# Communication

*class* **`ingeniamotion.communication.Communication`**(*motion_controller*)

Communication.

**`connect_servo_eoe`**(*ip*, *dict_path=None*, *alias='default'*, *port=1061*, *servo_status_listener=False*, *net_status_listener=False*)

Connect to target servo by Ethernet over EtherCAT

**Parameters:**

- **ip** ( `str` ) – servo IP.
- **dict_path** ( `Optional` [ `str` ]) – servo dictionary path.
- **alias** ( `str` ) – servo alias to reference it. `default` by default.
- **port** ( `int` ) – servo port. `1061` by default.
- **servo_status_listener** ( `bool` ) – Toggle the listener of the servo for its status, errors, faults, etc.
- **net_status_listener** ( `bool` ) – Toggle the listener of the network status, connection and disconnection.

**Raises:**

- TypeError – If the dict_path argument is missing.
- FileNotFoundError – If the dict file doesn't exist.
- ingenialink.exceptions.ILError – If the servo's IP or port is incorrect.

**Return type:** `None`

**`connect_servo_ethernet`**(*ip*, *dict_path*, *alias='default'*, *port=1061*, *connection_timeout=1*, *servo_status_listener=False*, *net_status_listener=False*)

Connect to target servo by Ethernet

**Parameters:**

- **ip** ( `str` ) – servo IP
- **dict_path** ( `str` ) – servo dictionary path.
- **alias** ( `str` ) – servo alias to reference it. `default` by default.
- **port** ( `int` ) – servo port. `1061` by default.
- **connection_timeout** ( `int` ) – Timeout in seconds for connection. `1` seconds by default.
- **servo_status_listener** ( `bool` ) – Toggle the listener of the servo for its status, errors, faults, etc.
- **net_status_listener** ( `bool` ) – Toggle the listener of the network status, connection and disconnection.

**Raises:**

- FileNotFoundError – If the dict file doesn't exist.
- ingenialink.exceptions.ILError – If the servo's IP or port is incorrect.

**Return type:** `None`

`connect_servo_virtual`(*dict_path=None*, *alias='default'*, *port=1061*, *connection_timeout=1*, *servo_status_listener=False*, *net_status_listener=False*)

Connect to the virtual drive using an ethernet communication.

**Parameters:**

- **dict_path** ( `Optional` [ `str` ]) – servo dictionary path.
- **alias** ( `str` ) – servo alias to reference it. `default` by default.
- **port** ( `int` ) – servo port. `1061` by default.
- **connection_timeout** ( `int` ) – Timeout in seconds for connection. `1` seconds by default.
- **servo_status_listener** ( `bool` ) – Toggle the listener of the servo for its status, errors, faults, etc.
- **net_status_listener** ( `bool` ) – Toggle the listener of the network status, connection and disconnection.

**Raises:**

- FileNotFoundError – If the dict file doesn't exist.
- ingenialink.exceptions.ILError – If the servo's IP or port is incorrect.

**Return type:** `None`

`connect_servo_eoe_service`(*ifname*, *dict_path*, *ip='192.168.3.22'*, *slave=1*, *port=1061*, *alias='default'*, *servo_status_listener=False*, *net_status_listener=False*)

Connect to target servo by Ethernet over EtherCAT

**Parameters:**

- **ifname** ( `str` ) – interface name. It should have format `\Device\NPF_[...]` .
- **dict_path** ( `str` ) – servo dictionary path.
- **ip** ( `str` ) – IP address to be assigned to the servo.
- **slave** ( `int` ) – slave index. `1` by default.
- **port** ( `int` ) – servo port. `1061` by default.
- **alias** ( `str` ) – servo alias to reference it. `default` by default.
- **servo_status_listener** ( `bool` ) – Toggle the listener of the servo for its status, errors, faults, etc.
- **net_status_listener** ( `bool` ) – Toggle the listener of the network status, connection and disconnection.

**Raises:**

- NotImplementedError – If this method is run in Linux.
- FileNotFoundError – If the dict file doesn't exist.
- ValueError – ip must be a subnetwork of 192.168.3.0/24
- ingenialink.exceptions.ILError – If the EoE service is not running
- ingenialink.exceptions.ILError – If the EoE service cannot be started on the network interface.

**Return type:** `None`

**connect_servo_eoe_service_interface_ip**(*interface_ip*, *dict_path*, *ip='192.168.3.22'*, *slave=1*, *port=1061*, *alias='default'*, *servo_status_listener=False*, *net_status_listener=False*)

Connect to target servo by Ethernet over EtherCAT

**Parameters:**

- **interface_ip** ( `str` ) – IP of the interface to be connected to.
- **dict_path** ( `str` ) – servo dictionary path.
- **ip** ( `str` ) – IP address to be assigned to the servo.
- **slave** ( `int` ) – slave index. `1` by default.
- **port** ( `int` ) – servo port. `1061` by default.
- **alias** ( `str` ) – servo alias to reference it. `default` by default.
- **servo_status_listener** ( `bool` ) – Toggle the listener of the servo for its status, errors, faults, etc.
- **net_status_listener** ( `bool` ) – Toggle the listener of the network status, connection and disconnection.

**Raises:**

- TypeError – If the dict_path argument is missing.
- IndexError – If interface index is out of range.
- FileNotFoundError – If the dict file doesn't exist.
- ValueError – ip must be a subnetwork of 192.168.3.0/24
- ingenialink.exceptions.ILError – If the EoE service is not running
- ingenialink.exceptions.ILError – If the EoE service cannot be started on the network interface.

**Return type:** `None`

**`connect_servo_comkit`**(*ip*, *coco_dict_path*, *moco_dict_path*, *alias='default'*, *port=1061*, *connection_timeout=1*, *servo_status_listener=False*, *net_status_listener=False*)

Connect to target servo using a COM-KIT

**Parameters:**

- **ip** ( `str` ) – servo IP
- **coco_dict_path** ( `str` ) – COCO dictionary path.
- **moco_dict_path** ( `str` ) – MOCO dictionary path.
- **alias** ( `str` ) – servo alias to reference it. `default` by default.
- **port** ( `int` ) – servo port. `1061` by default.
- **connection_timeout** ( `int` ) – Timeout in seconds for connection. `1` seconds by default.
- **servo_status_listener** ( `bool` ) – Toggle the listener of the servo for its status, errors, faults, etc.
- **net_status_listener** ( `bool` ) – Toggle the listener of the network status, connection and disconnection.

**Raises:**

- FileNotFoundError – If a dict file doesn't exist.
- ingenialink.exceptions.ILError – If the servo's IP or port is incorrect.

**Return type:** `None`

**`get_ifname_from_interface_ip`**(*address*)

Returns interface name based on the address ip of an interface.

**Parameters:**

- **address** ( `str` ) – ip expected adapter is expected to
- **configured with.** (*be*) –

**Raises:**

- ValueError – In case the input is not valid or the adapter
- **is not found.** –

**Return type:** `str`

**Returns:** Ifname of the controller.

**`get_ifname_by_index`**(*index*)

Return interface name by index.

**Parameters:** **index** ( `int` ) – position of interface selected in `get_interface_name_list()` .

**Return type:** `str`

**Returns:** Real name of selected interface. It can be used for functions `connect_servo_eoe_service()` and `connect_servo_ethercat()` .

**Raises:** IndexError – If interface index is out of range.

*static* **`get_interface_name_list`**()

Get interface list.

**Return type:** `List` [ `str` ]

**Returns:** List with interface readable names.

**`get_available_canopen_devices`**()

Return the list of available CAN devices (those connected and with drivers installed).

**Returns:**

{

CAN_DEVICE.KVASER: [0, 1] CAN_DEVICE.PCAN: [0]

}

**Return type:** Dict of available CAN devices and channels. For example

**`connect_servo_eoe_service_interface_index`**(*if_index*, *dict_path*, *ip='192.168.3.22'*, *slave=1*, *port=1061*, *alias='default'*, *servo_status_listener=False*, *net_status_listener=False*)

Connect to target servo by Ethernet over EtherCAT

**Parameters:**

- **if_index** ( `int` ) – interface index in list given by function `get_interface_name_list()` .
- **dict_path** ( `str` ) – servo dictionary path.
- **ip** ( `str` ) – IP address to be assigned to the servo.
- **slave** ( `int` ) – slave index. `1` by default.
- **port** ( `int` ) – servo port. `1061` by default.
- **alias** ( `str` ) – servo alias to reference it. `default` by default.
- **servo_status_listener** ( `bool` ) – Toggle the listener of the servo for its status, errors, faults, etc.
- **net_status_listener** ( `bool` ) – Toggle the listener of the network status, connection and disconnection.

**Raises:**

- TypeError – If the dict_path argument is missing.
- IndexError – If interface index is out of range.
- FileNotFoundError – If the dict file doesn't exist.
- ValueError – ip must be a subnetwork of 192.168.3.0/24
- ingenialink.exceptions.ILError – If the EoE service is not running
- ingenialink.exceptions.ILError – If the EoE service cannot be started on the network interface.

**Return type:** `None`

`scan_servos_eoe_service`(*ifname*)

Return a List of available servos.

**Parameters:** **ifname** ( `str` ) – interface name. It should have format `\Device\NPF_[...]` .

**Return type:** `List` [ `int` ]

**Returns:** Drives available in the target interface.

**Raises:**

- NotImplementedError – If this method is run in Linux.
- ingenialink.exceptions.ILError – If the EoE service is not running
- TypeError – If some parameter has a wrong type.

`scan_servos_eoe_service_interface_index`(*if_index*)

Return a list of available servos.

**Parameters:** **if_index** ( `int` ) – interface index in list given by function `get_interface_name_list()` .

**Return type:** `List` [ `int` ]

**Returns:** Drives available in the target interface.

**Raises:**
- IndexError – If interface index is out of range.
- ingenialink.exceptions.ILError – If the EoE service is not running

**`connect_servo_canopen`**(*can_device*, *dict_path*, *node_id*, *baudrate=<CAN_BAUDRATE.Baudrate_1M: 1000000>*, *channel=0*, *alias='default'*, *servo_status_listener=False*, *net_status_listener=False*)

Connect to target servo by CANOpen.

**Parameters:**
- **can_device** ( `CAN_DEVICE` ) – CANOpen device type.
- **dict_path** ( `str` ) – servo dictionary path.
- **node_id** ( `int` ) – node id. It's posible scan node ids with `scan_servos_canopen()` .
- **baudrate** ( `CAN_BAUDRATE` ) – communication baudrate. 1 Mbit/s by default.
- **channel** ( `int` ) – CANopen device channel. `0` by default.
- **alias** ( `str` ) – servo alias to reference it. `default` by default.
- **servo_status_listener** ( `bool` ) – Toggle the listener of the servo for its status, errors, faults, etc.
- **net_status_listener** ( `bool` ) – Toggle the listener of the network status, connection and disconnection.

**Raises:**
- FileNotFoundError – If either of the dict files doesn't exist.
- ingenialink.exceptions.ILError – If CANOpen device type, node id or channel is incorrect.

**Return type:** `None`

**`connect_servo_ethercat`**(*interface_name*, *slave_id*, *dict_path*, *alias='default'*, *servo_status_listener=False*, *net_status_listener=False*)

Connect to an EtherCAT slave.

**Parameters:**

- **interface_name** ( `str` ) – interface name. It should have format `\Device\NPF_[...]` .
- **slave_id** ( `int` ) – EtherCAT slave ID.
- **dict_path** ( `str` ) – servo dictionary path.
- **alias** ( `str` ) – servo alias to reference it. `default` by default.
- **servo_status_listener** ( `bool` ) – Toggle the listener of the servo for its status, errors, faults, etc.
- **net_status_listener** ( `bool` ) – Toggle the listener of the network status, connection and disconnection.

**Raises:** FileNotFoundError – If the dict file doesn't exist.

**Return type:** `None`

**connect_servo_ethercat_interface_index**(*if_index*, *slave_id*, *dict_path*, *alias='default'*, *servo_status_listener=False*, *net_status_listener=False*)

Connect to an EtherCAT slave.

**Parameters:**

- **if_index** ( `int` ) – interface index in list given by function `get_interface_name_list()` .
- **slave_id** ( `int` ) – EtherCAT slave ID.
- **dict_path** ( `str` ) – servo dictionary path.
- **alias** ( `str` ) – servo alias to reference it. `default` by default.
- **servo_status_listener** ( `bool` ) – Toggle the listener of the servo for its status, errors, faults, etc.
- **net_status_listener** ( `bool` ) – Toggle the listener of the network status, connection and disconnection.

**Raises:** IndexError – If interface index is out of range.

**Return type:** `None`

**connect_servo_ethercat_interface_ip**(*interface_ip*, *slave_id*, *dict_path*, *alias='default'*, *servo_status_listener=False*, *net_status_listener=False*)

Connect to an EtherCAT slave.

**Parameters:**

- **interface_ip** ( `str` ) – IP of the interface to be connected to.
- **slave_id** ( `int` ) – EtherCAT slave ID.
- **dict_path** ( `str` ) – servo dictionary path.
- **alias** ( `str` ) – servo alias to reference it. `default` by default.
- **servo_status_listener** ( `bool` ) – Toggle the listener of the servo for its status, errors, faults, etc.
- **net_status_listener** ( `bool` ) – Toggle the listener of the network status, connection and disconnection.

**Return type:** `None`

*static* `scan_servos_ethercat_with_info`(*interface_name*)

Scan a network adapter to get all connected EtherCAT slaves including slave information.

**Parameters:** **interface_name** ( `str` ) – interface name. It should have format `\Device\NPF_[...]`.

**Return type:** `OrderedDict` [ `int` , `SlaveInfo` ]

**Returns:** Dictionary of nodes available in the network and slave information.

**Raises:** TypeError – If some parameter has a wrong type.

`scan_servos_ethercat`(*interface_name*)

Scan a network adapter to get all connected EtherCAT slaves.

**Parameters:** **interface_name** ( `str` ) – interface name. It should have format `\Device\NPF_[...]`.

**Return type:** `List` [ `int` ]

**Returns:** List of EtherCAT slaves available in the network.

**Raises:** TypeError – If some parameter has a wrong type.

`scan_servos_ethercat_interface_ip`(*interface_ip*)

Scan a network adapter to get all connected EtherCAT slaves.

**Parameters:** **interface_ip** ( `str` ) – IP of the interface to be connected to.

**Return type:** `List` [ `int` ]

**Returns:** List of EtherCAT slaves available in the network.

**`scan_servos_ethercat_interface_index`**(*if_index*)

Scan a network adapter to get all connected EtherCAT slaves.

**Parameters:** **if_index** (`int`) – interface index in list given by function `get_interface_name_list()`.

**Return type:** `List`[`int`]

**Returns:** List of EtherCAT slaves available in the network.

**Raises:** IndexError – If interface index is out of range.

**`scan_servos_canopen_with_info`**(*can_device*, *baudrate=<CAN_BAUDRATE.Baudrate_1M: 1000000>*, *channel=0*)

Scan CANOpen device network to get all nodes including slave information.

**Parameters:**
- **can_device** (`CAN_DEVICE`) – CANOpen device type.
- **baudrate** (`CAN_BAUDRATE`) – communication baudrate. 1 Mbit/s by default.
- **channel** (`int`) – CANOpen device channel. `0` by default.

**Return type:** `OrderedDict`[`int`, `SlaveInfo`]

**Returns:** Dictionary of nodes available in the network and slave information.

**Raises:** TypeError – If some parameter has a wrong type.

**`scan_servos_canopen`**(*can_device*, *baudrate=<CAN_BAUDRATE.Baudrate_1M: 1000000>*, *channel=0*)

Scan CANOpen device network to get all nodes.

**Parameters:**
- **can_device** (`CAN_DEVICE`) – CANOpen device type.
- **baudrate** (`CAN_BAUDRATE`) – communication baudrate. 1 Mbit/s by default.
- **channel** (`int`) – CANOpen device channel. `0` by default.

**Return type:** `List`[`int`]

**Returns:** List of node ids available in the network.

**Raises:** TypeError – If some parameter has a wrong type.

**`disconnect`**(*servo='default'*)

Disconnect servo.

**Parameters:** **servo** ( `str` ) – servo alias to reference it. `default` by default.

**Return type:** `None`

`get_register`(*register*, *servo='default'*, *axis=1*)

Return the value of a target register.

**Parameters:**
- **register** ( `str` ) – register UID.
- **servo** ( `str` ) – servo alias to reference it. `default` by default.
- **axis** ( `int` ) – servo axis. `1` by default.

**Return type:** `Union` [ `int` , `float` , `str` ]

**Returns:** Current register value.

**Raises:**
- ingenialink.exceptions.ILAccessError – If the register access is write-only.
- IMRegisterNotExist – If the register doesn't exist.
- TypeError – If some parameter has a wrong type.

`set_register`(*register*, *value*, *servo='default'*, *axis=1*)

Set a value of a target register.

**Parameters:**
- **register** ( `str` ) – register UID.
- **value** ( `Union` [ `int` , `float` , `str` ]) – new value for the register.
- **servo** ( `str` ) – servo alias to reference it. `default` by default.
- **axis** ( `int` ) – servo axis. `1` by default.

**Raises:**
- TypeError – If the value is of the wrong type.
- IMRegisterNotExist – If the register doesn't exist.
- IMRegisterWrongAccess – If the register access is read-only.

**Return type:** `None`

`subscribe_net_status`(*callback*, *servo='default'*)

Add a callback to net status change event.

**Parameters:**
- **callback** ( `Callable` [[ `NET_DEV_EVT` ], `None` ]) – when net status changes callback is called.
- **servo** ( `str` ) – servo alias to reference it. `default` by default.

**Return type:** `None`

`unsubscribe_net_status`(*callback*, *servo='default'*)

Remove net status change event callback.

**Parameters:**
- **callback** (`Callable`[[`NET_DEV_EVT`], `None`]) – callback to remove.
- **servo** (`str`) – servo alias to reference it. `default` by default.

**Return type:** `None`

`subscribe_servo_status`(*callback*, *servo='default'*)

Add a callback to servo status change event.

**Parameters:**
- **callback** (`Callable`[[`SERVO_STATE`, `None`, `int`], `Any`]) – when servo status changes callback is called.
- **servo** (`str`) – servo alias to reference it. `default` by default.

**Return type:** `None`

`unsubscribe_servo_status`(*callback*, *servo='default'*)

Remove servo status change event callback.

**Parameters:**
- **callback** (`Callable`[[`SERVO_STATE`, `None`, `int`], `Any`]) – callback to remove.
- **servo** (`str`) – servo alias to reference it. `default` by default.

**Return type:** `None`

`load_firmware_canopen`(*fw_file*, *servo='default'*, *status_callback=None*, *progress_callback=None*, *error_enabled_callback=None*)

Load firmware via CANopen.

**Parameters:**

- **fw_file** ( `str` ) – Firmware file path.
- **servo** ( `str` ) – servo alias to reference it. `default` by default.
- **status_callback** ( `Optional` [ `Callable` [[ `str` ], `None` ]]) – callback with status.
- **progress_callback** ( `Optional` [ `Callable` [[ `int` ], `None` ]]) – callback with progress.
- **error_enabled_callback** ( `Optional` [ `Callable` [[ `bool` ], `None` ]]) – callback with errors enabled.

**Raises:** ValueError – If servo is not connected via CANopen.

**Return type:** `None`

`load_firmware_ecat`(*ifname*, *fw_file*, *slave=1*, *boot_in_app=None*, *password=None*)

Load firmware via ECAT.

**Parameters:**

- **ifname** ( `str` ) – interface name. It should have format `\Device\NPF_[...]`.
- **fw_file** ( `str` ) – Firmware file path.
- **slave** ( `int` ) – slave index. `1` by default.
- **boot_in_app** ( `Optional` [ `bool` ]) – true if the bootloader is included in the application, false otherwise. If None, the file extension is used to define it.
- **password** ( `Optional` [ `int` ]) – Password to load the firmware file. If `None` the default password will be used.

**Raises:**

- FileNotFoundError – If the firmware file cannot be found.
- ValueError – If the firmware file has the wrong extension.
- ingenialink.exceptions.ILFirmwareLoadError – If no slave is detected.
- ingenialink.exceptions.ILFirmwareLoadError – If the FoE write operation is not successful
- NotImplementedError – If FoE is not implemented for the current OS and architecture

**Return type:** `None`

`load_firmware_ecat_interface_index`(*if_index*, *fw_file*, *slave=1*, *boot_in_app=None*, *password=None*)

Load firmware via ECAT.

**Parameters:**

- **if_index** ( `int` ) – interface index in list given by function `get_interface_name_list()`.
- **fw_file** ( `str` ) – Firmware file path.
- **slave** ( `int` ) – slave index. `1` by default.
- **boot_in_app** ( `Optional` [ `bool` ]) – true if the bootloader is included in the application, false otherwise. If `None`, the file extension is used to define it.
- **password** ( `Optional` [ `int` ]) – Password to load the firmware file. If `None` the default password will be used.

**Raises:**

- IndexError – If interface index is out of range.
- FileNotFoundError – If the firmware file cannot be found.
- ingenialink.exceptions.ILFirmwareLoadError – If no slave is detected.
- ingenialink.exceptions.ILFirmwareLoadError – If the FoE write operation is not successful
- NotImplementedError – If FoE is not implemented for the current OS and architecture

**Return type:** `None`

**`load_firmware_ethernet`**(*ip*, *fw_file*, *ftp_user=None*, *ftp_pwd=None*)

Load firmware via Ethernet. Boot mode is needed to load firmware.

**Warning**

After functions ends, the servo will take a moment to load firmware. During the process, the servo will be not operative.

**Parameters:**

- **ip** ( `str` ) – servo IP.
- **fw_file** ( `str` ) – Firmware file path.
- **ftp_user** ( `Optional` [ `str` ]) – FTP user to connect with.
- **ftp_pwd** ( `Optional` [ `str` ]) – FTP password for the given user.

**Return type:** `None`

**`boot_mode_and_load_firmware_ethernet`**(*fw_file*, *servo='default'*, *ftp_user=None*, *ftp_pwd=None*)

Set servo to boot mode and load firmware. Servo is disconnected.

**Warning**

After functions ends, the servo will take a moment to load firmware. During the process, the servo will be not operative.

**Parameters:**

- **fw_file** ( `str` ) – Firmware file path.
- **servo** ( `str` ) – servo alias to reference it. `default` by default.
- **ftp_user** ( `Optional` [ `str` ]) – FTP user to connect with.
- **ftp_pwd** ( `Optional` [ `str` ]) – FTP password for the given user.

**Raises:** ValueError – If servo is not connected via Ethernet.

**Return type:** `None`

**`boot_mode`**(*servo='default'*)

Set servo to boot mode. Servo is disconnected.

**Parameters:** **servo** ( `str` ) – servo alias to reference it. `default` by default.

**Return type:** `None`

**`load_firmware_moco`**(*fw_file*, *servo='default'*)

Load firmware to the Motion Core.

**Warning**

After functions ends, the servo will take a moment to load firmware. During the process, the servo will be not operative.

**Parameters:**

- **fw_file** ( `str` ) – Firmware file path.
- **servo** ( `str` ) – servo alias to reference it. `default` by default.

**Raises:** ValueError – If servo is not connected via Ethernet.

**Return type:** `None`

**`boot_mode_moco`**(*servo='default'*)

Set the Motion Core to boot mode.

**Parameters:** **servo** ( `str` ) – servo alias to reference it. `default` by default.

**Return type:** `None`