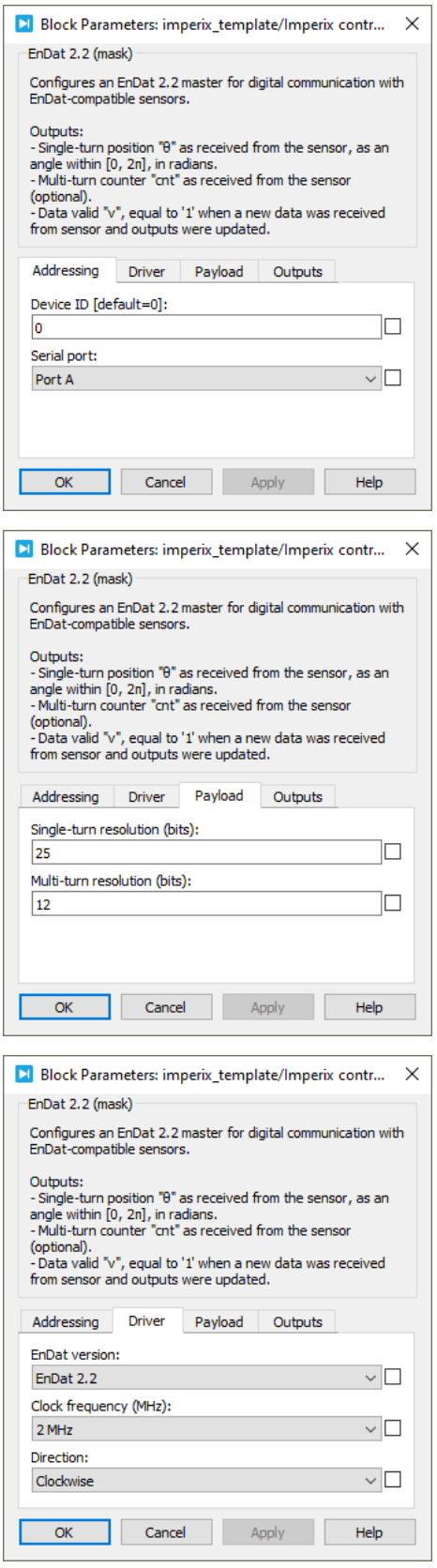
- The output θ is the single-turn position, in radians, in [0, 2π].
- The output `cnt` is the multi-turn counter, as received from the sensor. This port is hidden by default but it can be shown using the `Output multi-turn counter` checkbox.
- The output v is the data valid signal, set to 1 each time a new data are available.
- The input `sim` is a 2-dimensional vector used in simulation and represents the angle value (in radians) and multi-turn counter (-) computed by the simulation plant model.
- The > input signal needs to be connected to the sampling clock generated by the CONFIG block.

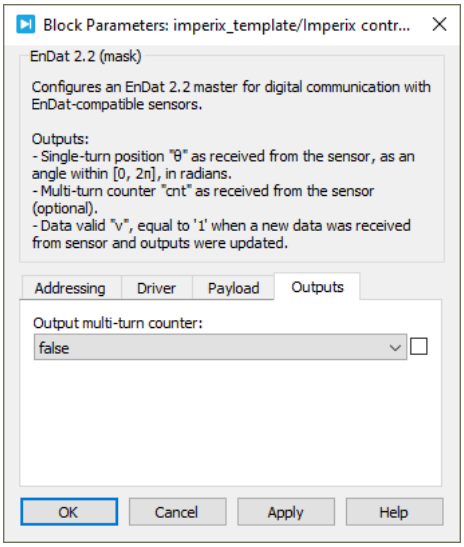


Parameters

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

- `Serial port` selects the serial port.
- `EnDat version` specifies the EnDat version of the sensor, either EnDat 2.2 or EnDat 2.1.
- `Clock frequency` specifies the transmission clock frequency for the BiSS-C communication, in MHz. The clock frequency cannot exceed 10 MHz, as defined by the BiSS-C standard.
- `Direction` specifies the direction as clockwise or counterclockwise. When counterclockwise is selected, the output angle θ is 2π-θ', where θ' is the angle received from the sensor.
- `Single-turn resolution` specifies the resolution (i.e., number of bits) for the single-turn position. The single-turn resolution cannot exceed 32 bits.
- `Multi-turn resolution` specifies the resolution (i.e., number of bits) for the multi-turn counter. The multi-turn resolution cannot exceed 32 bits.
- `Output multi-turn counter` shows or hides the multi-turn counter output `cnt`.





C++ functions

`EnDat_Init` — Initialize the EnDat master

```
void EnDat_Init(unsigned int port, unsigned int device);Code language: C++ (cpp)
```

This function initializes the serial port *port* of the device *device* for the EnDat communication protocol.

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

TPI Rev. F: Only serial port A is available.

Parameters

- `port`: Serial port, 0 (port A) or 1 (port B)
- `device`: imperix device to address in a multi-device configuration (default: 0)

`EnDat_ConfigureEnDatVersion` — Configure the EnDat version

```
void EnDat_ConfigureEnDatVersion(tRs485EndatVersion version, unsigned int port, unsigned int device);Code language: (
```

This function configures the EnDat version to 2.1 or 2.2. The default version is 2.2.

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

TPI Rev. F: Only serial port A is available.

Parameters

- `version`: EnDat version, 2.1 (RS485_ENDAT_21) or 2.2 (RS485_ENDAT_22)
- `port`: serial port, 0 (port A) or 1 (port B)
- `device`: imperix device to address in a multi-device configuration (default: 0)

`EnDat_ConfigureClkFrequency` — Configure the transmission clock frequency

```
void EnDat_ConfigureClkFrequency(tRs485EndatFreq clk_frequency, unsigned int port, unsigned int device);Code languag
```

This function configures the frequency of the transmission clock for the EnDat communication on the serial port *port* of the device *device*. The clock frequency be 100 kHz, 200 kHz, 1 MHz, 2 MHz, 4 MHz, 8 MHz or 16 MHz.

Frequencies higher than 2MHz should not be used with EnDat 2.1.

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

TPI Rev. F: Only serial port A is available.

Parameters

- `clk_frequency`: transmission clock frequency
  - 100kHz (RS485_ENDAT_100KHZ)
  - 200kHz (RS485_ENDAT_200KHZ)
  - 1MHz (RS485_ENDAT_1MHZ)
  - 2MHz (RS485_ENDAT_2MHZ)

    - 4MHz (RS485_ENDAT_4MHZ)
    - 8MHz (RS485_ENDAT_8MHZ)
    - 16MHz (RS485_ENDAT_16MHZ)
- `port`: serial port, 0 (port A) or 1 (port B)
- `device`: imperix device to address in a multi-device configuration (default: 0)

`EnDat_ConfigureDirection` — Configure the rotation direction

```
void EnDat_ConfigureDirection(tRs485Direction direction, unsigned int port, unsigned int device);Code language: C++ (
```

This function configures the direction for the EnDat communication on the serial port *port* of the device *device*, as *clockwise* or *counterclockwise*. When *counterclockwise* is selected, the output angle θ is 2π-θ', where θ' is the angle received from the sensor.

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

TPI Rev. F: Only serial port A is available.

Parameters

- `direction`: direction (RS485_CW or RS485_CCW)
- `port`: serial port, 0 (port A) or 1 (port B)
- `device`: imperix device to address in a multi-device configuration (default: 0)

`EnDat_ConfigureSingleturnResolution` — Configure the single-turn resolution

```
void EnDat_ConfigureSingleturnResolution(unsigned int resolution, unsigned int port, unsigned int device);Code langu
```

This function configures the single-turn resolution (i.e., number of bits of the single-turn position) for the EnDat communication on the serial port *port* of the device *device*.

The single-turn and multi-turn resolution fields are each limited to 32 bits, with a maximum combined size of 48 bits.

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

TPI Rev. F: Only serial port A is available.

Parameters

- `resolution`: single-turn resolution, i.e. number of bits of the single-turn position
- `port`: serial port, 0 (port A) or 1 (port B)
- `device`: imperix device to address in a multi-device configuration (default: 0)

`EnDat_ConfigureMultiturnResolution` — Configure the multi-turn resolution

```
void EnDat_ConfigureMultiturnResolution(unsigned int resolution, unsigned int port, unsigned int device);Code langua
```

This function configures the multi-turn resolution (i.e., number of bits of the multi-turn counter) for the EnDat communication on the serial port *port* of the device *device*.

The single-turn and multi-turn resolution fields are each limited to 32 bits, with a maximum combined size of 48 bits.

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

TPI Rev. F: Only serial port A is available.

Parameters

- `resolution`: multi-turn resolution, i.e. number of bits of the multi-turn counter
- `port`: serial port, 0 (port A) or 1 (port B)
- `device`: imperix device to address in a multi-device configuration (default: 0)

`EnDat_ReadFrame` — Read the current frame

```
void EnDat_ReadFrame(unsigned int port, unsigned int device);Code language: C++ (cpp)
```

This function triggers the CPU to read and interpret the last frame received from the sensor.

It must be called in the control interrupt routine.

TPI Rev. F: Only serial port A is available.

Parameters

- `port`: serial port, 0 (port A) or 1 (port B)
- `device`: imperix device to address in a multi-device configuration (default: 0)

`EnDat_GetPosition` — Get the sensor's position

```
float EnDat_GetPosition(unsigned int port, unsigned int device);Code language: C++ (cpp)
```

This function returns the single-turn position received from the sensor as an angle, in radians, in [0, 2π].

It must be called in the control interrupt routine, after *EnDat_ReadFrame*.

TPI Rev. F: Only serial port A is available.

Parameters

- `port`: serial port, 0 (port A) or 1 (port B)
- `device`: imperix device to address in a multi-device configuration (default: 0)

`EnDat_GetMultiturnCounter` — Get the sensor's multi-turn counter

```
unsigned int EnDat_GetMultiturnCounter(unsigned int port, unsigned int device);Code language: C++ (cpp)
```

This function returns the multi-turn counter received from the sensor.

It must be called in the control interrupt routine, after *EnDat_ReadFrame*.

TPI Rev. F: Only serial port A is available.

Parameters

- `port`: serial port, 0 (port A) or 1 (port B)
- `device`: imperix device to address in a multi-device configuration (default: 0)

`EnDat_GetDataValid` — Get the data valid flag

```
unsigned int EnDat_GetDataValid(unsigned int port, unsigned int device);Code language: C++ (cpp)
```

This function returns the data valid flag indicating if a new data was received from the sensor. The flag being '1' means that the outputs of *EnDat_GetPosition* and *EnDat_GetMultiturnCounter* were updated since last interruption. If the flag is '0', the outputs were not updated and have the same value.

It must be called in the control interrupt routine, after *EnDat_ReadFrame*.

TPI Rev. F: Only serial port A is available.

Parameters

- `port`: serial port, 0 (port A) or 1 (port B)
- `device`: imperix device to address in a multi-device configuration (default: 0)