KEBA AG
Active Contact Flange Provider

Version 1.0.0

User’s guide

July 18, 2012

NOTES:

[Revision history]

Version	Date	Contents
1.0.0	2012-7-18	First edition.

[Hardware]

Model	Version	Notes
ACF 110-04		ACF TCP/IP Ethernet Communication Interface version 1040SV_03.00.00
ACF 110-10		
ACF 120-05		
ACF 120-10		

Contents

1. Introduction

This documentation describes how to use the KEBA ACF Provider for an easily communication with the Active Contact Flange from KEBA. The Provider can be used with a PC or directly using the RC8 controller with the DENSO PacScript language.

The connection with the ACF is based on the TCP/IP Ethernet Communication Interface. The Provider is acting as a Client and the ACF as a Server waiting for commands.

2. Outline of provider

2.1. Outline

The ACF Provider offers one execution method by CaoController::Execute to send parameters and receive current status values using the Active Contact Flange TCP/IP Ethernet Communication Interface version 1040SV_03.00.00

Table 2-1 Active Contact Flange provider

File name	CaoProvACF.dll
ProgID	CaoProv.KEBA.ACF
Registry registration	regsvr32 [PATH]/ CaoProvACF.dll
Registry un-registration	regsvr32 /u [PATH]/ CaoProvACF.dll

2.2. Method and property

The ACF provider connects at the time when the method AddConnection.is executed.

2.2.1. CaoWorkspace::AddController method

Syntax

AddController (< bstrCtrlName:VT_BSTR > and < bstrProvName:VT_BSTR >
<bstrPcName:VT_BSTR > [,<bstrOption:VT_BSTR>])

bstrCtrlName :[in] Controller name arbitrary.

bstrProvName : [in] Provider name. (Fixed to ” CaoProv.KEBA.ACF”)

bstrPcName : [in] Execution machine name of provider

bstrOption : [in] Option character string

Following is a list of option string items.

Option	Meaning
Conn=< connection parameter >	Set the communication form and connection parameters
ID=< id information>	Information about ACF model type and software controller version

Conn option

The following describes the connection parameter string for the Conn option for Ethernet.

“eth:<IP Address>[:<Port No>]"

<IP Address>	:	Mandatory. Specify IP address. Example:"192.168.0.1"
<Port No>	:	Specify the port number to communicate: Example:"192.168.0.1:7070" (The Port for ACF is always 7070)

ID option

<ID>	:	Model type and software controller version. Example: ID=104003000

2.2.2. CaoController::Execute method

The command name is specified in the first argument and the parameter of the command is specified for the second argument. Please refer to Chapter 3 Command reference for details of each command.

Syntax Execute (<bstrCommandName:VT_BSTR>, [<vntParam: VT_VARIANT])

bstrCommandName	:	[in] Command name (VT_BSTR)
vntParam	:	[in] Parameter
Return value	:	[out] Return value of command (VT_R4 | VT_ARRAY or VT_BSTR)

3. Command reference

3.1. Controller class

Table 3-1 CaoController::Execute command list

Command	Function	Page
Send	Send new values to ACF	8

3.1.1. CaoController::Execute("Send") command

Syntax Send (<set_f_target>, <set_f_zero>, <set_t_ramp>, <set_payload>)

set_f_target >	:	[in] Force to be applied (Unit:N) (VT_R4)
<set_f_zero>	:	[in] Minimal force where to start applying a force ramp when contact is detected (VT_R4) Unit: N
<set_t_ramp>	:	[in] Time used to apply a force ramp between set_f_zero and set_f_target (VT_R4) Unit: Sec Range: 0 – 10
<set_payload>	:	Payload of the tool applied to the active contact flange (VT_R4) Unit: kg Range: ACF 110 = max. 10, ACF 120 = max. 50
<act_force>	:	[out] Current force the active contact flange is applying
<act_stroke>	:	[out] Current stroke of the active contact flange
<act_contact_state>	:	[out] Indicates if the active contact flange detects contact
<act_error_code>	:	[out] Bitmap of occurred errors

Example

```
float[] input = new float[] { 1.2f, 3.6f, 1.5f, 1.8f };
CaoController.Execute("Send", input);
```

4. Sample program

List 4-1 **Example.cs**

```
public class Program
{
        static void Main(string[] args)
        {
            CaoEngine _engine;
            CaoController _controller;
            float[] sendValues;
            float[] receivedValues;

            _engine = new CaoEngine();
            _controller = _engine.Workspaces.Item(0).AddController("KEBA",
"CaoProv.KEBA.ACF", "", "Conn=eth:192.168.0.1:7070, ID=1040030300");
            sendValues = new float[] { 1.6f, 5.2f, 2.2f, 1.3f };
            receivedValues = new float[4];

            for (int i = 0; i < 100; i++)
            {
                receivedValues = (float[])_controller.Execute("Send", sendValues);

                Console.WriteLine(string.Format(
                    "Result: {0}, {1}, {2}, {3}",
                    receivedValues[0],
                    receivedValues[1],
                    receivedValues[2],
                    receivedValues[3]));

                Thread.Sleep(20);
            }
 }
```

List 4-2 PacScriptExample.pcs

```
'!TITLE " PacScriptExample.pcs "
#define VT_R4                   4
Sub Main
            TakeArm    Keep = 0
            Dim SendValues As Variant
            Dim ReceiveValues As Variant
            Dim KebaController As Object

            '    Connect to KEBA ACF
            KebaController    =    CAO.AddController("KEBA",    "CaoProv.KEBA.ACF",    "",
"Conn=eth:127.0.0.1:7070, ID=1040030300")

            SendValues = CreateArray(4, VT_R4)
            SendValues(0) = 1.2
            SendValues(1) = 1.4
            SendValues(2) = 1.5
            SendValues(3) = 1.7

            ReceiveValues = CreateArray(4, VT_R4)
            ReceiveValues(0) = 0
            ReceiveValues(1) = 0
            ReceiveValues(2) = 0
            ReceiveValues(3) = 0

      for I1 = 0 to 5
            DELAY 20
            ReceiveValues = KebaController.Execute("Send", SendValues)
            Debug.Print "Received Values"
            Debug.Print ReceiveValues(0)
            Debug.Print ReceiveValues(1)
            Debug.Print ReceiveValues(2)
            Debug.Print ReceiveValues(3)
      next
End Sub
```