WinAC

OPC-Client

Anwenderdokumentation

V1.2 • November 2009

Applikationen & Tools

Answers for industry.

SIEMENS

Industry Automation und Drives Technologies Service & Support Portal

Dieser Beitrag stammt aus dem Internet Serviceportal der Siemens AG, Industry Automation und Drives Technologies. Durch den folgenden Link gelangen Sie direkt zur Downloadseite dieses Dokuments.

http://support.automation.siemens.com/WW/view/de/48355169

Bei Fragen zu diesem Beitrag wenden Sie sich bitte über folgende E-Mail-Adresse an uns:

online-support.automation@siemens.com

SIEMENS

SIMATIC
WinAC OPC-Client

Gewährleistung und Haftung

Hinweis

Die Applikationsbeispiele sind unverbindlich und erheben keinen Anspruch auf Vollständigkeit hinsichtlich Konfiguration und Ausstattung sowie jeglicher Eventualitäten. Die Applikationsbeispiele stellen keine kundenspezifischen Lösungen dar, sondern sollen lediglich Hilfestellung bieten bei typischen Aufgabenstellungen. Sie sind für den sachgemäßen Betrieb der beschriebenen Produkte selbst verantwortlich. Diese Applikationsbeispiele entheben Sie nicht der Verpflichtung zu sicherem Umgang bei Anwendung, Installation, Betrieb und Wartung. Durch Nutzung dieser Applikationsbeispiele erkennen Sie an, dass wir über die beschriebene Haftungsregelung hinaus nicht für etwaige Schäden haftbar gemacht werden können. Wir behalten uns das Recht vor, Änderungen an diesen Applikationsbeispielen jederzeit ohne Ankündigung durchzuführen. Bei Abweichungen zwischen den Vorschlägen in diesem Applikationsbeispiel und anderen Siemens Publikationen, wie z.B. Katalogen, hat der Inhalt der anderen Dokumentation Vorrang.

Für die in diesem Dokument enthaltenen Informationen übernehmen wir keine Gewähr.

Unsere Haftung, gleich aus welchem Rechtsgrund, für durch die Verwendung der in diesem Applikationsbeispiel beschriebenen Beispiele, Hinweise, Programme, Projektierungs- und Leistungsdaten usw. verursachte Schäden ist ausgeschlossen, soweit nicht z.B. nach dem Produkthaftungsgesetz in Fällen des Vorsatzes, der groben Fahrlässigkeit, wegen der Verletzung des Lebens, des Körpers oder der Gesundheit, wegen einer Übernahme der Garantie für die Beschaffenheit einer Sache, wegen des arglistigen Verschweigens eines Mangels oder wegen Verletzung wesentlicher Vertragspflichten zwingend gehaftet wird. Der Schadensersatz wegen Verletzung wesentlicher Vertragspflichten ist jedoch auf den vertragstypischen, vorhersehbaren Schaden begrenzt, soweit nicht Vorsatz oder grobe Fahrlässigkeit vorliegt oder wegen der Verletzung des Lebens, des Körpers oder der Gesundheit zwingend gehaftet wird. Eine Änderung der Beweislast zu Ihrem Nachteil ist hiermit nicht verbunden.

Inhaltsverzeichnis

Einführung

Ziel der Applikation:

Das Dokument beschreibt die Software **WinAC OPC-Client** für den Anwender.

1 Grundlegende Informationen

1.1 Aufgabenstellung

Der SIMATIC NET OPC Server ermöglicht es OPC-Client –Applikationen auf WinAC Daten zuzugreifen. WinAC RTX kann sich aber standardmäßig nicht mit anderen OPC-Servern verbinden

Diese Funktionalität wurde nun mit dem **WinAC OPC-Client** ergänzt. So ist es jetzt möglich, aus der WinAC RTX SPS Programm direkt mit einen anderen OPC-Server Daten auszutauschen. Einzige Voraussetzung ist, dass sich der OPC-Server auf demselben PC-System befindet wie die WinAC RTX Runtime.

1.2 Referenzsystem

Die in dieser Dokumentation beschriebene Applikation basiert auf folgendem Referenzsystem:

- SIMATIC Field PG M II mit WinAC RTX 2009-F
- SIMATIC NET 6.3 (nur zum Testzwecken als OPC-Server)
- SIMATIC Manager V5.4, SP4

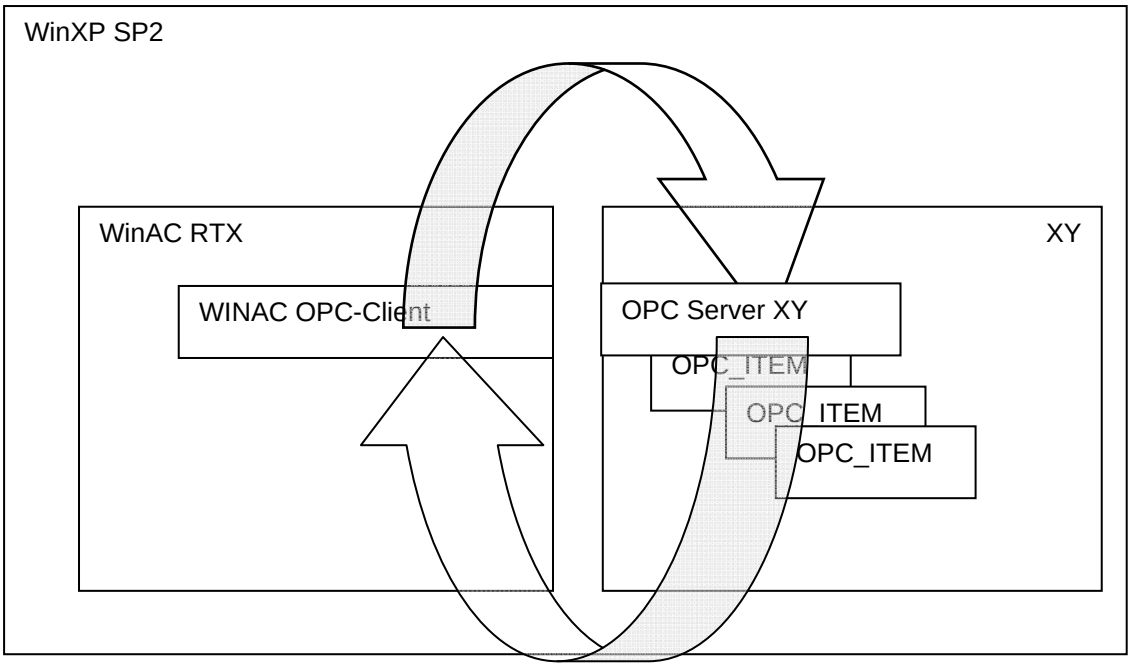
2 Übersicht

2.1 Funktionsumfang

WinAC OPC-Client ermöglicht der Kommunikation zwischen WinAC und einen beliebigen OPC Server die sich auf dem Zielsystem befinden. Das heißt nur lokale OPC-Server werden unterstützt.

Der WinAC OPC-Client kann bis zu 4000 verschiedenen Variablen am OPC-Server anmelden, diese lesen und auf diese schreiben.

Abbildung 2-1 Übersicht



3 Installation

3.1 Quickstart

Mit den folgenden Schritten lässt sich **WinAC OPC-Client** in Betrieb nehmen.

- Die WinAC_OPCClient.dll ins **system32**-Verzeichnis kopieren. (setup.bat)
- Kopieren von Komponenten aus dem Demo-Projekt in das Anwenderprojekt:
  - die FBs 65000-65007 einschließlich ihrer Instanz-DBs den UDT 4
  - die SFBs 65001 und 65002
- Ein OPC-Client Parameter DB erzeugen und mit OPC-Server Name und OPC Items (UDT4) parametrieren oder bestehendes Beispiel DB (DB2000) anpassen und erweitern.
- FB65000 OPC_INIT ausführen
- FB65003 OPC_CONN und FB65004 OPC_ADD_ITEM mit Werten aus Parameter DB aufrufen
- Wenn Verbindung aufgebaut ist und die Items angemeldet sind, zyklisch lesen bzw. schreiben mit FB65006 OPC_READ und FB65007 OPC_WRITE.

4 Die Anwenderschnittstelle

Als Anwenderschnittstelle dienen folgende S7-Funktionsbausteine

- FB65000 OPC_INIT → Initialisiert WinAC OPC Client
- FB65003 OPC_CONN → Verbinden mit dem OPC-Server
- FB65004 OPC_ADD_ITEM → Die OPC Items anmelden
- FB65006 OPC_READ → Lese alle „READ“ Variablen
- FB65007 OPC_WRITE → Schreibe alle “WRITE” Variablen

Als spezielle Datentyp für die OPC Items dient

- UDT4 OPC_ITEM_DATA → Parameter für eine OPC Variable

Die jeweilige FB- bzw. UDT- Nummer kann man an die Projekt Gegebenheiten angepasst werden.

4.1 Initialisierung FB65000 OPC_INIT

Dieser FB initialisiert die ODK-DLL („WinAC_OPCClient.dll“). Er muss einmalig (z.B. im OB100) aufgerufen werden. Erst nach einem Aufruf des **OPC_INIT** können die restlichen FBs des WinAC OPC- Clients benutzt werden.

Schnittstelle

Tabelle 4-1 Parameter des FBs OPC_INIT

Parameter	In/Out	Typ	Beschreibung
DB_OPC_CONN	In	Block_DB	Instanz-DB von OPC_CONN
DB_OPC_ADD_ITEM	In	Block_DB	Instanz-DB von OPC_ADD_ITEM
DB_OPC_READ	In	Block_DB	Instanz-DB von OPC_READ
DB_OPC_WRITE	In	Block_DB	Instanz-DB von OPC_WRITE
ERROR	Out	Bool	Fehler ist aufgetreten
STATUS	Out	Word	Status des Aufrufes (auszuwerten, wenn ERROR gesetzt)

Hier müssen alle Instanz DBs der restlichen OPC-Client FBs angegeben werden zwecks Initialisierung.

Rückgabeinformationen

Im Fehlerfall (z.B. ODK DLL auf dem PC-System nicht vorhanden) wird ein negativer Statuswert ausgegeben. Error Flag ist in diesem Fall gesetzt.

Um welche Fehler sich konkret handelt sehen Sie im Kapitel 6 „Fehlermeldungen“.

4.2 Verbindung aufbauen FB65003 – OPC_CONN

Dieser FB Baut eine Verbindung zwischen WinAC RTX und dem OPC-Server auf.

Schnittstelle

Tabelle 4-2 Parameter des OPC_CONN

Parameter	In/Out	Typ	Beschreibung
REQ	In	BOOL	Request Anforderung für die Verbindungsaufbau
OPC_SERVER	In	STRING	OPC-Server Name (z.B. „SIMATICNET.OPC“)
DONE	Out	BOOL	Fertig Flag
BUSY	Out	BOOL	Beschäftigt Flag
ERROR	Out	BOOL	Error Flag

Die Verbindung wird erst dann aufgebaut, wenn das „REQ“ Flag gesetzt ist. Während des Verbindungsaufbaus steht das „BUSY“ Flag an. Wenn die Funktion bearbeitet wurde, wird das „DONE“ Flag gesetzt. Im Fehlerfall wird zusätzlich das „ERROR“ Flag gesetzt.

Rückgabeinformationen

Im Fehlerfall (wenn das ERROR Flag ansteht) werden die Fehler Codes im Instanz-DB des FB (z.B. DB6503) gespeichert:

- Status → ODK-Fehler siehe Kap. 6.1
- ErrorCode → Anwender- bzw. Systemfehler siehe Kap. 0
- OPC_Error_Code → Die OPC Fehler Codes sind durch die OPC Foundation definiert. Man kann Sie z.B. unter http://www.advosol.us/OpcErrorLookup.aspx nachlesen.

4.3 OPC Items anmelden FB65004 – OPC_ADD_ITEMS

Dieser FB meldet die Items (Variablen) beim OPC Server an. Die einzelne OPC Items sind im entsprechenden UDT4 „OPC_ITEM_DATA“ beschrieben.

Datentyp UDT4 – OPC_ITEM_DATA

Bei diesem Datentyp handelt sich um eine Struktur mit folgendem Aufbau:

Tabelle 4-3 UDT4 OPC_ITEM_DATA Struktur:

Parameter	In/Out	Typ	Komentar
OPC_ITEM_NAME	In	STRING[254]	OPC-Item Name
dataType	In	BYTE	Datentyp siehe Tabelle 4-4
quantity	In	INT	Anzahl Werte (Array>1)
dbNumber	In	INT	DB Nummer (Wenn memoryArea Typ ein DB ist)
memoryArea	In	BYTE	Typ des Speicherbereichs: DB,

Parameter	In/Out	Typ	Komentar
			M , E, A,... siehe Tabelle 4-5
areaOffset	In	INT	Speicherbereich Byte Offset
bitNumber	In	BYTE	Speicherbereich Bit Offset
Read_Write_Mask	In	BYTE	Item to Read = 0; Write = 1; Read and Write = 2
Status	Out	DWORD	Status der Variable <>0 → Fehler bei der Variable.

„OPC_ITEM_NAME“ beschreibt den Namen der OPC-Variablen. Die restlichen Parameter beschreiben einen Speicherbereich innerhalb der WinAC RTX SPS oder Status bzw. Kommunikationsrichtung.

Die Syntax von „OPC_ITEM_NAME“ ist OPC-Server abhängig. Typ und länge des mit „OPC_ITEM_NAME“ beschriebenen Items muss mit der Angabe dataType, quantity, usw. übereinstimmen.

Folgende Datentypen (Parameter „**dataType“**) des Speicherbereichs werden unterstützt:

Tabelle 4-4 mögliche dataType:

Hex	Datatype
0x01	Bit
0x02	Byte
0x03	Char
0x04	Word
0x05	Integer
0x06	Double Word
0x07	Double Integer
0x08	Real
0x09	Date
0x0A	Tod
0x0B	Time
0x13	S7-String

Als Speicherbereich (Parameter „**memoryArea**“) kommen folgende Typen in Frage:

Tabelle 4-5 mögliche memoryArea:

Hex	Speichertyp
0x80	(P) Peripheral
0x81	(E) Process image
0x82	(A) Process image
0x83	(M) Flag
0x84	(DB) Data Block
0x01	(DS) Data Set
0x03	(SD) AS200:SYS-AREA
0x04	(S) AS200:stage bits
0x05	(SF) AS200:special flag
0x06	(AI) AS200:analog input
0x07	(AQ) AS200:analog output
0x08	(SZL) System State List

Welche OPC Items gelesen bzw. geschrieben werden, hängt von Parameter „**Read_Write_Mask**“ ab:

0 → Die Item-Daten werden von OPC-Server gelesen und in den WinAC Speicherbereich kopiert.

1 → Die Item-Daten werden aus dem Speicherbereich der WinAC geholt und zum OPC-Server geschrieben.

2 → Die Item Daten werden sowohl gelesen“ als auch geschrieben.

Wenn die Anmeldung eines Items am OPC-Server gescheitert ist, wird der Parameter **Status** mit der Fehlercode beschrieben. Später werden beim Lesen bzw. Schreiben nur die Items berücksichtigt deren Status = 0 ist. (heißt: fehlerfrei bei dem OPC-Server angemeldet)

Schnittstelle

Tabelle 4-6 Parameter des FBs OPC_ADD_ITEMS

Parameter	In/Out	Typ	Beschreibung
REQ	In	BOOL	Request Anforderung für die OPC_ADD_ITEM
OPC_ITEMS	In/Out	ANY	ANY-Zeiger auf Array von UDT4 (OPC_ITEM_DATA) Strukturen
DONE	Out	BOOL	Fertig Flag
BUSY	Out	BOOL	Beschäftigt Flag
ERROR	Out	BOOL	Error Flag

Die Items werden erst dann bei dem Server angemeldet, wenn das „REQ“ Flag gesetzt ist. Es wird die Länge des ANY Pointer (OPC_ITEMS) geprüft. Sollte diese

nicht mit dem mehrfachen der UDT4 (OPC_ITEM_DATA) Struktur entsprechen, wird die Funktion mit Fehler abgebrochen.

Bei der Anmeldung werden die fehlerhaften Items mit dem Fehlercode in Status-Parameter „markiert“. Trotzdem wird die Anmeldung von den restlichen Items fortgesetzt. Am Ende der Anmeldung wird das BUSY Flag zurückgesetzt und das „DONE“ Flag (eventuell auch das „ERROR“ Flag) gesetzt.

Rückgabeinformationen

Im Fehlerfall (wenn das ERROR Flag ansteht) werden die Fehler Codes im Instanz-DB des FB (z.B. DB6504) gespeichert:

Status	→	ODK-Fehler siehe Kap. 6.1
ErrorCode	→	Anwender- bzw. Systemfehler siehe Kap. 0
OPC_Error_Code	→	Die OPC Fehler Codes sind durch die OPC Foundation definiert. Man kann Sie z.B. unter http://www.advosol.us/OpcErrorLookup.aspx nachlesen.

4.4 OPC-Items Lesen FB65006 OPC_READ

Dieser FB liest die Daten aller zum Lesen angemeldeten OPC Items vom OPC-Server und speichert diese im parametrierten SPS Speicherbereich.

Schnittstelle

Tabelle 4-7 Parameter des FBs OPC_READ

Parameter	In/Out	Typ	Beschreibung
REQ	In	BOOL	Request Anforderung für die OPC_READ
DONE	Out	BOOL	Fertig Flag
BUSY	Out	BOOL	Beschäftigt Flag
ERROR	Out	BOOL	Error Flag

Alle „READ“ OPC-Items werden immer dann gelesen, wenn das „REQ“ Flag gesetzt ist:

Nachdem alle Items gelesen wurden wird das „DONE“ Flag gesetzt und das „BUSY“ Flag zurückgesetzt. Im Fehlerfall wird zusätzlich auch das „ERROR“ Flag gesetzt.

Rückgabeinformationen

Im Fehlerfall (wenn das ERROR Flag ansteht) werden die Fehler Codes im Instanz-DB des FB (z.B. DB6506) gespeichert:

Status	→	ODK-Fehler siehe Kap. 6.1
ErrorCode	→	Anwender- bzw. Systemfehler siehe Kap. 0
OPC_Error_Code	→	Die OPC Fehler Codes sind durch die OPC Foundation definiert. Man kann Sie z.B. unter http://www.advosol.us/OpcErrorLookup.aspx nachlesen.

4.5 OPC-Items schreiben FB65007 OPC_WRITE

Dieser FB holt die Daten aller zum Schreiben angemeldeten OPC Items aus parametrierter SPS Speicherbereich und schreibt diese zum OPC-Server.

Schnittstelle

Tabelle 4-8 Parameter des FBs OPC_WRITE

Parameter	In/Out	Typ	Beschreibung
REQ	In	BOOL	Request Anforderung für die OPC_WRITE
DONE	Out	BOOL	Fertig Flag
BUSY	Out	BOOL	Beschäftigt Flag
ERROR	Out	BOOL	Error Flag

Alle „WRITE“ OPC-Items werden immer dann geschrieben, wenn das „REQ“ Flag gesetzt ist:

Nachdem alle Items geschrieben wurden, wird das „DONE“ Flag gesetzt und das „BUSY“ Flag zurückgesetzt. Im Fehlerfall wird zusätzlich auch das „ERROR“ Flag gesetzt.

Rückgabeinformationen

Im Fehlerfall (wenn das ERROR Flag ansteht) werden die Fehler Codes im Instanz-DB des FB (z.B. DB6507) gespeichert:

Status	→	ODK-Fehler siehe Kap. 6.1
ErrorCode	→	Anwender- bzw. Systemfehler siehe Kap. 0
OPC_Error_Code	→	Die OPC Fehler Codes sind durch die OPC Foundation definiert. Man kann Sie z.B. unter http://www.advosol.us/OpcErrorLookup.aspx nachlesen.

5 Programmbeispiel

Im mitgelieferten Step7 Beispielprogram wird die Verwendung der WinAC OPC-Client FBs demonstriert.

Step1

Der OPC-Client Parameter DB (DB2000) definiert die OPC Items und den OPC Server:

Tabelle 5-1 OPC_ClientDB. OPC_SERVER_NAME:

OPC_SERVER_NAME	STRING [254]	'OPC.SIMATICNET'

→ **OPC Server** ist **„OPC.SIMATICNET“**

Tabelle 5-2 OPC_ClientDB.OPC_ITEM1:

DB Variable	TYP	Wert
OPC_ITEM_NAME	STRING]	'S7:[Tst]DB10,X0.0,16'
dataType	BYTE	B#16#1 → BOOL
quantity	INT	16
dbNumber	INT	0
memoryArea	BYTE	B#16#83 → Merkerbereich
areaOffset	INT	2000
bitNumber	BYTE	B#16#0
Read_Write_Mask	BYTE	B#16#1 → WRITE
Status	DWORD	DW#16#0

→**16 Bits** von **MX2000.0,16** in OPC Item **'S7:[Tst]DB10,X0.0,16'** schreiben

Tabelle 5-3 OPC_ClientDB.OPC_ITEM2:

DB Variable	TYP	Wert
OPC_ITEM_NAME	STRING	'S7:[Tst]DB10,CHAR2,2'
dataType	BYTE	B#16#3 → Char
quantity	INT	2
dbNumber	INT	0
memoryArea	BYTE	B#16#83 → Merkerbereich
areaOffset	INT	2002
bitNumber	BYTE	B#16#0
Read_Write_Mask	BYTE	B#16#1 → WRITE
Status	DWORD	DW#16#0

→**2 CHAR** von **MB2002** in OPC Item **'S7:[Tst] DB10,CHAR2,2'** schreiben

Tabelle 5-4 OPC_ClientDB.OPC_ITEM3:

DB Variable	TYP	Wert
OPC_ITEM_NAME	STRING	'S7:[Tst]MX2022.0,7'
dataType	BYTE	B#16#1 → BOOL
quantity	INT	7
dbNumber	INT	10
memoryArea	BYTE	B#16#84 → Datenbaustein
areaOffset	INT	22
bitNumber	BYTE	B#16#0
Read_Write_Mask	BYTE	B#16#0 → READ
Status	DWORD	DW#16#0

→**7 Bits** von OPC Item **'S7:[Tst]MX2022.0,7'** lesen und in **DB10.DBX22.0** speichern

Tabelle 5-5 OPC_ClientDB.OPC_ITEM9:

DB Variable	TYP	Wert
OPC_ITEM_NAME	STRING	'S7:[Tst]MREAL2028,3'
dataType	BYTE	B#16#8 →REAL
quantity	INT	3
dbNumber	INT	10
memoryArea	BYTE	B#16#84 → Datenbaustein
areaOffset	INT	28
bitNumber	BYTE	B#16#0
Read_Write_Mask	BYTE	B#16#0 → READ
Status	DWORD	DW#16#0

→**3 REAL** von OPC Item **'S7:[Tst]MREAL2028,3'** lesen und in **DB10.DBD28** speichern

usw. (siehe DB2000 in Beispielprojekt)

Step2

Im OB100 wird OPC Client initialisiert.

Tabelle 5-6 OB100 Initialisieren

```
CALL  "OPC_INIT" , "DB_INIT_OPC"                    //Init OPC Client
     DB_OPC_CONN    :="DB_OPC_CONN"
     DB_OPC_ADD_ITEM:="DB_OPC_ADD_ITEMS"
     DB_OPC_READ    :="DB_OPC_READ"
     DB_OPC_WRITE   :="DB_OPC_WRITE"
     ERROR          :="Init_Error"
     STATUS         :=

     L     0                                        //Reset Status Flags
     T     "Flags"

     UN    "Init_Error"                             // Init error ?
     SPB   Ende                                     // N -->

     CALL  "STP"                                    // Stop PLC


Ende: BE
```

Hier wird der FB „OPC_INIT“ ausgeführt. Als Parameter wurden alle Instanz DBs der restlichen WinAC OPC-Client FBs übergeben.

Zusätzlich werden alle Status –Flags (Connected, Added,...) zurückgesetzt. Im Fehlerfall wird die CPU in Stop versetzt.

Step3:

Im OB1 wird die Verbindung mit dem OPC-Server hergesetellt, alle Items aus den Parametrier- Datenbaustein angemeldet und zyklisch gelesen bzw. geschrieben:

Tabelle 5-7 OB1 mit OPC Server verbinden, Items anmelden und lesen/schreiben:

```
      U     "Connected"                                          //If connected jump to mrk0
      SPB   mrk0

      CALL  "OPC_CONN" , "DB_OPC_CONN"                           //Connect OPC Server
       REQ       :="Connect_OPC"
       OPC_SERVER:="OPCClient_DB".OPC_SERVER_NAME
       DONE      :="Connected"
       BUSY      :="Connect_Busy"
       ERROR     :="Conn_Error"

mrk0: U     "Add_Item_Done"                                     //If items added jump to mrk1
      SPB   mrk1

      CALL  "OPC_ADD_ITEM" , "DB_OPC_ADD_ITEMS"                  //Else add items
       REQ      :="Connected"
       DONE     :="Add_Item_Done"
       BUSY     :="Add_Item_Busy"
       ERROR    :="Add_Item_Error"
       OPC_ITEMS:=P#DB2000.DBX256.0 BYTE 2720

mrk1: CALL  "OPC_READ" , "DB_OPC_READ"                         //Read all "READ" items
      REQ  :="Add_Item_Done"
      DONE :="Read_Done"
      BUSY :="Read_Busy"
      ERROR:="Read_Error"

      UN    "Read_Done"                                          // Read counter
      SPB   mrk2
      L     "Read_Counter"
      L     1
      +I
      T     "Read_Counter"

mrk2: CALL  "OPC_WRITE" , "DB_OPC_WRITE"                       // Write all "WRITE" Items
      REQ  :="Add_Item_Done"
      DONE :="Write_Done"
      BUSY :="Write_Busy"
      ERROR:="Write_Error"

      UN    "Write_Done"                                         //Write counter
      SPB   ende
      L     "Write_Counter"
      L     1
      +I
      T     "Write_Counter"

ende: BE
```

Step4

Über die Variablentabelle VAT1 kann man im Beispiel Projekt die verschiedenen Funktionen des OPC-Clients steuern und beobachten.

6 Fehlermeldungen

Der WinAC OPC-Client kann drei verschiedene Klassen von Fehlermeldungen liefern. Alle Fehlermeldungen werden in Instanz-DBs gespeichert.

- Code im Instanz-DB **ODK_Status** gemäß WinAC-ODK (siehe Kapitel 6.1 in diesem Dokument)
- Code im Instanz-DB **ODK_OUT. ErrorCode** als Anwender-/System Fehler (siehe Kapitel **Fehler! Verweisquelle konnte nicht gefunden werden.** in diesem Dokument).
- Code im Instanz-DB **ODK_OUT. OPC_ErrorCode** als OPC-Fehler (Die OPC Fehler Codes sind durch die OPC- Foundation definiert. Man kann Sie z.B. unter http://www.advosol.us/OpcErrorLookup.aspx nachlesen)

Nach außen werden die Fehler über das **Error Flag** kommuniziert.

6.1 Fehlercodes vom WinAC ODK 4.1

Der WinAC IP Treiber wurde mit dem WinAC ODK (Open Development Kit) entwickelt. Das ODK kann ebenfalls Fehlercodes generieren, die im **STATUS** der FBs zurückgegeben werden, aber nicht in der Dokumentation der FBs beschrieben sind.

6.1.1 Error Codes für SFB65001 CREA_COM

Diese Fehlermeldungen können nur vom FB **TINIT** zurückgegeben werden.

Tabelle 6-1 WinAC ODK Fehlermedlungen für CREA_COM

Error Code	Symbol	Description
0	NO_ERRORS	Success
0x807F	ERROR_INTERNAL	An internal error occurred.
0x8001	E_EXCEPTION	An exception occurred.
0x8102	E_CLSID_FAILED	The call to CLSIDFromProgID failed.
0x8103	E_COINITIALIZE_FAILED	The call to CoInitializeEx failed.
0x8104	E_CREATE_INSTANCE_FAILED	The call to CoCreateInstance failed.
0x8105	E_LOAD_LIBRARY_FAILED	The library failed to load.
0x8106	E_NT_RESPONSE_TIMEOUT	A Windows response timeout occurred.
0x8107	E_INVALID_OB_STATE	Controller is in an invalid state for scheduling an OB.
0x8108	E_INVALID_OB_SCHEDULE	Schedule information for OB is invalid.
0x8109	E_INVALID_INSTANCEID	Instance ID for SFB65001 call is invalid.
0x810A	E_START_ODKPROXY_FAILED	Controller could not load proxy DLL.
0x810B	E_CREATE_SHAREMEM_FAILED	The WinAC controller could not create or initialize shared

Error Code	Symbol	Description
		memory area.
0x810C	E_OPTION_NOT_AVAILABLE	Attempt to access unavailable option ocurred.

6.1.2 Error Codes für SFB65002 EXEC_COM

Diese Fehlermeldungen können von allen FBs zurückgegeben werden.

Tabelle 6-2 WinAC ODK Fehlermedlungen für EXEC_COM

Error Code	Symbol	Description
0	NO_ERRORS	Success
0x807F	ERROR_INTERNAL	An internal error occurred.
0x8001	E_EXCEPTION	An exception occurred.
0x8002	E_NO_VALID_INPUT	Input: the ANY pointer is invalid.
0x8003	E_INPUT_RANGE_INVALID	Input: the ANY pointer range is invalid.
0x8004	E_NO_VALID_OUTPUT	Output: the ANY pointer is invalid.
0x8005	E_OUTPUT_RANGE_INVALID	Output: the ANY pointer range is invalid.
0x8006	E_OUTPUT_OVERFLOW	More bytes were written into the output buffer by the extension object than were allocated.
0x8007	E_NOT_INITIALIZED	ODK system has not been initialized: no previous call to SFB65001 (CREA_COM).
0x8008	E_HANDLE_OUT_OF_RANGE	The supplied handle value does not correspond to a valid extension object.
0x8009	E_INPUT_OVERFLOW	More bytes were written into the input buffer by the extension object than were allocated.

6.2 Anwender-/System Fehler

Die folgende Fehler können von Treiber des OPC Client zurückgegeben werden.

Tabelle 6-3

Fehlercode	Uhrsache
0x80000001	Fehler bei der Verbindungsaufbau mit OPC-Server
0x80000002	Fehler beim Anlegen der OPC-Items
0x80000008	Fehler beim Lesen von OPC Item
0x80000010	Fehler beim Schreiben von OPC Item
0x80000100	Fehler beim ODK Input/Output
0x80000200	Fehler bei Aufruf von Asynchronfunktion (ODK Intern)
0x80001000	Falsche Länge von ANY bei der OPC_ADD_ITEM Funktion oder der Maximale Anzahl der Items ist schon erreicht (4000 Items max)
0x80002000	Verbindung mit dem Server ist noch nicht hergestellt!
0x80004000	Speicherbereich Länge entspricht nicht der Länge der gelesenen OPC Item
0x80008000	Länge der geschriebener OPC Item entspricht nicht dem Speicherbereich
0x80020000	Es ist noch kein Item angemeldet das gelesen/geschrieben werden kann.

7 Historie

Tabelle 7-1

Version	Datum	Bemerkung
V1.0	18.04.08	Erste Version
V1.1	02.04.09	Getestet mit WinAC RTX 2008
V1.2	02.11.09	Getestet mit WinAC RTX--F 2009