Emerson Process FBx000 Series Communications Driver

Information Sheet for Crimson v3.1+

Compatible Devices

Emerson FB1000/2000/3000 Series Flow Computers and RTUs with Ethernet.

Verified Devices

FB2200
FB3000

Overview

The Emerson Process FBx000 Series communications driver provides access to data tags in Emerson Process FB Series devices. The driver uses Field Tools configuration files to ensure Crimson data tags match those in the FB Series device.

Operation

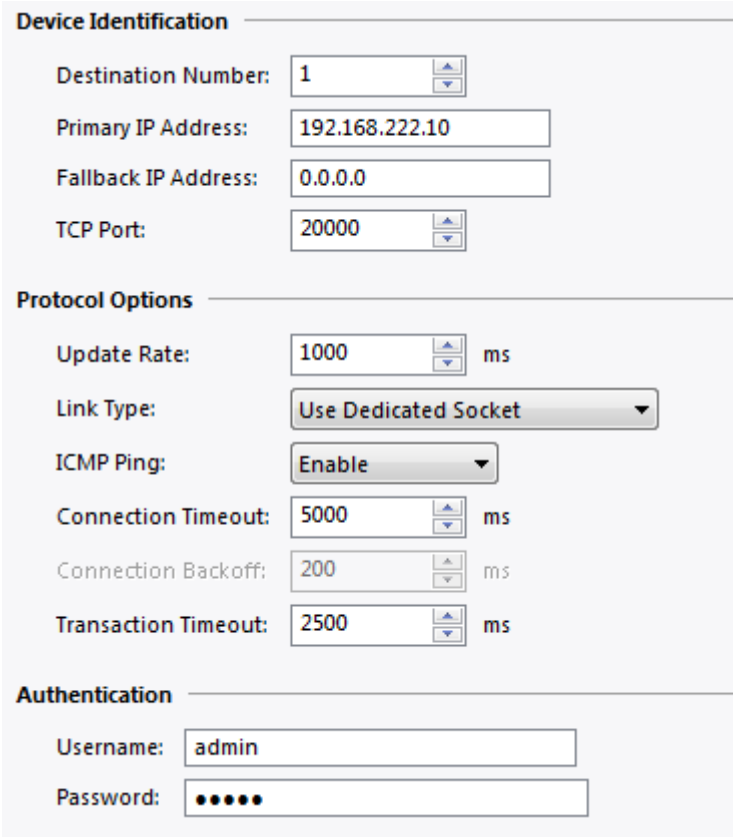
The protocol used by the Emerson FBx000 Series device supports read access to data items via single or batched transactions, the latter being far more efficient as multiple data items can be combined in a batched transaction. Batched transactions are further optimized by using an Object, Parameter, Instance (OPI) reference, rather than the data tag textual name, in the request. As OPIs are only available at runtime, these are collected from the Emerson device once communication with the Crimson device has been established. This can take a number of seconds depending on the complexity of the configuration file, during which time data is not retrieved. However, this process occurs only once on system start up.

This driver has been implemented as a caching driver meaning that mapped Crimson data objects are serviced once per communications scan. The data cache is optimized for context such that only data items that are required to be accessed are actually read from the FB device during the scan. You might experience a few seconds data update delay on a context change such as a Display Page transition. Unused data objects remain on the cache for 30 update periods when they will expire and are purged from the data cache.

Driver Configuration

- *Source Number* - The DNP3 address that the driver should use for all communications.

Device Configuration



Device Identification

- *Destination Number* – The Flow Computer Address as configured under the DNP3 tab in the Communications configuration in Emerson Field Tools for the FB device. This number can be changed at runtime using the DevCtrl(..) user function, details of which can be found later in this document.

- *Primary IP Address* – The primary IP address to use to establish Ethernet communications.

- *Fallback IP Address* – An optional fallback IP address. This is used in conjunction with the DevCtrl(..) function to switch between the primary

and fallback IP address, details of which can be found later in this document.

- *TCP Port* – The TCP/IP used for DNP3 communications by the FB device.

Protocol Options

- *Update Rate* – The driver will periodically update the values for tags when they are referenced. This period is configurable, lower values will give fresher tag values, but may increase the CPU load on the FB device.

Authentication

- *Username* – A username for a configured user in the FB device.

- *Password* – The corresponding password for the user.

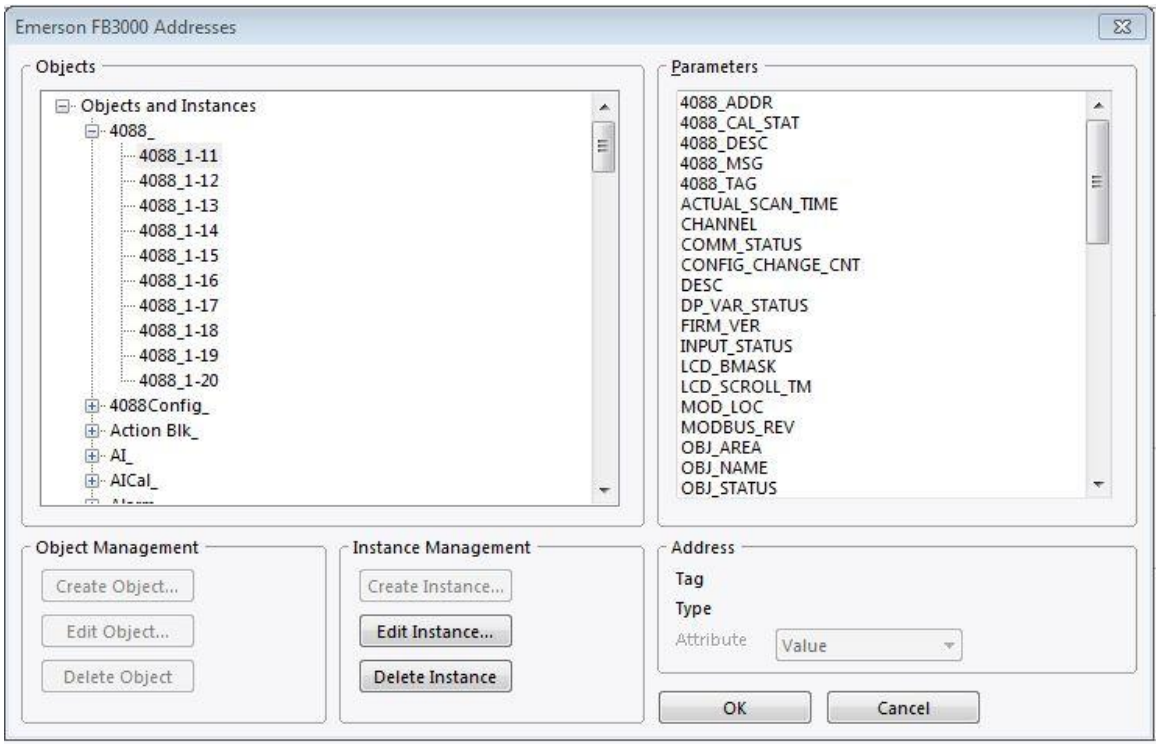
Note that if incorrect credentials are supplied, the user may be locked out for a period of time, during which communications cannot be established.

Tag Management

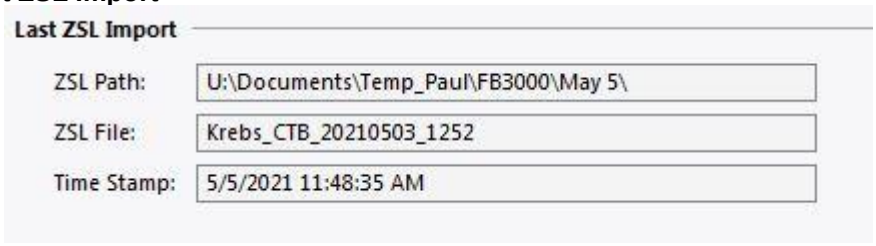
- Clicking *Import Parameters* will allow you to select and import the Field Tools configuration file for the target FB device.

- Clicking *Manage Parameters* will invoke the following dialog that allows you to view the Objects, Parameters and Instances available in the target FB device.

The Emerson FB3000 Addresses dialog allows you to edit data tags but this approach is not advisable as the configurations in the FB device and the Crimson device must match for effective error-free communication. Manual tag creation is described in a later section for information only.



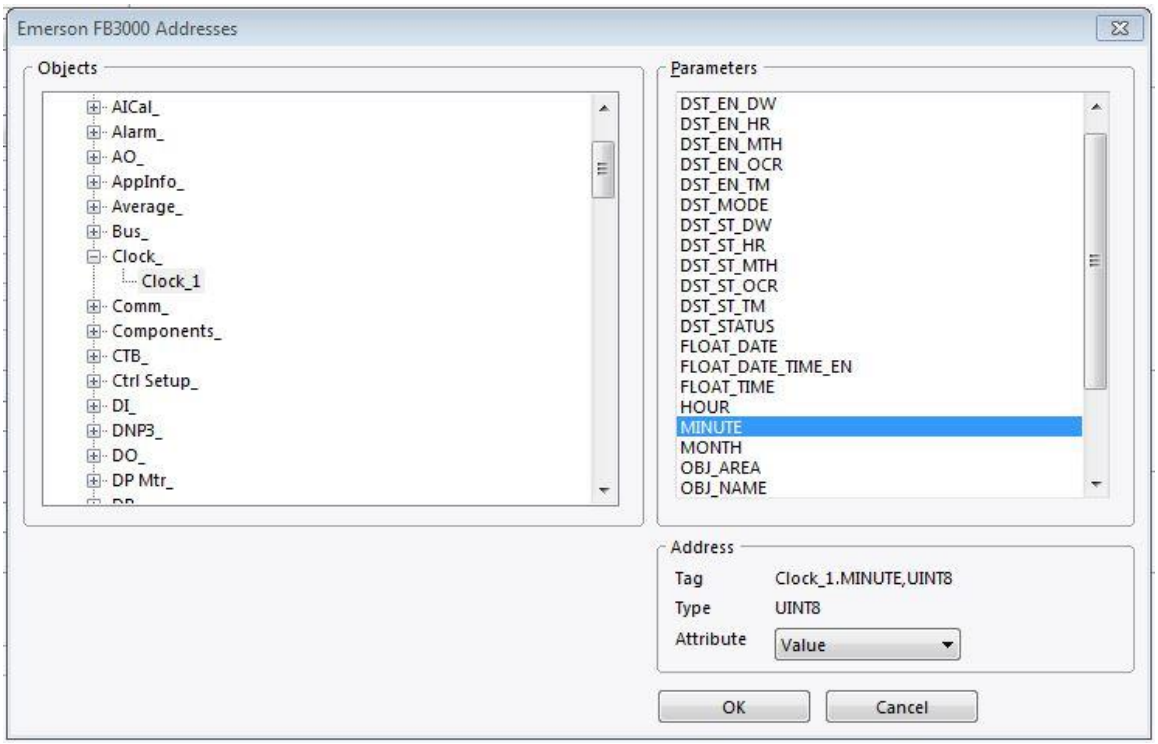
Last ZSL Import



- *ZSL Path* – The Path of the currently imported Field Tools configuration file.
- *ZSL File* – The File name of the currently imported Field Tools configuration file.
- *Time Stamp* – The time stamp of the currently imported Field Tools configuration file.

Address Selection

When mapping Crimson data tags to tags in the Emerson device, the Address Selection dialog will appear:



Expand the desired object and highlight an instance. The available members for that instance will be loaded into the list box on the right. Select a member, and the address information will be populated. Select the desired attribute for the tag, and click OK to map the tag.

Note, the following limitations.

1. This driver supports up to 240 arrays type, including STRING, DOUBLE and INT64 data types.
2. Only the *Value* attribute is supported.

Accessing 64-bit Data

64-bit data, such as the DOUBLE and INT64 datatypes must use the Crimson 64-bit math functions to display or set properly. These functions are described in detail in the Crimson reference manual.

To read a single 64-bit tag, create a Crimson tag as an array with an extent of 2. Map the tag the desired FB3000 64-bit parameter. To display the value, use the first element of the tag in a display function such as AsTextR64. For example, the following code will decode a 64-bit DOUBLE parameter into a string that can be used to display the value:

```
cstring value = AsTextR64(myDouble[0]);
```

To write a single 64-bit tag, create a Crimson tag as an array with an extent and map it to the desired 64-bit DOUBLE parameter. Use an editable Crimson string tag to set the desired value, then use the AsTextR64 function as below:

```
cstring value = “1.234”;

TextToR64(value, myDouble[0]);
```

Changing the Target Device at Runtime with the DevCtrl(..) Function

DevCtrl(..) is a Crimson user function to set or change properties for a specific comms device at runtime. Using this function, the FBx000 Series driver supports changing the target’s primary or secondary IP address, or to change the target’s destination number.

Function Code	Function
1	Set primary IP address
2	Set destination number
5	Set secondary IP address

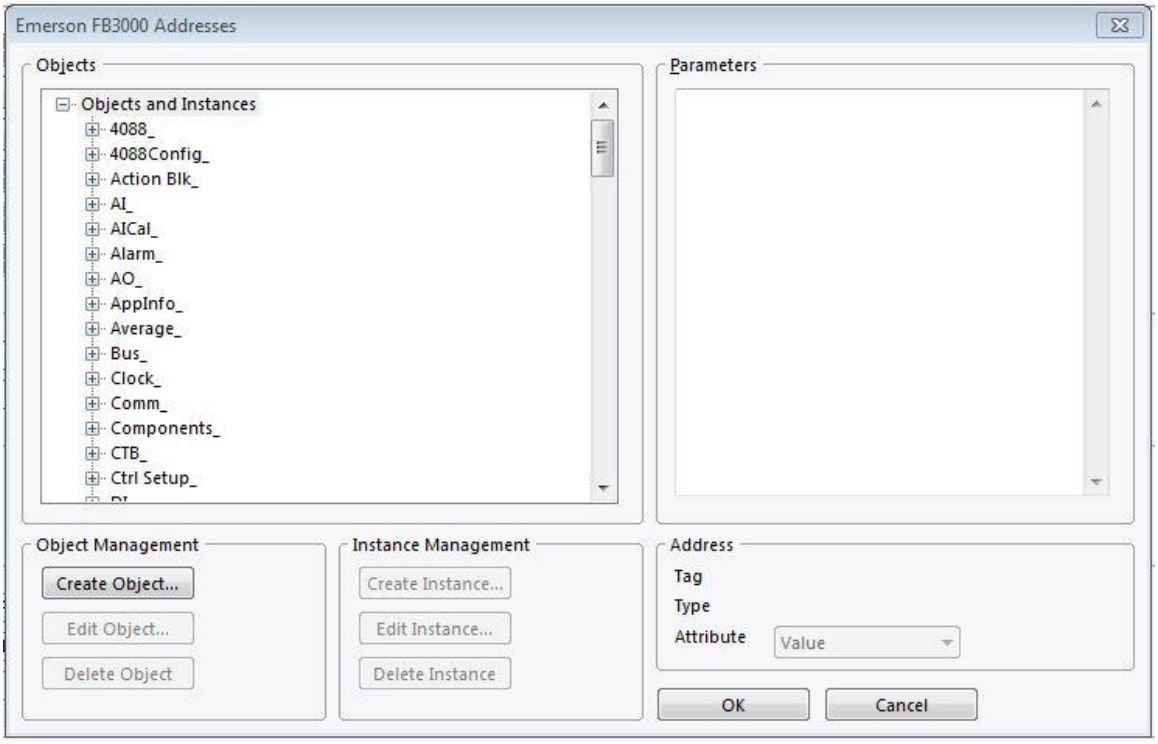
The DevCtrl function takes three arguments: the device number, function code, and a string argument for the function code. See the Crimson user manual for details on determining the device number and further information on the DevCtrl function. For example, the following code will change the secondary IP address for comms device 1:

```
DevCtrl(1, 5, “192.168.1.50”);
```

Manual Tag Creation

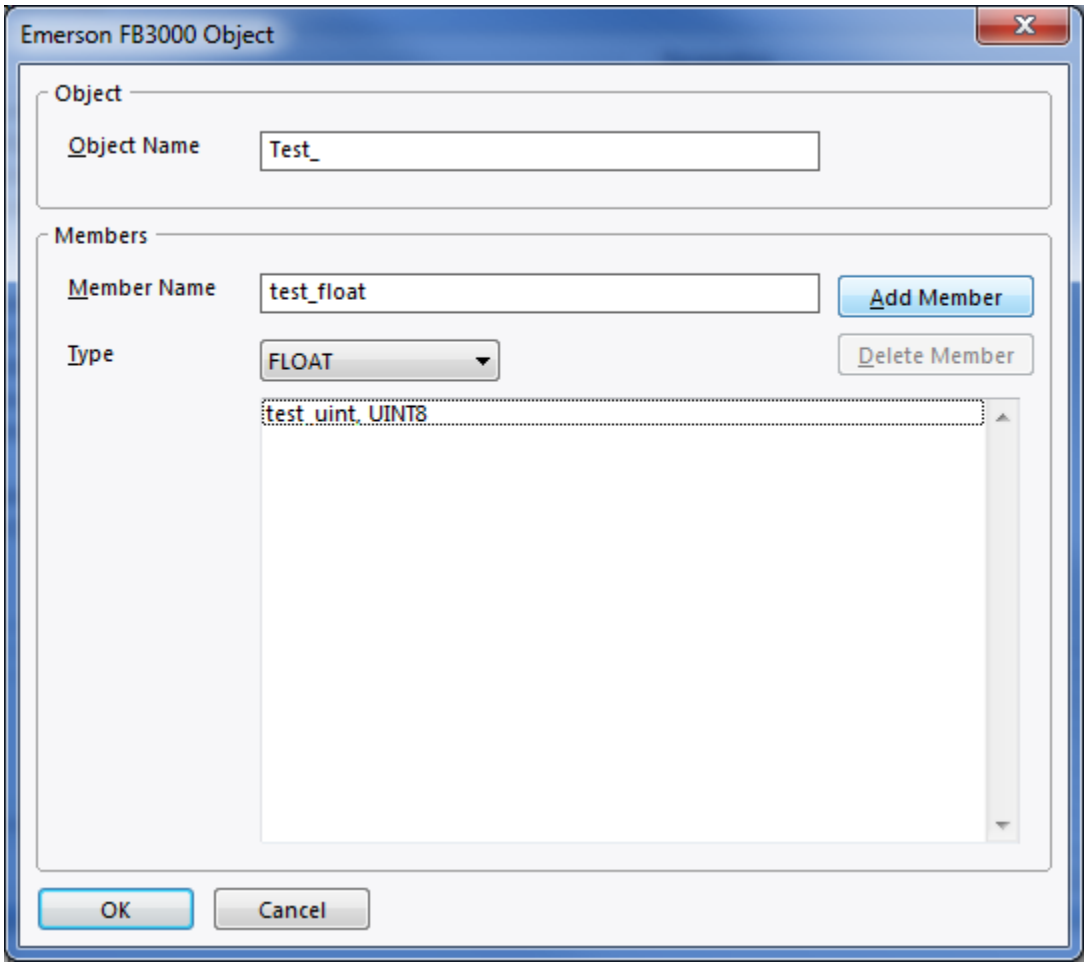
Tags can also be created manually without the need for a corresponding configuration file. This also allows access to parameters that aren’t found in the configuration file.

On clicking the *Manage parameters* link, a dialog will appear, showing all currently configured objects and object instances.



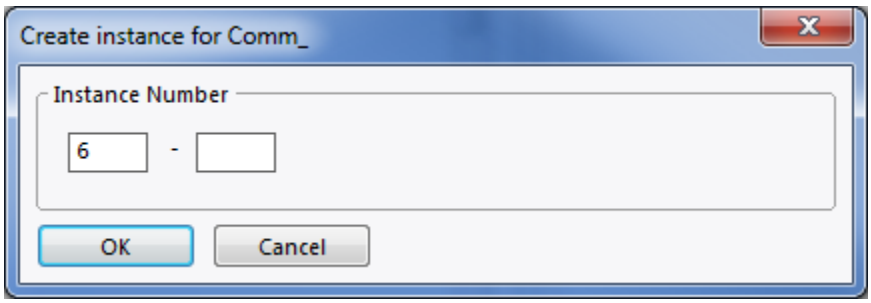
To create a new object definition, click the Create Object… button. This will launch a new dialog. In the Object Creation dialog, enter a name for the object. Typically, an object name will end with a trailing underscore. Then, click the Add Member button to add members to the object definition. Type in a name for the member, and select its type from the Type drop-down menu. Click OK to add the member to the object. When the object definition is complete, click OK to save the object.

Highlighting an object name and clicking Edit Object… will bring up the same dialog as above, but already filled with the details of the selected object. From here, the object can have new data members added or deleted in the same manner.



To create an instance of an object, highlight the object name in the tree, then click the Create Instance… button. Enter the number for the instance. If the instance has a hyphen in the name, enter the number that follows the hyphen in the box on the right. Click OK to finish creating the instance.

An instance can be edited by clicking the Edit Instance… button. This will bring up the same dialog used when creating an instance, but with the details of the instance displayed. They can be changed in the same manner as creating a new instance. Click the OK button to save changes or Cancel to discard them.



Objects and instances can be deleted with the Delete Object and Delete Instance buttons, respectively.

Ethernet Cable Information

Standard Ethernet Cable

Revision History

10/26/2020 – Created.
06/04/2021 – Finalized.