Application Note

FESTO

YJKP - System diagnosis

YJKP

Description of the diagnosis elements

100260

Title ..... YJKP - System diagnosis
Version 1.10
Document no. 100260
Original ..... en
Author ..... Festo

Last saved 05.09.2019

Copyright Notice

Legal Notice

Table of contents

1 Components/Software used 5

2 Process diagnosis 6

3 Device diagnosis 10

3.1 Using WebVisu 10

3.2 Using Host 12

4 Interface diagnosis 14

1 Components/Software used

Type/Name	Version Software/Firmware	Date of manufacture
Servo press kit YJKP	general	--
Application software YJKP (GSAY-A4-F0-Z4-1.3.5)	V1.3.5	--
Firmware controller (CECC-X)	V3.4.6	--
Firmware motor controller (CMMP-AS)	V4.0.1501.2.4	--

Table 1.1: 1 Components/Software used

2 Process diagnosis

Process diagnosis shows you more details about the performed operations, such as the number of processed parts, successful operations, failed operations.. etc.

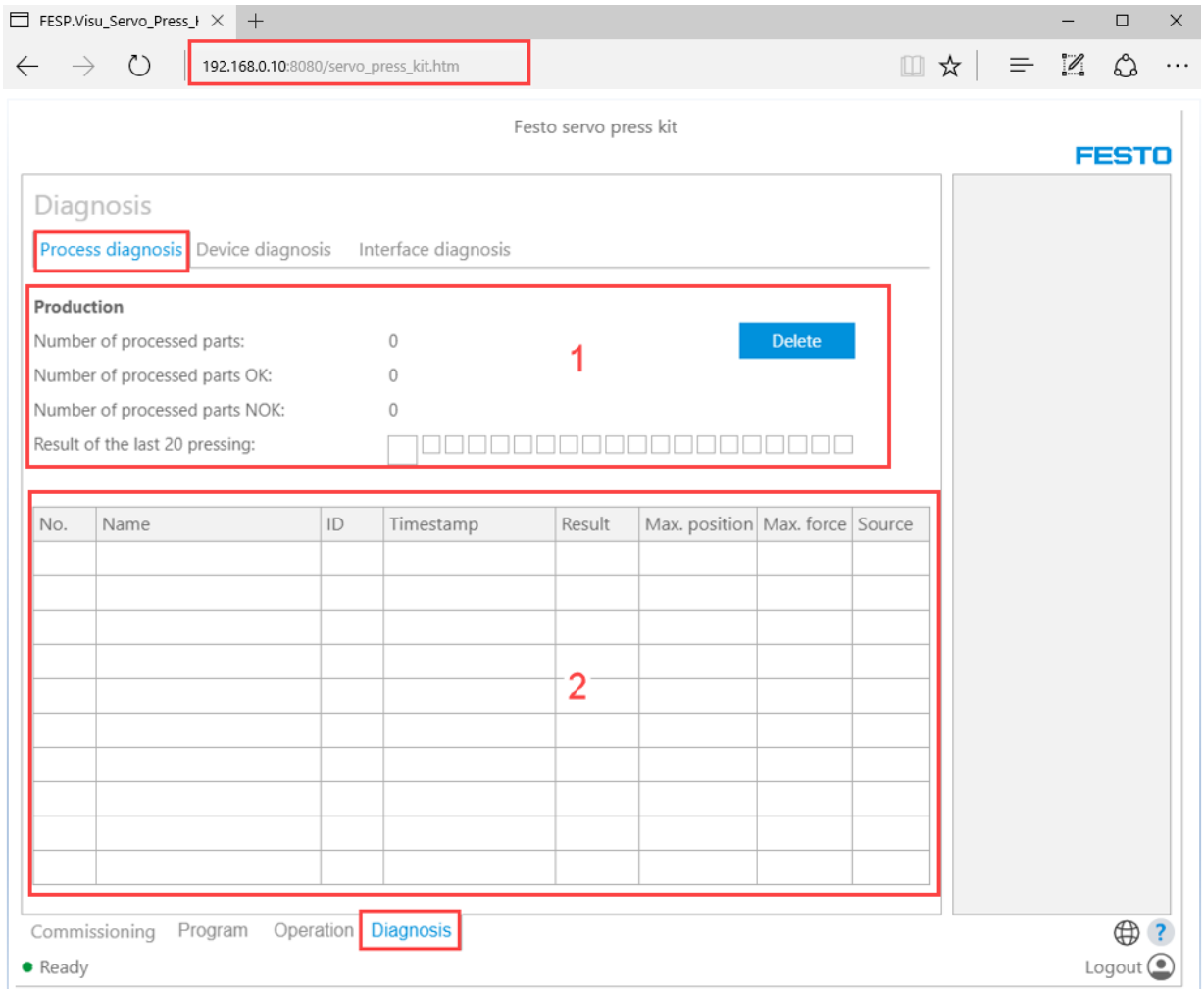
Open a browser and start the WebVisu of the YJKP.

In Browser: ‹IP of the CECC-X›:8080/servo_press_kit.htm

The visualization of the servo-press kit is opened with 4 tabs :

- Commissioning / Program : Not active
- Operation / Diagnosis: Active

Go to process diagnosis tab -› process diagnosis



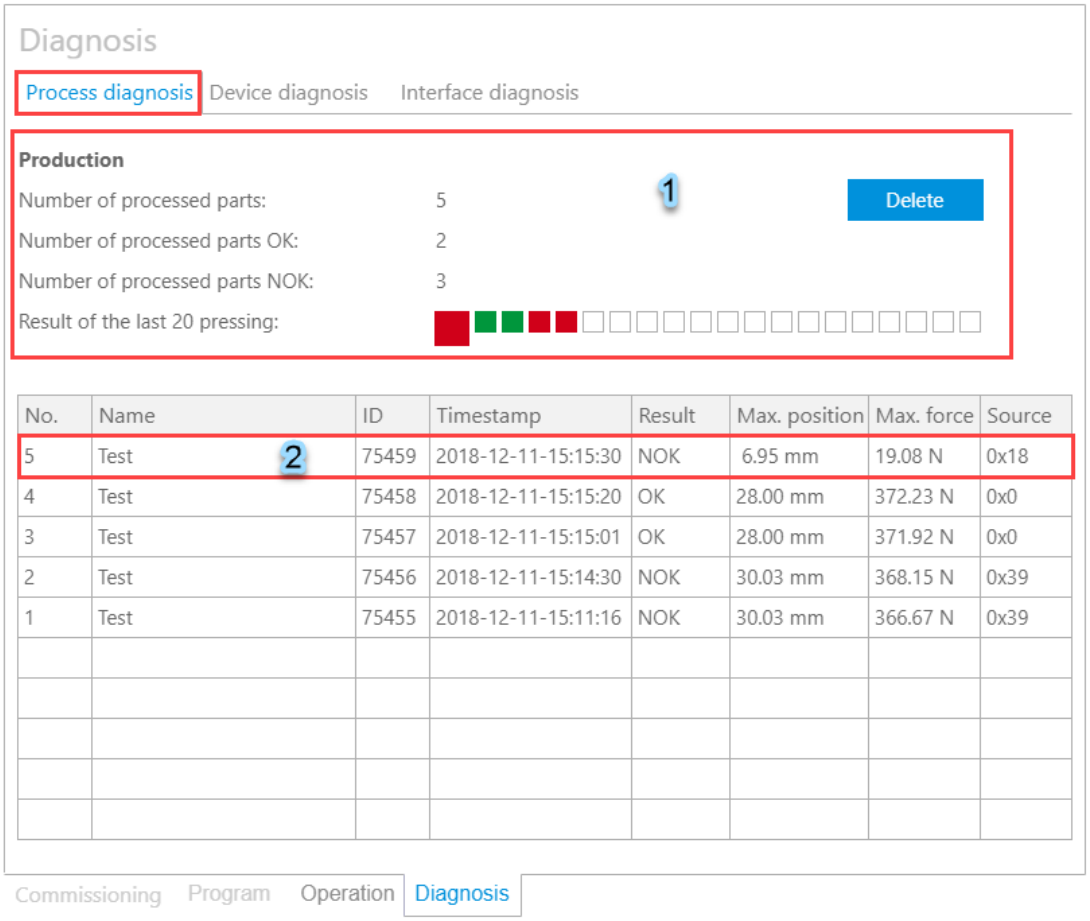
This screen consists of two parts:

1. Production: contains information about the processed parts.
   a. Number of processed parts: Total number of processed parts.
   b. Number of processed parts Ok: Number of successfully processed parts.
   c. Number of processed parts NOK: Number of failed processed parts.
   d. Result of the last 20 pressing : Status of the last 20 operations.
   e. Delete : Delete all operation results.

2. Table: contains details about each process:
   a. No: Number of the press process.
   b. Name: The name of the press program.
   c. ID: continuous production counter
   d. Timestamp: Execution time.
   e. Result : Result of the process (OK/NOK).
   f. Max.Position: Maximum position reached in this operation.
   g. Max.Force: Maximum force value reached in this operation.
   h. Source: NOK source in case of an NOK-part.

Example:

Suppose that the process diagnosis screen for a press program was as follows:



No.	Name	ID	Timestamp	Result	Max. position	Max. force	Source
5	Test	75459	2018-12-11-15:15:30	NOK	6.95 mm	19.08 N	0x18
4	Test	75458	2018-12-11-15:15:20	OK	28.00 mm	372.23 N	0x0
3	Test	75457	2018-12-11-15:15:01	OK	28.00 mm	371.92 N	0x0
2	Test	75456	2018-12-11-15:14:30	NOK	30.03 mm	368.15 N	0x39
1	Test	75455	2018-12-11-15:11:16	NOK	30.03 mm	366.67 N	0x39

This screen gives these information:

1. Production:
   a. Number of processed parts: 5
   b. Number of processed parts Ok : 3
   c. Number of processed parts NOk : 2
   d. Overview of the last 20 pressing result in order of their appearance.

2. Table (example of the last pressing operation):
   a. No: Last pressing operation (after a reboot of the system for instance) is number 5
   b. Name: program name is “Test”
   c. ID: number 75459 overall pressed part
   d. Timestamp: 2018-12-11-15:15:30
   e. Result : NOK
   f. Max.Position: 6.95 mm
   g. Max.Force: 19.08 N
   h. Source: Move the mouse over the code (0x18), a new screen will show up describing the reason for the NOK part. In this case the program was interrupted in step 1 by an external termination (xAbort / X3.5 := false)

NOK Reason

Termination of the press process

Step: 1

Note

If you are using a host control to control the press process, these information are also available.

You will need to use two function block:

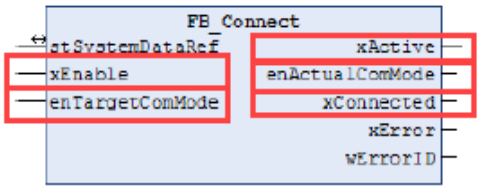
1. FB_Connect

   Required inputs:

   - xEnable := true;
   - enTargetComMode := 1;

   Required outputs:

   - xActive := true;
   - enActualComMode := 1;
   - xConnected := true;



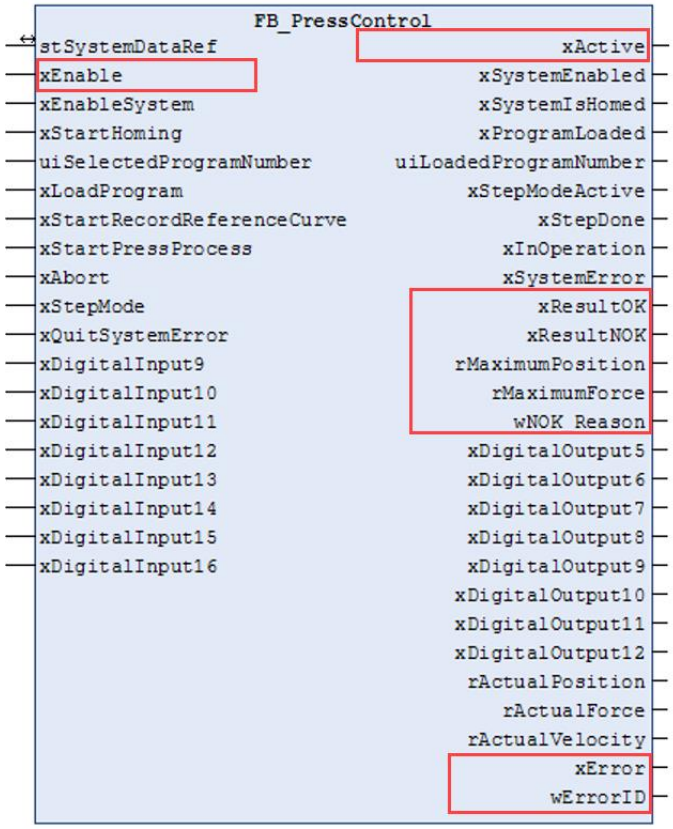
2. FB_PressControl

   Required inputs:

   - xEnable := true;

   Required outputs:

   - xActive := true;
   - xResultOk := ; *(True if the process was successful)*
   - xResultNok:= ; *(True if the process failed)*
   - rMaximumPosintion:=; *(The greatest position reached in this operation)*
   - rMaximumForce:=; *(The greatest force value reached in this operation)*
   - wNOK_Reason:= ; *(The reason of the operation failure)*
   - *xError:=;* *(Error occurred during action)*
   - *wErrorID:=;* *(Contains detailed error information in case of an error)*



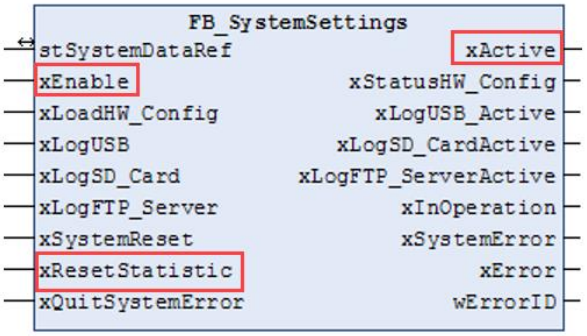
If you like to delete all operation results, use the FB_SystemSettings:

Required inputs:

- xEnable := true;
- xResetStatistic:=True;

Required outputs:

- xActive := true;



3 Device diagnosis

3.1 Using WebVisu

Device diagnosis provides you information about current system errors.

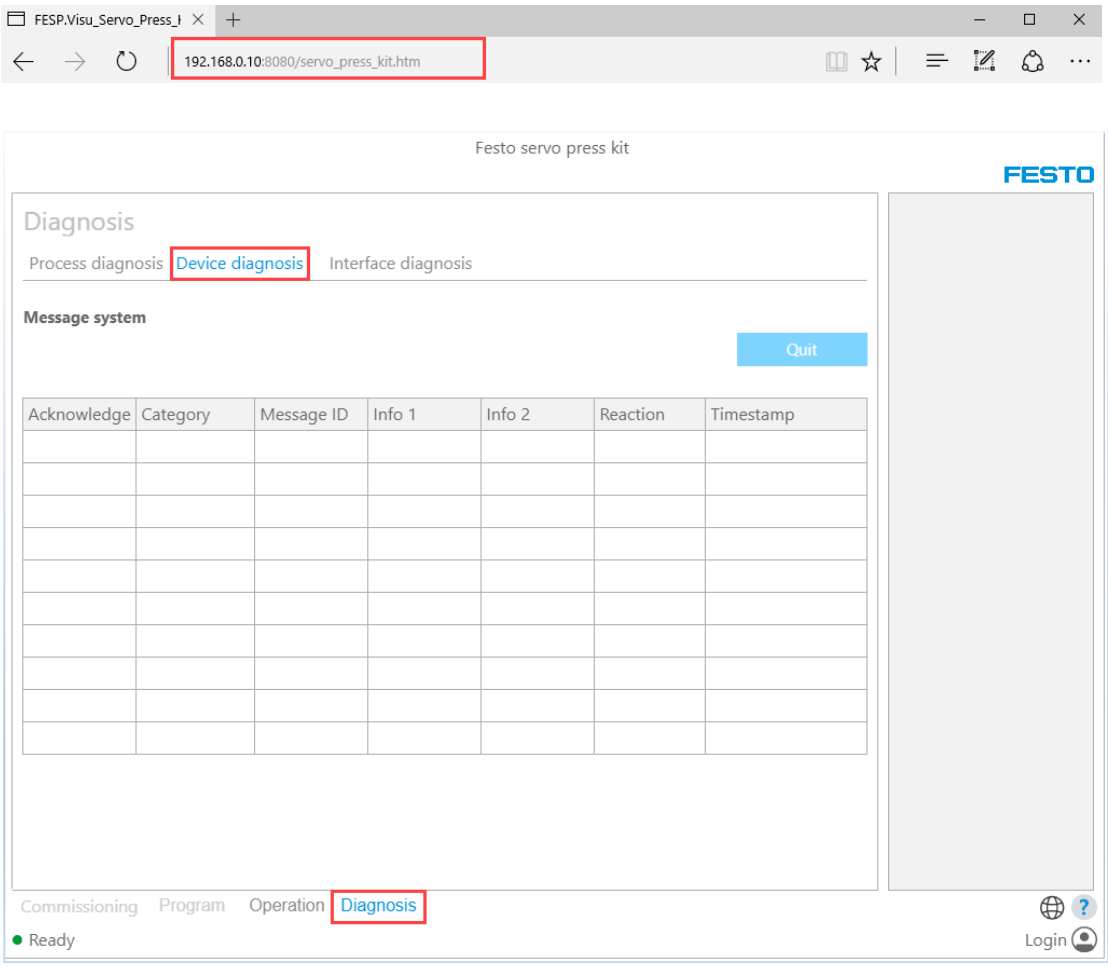
Open a browser and start the WebVisu of the YJKP.

In Browser: ‹IP of the CECC-X›:8080/servo_press_kit.htm

The visualization of the servo-press kit is opened with 4 tabs :

- Commissioning / Program : Not active
- Operation / Diagnosis: Active

Go to Diagnosis tab -› Device diagnosis

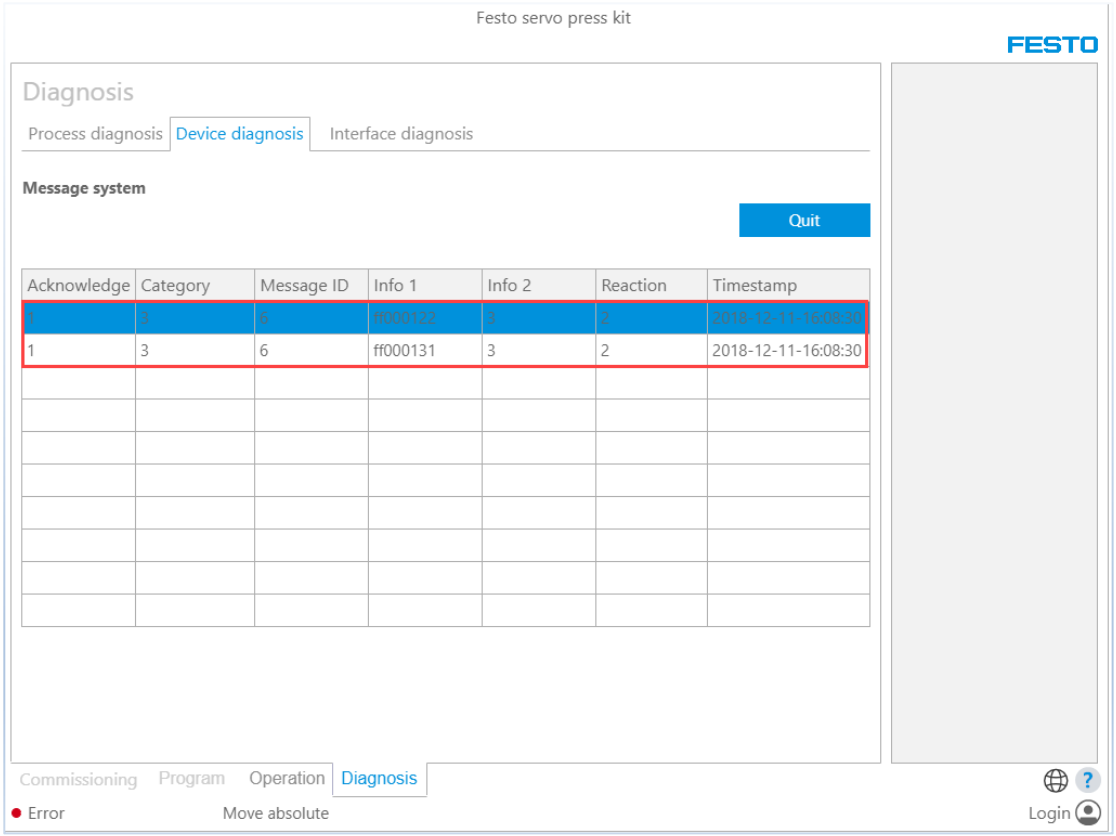


This screen consists of:

1. Table : contains information about a current error.
2. Quit: quits the system error one by one.

Example:

Suppose that the device diagnosis screen for a press program was as follows:



As shown, these errors are represented coded.

Theses messages appear in the order in which they occur, which means that the top message represents the last error. Conversely, the message in the bottom collums is usually the root cause.

If the timestamp, like in this example, is the same, the messages should be read in a common context.

Move the mouse over a one of these codes and a screen with detailled information will show up.

In this example we will get the following screen over the first message:

For the operator it's important to know, which error occured, why it came up and how to solve it.
Here we get already a lot of information:

- Error: Move absolute, so the system tried to execute a movement to an absolute position.
- Source: Error reason. In this example, the motor controller is not enabled.
- Action: How to solve it?
  1. Activate motor controller (**Input X17.4.2** or via a **function block (xEnableSystem)**).
  2. Acknowledge error (**Input X3.4** or via a **function block (xQuitSystemError)**).

The “Reaction” is an internal information and can be ignored by the operator.

All information can be also found in the online help. Here you can focus the collums “Info1” and “Info2”, which define the error message in detail.

ff000122 / 3:

0xFF000122	0x00000003	Move absolute	– Motor controller not released	• Enable motor controller • Acknowledge error

ff000131 / 3:

0xFF000131	0x00000003	Sequencer	– Error in position mode	• Acknowledge error

With these information we know, that a problem occurred, while the sequencer was executed in a position mode step.

In position mode, the system will either do an absolute or relative movement. In this case it was an absolute positioning. This movement couldn’t be carried out, because the motor controller was not released.

3.2 Using Host

If you use a host control to control the press process, two function blocks are required to see the device status. A third one is needed, if you want to get detailed information.

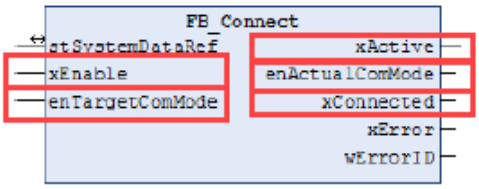
1. FB_Connect

   Required inputs:

   - xEnable := true;
   - enTargetComMode := 1;

   Required outputs:

   - xActive := true;
   - enActualComMode := 1;
   - xConnected := true;



2. FB_Status / FB_PressControl / FB_Manual

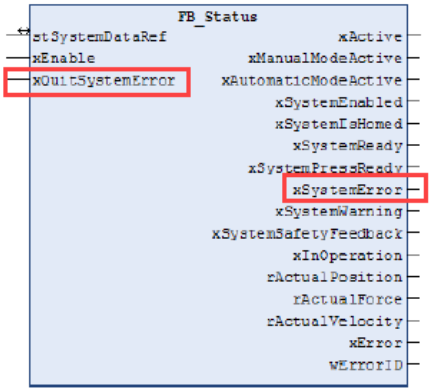
   Each of these function blocks has an output, which signals an error and an input to acknowledge it.

   Required inputs:

   - xEnable := true;
   - xQuitSystemError:= ; *(Quit system error command)*

   Required outputs:

   - xActive := true;
   - *xSystemError:=;* *(true in case a system error)*



3. FB_ReadWriteObject

Detailed information are available in a specific object (0x2200), which can be read with this function block.

Most important subindexes of this object are 0x04 and 0x05, which represent the exact error code.

Message	0x2200	0x00	31	UDINT	4	RO
Message ack	0x2200	0x01	--	BOOL	1	RO
Message id	0x2200	0x02	--	DWORD	4	RO
Message table	0x2200	0x03	--	WORD	2	RO
Message ident additional 01	0x2200	0x04	--	DWORD	4	RO
Message ident additional 02	0x2200	0x05	--	DWORD	4	RO
Message source device	0x2200	0x06	--	UINT	2	RO
Message source system	0x2200	0x07	--	UINT	2	RO
Message reaction	0x2200	0x08	--	UINT	2	RO
Message category	0x2200	0x09	--	UINT	2	RO
Message timestamp	0x2200	0x0A	--	DWORD	4	RO

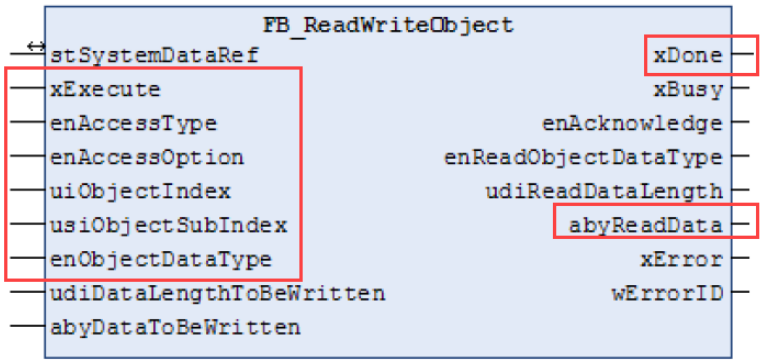
Example:

Required inputs:

- enAccessType := 0x00 (read)
- enAccessOption := 0x00 (single)
- uiObjectIndex := 0x2200 (object message)
- usiObjectSubIndex := 0x04 (subindex for additional info 1)
- enObejctDataType := 0x0A (DWORD)
- xExecute := ; true (object is read with rising edge)

Required outputs:

- xDone := true;
- abyReadDate := ; (read data)



4 Interface diagnosis

Interface diagnosis shows the status of the CECC-X inputs/outputs, and some information about the communication process when using a host controller to control the press process.

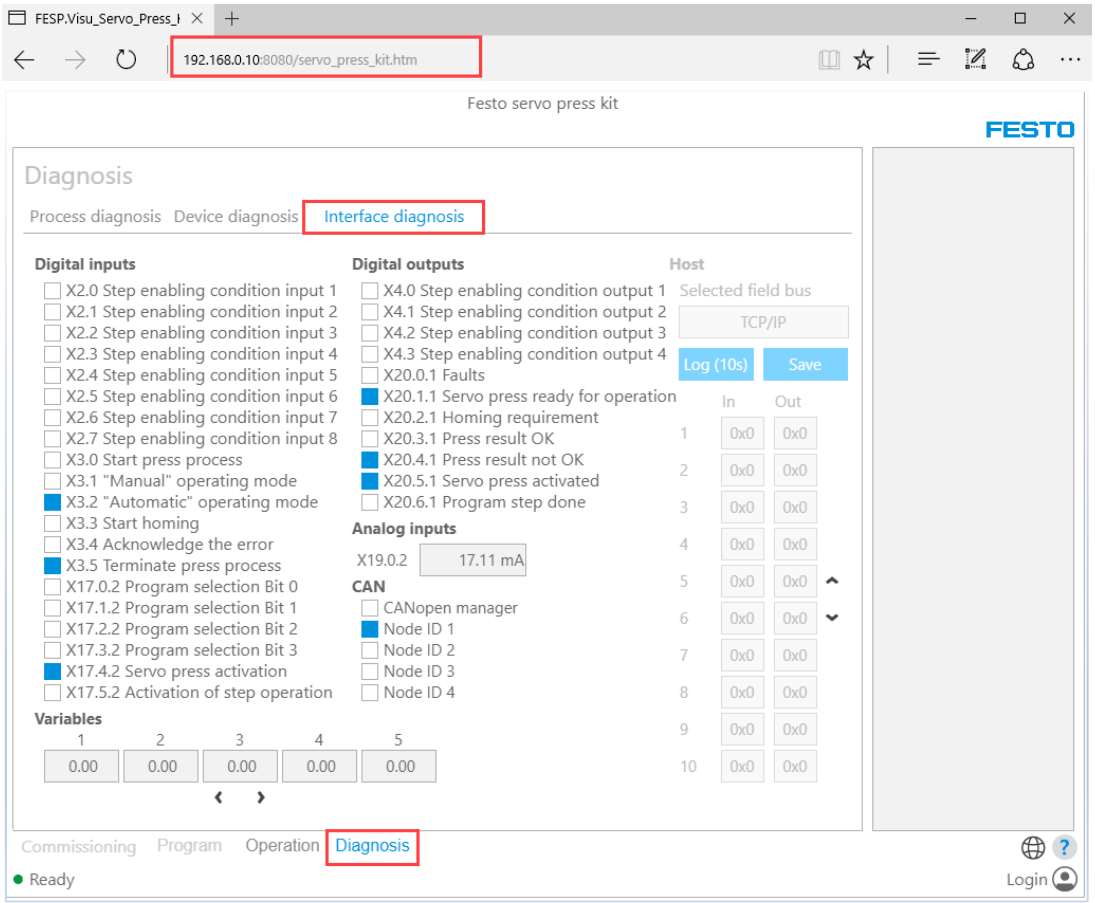
Open a browser and start the WebVisu of the YJKP.

In Browser: ‹IP of the CECC-X›:8080/servo_press_kit.htm

The visualization of the servo-press kit is opened with 4 tabs :

- Commissioning / Program : Not active
- Operation / Diagnosis: Active

1. Go to Diagnosis -› Interface diagnosis.



This screen consists of:

- **Digital inputs:** Status of the CECC-X digital inputs.
- **Digital outputs:** Status of the CECC-X digital outputs.
- **Analog inputs:** Value of the sensor, that is connected to the X19.0.2 controller analog input.
- **CAN:** ID Node of the Motor controller, that is connected to the controller CECC-X.
- **Variable:** Values of the variables, which can be written over a Host-PLC or within the sequencer using the function “Variable”.
- **Host:** If the system is controlled with a Host-PLC, this part will be active. It provides bytewise information about the input and outputdata on the interface. This communication can be logged for 10s, which is mainly used for debugging purposes for instance in case of communication problems. The log-file is saved on the SD-card.