



### **CECC-LK as IO-Link Master connected to Balluff RFID Device and data carriers**

The application note contains a step by step explanation how to configure a Balluff RFID read/write system as IO-Link device connected to CECC-LK in Codesys V3

Title .....CECC-LK as IO-Link Master connected to Balluff RFID Device and data carriers  
Version ..... 1.10  
Document no. .... 100126  
Original .....en  
Author .....Festo  
  
Last saved ..... 01.12.2016

## Copyright Notice

This documentation is the intellectual property of Festo AG & Co. KG, which also has the exclusive copyright. Any modification of the content, duplication or reprinting of this documentation as well as distribution to third parties can only be made with the express consent of Festo AG & Co. KG.

Festo AG & Co KG reserves the right to make modifications to this document in whole or in part. All brand and product names are trademarks or registered trademarks of their respective owners.

## Legal Notice

Hardware, software, operating systems and drivers may only be used for the applications described and only in conjunction with components recommended by Festo AG & Co. KG.

Festo AG & Co. KG does not accept any liability for damages arising from the use of any incorrect or incomplete information contained in this documentation or any information missing therefrom.

Defects resulting from the improper handling of devices and modules are excluded from the warranty.

The data and information specified in this document should not be used for the implementation of safety functions relating to the protection of personnel and machinery.

No liability is accepted for claims for damages arising from a failure or functional defect. In other respects, the regulations with regard to liability from the terms and conditions of delivery, payment and use of software of Festo AG & Co. KG, which can be found at [www.festo.com](http://www.festo.com) and can be supplied on request, shall apply.

All data contained in this document do not represent guaranteed specifications, particularly with regard to functionality, condition or quality, in the legal sense.

The information in this document serves only as basic information for the implementation of a specific, hypothetical application and is in no way intended as a substitute for the operating instructions of the respective manufacturers and the design and testing of the respective application by the user.

The operating instructions for Festo products can be found at [www.festo.com](http://www.festo.com).

Users of this document (application note) must verify that all functions described here also work correctly in the application. By reading this document and adhering to the specifications contained therein, users are also solely responsible for their own application.

# Table of contents

|          |  |          |
|----------|--|----------|
| <b>1</b> | <b>Components/Software/ IP address used.....</b> | <b>4</b> |
| 1.1      | Recommended manuals / IODD .....                 | 4        |
| 1.2      | Topology .....                                   | 5        |
| 1.3      | Wiring .....                                     | 5        |
| <b>2</b> | <b>Configuring the IO-Link master .....</b>      | <b>7</b> |
| 2.1      | IO-Link configuration .....                      | 7        |
| 2.2      | Configuration of Parameters and Testing .....    | 9        |
| 2.3      | Programming in Codesys .....                     | 12       |
| 2.3.1    | Program to read parameter by coding.....         | 12       |
| 2.3.2    | Program to write parameter by coding.....        | 13       |
| 2.3.3    | Read stored data from data carrier .....         | 15       |

## 1 Components/Software/ IP address used

| Type/Name                          | Version Software/Firmware | IP address    | Subnet mask |
|------------------------------------|---------------------------|---------------|-------------|
| CECC-LK                            | V 1.4.0.1                 | 192.168.0.20  | 255.255.0.0 |
| Laptop                             | --                        | 192.168.0.100 | 255.255.0.0 |
| Codesys V3.5                       | SP7 Patch 4               | --            | --          |
| BIS0103<br>BIS M-451-072-001-07-S4 | V1.70                     | --            | --          |
| BIS0045<br>BIS M-111-02/L          | --                        | --            | --          |
| BIS0046<br>BIS M-112-02/L          | --                        | --            | --          |

Table 1.1: 1 Components/Software used

### 1.1 Recommended manuals / IODD

CECC manual:

[https://www.festo.com/net/SupportPortal/Files/407042/CECC\\_2014-03a\\_8036062g1.pdf](https://www.festo.com/net/SupportPortal/Files/407042/CECC_2014-03a_8036062g1.pdf)

Target Support:

[https://www.festo.com/net/en-gb\\_gb/SupportPortal/Downloads/415525/443515/CECC\\_3.5.7.159\(ad778b5e1029\).package](https://www.festo.com/net/en-gb_gb/SupportPortal/Downloads/415525/443515/CECC_3.5.7.159(ad778b5e1029).package)

BIS0103, BIS M-451-072-001-07-S4 manual:

[http://asset.balluff.com/std.lang.all/pdf/binary/870554\\_000\\_04\\_DOK.pdf](http://asset.balluff.com/std.lang.all/pdf/binary/870554_000_04_DOK.pdf)

Datasheet:

[http://asset.balluff.com/std.lang.all/pdf/datasheet/6\\_/gl/Datasheet\\_BIS0103\\_228506\\_GL.pdf](http://asset.balluff.com/std.lang.all/pdf/datasheet/6_/gl/Datasheet_BIS0103_228506_GL.pdf)

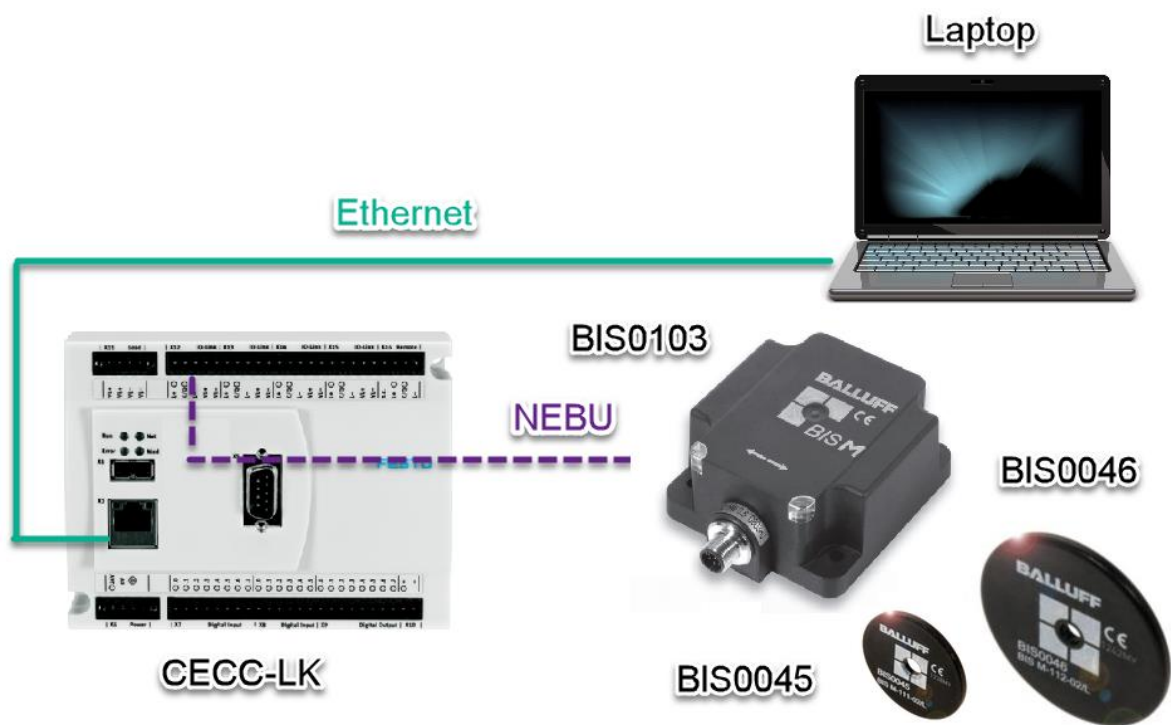
IODD file:

[http://asset.balluff.com/std.lang.all/zip/binary/918408\\_000\\_00\\_DRF.zip](http://asset.balluff.com/std.lang.all/zip/binary/918408_000_00_DRF.zip)

NEBU catalog:

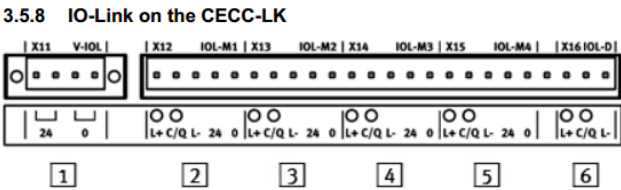
[https://www.festo.com/net/en-gb\\_gb/SupportPortal/Downloads/272418/208021/nebu\\_en.pdf](https://www.festo.com/net/en-gb_gb/SupportPortal/Downloads/272418/208021/nebu_en.pdf)

1.2 Topology



1.3 Wiring

IO-Link is a point to point communication (1 Master -> 1 Slave). In our example, we use port 1 of IO-Link master from CECC-LK. Festo has the NEBU cable series to connect to the IO-Link device and master. To get the correct wiring, please check the following pin assignments.  
CECC-LK:



- 1 IO-Link load voltage supply X11
- 2...5 IO-Link master ports for connecting IO-Link devices X12 ... X15
- 6 IO-Link device port for connecting the CECC-LK as an IO-Link device X16

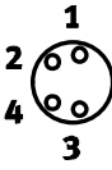
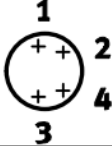
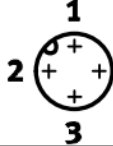
Figure: CECC-LK, top manifold rail with IO-Link interfaces

| IO-Link | Pin        | Signal | Comment   |
|---------|------------|--------|---|
| 1       | X11.1      | 24     | Connection for load voltage supply for IO-Link master ports <sup>1)</sup> : UA+       |
|         | X11.2      |        |   |
|         | X11.3      | 0      | Connection for load voltage supply for IO-Link master ports <sup>1)</sup> : UA– (GND) |
|         | X11.4      |        |   |
| 2... 5  | X12...15.1 | L+     | 24 V  |
|         | X12...15.2 | C/Q    | IO-Link communication signal  |
|         | X12...15.3 | L–     | 0 V   |
|         | X12...15.4 | 24     | UA+   |
|         | X12...15.5 | 0      | UA–   |
| 6       | X16.1      | L+     | 24 V  |
|         | X16.2      | C/Q    | IO-Link communication signal  |
|         | X16.3      | L–     | 0 V   |

1) Port class B

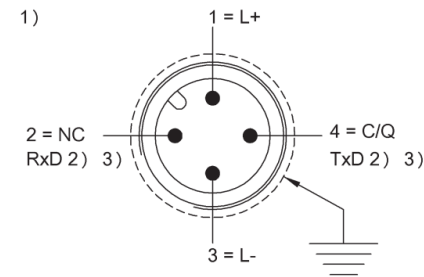
Table: Pin allocation of the IO-Link interfaces on the CECC-LK

NEBU:

| Electrical connection: socket, 4-pin, M8 – plug, 4-pin                            |   |    |   |   |   |
|---|---|----|---|---|---|
|  | 1 | BN | 1 |  |  |
|   | 2 | WH | 2 |   |   |
|   | 3 | BU | 3 |   |   |
|   | 4 | BK | 4 |   |   |

BIS0103:

BIS M-451-072-001-07-S4  
BIS0103

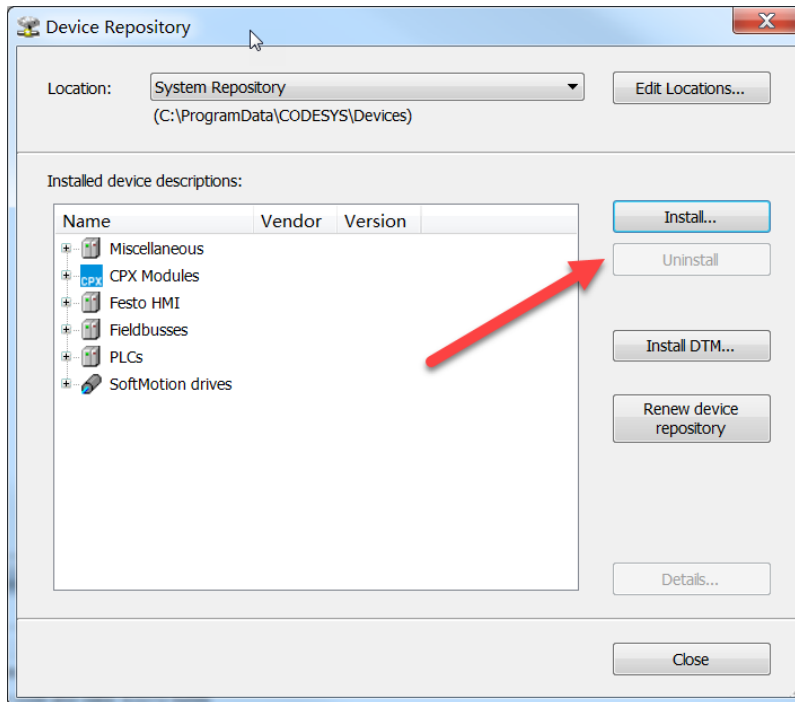


- 1) View towards connector
- 2) Service
- 3) (Only for Balluff Service)

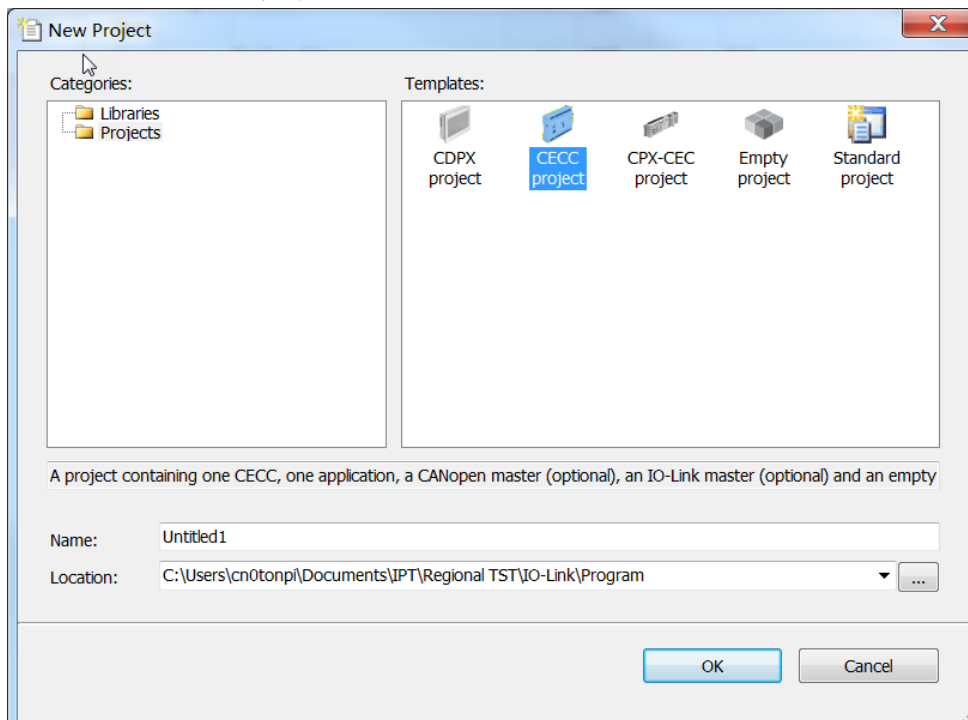
## 2 Configuring the IO-Link master

### 2.1 IO-Link configuration

Download the IODD file of Balluff RFID read/write device and install it in the Codesys

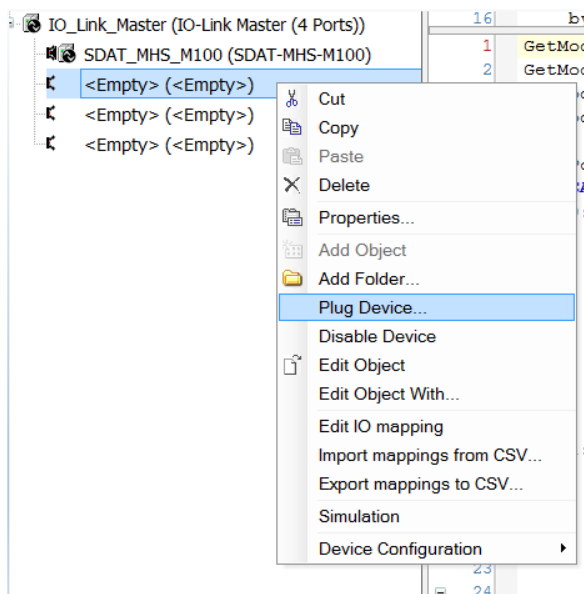


New an IO-Link master project, select CECC-LK and activate IO-Link Master

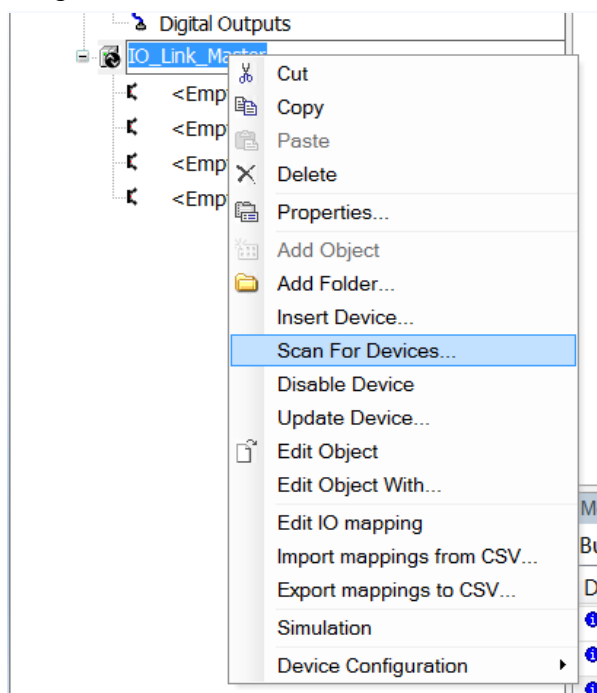


You can either right click the port -> Plug Device to add sensor manually, I use port 2 here.

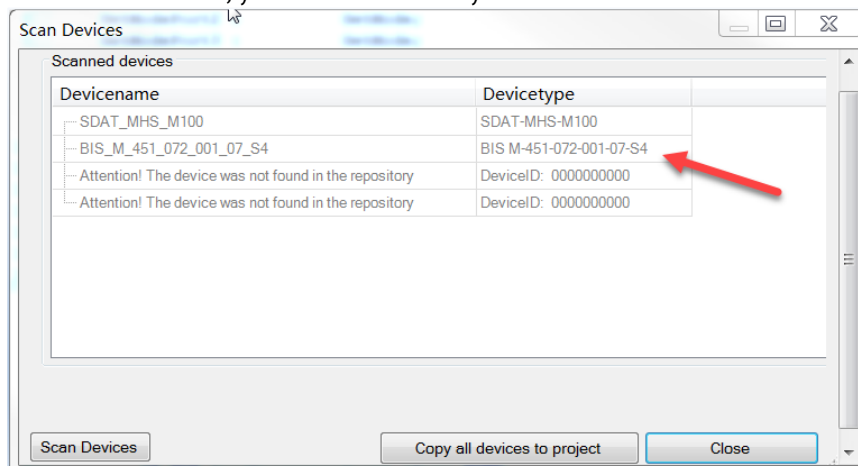
## Configuring the IO-Link master



Or right click the IO\_Link\_Master -> Scan For Devices



And after a moment, you will automatically find the device



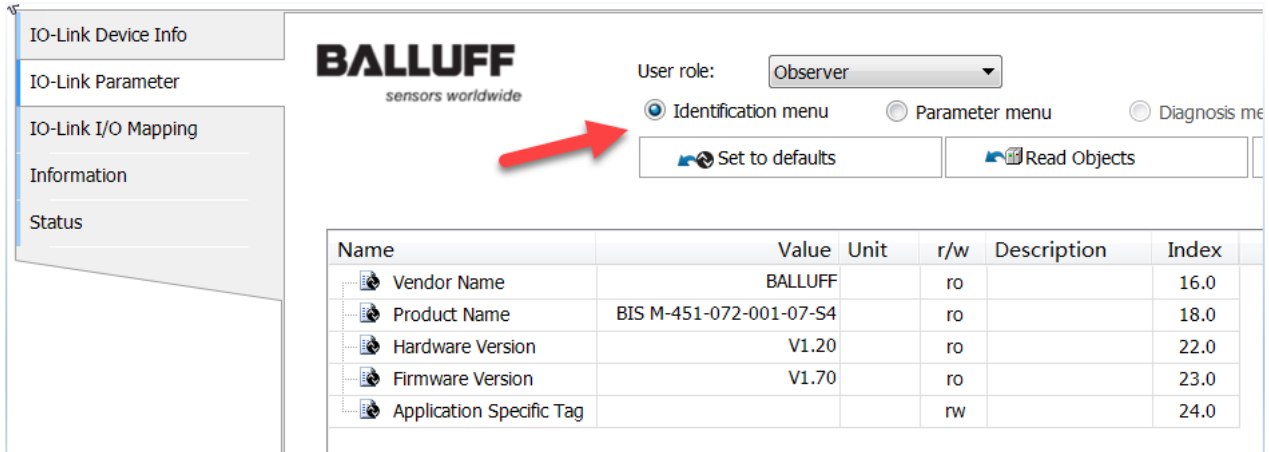


## 2.2 Configuration of Parameters and Testing

If BIS0103 is working properly, you will find the C/Q of CECC-LK is illuminated, and this read/write device's green LED flashes.

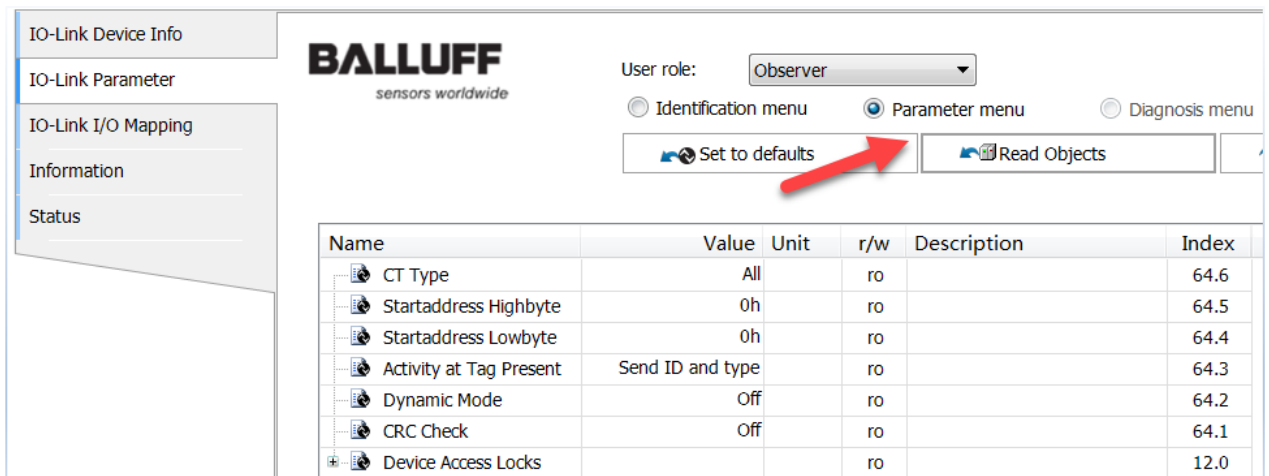
| LED   | Status                                  | Function                  |
|-------|---|---------------------------|
| LED 1 | Green                                   | Power                     |
| LED 2 | Yellow                                  | Data carrier detected     |
| LED 1 | Green flashing<br>(1 s on / 100 ms off) | IO-Link connection active |

Go to the BIS0103 IO-Link configurator, and activate the index column, you will see the parameters saved in Identification Menu and Parameter Menu.



The screenshot shows the Balluff IO-Link configurator interface. On the left is a sidebar with menu items: IO-Link Device Info, IO-Link Parameter, IO-Link I/O Mapping, Information, and Status. The main area features the Balluff logo and 'sensors worldwide' text. Below this, the 'User role' is set to 'Observer'. Three radio buttons are present: 'Identification menu' (selected), 'Parameter menu', and 'Diagnosis menu'. Below the radio buttons are two buttons: 'Set to defaults' and 'Read Objects'. A red arrow points to the 'Identification menu' radio button. Below this is a table with the following data:

| Name                     | Value                   | Unit | r/w | Description | Index |
|--------------------------|-------------------------|------|-----|-------------|-------|
| Vendor Name              | BALLUFF                 |      | ro  |             | 16.0  |
| Product Name             | BIS M-451-072-001-07-S4 |      | ro  |             | 18.0  |
| Hardware Version         | V1.20                   |      | ro  |             | 22.0  |
| Firmware Version         | V1.70                   |      | ro  |             | 23.0  |
| Application Specific Tag |                         |      | rw  |             | 24.0  |



The screenshot shows the Balluff IO-Link configurator interface with the 'Parameter menu' selected. The 'User role' is still 'Observer'. The 'Parameter menu' radio button is now selected, and a red arrow points to it. The 'Set to defaults' and 'Read Objects' buttons are still present. Below this is a table with the following data:

| Name                    | Value            | Unit | r/w | Description | Index |
|-------------------------|------------------|------|-----|-------------|-------|
| CT Type                 | All              |      | ro  |             | 64.6  |
| Startaddress Highbyte   | 0h               |      | ro  |             | 64.5  |
| Startaddress Lowbyte    | 0h               |      | ro  |             | 64.4  |
| Activity at Tag Present | Send ID and type |      | ro  |             | 64.3  |
| Dynamic Mode            | Off              |      | ro  |             | 64.2  |
| CRC Check               | Off              |      | ro  |             | 64.1  |
| Device Access Locks     |                  |      | ro  |             | 12.0  |

There is no description in the Balluff's IODD file, therefore, we get the description from manual. It is called SPDU in Balluff's manual.

|                     | SPDU              |                | Object name       | Length   | Information                               |
|---------------------|-------------------|----------------|-------------------|----------|---|
|                     | Index             | Subindex       |                   |          |   |
| Identification data | 0 <sub>hex</sub>  | 8<br>9         | Vendor ID         | 2 bytes  | Balluff Vendor ID = 0378 <sub>hex</sub>   |
|                     |                   | 10<br>11<br>12 | Device ID         | 3 bytes  | Balluff Device ID = 0602xx <sub>hex</sub> |
|                     | 10 <sub>hex</sub> | 0              | Vendor name       | 7 bytes  | Balluff                                   |
|                     | 11 <sub>hex</sub> | 0              | Vendor text       | 15 bytes | www.balluff.com                           |
|                     | 12 <sub>hex</sub> | 0              | Product name      | 23 bytes | Device designation                        |
|                     | 13 <sub>hex</sub> | 0              | Product ID        | 7 bytes  | Ordering code                             |
|                     | 14 <sub>hex</sub> | 0              | Product text      | 27 bytes | IO-Link RFID read-write head              |
|                     | 16 <sub>hex</sub> | 0              | Hardware revision | 5 bytes  | Hardware version                          |
|                     | 17 <sub>hex</sub> | 0              | Firmware revision | 5 bytes  | Firmware version                          |

|                | Access            |                  | Description                             | Data width | Value range   | Factory setting |
|----------------|-------------------|------------------|---|------------|---|-----------------|
|                | Index             | Subindex         |   |            |   |                 |
| Parameter data | 40 <sub>hex</sub> | 1 <sub>hex</sub> | CRC yes/no                              | 1 byte     | 0 = without CRC<br>1 = with CRC   | 0               |
|                | 40 <sub>hex</sub> | 2 <sub>hex</sub> | Dynamic mode - yes/no                   | 1 byte     | 0 = no<br>1 = yes   | 0               |
|                | 40 <sub>hex</sub> | 3 <sub>hex</sub> | Action if tag present                   | 1 byte     | 0 = no action<br>1 = serial number and tag type<br>7 = automatically read 8 bytes of data beginning at a set start address after subindex 4 and 5 | 1               |
|                | 40 <sub>hex</sub> | 4 <sub>hex</sub> | Low byte of start address for autoread  | 2 bytes    | Observe data-carrier specifications.  | 0               |
|                | 40 <sub>hex</sub> | 5 <sub>hex</sub> | High byte of start address for autoread |            |   |                 |
|                | 40 <sub>hex</sub> | 6 <sub>hex</sub> | Used data-carrier type                  | 1 byte     | 00 <sub>hex</sub> =ALL<br>FE <sub>hex</sub> =BIS M1_ _-01<br>FF <sub>hex</sub> =BIS M1_ _-02  | 0               |

The BIS0103 has 32 bytes output & input, which is very huge, it reaches maximum process data size of the IO-Link defined. You can read & write process data directly in Codesys.

| IO-Link Device Info   |  | Channels            |        |             |        |             |         |               |                 |      |         |
|-----------------------|--|---------------------|--------|-------------|--------|-------------|---------|---------------|-----------------|------|---------|
| IO-Link Parameter     |  | Variable            | Map... | Channel     | Add... | Type        | Defa... | Current Value | Prepared Val... | Unit | Desc... |
| IO-Link I/O Mapping   |  | Process Data Inputs |        |             |        |             |         |               |                 |      |         |
| Information<br>Status |  |                     |        | Bitheader 2 |        | %IB5 USINT  |         | 128           |                 |      | 0..255  |
|                       |  |                     |        | Byte 30     |        | %IB6 USINT  |         | 0             |                 |      | 0..255  |
|                       |  |                     |        | Byte 29     |        | %IB7 USINT  |         | 0             |                 |      | 0..255  |
|                       |  |                     |        | Byte 28     |        | %IB8 USINT  |         | 0             |                 |      | 0..255  |
|                       |  |                     |        | Byte 27     |        | %IB9 USINT  |         | 0             |                 |      | 0..255  |
|                       |  |                     |        | Byte 26     |        | %I... USINT |         | 0             |                 |      | 0..255  |
|                       |  |                     |        | Byte 25     |        | %I... USINT |         | 0             |                 |      | 0..255  |
|                       |  |                     |        | Byte 24     |        | %I... USINT |         | 0             |                 |      | 0..255  |
|                       |  |                     |        | Byte 23     |        | %I... USINT |         | 0             |                 |      | 0..255  |
|                       |  |                     |        | Byte 22     |        | %I... USINT |         | 0             |                 |      | 0..255  |
|                       |  |                     |        | Byte 21     |        | %I... USINT |         | 0             |                 |      | 0..255  |
|                       |  |                     |        | Byte 20     |        | %I... USINT |         | 0             |                 |      | 0..255  |

Description is again only available in the manual. To understand, for example, let us put a data carrier BIS0045 close to the range of it, and check the 1<sup>st</sup> bit string.

**Input buffer:**

| Bit No.                            | 7                                       | 6  | 5  | 4 | 3  | 2  | 1  | 0  |
|------------------------------------|---|----|----|---|----|----|----|----|
| Subaddress                         |   |    |    |   |    |    |    |    |
| 00 <sub>hex</sub> - 1st bit string | BB                                      | HF | TO |   | AF | AE | AA | CP |
| 01 <sub>hex</sub>                  | Error code or data or high-byte version |    |    |   |    |    |    |    |
| 02 <sub>hex</sub>                  | Data or low-byte version                |    |    |   |    |    |    |    |
| 03 <sub>hex</sub>                  | Data                                    |    |    |   |    |    |    |    |
| 04 <sub>hex</sub>                  | Data                                    |    |    |   |    |    |    |    |
| 05 <sub>hex</sub>                  | Data                                    |    |    |   |    |    |    |    |
| 06 <sub>hex</sub>                  | Data                                    |    |    |   |    |    |    |    |
| 07 <sub>hex</sub>                  | Data                                    |    |    |   |    |    |    |    |
| 08 <sub>hex</sub>                  | Data                                    |    |    |   |    |    |    |    |
| Last byte - 2nd bit string         | BB                                      | HF | TO |   | AF | AE | AA | CP |

**Explanations on the input buffer using 10 bytes as an example:**

| Subaddress        | Bit name       | Meaning         | Function description   |
|-------------------|----------------|-----------------|--|
| 00 <sub>hex</sub> | 1st bit string |                 |  |
|                   | BB             | Power           | 1 = Device is ready<br>0 = Device is in ground state   |
|                   | HF             | Head Failure    | 1 = Head is turned off<br>0 = Head is turned on  |
|                   | TO             | Toggle bit      | A state change during a job indicates that the read/write device is ready to transfer other data |
|                   | AF             | Job error       | 1 = Job incorrectly processed<br>0 = Job processed without errors                                |
|                   | AE             | Job end         | 1 = Job processed without errors<br>0 = No job or job running                                    |
|                   | AA             | Job accepted    | 1 = The job was detected and accepted. Is being processed.<br>0 = No job active                  |
|                   | CP             | Codetag Present | Data carrier is in the read range of the read/write head   |
|                   |                |                 | No data carrier in read range  |

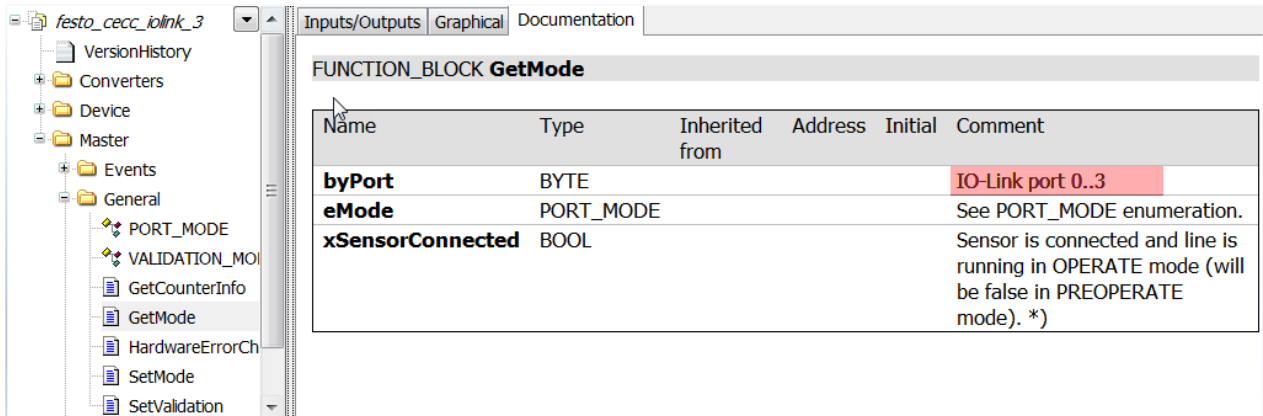
We can see the BB and CP is TRUE, the others are FALSE, which means “device is ready” and “data carrier is in the read range of the read/write head”.

|                     |                      |        |             |        |       |         |               |
|---------------------|----------------------|--------|-------------|--------|-------|---------|---------------|
| IO-Link Device Info | Channels             |        |             |        |       |         |               |
|                     | Variable             | Map... | Channel     | Add... | Type  | Defa... | Current Value |
|                     |                      |        | Byte 11     | %I...  | USINT |         | 0             |
|                     |                      |        | Byte 10     | %I...  | USINT |         | 0             |
|                     |                      |        | Byte 9      | %I...  | USINT |         | 0             |
|                     |                      |        | Byte 8      | %I...  | USINT |         | 123           |
|                     |                      |        | Byte 7      | %I...  | USINT |         | 148           |
|                     |                      |        | Byte 6      | %I...  | USINT |         | 179           |
|                     |                      |        | Byte 5      | %I...  | USINT |         | 74            |
|                     |                      |        | Byte 4      | %I...  | USINT |         | 83            |
| IO-Link Parameter   |                      |        | Byte 3      | %I...  | USINT |         | 1             |
|                     |                      |        | Byte 2      | %I...  | USINT |         | 8             |
|                     |                      |        | Byte 1      | %I...  | USINT |         | 224           |
|                     |                      |        | Bitheader 1 | %I...  | USINT |         | 129           |
|                     |                      |        | Bit0        | %I...  | BOOL  | FALSE   | TRUE          |
|                     |                      |        | Bit1        | %I...  | BOOL  | FALSE   | FALSE         |
|                     |                      |        | Bit2        | %I...  | BOOL  | FALSE   | FALSE         |
|                     |                      |        | Bit3        | %I...  | BOOL  | FALSE   | FALSE         |
|                     |                      |        | Bit4        | %I...  | BOOL  | FALSE   | FALSE         |
|                     |                      |        | Bit5        | %I...  | BOOL  | FALSE   | FALSE         |
| IO-Link I/O Mapping |                      |        | Bit6        | %I...  | BOOL  | FALSE   | FALSE         |
|                     |                      |        | Bit7        | %I...  | BOOL  | FALSE   | TRUE          |
|                     | Process Data Outputs |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
| Information         |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
| Status              |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |
|                     |                      |        |             |        |       |         |               |

## 2.3 Programming in Codesys

### 2.3.1 Program to read parameter by coding

First we test the IO-Link connection by the Function Block GetMode.



Being noticed that we are using IOL port 2, but in this FB we should use port number “1”!

If the connection is OK, you get the return value “True”.

```

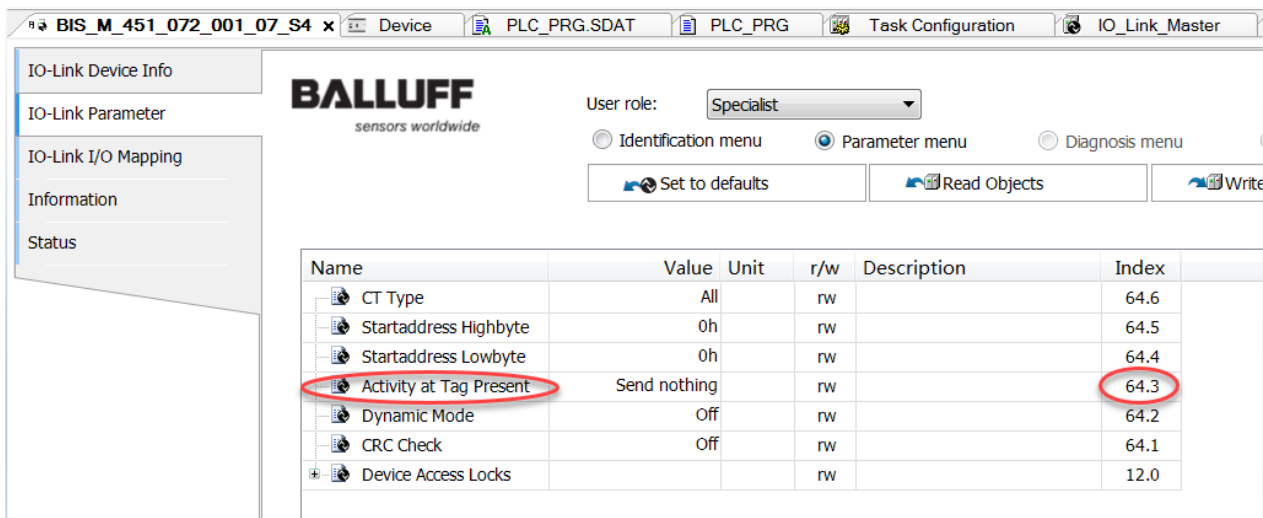
1 GetModePort1 (byPort 16#00 := 0, xSensorConnected TRUE => xPort1Connected TRUE );
2 GetModePort2 (byPort 16#01 := 1, xSensorConnected TRUE => xPort2Connected TRUE );
3 GetModePort3 (byPort 16#02 := 2, xSensorConnected FALSE => xPort3Connected FALSE );
4 GetModePort4 (byPort 16#03 := 3, xSensorConnected FALSE => xPort4Connected FALSE );

```

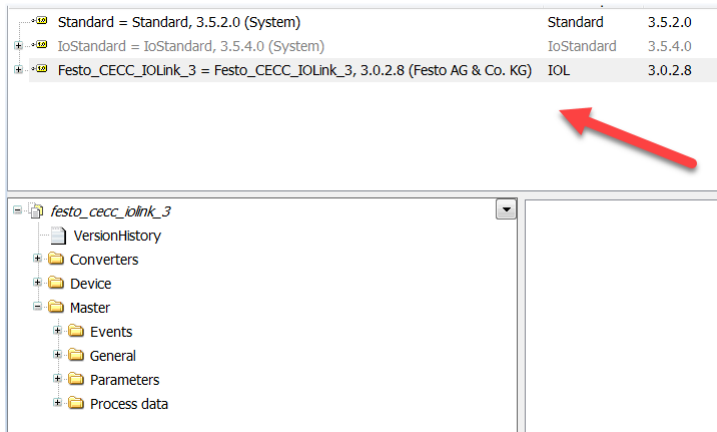
We want the data carrier to send data once the data is in the range of BIS0103.

This is controlled by the parameter Activity at Tag Present at Index 64.3.

In the screenshot, we can see the parameter has currently value of “Send nothing”. Therefore we need to program to firstly read the parameter in code, and then change it accordingly if necessary.



### Import the library for IOL CECC-LK



### Use FB readParameter to get the value

```

1 IF xPort2Connected THEN
2   CASE byStepPort2 OF
3     0: //check the parameter Activity at Tag Present
4       IF NOT readParameterPort2.xBusy THEN
5         readParameterPort2(xExecute := TRUE, byPort := 1, wIndex := 64, bySubindex := 3, pbyData := ADR(byActivity));
6       ELSE
7         ReadParameterPort2(xExecute := FALSE);
8       END_IF
9
10      IF ReadParameterPort2.xDone THEN
11        IF byActivity = 1 THEN
12          byStepPort2 := 2;
13        ELSE
14          byStepPort2 := 1;
15        END_IF
16      END_IF

```

The right value of "send UID" is 1, we check this value

### 2.3.2 Program to write parameter by coding

When the data carrier does not have the right parameter to send its UID, then we should use the FB WriteParameter to set the parameter with Index 64.3.

#### Coding and write operation ok

```

1: //set the mode if mode is not read UID
   byActivityM16#01 := 1; //set the mode to read UID
   IF NOT writeParameterPort2.xBusy THEN
     writeParameterPort2(xExecute := TRUE, byPort16#01 := 1, wIndex16#0040 := 64, bySubindex16#03 := 3, pbyData16#4048B364 := ADR(byActivityM16#01), by
   ELSE
     WriteParameterPort2(xExecute := FALSE);
   END_IF
   IF WriteParameterPort2.xDone16#02 := 2;
   ELSIF WriteParameterPort2.byErrorCode16#00 <> 0 THEN
     byStepPort216#02 := 99;
   END_IF
2:
;
99: //Error handling
;

```

OK and no error

Check the parameter again, and we see the value of this parameter has already changed to required one.

**BALLUFF**  
sensors worldwide

User role: Specialist

☐ Identification menu
 ☒ Parameter menu
 ☐ Diagnosis menu

**Read Objects again** →

| Name                    | Value            | Unit | r/w | Description | Index |
|-------------------------|------------------|------|-----|-------------|-------|
| CT Type                 | All              |      | rw  |             | 64.6  |
| Startaddress Highbyte   | 0h               |      | rw  |             | 64.5  |
| Startaddress Lowbyte    | 0h               |      | rw  |             | 64.4  |
| Activity at Tag Present | Send ID and type |      | rw  |             | 64.3  |
| Dynamic Mode            | Off              |      | rw  |             | 64.2  |
| CRC Check               | Off              |      | rw  |             | 64.1  |
| Device Access Locks     |                  |      | rw  |             | 12.0  |

### 2.3.3 Read stored data from data carrier

Let's try to read first 16 bytes.



#### Note:

As per experiment, you can still send data by sending the command 01 in address 01hex no matter in which "Activity at the Tag Present"!

#### Output buffer:

| Subaddress \ Bit No.               | 7                                   | 6  | 5  | 4 | 3 | 2  | 1 | 0  |
|------------------------------------|-------------------------------------|----|----|---|---|----|---|----|
| 00 <sub>hex</sub> - 1st bit string |                                     | TI | KA |   |   | GR |   | AV |
| 01 <sub>hex</sub>                  | Command designator or data          |    |    |   |   |    |   |    |
| 02 <sub>hex</sub>                  | Start address (low byte) or data    |    |    |   |   |    |   |    |
| 03 <sub>hex</sub>                  | Start address (high byte) or data   |    |    |   |   |    |   |    |
| 04 <sub>hex</sub>                  | Number of bytes (low byte) or data  |    |    |   |   |    |   |    |
| 05 <sub>hex</sub>                  | Number of bytes (high byte) or data |    |    |   |   |    |   |    |
| 06 <sub>hex</sub>                  | Data                                |    |    |   |   |    |   |    |
| 07 <sub>hex</sub>                  | Data                                |    |    |   |   |    |   |    |
| 08 <sub>hex</sub>                  | Data                                |    |    |   |   |    |   |    |
| Last byte - 2nd bit string         |                                     | TI | KA |   |   | GR |   | AV |

#### Explanations on the output buffer using 10 bytes as an example:

| Subaddress        | Bit name           | Meaning     | Function description   |
|-------------------|--------------------|-------------|--|
| 00 <sub>hex</sub> | 1st bit string     |             |  |
|                   | TI                 | Toggle bit  | A state change during a job indicates that the controller is ready to receive additional data made available by the read/write device. |
|                   | KA                 | Head on/off | 1 = Head off (read/write head switched off)<br>0 = Head on (read/write head in operation)  |
|                   | GR                 | Basic state | 1 = Software reset - causes the BIS to switch to the ground state<br>0 = Normal operation  |
|                   | AV                 | Job         | 1 = New job pending<br>0 = No new job or job no longer pending   |
| 01 <sub>hex</sub> | Command designator |             | 00 <sub>hex</sub> = No command   |
|                   |                    |             | 01 <sub>hex</sub> = Read data carrier  |
|                   |                    |             | 02 <sub>hex</sub> = Write data carrier   |
|                   |                    |             | 12 <sub>hex</sub> = Initialize the CRC_16 data check on the data carrier   |
|                   |                    |             | 32 <sub>hex</sub> = Write a constant value on the data carrier   |

**Be careful that both AV in first byte and last byte must be activated!** By this device, parameters of in process data are redundant, so you have to set both, otherwise it will cause error.

```

2 | //read first 8 byte from data carrier
  | IF NOT xInitStep2 TRUE THEN
  |   %QB31.16#01 := 16#01; //Output Address 01
  |   %QB30.16#00 := 16#00; //Output Address 02
  |   %QB29.16#00 := 16#00; //Output Address 03
  |   %QB28.16#10 := 16#10; //Output Address 04
  |   %QX32.0 FALSE := TRUE; //Set AV True: pend new job
  |   %QX1.0 FALSE := TRUE; //Set AV true

```

Here the meaning of parameters is: read the 16 bytes of the data carrier starting from the address 0 (internal address of data carrier!).

If everything is correct you will get activated AA and AE, and the first 16 bytes from the data carrier in **Process Data address** addressed from 01hex to 16hex..

As we can see in the software

|                     |                     |         |             |         |       |               |               |
|---------------------|---------------------|---------|-------------|---------|-------|---------------|---------------|
| IO-Link Device Info | Channels            |         |             |         |       |               |               |
| IO-Link Parameter   | Variable            | Mapping | Channel     | Address | Type  | Default Value | Current Value |
| IO-Link I/O Mapping | Process Data Inputs |         | Bitheader 2 | %IB5    | USINT |               | 16#86         |
| Information         |                     |         | Bit0        | %IX5.0  | BOOL  | FALSE         | FALSE         |
| Status              |                     |         | Bit1        | %IX5.1  | BOOL  | FALSE         | TRUE          |
|                     |                     |         | Bit2        | %IX5.2  | BOOL  | FALSE         | TRUE          |
|                     |                     |         | Bit3        | %IX5.3  | BOOL  | FALSE         | FALSE         |
|                     |                     |         | Bit4        | %IX5.4  | BOOL  | FALSE         | FALSE         |
|                     |                     |         | Bit5        | %IX5.5  | BOOL  | FALSE         | FALSE         |
|                     |                     |         | Bit6        | %IX5.6  | BOOL  | FALSE         | FALSE         |
|                     |                     |         | Bit7        | %IX5.7  | BOOL  | FALSE         | TRUE          |

Now, all the data from data carrier are:

| Variable | Mapping | Channel | Address | Type  | Default Value | Current Value |
|----------|---------|---------|---------|-------|---------------|---------------|
| *-*      |         | Byte 19 | %IB17   | USINT |               | 16#00         |
| *-*      |         | Byte 18 | %IB18   | USINT |               | 16#00         |
| *-*      |         | Byte 17 | %IB19   | USINT |               | 16#00         |
| *-*      |         | Byte 16 | %IB20   | USINT |               | 16#00         |
| *-*      |         | Byte 15 | %IB21   | USINT |               | 16#00         |
| *-*      |         | Byte 14 | %IB22   | USINT |               | 16#00         |
| *-*      |         | Byte 13 | %IB23   | USINT |               | 16#00         |
| *-*      |         | Byte 12 | %IB24   | USINT |               | 16#00         |
| *-*      |         | Byte 11 | %IB25   | USINT |               | 16#00         |
| *-*      |         | Byte 10 | %IB26   | USINT |               | 16#00         |
| *-*      |         | Byte 9  | %IB27   | USINT |               | 16#99         |
| *-*      |         | Byte 8  | %IB28   | USINT |               | 16#99         |
| *-*      |         | Byte 7  | %IB29   | USINT |               | 16#99         |
| *-*      |         | Byte 6  | %IB30   | USINT |               | 16#99         |
| *-*      |         | Byte 5  | %IB31   | USINT |               | 16#99         |
| *-*      |         | Byte 4  | %IB32   | USINT |               | 16#99         |
| *-*      |         | Byte 3  | %IB33   | USINT |               | 16#99         |
| *-*      |         | Byte 2  | %IB34   | USINT |               | 16#00         |
| *-*      |         | Byte 1  | %IB35   | USINT |               | 16#00         |

You have the possibility to read data in code.

```

IF %IX5.1 FALSE AND %IX5.2 FALSE THEN
  arrOldReadData[0] 16#00 := %IB35 16#00;
  arrOldReadData[1] 16#00 := %IB34 16#00;
  arrOldReadData[2] 16#99 := %IB33 16#99;
  arrOldReadData[3] 16#99 := %IB32 16#99;
  arrOldReadData[4] 16#99 := %IB31 16#99;
  arrOldReadData[5] 16#99 := %IB30 16#99;
  arrOldReadData[6] 16#99 := %IB29 16#99;
  arrOldReadData[7] 16#99 := %IB28 16#99;
  arrOldReadData[8] 16#99 := %IB27 16#99;
  arrOldReadData[9] 16#00 := %IB26 16#00;
  arrOldReadData[10] 16#00 := %IB25 16#00;
  arrOldReadData[11] 16#00 := %IB24 16#00;
  arrOldReadData[12] 16#00 := %IB23 16#00;
  arrOldReadData[13] 16#00 := %IB22 16#00;
  arrOldReadData[14] 16#00 := %IB21 16#00;
  arrOldReadData[15] 16#00 := %IB20 16#00;
  byStepPort2 16#04 := 3;
END_IF

```

After the read of data is finished, AV should be set to 0(release the new job pending)

```

68 3: IF NOT xInitStep3 TRUE THEN
69   %QX32.0 FALSE := FALSE; //Set AV false: release pending new job
70   %QX1.0 FALSE := FALSE; //Set AV false
71   xInitStep3 TRUE := TRUE;
72   END_IF

```



If everything is correct, you get AA & AE from Process Data Input deactivated.

| Variable            | Mapping | Channel     | Address | Type  | Default Value | Current Value |
|---------------------|---------|-------------|---------|-------|---------------|---------------|
| Process Data Inputs |         |             |         |       |               |               |
|                     |         | Bitheader 2 | %IB5    | USINT |               | 16#81         |
|                     |         | Bit0        | %IX5.0  | BOOL  | FALSE         | TRUE          |
|                     |         | Bit1        | %IX5.1  | BOOL  | FALSE         | FALSE         |
|                     |         | Bit2        | %IX5.2  | BOOL  | FALSE         | FALSE         |
|                     |         | Bit3        | %IX5.3  | BOOL  | FALSE         | FALSE         |
|                     |         | Bit4        | %IX5.4  | BOOL  | FALSE         | FALSE         |
|                     |         | Bit5        | %IX5.5  | BOOL  | FALSE         | FALSE         |
|                     |         | Bit6        | %IX5.6  | BOOL  | FALSE         | FALSE         |
|                     |         | Bit7        | %IX5.7  | BOOL  | FALSE         | TRUE          |

AE, AA  
released

So the whole process of reading data is finished.

And the process of writing data is quite similar, so I do not write the process in this note.