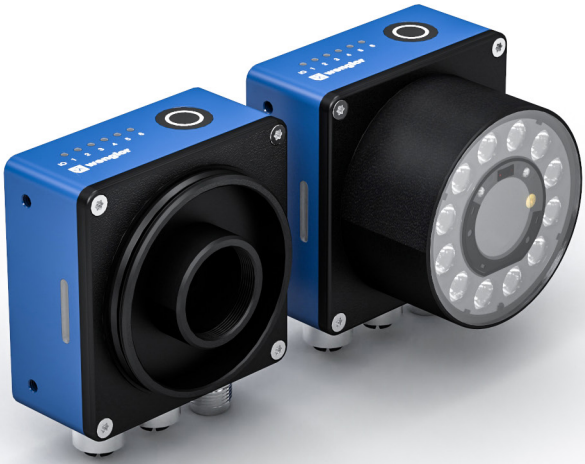
EN

WenglorTcpIP Library

Integration of B60 Smart Camera with Opcon, CoDeSys V2 and TwinCAT 2



Interface Protocol

Subject to change without notice
Available as PDF version only
Version 1.0.0
Status: 19.03.2025
www.wenglor.com

Table of contents

1. Change Index of Operating Instructions 3

2. Copyrights 3

3. The “TcpIpWenglorFb” 3

3.1 General Information for handling the FB 3
3.2 Data structures of the FB 4
3.2.1 TcpWenglorParaStruct 4
3.3 Transfer parameters of the FB 5
3.4 Funktions and Examples 7
3.4.1 Declaration of the Instance of the FB 7
3.4.2 Parameterization and Request of the Instance of the FB 7
3.4.3 “trigger” Order 8
3.4.4 “acqOn” Order 9
3.4.5 “acqOff” Order 10
3.4.6 “jobLoad” Order 11
3.4.7 “readValue” Order 12

4. Additional required Hardware and Software 12

1. Change Index of Operating Instructions

Version	Date	Description	Compatibility
1.0.0	19.03.2025	Initial version of WenglorTcpIp.lib operating instructions	Firmware MVC 1.0.0 Software: wenglor uniVision 3.3.0

2. Copyrights

3. The “TcpIpWenglorFb”

With this function block, Wenglor industrial cameras can be operated by Opcon, CodeSys V2 and TwinCAT 2 automation systems via Tcp/Ip protocol.

3.1 General Information for handling the FB

The FB requires parameterization. The structure of the required parameters is created under “TcpWenglor-ParaStruct”. The function block will be active when all parameters are assigned. In the first cycle, the internal variables are initialized, the values of the parameters are accepted and checked for plausibility. If everything is OK, the “rdyCmd” output is set to TRUE.

The FB must run every PLC cycle. This is firstly because several PLC cycles are needed to execute almost all commands and secondly the device status needs to be checked continuously.

FB has Boolean inputs for orders. The FB starts the order when it detects a rising edge on the order input. If the order is processed without an error, then “done” output is set to TRUE. If there is an error, “fault” output is set to TRUE and at the same time the corresponding error number is set to “fltNo” output.

If there is a “fltNo” not equal to 0 detected while waiting for “done”, then the order NOK is processed. If the order should be repeated, then the order input is to be set to FALSE, the error acknowledged and the order input reset to TRUE. As long as the order input is set to TRUE, the error number at “fltNo” output cannot be set to 0.

3.2 Data structures of the FB

3.2.1 TcpWenglorParaStruct

Syntax

```
TYPE TcpWenglorParaStruct :
STRUCT
   ipAddress         : STRING(15);
   confPort          : DINT;
   processPort       : DINT;
   preamble          : STRING;
   postamble         : STRING;
   delimiter         : STRING;
   webVisualization  : STRING;
END_STRUCT
END_TYPE
```

Description

Contains the parameters for the FB

Field

Field	Meaning
ipAddress	IP address of B60 Smart Camera
confPort	Port number for configuration communication with B60 Smart Camera
processPort	Port number for process communication with B60 Smart Camera
preamble	Prefix for process data
postamble	Suffix for process data
delimiter	Separator for process data
webVisualization	Address of camera web visualization

3.3 Transfer parameters of the FB

Function
TcpWenglorFb

Syntax

```
FUNCTION_BLOCK TcpWenglorFb
VAR_INPUT
   pQuit            : POINTER TO BOOL :=16#FFFFFFFF;
   par              : TcpWenglorParaStruct;
   trigger          : BOOL;
   acqOn            : BOOL;
   acqOff           : BOOL;
   jobLoad          : BOOL;
   readValue        : BOOL;
   newJobName       : STRING;
   readValuePath    : STRING;
END_VAR

VAR_OUTPUT
   fltNo            : TcpWenglorFaultEnum;
   ready            : BOOL;
   rdyCmd           : BOOL;
   done             : BOOL;
   connConfPort     : BOOL;
   connProcessPort  : BOOL;
   resultData       : ARRAY [1..8] OF STRING;
   deviceStatus     : STRING(200);
   processStatus    : STRING(200);
   value            : STRING;
   actJobName       : STRING;
END_VAR
```

Input parameter

Parameter name	Meaning
pQuit	Pointer to a variable of type Boolean. If the order inputs are FALSE, an existing error (fltNo<>0) is cleared wit this input.
par	Parameter structure of the type “TcpWenglorParaStruct”
trigger	Starts order to trigger
acqOn	Starts data acquisition on order
acqOff	Starts data acquistion off order
jobLoad	Starts load job order
readValue	Starts read value from a parameter order
newJobName	New job name to load

Output parameter

Parameter name	Meaning
fitNo	Error number 0: no error < 0: error present
ready	Displays whether the FB is on operation
rdyCmd	Displays whether the FB is ready to start an order
done	Displays that active order was ended successfully
connConfPort	Displays configuration port is connected
connProcessPort	Displays process port is connected
resultData	Contains the result data after successful trigger order
deviceStatus	Displays detailed info about device status
processStatus	Displays detailed info about status of Processing Instance
value	Contains the value after successful read value order
actJobName	Displays actual job name

Description
With one instance of this FB a device is controlled.

Return (fltNo)

NO_FAULT	There is no error present.
FLT_CONN_TIMEOUT	There is a connection timeout error on at least on of the ports.
FLT_SEND_DATA	There is an error while sending data.
FLT_RECEIVE_DATA_TIMEOUT	There is a timeout error while waiting to receive data
FLT_RECEIVE_DATA	There is an error while receiving data
FLT_CMD_NOT_SUCCESSFUL	According to the response from the device, there is an error with the command.

3.4 Funktions and Examples

A B60 Smart Camera should be active in the system.

3.4.1 Declaration of the Instance of the FB

In order to control the device, at least one instance of the FB is defined in the global data range.

```
VAR_GLOBAL
   TcpIpWenglor : TcpIpWenglorFb;
END_VAR
```

3.4.2 Parameterization and Request of the Instance of the FB

At first cycle an error quit signal should be defined as a pointer input to FB.
After the parameterization FB is called at every cycle.

```
VAR
   first : BOOL := FALSE;
END_VAR

IF ( first = FALSE )
THEN
   TcpIpWenglor.pQuit := ADR(State.Gen.FltReset);
ELSE
   TcpIpWenglor(par := StationData.Wenglor);
END_IF
```

3.4.3 “trigger” Order

A trigger order is started in the following way:

1. Activate rising edge on the „trigger“ input.
2. Wait until “done” output TRUE or “fltNo” is not equal to 0.
3. Analyse “resultData[1..8]” outputs.
4. Set “trigger” input back to FALSE.

```
CASE step OF
   0: (* start trigger command *)
      TcpIpWenglor.trigger := TRUE;
      step := step + 1;

   1: (* wait for order done or failure *)
      IF ( TcpIpWenglor.done = TRUE ) OR
         ( TcpIpWenglor.fltNo <> 0      )
      THEN
        TcpIpWenglor.trigger := FALSE;


        IF ( TcpIpWenglor.fltNo = OK )
        THEN
          Step := Step + 1;
        ELSE
          HandleFault();
        END_IF
      END_IF
   ELSE
      ;
END_CASE
```

3.4.4 “acqOn” Order

A data acquisition on order is started in the following way:

1. Activate rising edge on the „acqOn“ input.
2. Wait until “done” output TRUE or “fltNo” is not equal to 0.
3. Set “acqOn” input back to FALSE.

```
CASE step OF
   0: (* start data acqusition on command *)
      TcpIpWenglor.acqOn := TRUE;
      step := step + 1;

   1: (* wait for order done or failure *)
      IF ( TcpIpWenglor.done = TRUE ) OR
         ( TcpIpWenglor.fltNo <> 0      )
      THEN
        TcpIpWenglor.acqOn := FALSE;


        IF ( TcpIpWenglor.fltNo = OK )
        THEN
          Step := Step + 1;
        ELSE
          HandleFault();
        END_IF
      END_IF
   ELSE
      ;
END_CASE
```

3.4.5 “acqOff” Order

A data acquisition off order is started in the following way:

1. Activate rising edge on the „acqOff“ input.
2. Wait until “done” output TRUE or “fltNo” is not equal to 0.
3. Set “acqOff” input back to FALSE.

```
CASE step OF
   0: (* start data acqusition off command *)
      TcpIpWenglor.acqOff := TRUE;
      step := step + 1;

   1: (* wait for order done or failure *)
      IF ( TcpIpWenglor.done = TRUE ) OR
         ( TcpIpWenglor.fltNo <> 0      )
      THEN
        TcpIpWenglor.acqOff := FALSE;


        IF ( TcpIpWenglor.fltNo = OK )
        THEN
          Step := Step + 1;
        ELSE
          HandleFault();
        END_IF
      END_IF
   ELSE
      ;
END_CASE
```

3.4.6 “jobLoad” Order

A job load order is started in the following way:

1. Write the new job name to the “newJobName” input.
2. Activate rising edge on the “jobLoad” input.
3. Wait until “done” output TRUE or “fltNo” is not equal to 0.
4. Check “actJobName” output.
5. Set “jobLoad” input back to FALSE.

```
CASE step OF
   0: (* start job load command *)
      TcpIpWenglor.newJobName := newJobName;
      TcpIpWenglor.jobLoad:= TRUE;
      step := step + 1;

   1: (* wait for order done or failure *)
      IF ( TcpIpWenglor.done = TRUE ) OR
         ( TcpIpWenglor.fltNo <> 0      )
      THEN
        TcpIpWenglor.jobLoad := FALSE;


        IF ( TcpIpWenglor.fltNo = OK )
        THEN
          Step := Step + 1;
        ELSE
          HandleFault();
        END_IF
      END_IF
   ELSE
      ;
END_CASE
```

3.4.7 “readValue” Order

A read value order is started in the following way:

1. Write the parameter path to the “readValuePath” input
2. Activate rising edge on the “readValue” input.
3. Wait until “done” output TRUE or “fltNo” is not equal to 0.
4. Analyze “value” ouput.
5. Set “readValue” input back to FALSE.

```
CASE step OF
   0: (* start job load command *)
      TcpIpWenglor.readValuePath := readValuePath;
      TcpIpWenglor.readValue:= TRUE;
      step := step + 1;

   1: (* wait for order done or failure *)
      IF ( TcpIpWenglor.done = TRUE ) OR
         ( TcpIpWenglor.fltNo <> 0      )
      THEN
        TcpIpWenglor.readValue := FALSE;


        IF ( TcpIpWenglor.fltNo = OK )
        THEN
          Step := Step + 1;
        ELSE
          HandleFault();
        END_IF
      END_IF
   ELSE
      ;
END_CASE
```

4. Additional required Hardware and Software

The B60 Smart Camera and PLC must be connected to the same Ethernet network. There is no need for any additional hardware.

The “WenglorTcpIp.lib” uses the following standard CoDeSys libraries.

```
S02\SysLibs\Standard.LIB 10.2.16 15:20:42
S02\SysLibs\SysLibMem.lib 10.2.16 15:20:42
S02\SysLibs\SysLibSockets.lib 10.2.16 15:20:42
S02\SysLibs\SysLibCallback.lib 10.2.16 15:20:42
```

The “WenglorTcpIp.lib” library functional test was performed with the following hardware and PLC system

Hardware	PLC system
Rexroth VPB40.3	Opcon - HMI Modulo