ETH out - Ethernet output mailbox

SD019 | Posted on April 2, 2021 | Updated on July 24, 2025

Stéphane LOVEJOY
Senior Software Developer
imperix

Table of Contents

The Ethernet output mailbox block allows sending up to 1024 bytes of data via Ethernet using the UDP/IP protocol (in SDK versions prior to 2025.2, the size limit was 32-bit). To receive data via Ethernet using the UDP/IP protocol, the Ethernet input mailbox should be used.

It supports two operating modes:

- **On-demand mode**: the user manually triggers the message transmissions.
- **Periodical mode**: the message is sent periodically, whether the data has been changed or not. The user can configure the transmission frequency.

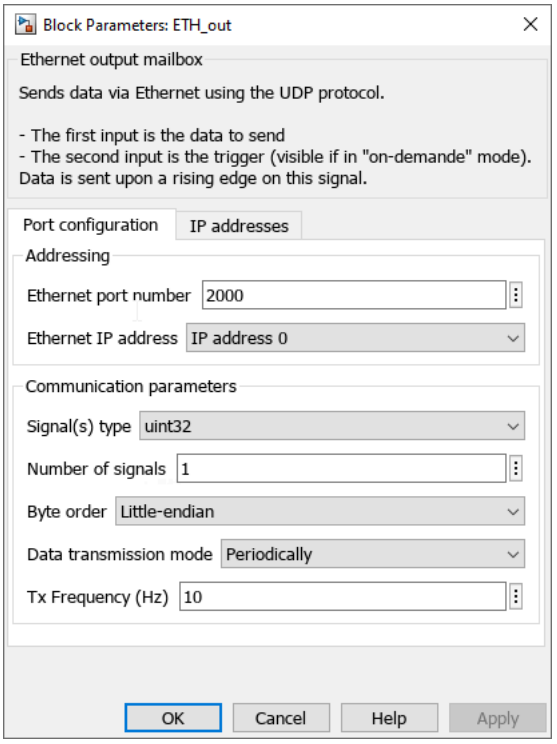
Simulink block

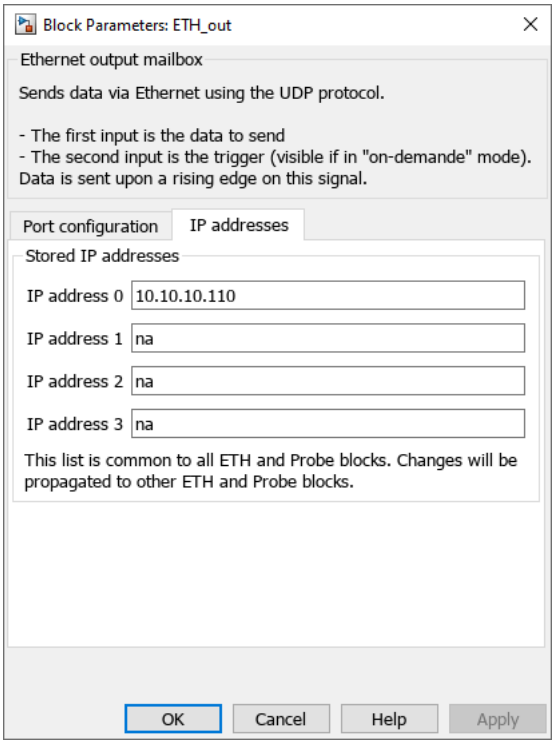
Signal specification

- The first input is the data to send. The accepted data type is specified by the `Signal coding format` parameter. This data can be provided as a vector, with its size defined by the `Number of signals` parameter. The maximum supported data size is 1024 bytes. (e.g. with `Signal coding format` set to uint32 and `Number of signals` set to 100, the data size will be 4 x 100 = 400 bytes)
- The second input is the *send data* signal. It is used to initiate a data transmission when the **on-demand** mode has been selected. Data is sent upon a rising edge on this signal.

Parameters

- `Ethernet port number`: sets the destination port to which the data will be sent.
- `Ethernet IP address` : selects the IP address to whom the data will be sent.
- `Signal coding format`: defines the data type accepted in the data input (int8, int16, int32, uint8, uint16, uint32, float32, or float64).
- `Number of signals`: specifies the vector size of the data to be sent.
- `Byte order` : defines the byte order in which the data will be sent. (little-endian or big-endian)
- `Tx frequency` : sets the data transmission frequency if the **periodical mode** has been selected.

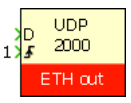




PLECS block

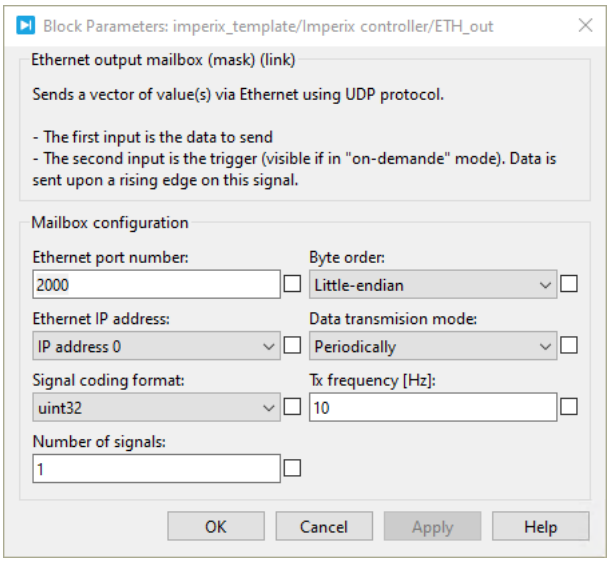
Signal specification

- The first input is the data to send. The accepted data type is specified by the `Signal coding format` parameter. This data can be provided as a vector, with its size defined by the `Number of signals` parameter. The maximum supported data size is 1024 bytes. (e.g. with `Signal coding format` set to uint32 and `Number of signals` set to 100, the data size will be 4 x 100 = 400 bytes)
- The second input is the *send data* signal. It is used to initiate a data transmission when the **on-demand** mode has been selected. Data is sent upon a rising edge on this signal.



Parameters

- `Ethernet port number`: sets the destination port to which the data will be sent.
- `Ethernet IP address`: selects the IP address to whom the data will be sent.
- `Signal coding format`: defines the data type accepted in the data input (int8, int16, int32, uint8, uint16, uint32, float32, or float64).
- `Number of signals`: specifies the vector size of the data to be sent.
- `Byte order`: defines the byte order in which the data will be sent. (little-endian or big-endian)
- `Tx frequency`: sets the data transmission frequency if the **periodical mode** has been selected.



The Ethernet IP addresses can be configured in the Imperix Controllers' target window (*Coder → Coder option → Target*).

C++ functions

`Eth_ConfigureOutputMailbox` **— Configure an Ethernet output mailbox**

```
bool Eth_ConfigureOutputMailbox(unsigned int mailboxId, unsigned int port, const char* ipAddress, float maxTxFrequ
```

Configures an Ethernet UDP output mailbox.

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

Parameters

- `mailboxId`: a unique id used to distinguish mailboxes from each other. This id must be unique throughout the code for all ETH and CAN input/output mailboxes.
- `port`: the destination UDP port number to which the data will be sent.
- `ipAddress`: the destination IP address to whom the data will be sent.
- `maxTxFrequency`: maximal frequency at which data can be sent. The frequency must be a multiple of the main interrupt frequency. If the requested frequency is not achievable, it will automatically be set to the closest valid frequency.
- `endianness`: defines the bytes order. BIG_ENDIAN (most significant byte first) or LITTLE_ENDIAN (least significant byte first)
- `msgLength`: size in bytes of the message, for best perfomance it is advised to keep it under 1024 bytes

Return value

- `bool`: returns false if too many output mailboxes were created or if the `port` was already assigned to another ETH output mailbox.

`Eth_Write` **— Write**

```
int Eth_Write(unsigned int mailboxId, void* data, size_t size);Code language: C++ (cpp)
```

```
int Eth_Write(unsigned int mailboxId, unsigned int dataLow);Code language: C++ (cpp)
```

```
int Eth_Write(unsigned int mailboxId, int dataLow);Code language: C++ (cpp)
```

```
int Eth_Write(unsigned int mailboxId, float dataLow);Code language: C++ (cpp)
```

These functions are used to send data on Ethernet using UDP.

They have to be called during the control interrupt.

Parameters

- `maildboxId`: a unique ID used to distinguish mailboxes from each other. This ID must be unique throughout the code for all ETH and CAN input/output mailboxes.
- `data`: data which will be sent.
- `size`: size in bytes of the data to copy, must match the size declared during mailbox creation

Return value

- `int`: returns 1 if the data has been loaded in the write buffer. Returns 0 otherwise.