

# applications & TOOLS

**PC-basierte Automatisierung mit  
SIMATIC WinAC**

Anbindung von Windows Applikationen an WinAC  
RTX mittels WinAC ODK am Beispiel von SIMATIC  
Vision Sensoren

**SIEMENS**

## Gewährleistung, Haftung und Support

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

Unsere Haftung, gleich aus welchem Rechtsgrund, für durch die Verwendung der in diesem Dokument beschriebenen Beispiele, Hinweise, Programme, Projektierungs- und Leistungsdaten usw. verursachte Schäden ist ausgeschlossen, soweit nicht z.B. nach dem Produkthaftungsgesetz in Fällen des Vorsatzes, der grober 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.

Die Applikationsbeispiele sind unverbindlich und erheben keinen Anspruch auf Vollständigkeit hinsichtlich Konfiguration und Ausstattung sowie jeglicher Eventualitäten. Sie stellen keine kundenspezifische 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 dieses Applikationsbeispiels erkennen Sie an, dass Siemens über die oben beschriebene Haftungsregelung hinaus nicht für etwaige Schäden haftbar gemacht werden kann. Wir behalten uns das Recht vor, Änderungen an diesem Applikationsbeispiel 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.

**Copyright © 2005 Siemens A&D. Weitergabe oder Vervielfältigung dieser Applikationsbeispiele oder Auszüge daraus sind nicht gestattet, soweit nicht ausdrücklich von Siemens A&D zugestanden.**

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

<mailto:csweb@ad.siemens.de>

## Vorwort

### Ziel der Applikation

Zum schnellen Einstieg in die PC-basierte Automatisierung mit SIMATIC WinAC wurden acht Beispiele entwickelt. Sie bestehen aus Beispielcode und einer ausführlichen Dokumentation. Anhand dieser Beispiele können Sie als Anwender sich aufgabenbezogen in die einzelnen Themen einarbeiten.

### Kerninhalte dieser Applikation

Folgende Kernpunkte werden in dieser Applikation behandelt:

- Grundlagen zur Software-SPS SIMATIC WinAC RTX
- Grundlagen zur offenen Programmierschnittstelle WinAC ODK
- Windows Programmierung: Mechanismen zur Synchronisation
- Windows Programmierung: Mechanismen zum Datenaustausch

### Abgrenzung

Diese Applikation enthält keine vertiefende Beschreibung

- zu SIMATIC Vision Sensoren
- zu Bildverarbeitungs-Mechanismen
- zu Grundlagen der Windows Programmierung

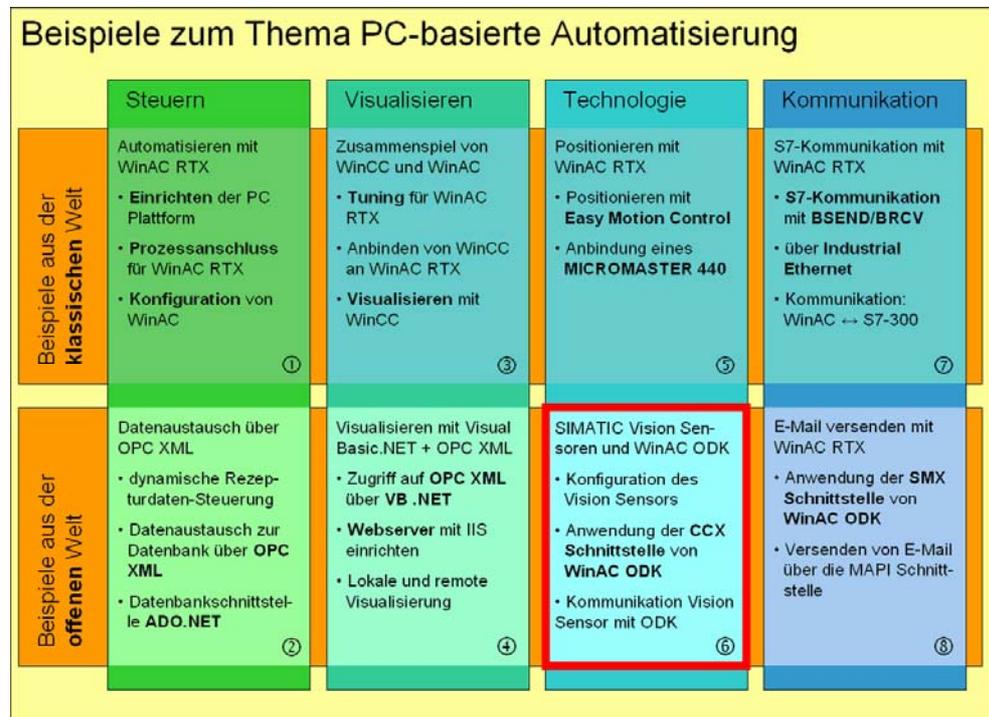
Grundlegende Kenntnisse über diese Themen werden voraus gesetzt.

### Die einzelnen Beispiele

Um PC-basierte Automatisierung optimal einsetzen zu können, haben wir für die vier typischen Aufgabenbereiche (Steuern, Kommunikation, Visualisierung, Technologie) jeweils ein Beispiel aus der „klassischen“ SPS-Welt und jeweils eines aus der „offenen“ PC-Welt entwickelt.

Die folgende Abbildung zeigt alle acht Beispiele mit ihrer Zuordnung zu den jeweiligen Aufgabenbereichen. Das vorliegende Beispiel zum Thema „SIMATIC Vision Sensor und WinAC ODK“ ist mit einem roten Rand markiert.

Abbildung 1-1



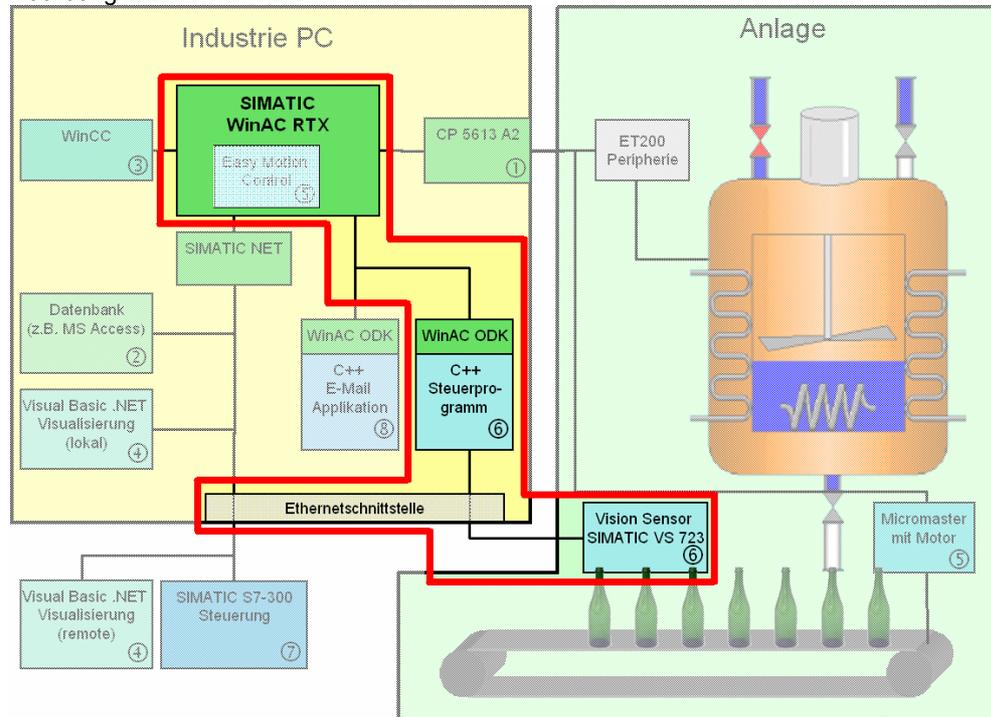
## Grundlage der Beispiele

Als gemeinsamer Aufhänger ist allen Beispielen ein fiktiver „Mischprozess“ zu Grunde gelegt. Anhand dieses „Mischprozesses“ werden die unterschiedlichen Aufgabenstellungen und Automatisierungskomponenten aus dem Produktspektrum der PC-basierten Automatisierung appliziert.

## Anlagenbild

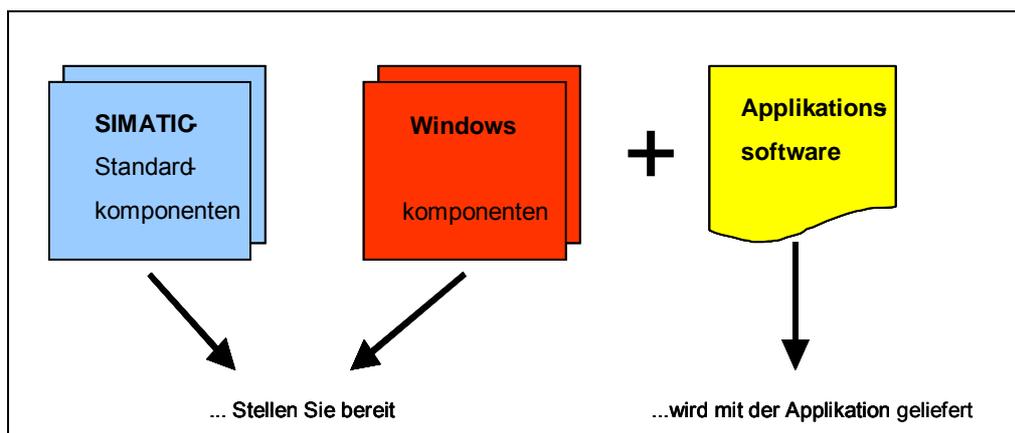
Folgendes Bild zeigt das Anlagenbild des „Mischprozesses“. Der rote Rahmen zeigt Ihnen, welche Komponenten in dem vorliegenden Beispiel behandelt werden.

Abbildung 1-2



## Lösungsgrundansatz der vorliegenden Applikation

Für die Applikation werden unterschiedliche Komponenten (Hard- und Software) benötigt. Diese werden zum Teil mit dieser Applikation mitgeliefert, teilweise stellen Sie die Komponenten zur Verfügung.



Um welche Komponenten es dabei im Einzelnen geht und wie deren Zusammenspiel miteinander funktioniert, wird in dieser Dokumentation vermittelt.

## Aufbau des Dokuments

Diese Dokumentation ist in folgende Hauptteile gegliedert.

Teil	Beschreibung	Hinweis
A1	Im Teil A1 erfahren Sie alles, um sich einen Überblick zu verschaffen. Sie lernen die verwendeten Komponenten (Standard Hard- und Softwarekomponenten und die zusätzlich entwickelte Software) kennen. Die dargestellten Funktionseckdaten zeigen die Leistungsfähigkeit der vorliegenden Applikation.	
A2	Im Teil A2 wird auf die detaillierten Funktionsabläufe der beteiligten Hard- und Softwarekomponenten eingegangen. Sie benötigen diesen Teil nur, wenn Sie den Detailablauf und das Zusammenspiel der Lösungskomponenten kennen lernen wollen.	Sie können diesen Teil überspringen, wenn Sie die Applikation zunächst einmal anhand der Step-by-Step-Anweisungen testen wollen
B	Der Teil B führt Sie Schritt für Schritt durch den Aufbau und die Inbetriebnahme der Applikation.	
C	Der Teil C ist dann interessant, wenn Sie auf Basis der vorliegenden Software eine Erweiterung oder Anpassung an Ihre Anlage vornehmen möchten.	
D	Im Teil D „Anhang“ finden Sie weiter führende Informationen, wie z. B. Literaturangaben.	

## Lerninhalte dieser Applikation

Nach dem Studium dieser Applikation haben Sie folgende Punkte gelernt:

- Grundlagen PC basierter Steuerungstechnik
- Leistungsmerkmale von SIMATIC WinAC RTX, der PC basierten Steuerung von Siemens
- Echtzeitfähigkeit mit Windows Betriebssystemen
- Grundlagen zum Windows Betriebssystem und seinen Mechanismen für Datenaustausch und Synchronisation
- Grundlagen des WinAC **Open Development Kit (ODK)**, einer Programmierschnittstelle zur WinAC RTX

## Referenz zum Automation and Drives Service & Support

Dieser Beitrag stammt aus dem Internet Applikationsportal des Automation and Drives Service & Support. Durch den folgenden Link gelangen Sie direkt zur Downloadseite dieses Dokuments.

<http://support.automation.siemens.com/WW/view/de/21572937>

## Inhaltsverzeichnis

<b>Inhaltsverzeichnis.....</b>	<b>7</b>
<b>Teil A1: Applikationsbeschreibung.....</b>	<b>9</b>
<b>1 Automatisierungsaufgabe .....</b>	<b>10</b>
<b>2 Die Automatisierungslösung.....</b>	<b>11</b>
2.1 Übersicht zur Gesamtlösung .....	11
2.2 Beschreibung der Funktionalitäten der Applikation .....	13
2.3 Alternative Anwendungen.....	16
2.4 Erforderliche Komponenten .....	17
<b>3 Leistungseckdaten .....</b>	<b>20</b>
<b>Teil A2: Funktionsmechanismen.....</b>	<b>21</b>
<b>4 Funktionsmechanismen.....</b>	<b>22</b>
4.1 Grundlagen: PC basierte Steuerung mit SIMATIC .....	22
4.1.1 SIMATIC WinAC Grundlagen .....	23
4.1.2 WinAC RTX Grundlagen.....	23
4.1.3 WinAC ODK Grundlagen .....	26
4.1.4 Funktionsweise der Custom Code Extension (CCX) .....	28
4.2 Grundlagen: Software-Applikationen unter Windows .....	33
4.2.1 Windows Mechanismen zur Synchronisation .....	34
4.2.2 Windows Mechanismen zum Datenaustausch .....	36
<b>Teil B: Installation der Beispielapplikation .....</b>	<b>38</b>
<b>5 Installation der Hard- und Software .....</b>	<b>39</b>
5.1 Vorbereitende Installation .....	39
5.2 Hardwareinstallation .....	40
5.3 Installation von WinAC RTX .....	41
5.4 Installation von SPECTATION .....	41
5.5 Installation der Applikationssoftware-Komponenten.....	42
5.6 Konfiguration des PC.....	44
5.7 Konfiguration und Programmierung des Vision Sensors .....	45
5.8 Konfiguration und Programmierung von WinAC RTX.....	48
<b>6 Bedienung der Applikation .....</b>	<b>51</b>
<b>Teil C: Programmbeschreibung .....</b>	<b>55</b>
<b>7 Erläuterung zum Code der einzelnen Elemente .....</b>	<b>59</b>
7.1 Initialisierung des WinAC ODK CCX .....	59
7.2 Beispiel für einen synchronen Aufruf des WinAC ODK CCX.....	61
7.3 Beispiel für einen asynchronen Aufruf des WinAC ODK CCX.....	66
7.4 Beispiel für eine ODK Funktion, die im Hintergrund läuft .....	72

<b>8</b>	<b>Anpassungen der Programme .....</b>	<b>78</b>
8.1	Anpassungen im S7-Programm.....	78
8.2	Anpassung der C++ Applikationen .....	79
8.2.1	Anpassungen in der Windows Applikation.....	79
8.2.2	Anpassungen in der ODK Applikation .....	80
<b>Teil D: Anhang .....</b>	<b>.....</b>	<b>82</b>
<b>9</b>	<b>Literaturangaben .....</b>	<b>82</b>

SIMATIC Vision Sensoren und WinAC ODK

## Teil A1: Applikationsbeschreibung

### Übersicht

#### Inhalt Teil A1

Im Teil A1 erfahren Sie alles, um sich einen Überblick zu verschaffen. Sie lernen die verwendeten Komponenten (Standard Hard- und Softwarekomponenten und die zusätzlich entwickelte Software) kennen.

Die dargestellten Funktionseckdaten zeigen die Leistungsfähigkeit der vorliegenden Applikation.

#### Ziel Teil A1

Der Teil A1 dieses Dokuments soll dem Leser

- das Automatisierungsproblem klarmachen
- eine Lösungsmöglichkeit aufzeigen
- die Leistungsfähigkeit der Gesamapplikation aufzeigen.

#### Behandelte Themen

Kap.	Titel	Seite
1	Automatisierungsaufgabe	10
2	Die Automatisierungslösung	11
2.1	Übersicht zur Gesamtlösung	11
2.2	Beschreibung der Funktionalitäten der Applikation	13
2.3	Alternative Anwendungen	16
2.4	Erforderliche Komponenten	17
3	Leistungseckdaten	20

SIMATIC Vision Sensoren und WinAC ODK

## 1 Automatisierungsaufgabe

### Anforderung

Im Bereich der industriellen Automatisierung werden häufig Industrie PC's eingesetzt, die zur Visualisierung dienen. Parallel dazu existiert eine Steuerung, die die Anlage steuert. Häufig stellt sich das Problem, dass Hardware eingesetzt werden soll, die mit dem PC verbunden werden muss und mit der Steuerung Daten austauschen soll.

### Allgemeine Aufgabenstellung

Funktionalitäten aus bestehenden, speziellen Windows Steuerungsprogrammen (die als Quellcode vorliegen) sollen in ein SIMATIC Steuerungsprogramm integriert werden. Als Steuerung bietet sich in diesem Fall SIMATIC WinAC RTX an.

### Allgemeine Technologische Aufgabenstellung

Diese Applikation soll zeigen, wie SIMATIC WinAC RTX mit anderen, eigenständigen Windows Applikationen interagieren kann. Dazu sollen Windows Standard-Mechanismen genutzt werden.

Im Mittelpunkt dieser Applikation steht ganz allgemein die Anbindung einer Windows-Applikation an die Steuerung WinAC RTX mittels des **Open Development Kits WinAC ODK**.

### Konkrete Technologische Aufgabenstellung

Die Technologische Aufgabe besteht darin, Flaschen die auf einem Förderband laufen, auf Ihre Qualität hin zu überwachen. Dazu soll ein Vision Sensor VS 723 genutzt werden. Er soll die Prüfung selbsttätig ausführen und Gut- oder Schlecht-Information liefern.

Auf dem PC soll eine Windows Applikation (im folgenden Kamera Applikation genannt) laufen, die die Bilder des Vision Sensors visualisiert, ihn steuert und die Ergebnisse an WinAC RTX meldet.

### Anforderungen an die Lösung

- Der Vision Sensor soll von der Kamera-Applikation aus voll bedienbar sein.
- Alle Funktionen müssen auch vom S7-Programm in der WinAC RTX aus bedienbar sein.
- Zwei verschiedenen Flaschentypen müssen konfigurierbar sein.
- Das Resultat einer Prüfung muss an WinAC RTX als binärer Wert gemeldet werden.
- Bilder fehlerhafter Prüflinge sollen auf Festplatte gespeichert werden
- Standardisierte Kommunikationsmechanismen sollen genutzt werden.
- Prüfaufgaben sollen vom Vision Sensor eigenständig bearbeitet werden

SIMATIC Vision Sensoren und WinAC ODK

## 2 Die Automatisierungslösung

### Einleitung

In diesem Kapitel erfahren Sie konkret, wie die vorliegende Applikation die in Kapitel 1 genannte Automatisierungsaufgabe löst. Dabei wird aufgezeigt, was die Applikation leisten kann, welche Teilmodule sie enthält und wie diese funktionieren. Die Beschreibung des Funktionsmechanismus hält sich dabei bewusst auf einem globalen Niveau. Ein tiefer gehendes Verständnis wird Ihnen im Teil A2 dieser Dokumentation vermittelt, den Sie allerdings nur dann benötigen, wenn Sie Hintergrundwissen, den Detailablauf und das Zusammenspiel der Lösungskomponenten kennen lernen wollen.

### Inhalt dieses Kapitels

Kap.	Titel	Seite
2.1	Übersicht zur Gesamtlösung	11
2.2	Beschreibung der Funktionalitäten der Applikation	13
2.3	Alternativlösungen	16
2.4	Erforderliche Komponenten	17

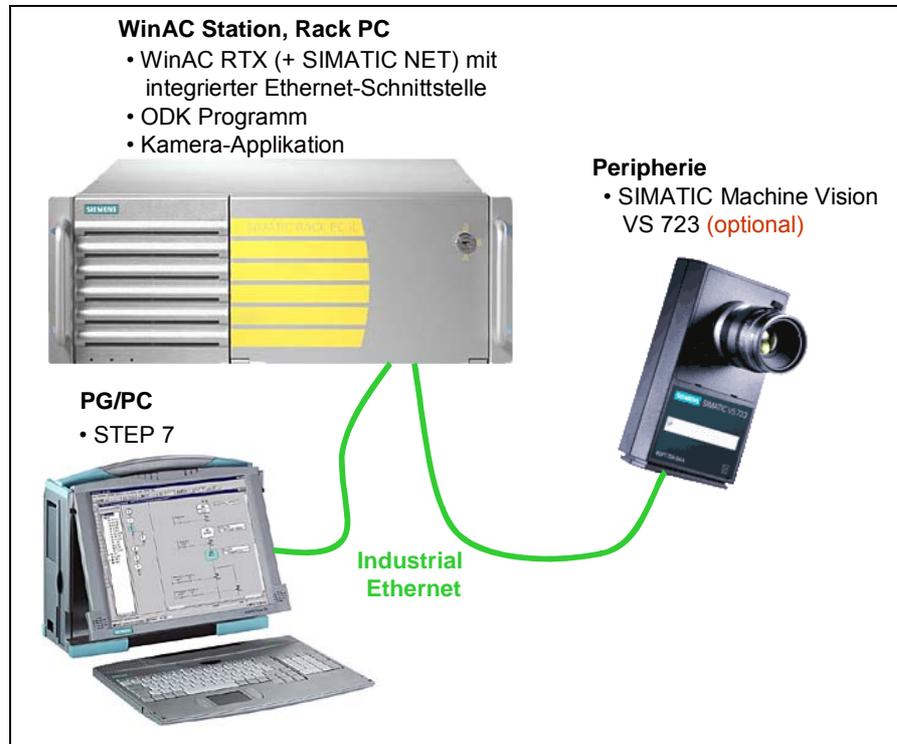
### 2.1 Übersicht zur Gesamtlösung

#### Darstellung der beteiligten Komponenten

Das folgende Übersichtsbild zeigt den HW-Aufbau der Beispielapplikation und der darauf befindlichen Standard- und Anwendersoftware-Komponenten. Bitte beachten Sie, dass sie die Applikation auch ohne Kamera in Betrieb nehmen können. Hierfür gibt es einen Simulationsmodus.

## SIMATIC Vision Sensoren und WinAC ODK

Abbildung 2-1



### Hinweis

Sie benötigen den Vision Sensor VS 723 nicht unbedingt, um dieses Beispiel in Betrieb zu nehmen. Dazu existiert ein Simulations-Modus in der Windows Applikation.

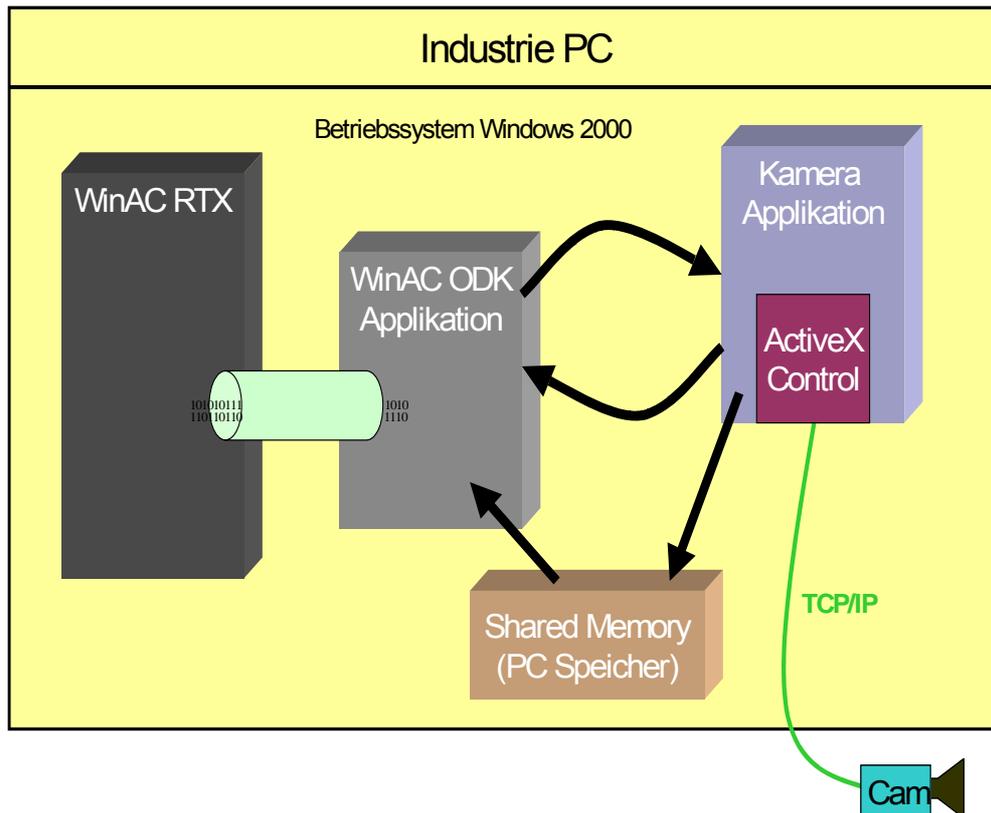
SIMATIC Vision Sensoren und WinAC ODK

## 2.2 Beschreibung der Funktionalitäten der Applikation

### Lösungselemente der Applikation

Die Applikation besteht aus mehreren Lösungselementen, die eng miteinander verknüpft sind:

Abbildung 2-2



Copyright © Siemens AG 2005 All rights reserved  
 21572937\_WinAC\_TO\_V11\_DOKU\_d.doc  
**Fehler! Unbekannter Name für Dokument-Element.**

### Beschreibung der Lösungselemente

Tabelle 2-1

Nr.	Element	Beschreibung
1	SIMATIC WinAC RTX	Eine Steuerung mit Echtzeitfähigkeit. Sie übernimmt die eigentliche Steuerung der Anlage
2	Kamera	Sie führt Qualitätsprüfungen an Glasflaschen durch und meldet das Testergebnis über eine TCP/IP Verbindung an den PC

SIMATIC Vision Sensoren und WinAC ODK

3	Kamera Applikation	Die Kamera Applikation <ul style="list-style-type: none"> <li>• steuert die Kamera (mit Hilfe einer Softwarekomponente, einem so genannten ActiveX Control ) und liest deren Testergebnisse</li> <li>• tauscht Daten mit WinAC RTX aus</li> <li>• speichert Bilder der Kamera auf der Festplatte</li> </ul>
4	WinAC ODK Applikation	Die ODK Applikation transferiert die Daten zwischen der Kamera Applikation und WinAC RTX und umgekehrt. Sie enthält in diesem Beispiel keine Steuerungsfunktionalität. Dies wäre jedoch möglich
5	Shared Memory	Der Shared Memory ist ein Teil des PC-Hauptspeichers (vergleichbar mit einer Datei), der als Medium für den Datenaustausch dient

### Ablauf ausgewählter Kernfunktionalitäten

Die folgende Tabellen beschreiben, welche Schritte in der Applikation ablaufen, damit bestimmte Funktionalitäten abgearbeitet werden:

#### Setzen des Status der Kamera

Tabelle 2-2

Schritt	Aktion	Erläuterung
1	WinAC RTX schickt ein Kommando an die ODK Applikation, die Daten über den zu setzenden Modus enthält	Das Kommando wird über einen SFB Aufruf in WinAC RTX abgesetzt, wobei die Daten in einem ANY Pointer übergeben werden
2	Die ODK Applikation erhält den Auftrag und speichert die Daten zwischen. Es wird ein parallel laufender Programmteil gestartet.	Parallel lauffähige Programmteile bezeichnet man in der Windows Terminologie als „Threads“ oder „Prozesse“.
3	Der parallel laufende Programmteil sendet eine Windows Nachricht mit Daten an die Kamera Applikation	Windows Nachrichten werden an einen spezifizierten Empfänger gesendet und vom Betriebssystem zugestellt
4	Die Kamera Applikation erhält die Windows Nachricht und speichert die Daten	Dazu muss in der Applikation eine Behandlungsroutine für Windows Nachrichten definiert sein
5	Über das ActiveX Control sendet die Kamera Applikation das Kommando an die Kamera	Das ActiveX Control ist eine Software Komponente, die speziell für die Kommunikation mit SIMATIC Vision Sensoren entwickelt wurde.

SIMATIC Vision Sensoren und WinAC ODK

Lesen der Anzahl geprüfter Flaschen aus der Kamera

Tabelle 2-3

Schritt	Aktion	Erläuterung
1	Die Kamera sendet nach jeder erfolgten Prüfung die Anzahl geprüfter Flaschen an die Kamera Applikation	Diese Kommunikation erfolgt wieder über das ActiveX Control
2	Die Kamera Applikation schreibt diesen Zählerstand in den Shared Memory	Shared Memory wird unter Windows wie eine Datei behandelt
3	Die ODK Applikation prüft zyklisch, ob sich der Stand des Flaschenzählers im Shared Memory erhöht hat. Der Wert wird ausgelesen	
4	Der ausgelesene Wert wird von der ODK Applikation direkt in eine Merkervariable in WinAC RTX geschrieben	Die ODK Applikation hat Zugriff auf fast alle Speicherbereiche in WinAC RTX.

**Vorteile des Lösungsbeispiels**

- Beliebige Hard- und Softwarekomponenten können in SIMATIC Steuerungsprogramme eingebunden werden
- Die Kamera Applikation kann sehr einfach durch beliebige andere Windows Applikationen ausgetauscht werden
- Über Shared Memory können große Datenmengen sehr performant ausgetauscht werden
- Nutzung aller Leistungsmerkmale, die das Betriebssystem Windows Ihnen bietet
- Die Lösung beinhaltet eine einfache Visualisierung
- Für die spätere Fehleranalyse werden Bilder fehlerhafter Teile auf der Festplatte gespeichert

**Vorteile PC basierter Automatisierung von Siemens**

- Kostenersparnis, da die Steuerung auf dem PC läuft und keine externe Steuerung benötigt wird
- Nutzung vorhandener, standardisierter Hardware und standardisierter Schnittstellen
- Hohe Flexibilität bei geringen Kosten
- Einfache Anbindung der Steuerung an Standard-Software (z.B. Microsoft Office) und eigene Applikationen (VB, C++, C#, ...)
- Nutzung der fortschreitenden Innovationen im PC Bereich
- Nutzung bekannter Engineering Tools (z.B. STEP 7)

## SIMATIC Vision Sensoren und WinAC ODK

- Optimale Integration von Steuerung, Visualisierung, Kommunikation und Technologie

### 2.3 Alternative Anwendungen

#### Alternative Softwarelösungen

Mit den dargestellten Mechanismen können beliebige Software Module in den Steuerungszyklus von WinAC RTX eingebunden werden, wie z.B.:

- Regler (z.B. PI Regler, in C++ implementiert)
- Anbindungen an Datenbanken
- Messwerterfassungs-Software
- Statistik Tools

#### Alternative Hardwarelösungen

Hardwarekomponenten von Siemens und von Fremdherstellern können mit WinAC RTX kommunizieren, wie z.B.:

- Barcodeleser
- Messwerterfassungskarten
- Modems
- Roboter

SIMATIC Vision Sensoren und WinAC ODK

## 2.4 Erforderliche Komponenten

Nachfolgende Tabellen enthalten die Hard- und Software-Komponenten für die jeweiligen Stationen. Die unten aufgeführten Komponenten können Sie direkt in der Siemens A&D Mall unter [www.ad.siemens.com/mall](http://www.ad.siemens.com/mall) bestellen.

### Hardware-Komponenten

Für diese Applikation benötigen Sie lediglich einen Industrie PC und optional eine Kamera aus der Machine Vision Serie.

Tabelle 2-4

Komponente	Anz.	MLFB/Bestellnummer	Hinweis
Industrie PC SIMATIC Rack PC IL 40 S	1	6AG4011-0CA21-0JX0	Konfigurator: siehe FAQ ID <a href="#">17128155</a>
SIMATIC VS723 Visionsensor	1	6GF1723-0AA	Optional; 640 x 480 Pixel s/w
Stromkabel für Vision Sensor VS72x	1	6GF9002-2AD	Nur falls VS723 genutzt wird
Ethernetkabel gekreuzt	1	je nach Hersteller	Nur falls VS723 genutzt wird
Objektiv XENOPLAN 1,4/23 MM für VS72x	1	6GF9001-1AL	Nur falls VS723 genutzt wird
Stromversorgung PS307 2A	1	307-1BA00-0AA0	Nur falls VS723 genutzt wird; Es werden nur 250 mA benötigt
Programmiergerät Power PG	1	6ES7751-.....-....	Konfigurator: siehe FAQ ID <a href="#">17128155</a> ; CP 5611 integriert

### Software-Komponenten

Tabelle 2-5

Komponente	Anz.	MLFB/Bestellnummer	Hinweis
SIMATIC WinAC RTX V4.1	1	6ES7671-0RC04-0YA0	
SIMATIC NET IE SOFTNET-S7 V6.2	1	6GK1716-0HB62-3AA4	Ohne SIMATIC NET kann WinAC RTX nicht installiert werden
STEP 7 V5.3	1	6ES7810-4CC07-0YA5	Enthält NCM S7 Ethernet
SPECTION 2.6 Projektiersoftware	1	6GF8007-3AA26	Nur falls VS723 genutzt wird

## SIMATIC Vision Sensoren und WinAC ODK

### Applikationssoftware-Komponenten

Die vorliegende Beispielapplikation besteht aus folgenden Komponenten.

Weitere Informationen zur Inbetriebnahme von Hard- und Software entnehmen Sie dem Kapitel „Installation der Hard- und Software“.

Tabelle 2-6

Komponente	Hinweis
21572937_WinAC_TO_CODE_v11.zip Darin sind enthalten :	Diese Datei enthält ein Setup. Dieses installiert die nachfolgend aufgelisteten Komponenten für Sie.
ODK_Application	Der Quellcode zur Erstellung des ODK Programms, erstellt mit dem ODK Wizard und implementiert in C++; Die fertig generierte Bibliothek liegt ebenfalls bei.
STEP7_Prj	S7 Projekt mit Konfiguration und Code, der direkt in WinAC RTX geladen werden kann.
Cam_Config	Die Konfiguration des Vision Sensors VS723; Kann mit SIMATIC Spectation direkt in den Sensor geladen werden
Cam_Application	Der Quellcode und die ausführbare Datei der Kameraapplikation
VS72x_ActiveX	Ein ActiveX Control, das zur Ansteuerung der Kamera dient. Es kann sehr einfach in Visual Basic oder C++ Projekte integriert werden.
21572937_WinAC_TO_v11_DOKU_d.pdf	Dieses Dokument

### Aufwand Programmierung/Projektierung

- Für die Implementierung einer vergleichbaren Applikation wird bei mittleren SIMATIC Vorkenntnissen ein Aufwand von einem Manntag für die Programmierung des S7-Programms angesetzt.
- Für die Programmierung der Kamera Applikation sind, bei guten Windows Programmierkenntnissen, ca.12 Manntage zu kalkulieren
- Die Erstellung des ODK Quellcodes schlägt bei guten Windows Programmierkenntnissen mit ca. 5 Manntagen zu Buche.
- Die Konfiguration des SIMATIC Vision Sensors VS 723 benötigt bei guten Grundkenntnissen einen Manntag
- Für die Integration aller Komponenten sind ca. 4 Tage anzusetzen.

## SIMATIC Vision Sensoren und WinAC ODK

### Hinweis

Die oben geschätzten Aufwände sind grobe Richtwerte und können je nach Vorkenntnissen deutlich größer oder kleiner ausfallen. Bei der Durchführung solcher Projekte müssen S7-Programmierer eng mit den Windows-Programmierern zusammenarbeiten, da beide stark miteinander verzahnt sind.

SIMATIC Vision Sensoren und WinAC ODK

## 3 Leistungseckdaten

### Eckdaten von Systemsoftware und Projektierung

Nachfolgende Tabelle informiert über die Eckdaten der Systemsoftware und Projektierung. Sie erhalten damit einen Überblick über die Leistungsfähigkeit dieser Applikation und ihrer Komponenten.

Tabelle 3-1

Merkmale	Wert	Hinweis
Max. Datengröße bei Übertragung von WinAC → ODK Applikation	64 kB	Max. Größe eines DB in WinAC RTX
Max. Datengröße bei Übertragung von der ODK Applikation zur Windows Applikation und umgekehrt	2 GB	Bei Einsatz von Shared Memory; Max. Größe einer Datei unter Windows 32
WinAC RTX Reaktionszeit auf Interrupts von der ODK Applikation	Im $\mu$ s Bereich	Abhängig von der Prozessorgeschwindigkeit und der ablaufenden Software
Reaktionszeit von Windows Programmen auf Nachrichten von anderen Programmen	10 – 100 $\mu$ s	Gemessen mit SendMessage <sup>1</sup> Funktion; Abhängig von der Prozessorgeschwindigkeit und der ablaufenden Software

<sup>1</sup> SendMessage ist eine Windows Funktion, die Nachrichten an andere Windows Programme sendet und den Sender so lange blockiert, bis der Empfänger die Nachricht empfangen hat. PostMessage hat die gleiche Funktion, ohne jedoch den Sender zu blockieren.

Von Verwendung der PostMessage Funktion wird jedoch abgeraten, da die Laufzeit einer Nachricht sehr stark abhängig ist von der Anzahl der momentan geöffneten Windows Applikationen und der Systemauslastung.

SIMATIC Vision Sensoren und WinAC ODK

## Teil A2: Funktionsmechanismen

### Ziel Teil A2

Der Teil A2 dieses Dokuments soll dem Leser

- alle vorkommenden Funktionselemente verständlich machen
- die wieder verwendbaren Komponenten aufzeigen
- entsprechendes Hintergrundwissen vermitteln

### Inhalt Teil A2

Kap.	Titel	Seite
4	Funktionsmechanismen	22
4.1	Grundlagen: PC basierte Steuerung mit SIMATIC	22
4.2	Grundlagen: Software-Applikationen unter Windows	33

## 4 Funktionsmechanismen

### Was steht hier ?

Hier finden Sie Informationen

- zu den Grundlagen PC basierter Steuerungstechnik
- zur Steuerung WinAC RTX
- zum WinAC **O**pen **D**evelopment **K**it (WinAC ODK)
- zu relevanten Windows-Mechanismen (Synchronisation und Datenaustausch)
- zum Zusammenspiel der einzelnen Komponenten

### Was können Sie damit anfangen ?

Grundsätzlich ist die vorliegende Applikation sofort einsatzfähig. Mit der Installationsanleitung können Sie die Applikation in Betrieb nehmen, ohne dieses Kapitel durchgearbeitet zu haben. Wollen Sie jedoch verstehen, was hinter den Kulissen geschieht, dann ist die Lektüre dieses Kapitels empfehlenswert. Auch wenn Sie bestimmte Module der Applikation für Ihre Anforderungen variieren möchten, benötigen Sie tiefer gehende Informationen.

### Aufbau dieses Kapitels

Nachfolgend finden Sie zur Orientierung den Aufbau dieses Kapitels.

Tabelle 4-1

Kap.	Titel	Seite
4	Funktionsmechanismen	22
4.1	Grundlagen: PC basierte Steuerung mit SIMATIC	22
4.2	Grundlagen: Software-Applikationen unter Windows	33

## 4.1 Grundlagen: PC basierte Steuerung mit SIMATIC

### Überblick

In diesem Kapitel werden die Mechanismen erklärt, die der Beispielapplikation zu Grunde liegen. Diese Mechanismen werden in verallgemeinerter Form dargestellt, damit Sie leicht auf ähnliche Applikationen anwendbar sind.

Sie lernen hier kennen, was das WinAC ODK ist, wie die Steuerung WinAC RTX Daten mit dem ODK Programm austauscht und wie dieses mit einer Windows Applikation über Windows Standardmechanismen Daten austauscht.

## SIMATIC Vision Sensoren und WinAC ODK

### 4.1.1 SIMATIC WinAC Grundlagen

#### Produkte aus der SIMATIC WinAC Reihe

Die SIMATIC WinAC Reihe setzt sich aus folgenden Produkten zusammen:

- WinAC RTX
- WinAC Slot
- WinAC PN Option
- WinAC ODK

All diese Produkte sind selbstverständlich voll in das Totally Integrated Automation Konzept (TIA) integriert und somit mit STEP 7 programmierbar.

Da es sich um reine Software Produkte handelt, werden WinAC Basis, WinAC RTX und WinAC PN unter dem Begriff Software WinAC zusammengefasst.

#### Beschreibung WinAC RTX

Zusätzlich zu den Vorteilen der WinAC Basis bietet die Software Steuerung WinAC RTX deterministische Eigenschaften. Dies wird gewährleistet durch die Windows Realtime Extension RTX.

Das vorliegende Applikationsbeispiel basiert auf WinAC RTX. In den folgenden Kapiteln erfahren Sie mehr zu diesem Produkt.

#### WinAC Slot

WinAC Slot in den beiden Ausprägungen 412 und 416 ist eine PC basierte Steuerung in Form einer PCI-Einsteckkarte. Diese Karte hat bereits serienmäßig eine DP und eine MPI Schnittstelle integriert. Die Verbindung mit dem optionalen Power Supply Extension Board verschafft der Steuerung zusätzliche Sicherheiten gegen den Ausfall der PC Hard- und Software.

#### WinAC ODK

Das WinAC Open Development Kit ist ein Software Produkt, das Schnittstellen zwischen Windows Applikationen und WinAC Steuerungen anbietet. Somit lassen sich auf einfachste Art und Weise Windows Applikationen erstellen, die Daten in beiden Richtungen mit WinAC Steuerungen austauschen.

Das WinAC ODK steht im Mittelpunkt dieser Applikation.

### 4.1.2 WinAC RTX Grundlagen

#### Was ist WinAC RTX?

Das Windows Automation Center mit Echtzeiterweiterung (WinAC RTX) bietet die Funktionalität einer speicherprogrammierbaren Steuerung (SPS) in einer PC-basierten Umgebung, z.B. unter Windows XP. Funktional ist WinAC RTX mit einer S7-400 vergleichbar.

## SIMATIC Vision Sensoren und WinAC ODK

### Die Steuerung WinAC RTX

WinAC RTX führt das S7-Programm auf dem PC aus und steuert die angeschlossene dezentrale Peripherie. WinAC RTX ist vollständig code-kompatibel mit anderen SIMATIC S7 Steuerungen. Damit sind vorhandene S7-Programme einer S7-300 oder S7-400 Steuerung auch auf WinAC RTX ablauffähig.

### Die Echtzeiterweiterung RTX

WinAC RTX nutzt die Echtzeiterweiterungen (RTX) für Windows der Firma Ardence (ehemals VenturCom). Diese Erweiterungen ermöglichen folgende Funktionen:

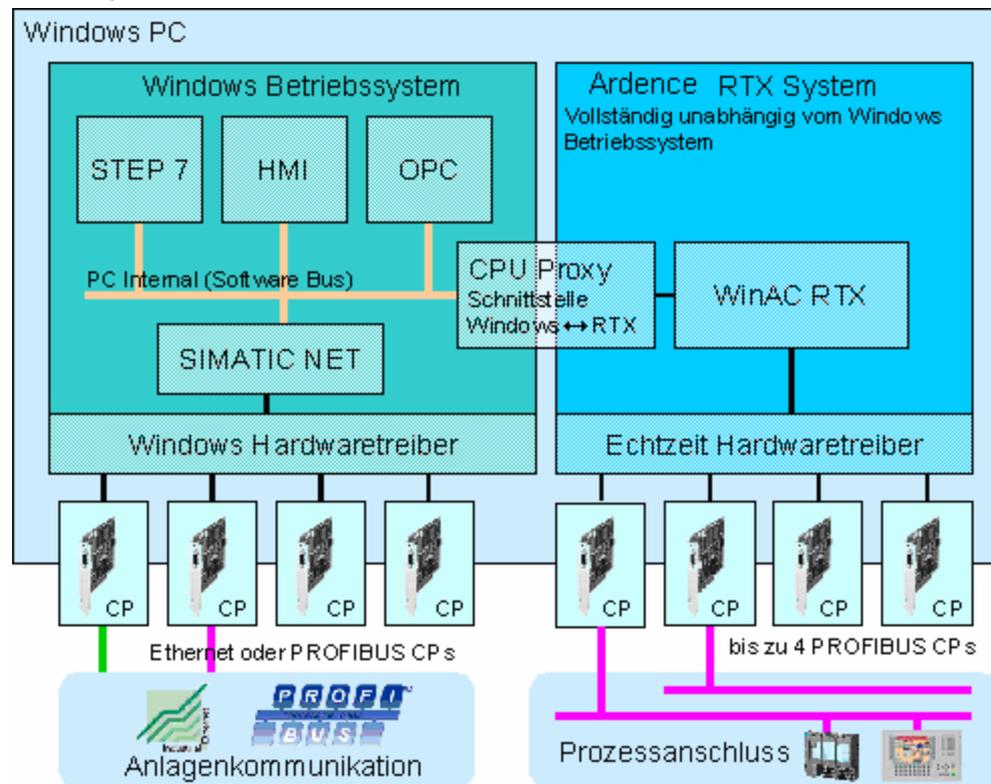
- Deterministischer Betrieb mit vorhersagbaren Reaktionszeiten.
- Isochrone Modus (Äquidistanz, konstante Buszykluszeit)
- Bei einem Windows-Systemausfall („Bluescreen“) läuft die Prozesssteuerung von WinAC RTX unabhängig vom Betriebssystem weiter und kann damit ordnungsgemäß heruntergefahren werden (über Alarm-OBs).

### Softwarearchitektur

Das folgende Bild zeigt, wie ein WinAC RTX System aufgebaut ist. Hierbei wird die Trennung zwischen Windows Betriebssystem und dem Echtzeitsystem RTX von Ardence, auf welchem WinAC RTX basiert, deutlich.

## SIMATIC Vision Sensoren und WinAC ODK

Abbildung 4-1



### Prozessanschluss für WinAC RTX

Der Prozessanschluss von WinAC RTX erfolgt über PROFIBUS DP. Dazu werden PROFIBUS Kommunikationsprozessoren (z.B. PCI-Steckkarte CP 5613, in Zukunft auch der CP 5611) in den PC gesteckt und als Schnittstellenmodule (IF-Module) in der HW-Konfig projiziert (Analog zur S7-400).

Die als IF-Module projizierten CPs werden über Echtzeit Hardwaretreiber angesprochen. Dies ermöglicht den Isochron-Modus (Äquidistanz der Buszyklen) für die angeschlossenen PROFIBUS-DP Mastersysteme.

Für WinAC RTX lassen sich bis zu vier CPs als IF-Module projizieren.

### Anlagenkommunikation mit WinAC RTX

Die Anlagenkommunikation mit WinAC RTX erfolgt über CPs, die in den PC gesteckt werden. Diese CPs werden über die HW-Konfig der SIMATIC PC-Station projiziert.

Für die Anlagenkommunikation von SIMATIC S7 Stationen untereinander wird die S7-Kommunikation eingesetzt. Um die S7-Kommunikation zu nutzen, muss auf dem PC, auf welchem WinAC RTX läuft, auch die SIMATIC NET PC Software installiert und lizenziert werden (Die aktuelle SIMATIC NET CD wird zusammen mit WinAC RTX geliefert).

SIMATIC Vision Sensoren und WinAC ODK

## PG-Routing mit WinAC RTX

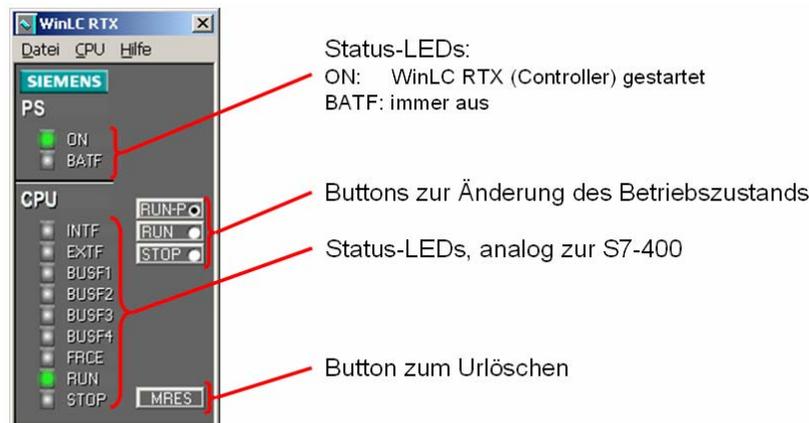
WinAC RTX unterstützt das PG-Routing. Damit kann STEP 7 über die CPs einer WinAC PC-Station S7-Stationen in anderen Subnetzen erreichen.

## Das WinAC RTX Bedien-Panel

Das Bedien-Panel von WinAC RTX ist dem Aussehen einer herkömmlichen S7-400-Steuerung angelehnt. Es besitzt die gleichen Status-LEDs sowie Buttons zur Betriebszustandsänderung.

Das Bedien-Panel kann während des Betriebs des WinAC RTX geschlossen werden, dabei läuft die Steuerung weiter. Zum erneuten Öffnen des Bedien-Panels reicht ein Doppelklick auf das Symbol  in der Taskleiste.

Abbildung 4-2



### 4.1.3 WinAC ODK Grundlagen

#### Funktionalität des WinAC ODK

Das WinAC **Open Development Kit** ist ein Software Werkzeug, mit dessen Hilfe eine Windows Applikation (im folgenden ODK Applikation genannt) erstellt werden kann, deren Funktionen direkt aus WinAC RTX heraus aufgerufen werden können. Damit kann der volle Leistungsumfang, den das Betriebssystem Windows und darauf aussetzende Applikationen bieten, in ein WinAC Steuerungsprogramm eingebunden werden.

Das ODK generiert dazu ein Grundgerüst in der Programmiersprache C++, in das hinein der Anwender seine gewünschten Funktionalitäten programmieren kann. Dieses Grundgerüst vereinfacht den Einstieg in die Programmierung stark. Es muss jedoch nicht zwingend genutzt werden.

#### Was ist WinAC ODK V 4.1?

Das WinAC ODK V 4.1 ist die Zusammenführung der beiden Vorgängerprodukte WinAC Slot T-Kit und WinAC ODK V 4.1. Sie benötigen

## SIMATIC Vision Sensoren und WinAC ODK

nun nur noch ein Produkt zur Programmierung von Windows Applikationen für WinAC Slot und Software WinAC (Basis und RTX).

### Komponenten des WinAC ODK V 4.1

In WinAC ODK V 4.1 sind drei Komponenten enthalten, die im folgenden beschrieben werden:

- Custom Code Extension Interface (CCX)
- Shared Memory Extension Interface (SMX)
- Controller Management Interface (CMI)

#### Custom Code Extension Interface (CCX)

Das CCX ist der Nachfolger des WinAC ODK V 4.1. Es ermöglicht einen Zugriff auf Daten in WinAC RTX sowohl vom S7-Programm zur ODK Applikation als auch in die andere Richtung. Der S7-Programmierer nutzt für diese Zugriffe SFB Aufrufe.

Das CCX verwenden Sie, wenn Sie C++ Funktionen direkt und als Teil des Steuerungszykluses von WinAC RTX aufrufen möchten. Es liegt dieser Applikation zu Grunde und wird ausführlich beschrieben. Es kann nur zusammen mit WinAC RTX genutzt werden.

#### Shared Memory Extension Interface (SMX)

Das SMX löst das WinAC T-Kit ab. Auch hier werden Schnittstellen zur Verfügung gestellt, die den Datenaustausch zwischen WinAC und der ODK Applikation ermöglichen. Für den S7-Programmierer stellt sich dieser Datenaustausch als Zugriff auf Peripheriedaten dar.

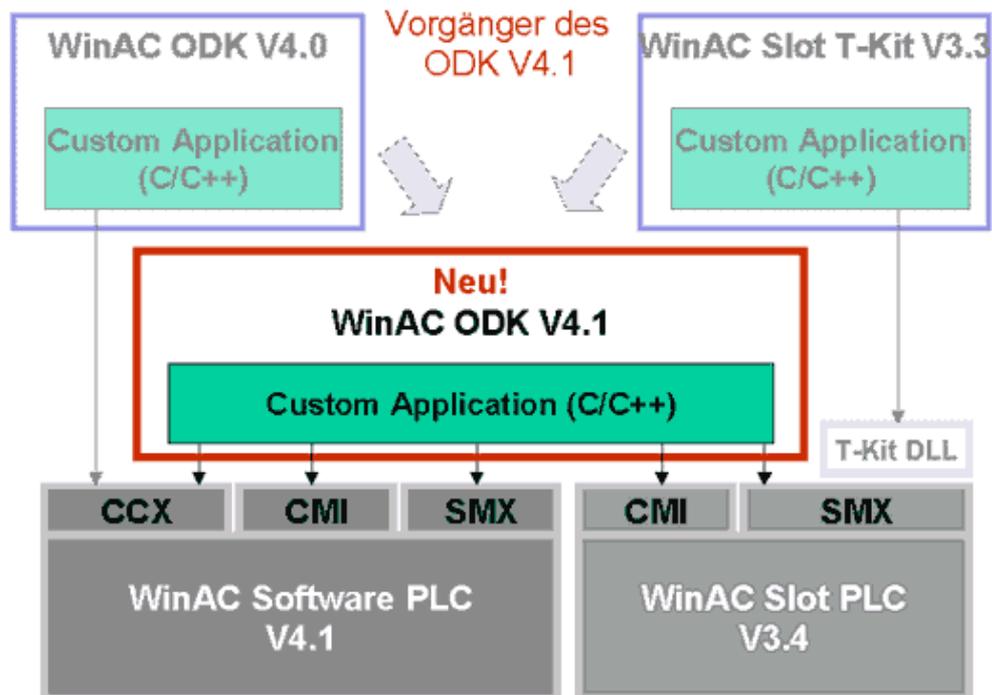
Das SMX nutzen Sie vor allem für PC-Applikationen, die große Datenmengen performant mit WinAC austauschen. Es ist sowohl bei WinAC RTX als auch bei WinAC Slot anwendbar.

#### Bild: Komponenten des WinAC ODK V 4.1

Folgendes Bild zeigt, welche Komponente mit welcher WinAC Variante nutzbar ist.

## SIMATIC Vision Sensoren und WinAC ODK

Abbildung 4-3



### Lieferumfang des WinAC ODK V4.1

Das WinAC ODK besteht aus folgenden Teilen:

- Ein Assistent, der auf Basis der Eingaben des Anwenders automatisch ein Grundgerüst in der Programmiersprache C++ generiert. Dies erfolgt wahlweise für das CCX oder SMX
- Eine S7-Bibliothek mit zwei S7-Bausteinen (SFB 65001 und 65002), die die Initialisierung und den Aufruf der ODK Applikationen ermöglichen. Sie sind direkt in WinAC RTX ablauffähig und werden nur beim CCX benötigt.
- Header Dateien, die für die Programmierung in C++ benötigt werden
- Bibliotheken und Header Dateien für das CMI
- Eine ausführliche Dokumentation in elektronischer Form und Beispielprogramme.

#### 4.1.4 Funktionsweise der Custom Code Extension (CCX)

##### Grafische Darstellung der Funktionsweise des CCX

Das Bild zeigt, wie die ODK Applikation (die verschiedene ODK Funktionen beinhaltet) mit der WinAC und mit anderen Windows-Applikationen

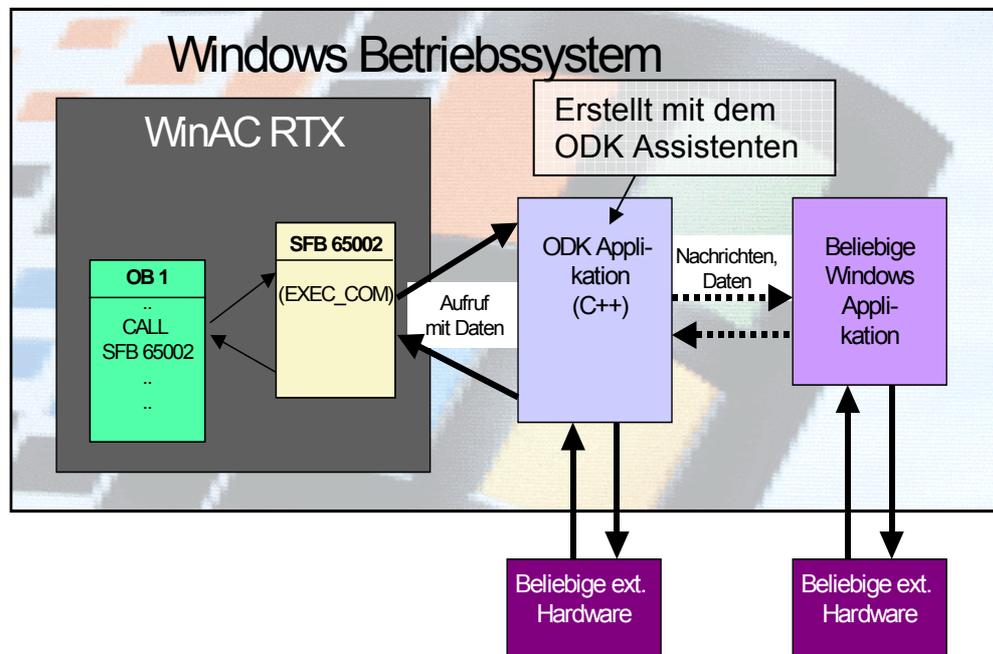
## SIMATIC Vision Sensoren und WinAC ODK

zusammenhängt. Es ist auch ersichtlich, wie externe Hardware mit Hilfe des ODK genutzt werden kann.

Die violett dargestellte Windows Applikation ist optional. Der Entwickler hat die Möglichkeit, seine Funktionalität direkt in der ODK Applikation zu implementieren, was zum Beispiel für Regler oder Datenbankanwendungen durchaus sinnvoll ist. Dann wird die Windows Applikation nicht benötigt.

Häufig gilt es jedoch, bestehende Windows Applikationen in die Steuerungsaufgabe mit einzubeziehen. Auch für Applikationen, die sichtbare Benutzeroberflächen (Fenster) besitzen, ist die Lösung mit einer eigenen Windows Applikation aus programmieretechnischen Gründen empfehlenswert.

Abbildung 4-4



Copyright © Siemens AG 2005 All rights reserved  
 21572937\_WinAC\_TO\_V11\_DOKU\_d.doc  
**Fehler! Unbekannter Name für Dokument-  
 Eigenschaften.**

### Die ODK Applikation aus Sicht des WinAC Projektors

Die Aufrufe von ODK Funktionen sind mit Aufrufen von SFBs und OBs in einer klassischen Steuerung vergleichbar. Dabei werden drei Aufrufarten unterschieden:

- Synchroner Aufruf einer ODK Funktion: Die Funktion ist bereits nach einem Aufruf vollständig abgearbeitet
- Asynchroner Aufruf einer ODK Funktion: Die Funktion benötigt mehrere Aufrufe, bis sie vollständig abgearbeitet ist
- Ausführung von kontinuierlichen ODK Funktionen im Hintergrund (Monitoring): Die Funktion läuft ständig im Hintergrund und führt Überwachungsfunktionen aus.

## SIMATIC Vision Sensoren und WinAC ODK

Welcher der drei Funktionstypen vorliegt, entscheidet allein die Art der Implementierung in der ODK Applikation.

Einen Sonderfall des Aufrufs stellt die Initialisierung der ODK Applikation dar. Diese erfolgt einmalig vor der ersten Nutzung von ODK Funktionen.

Initialisierung und die drei Aufrufarten werden nachfolgend beschrieben.

### Initialisierung der ODK Applikation

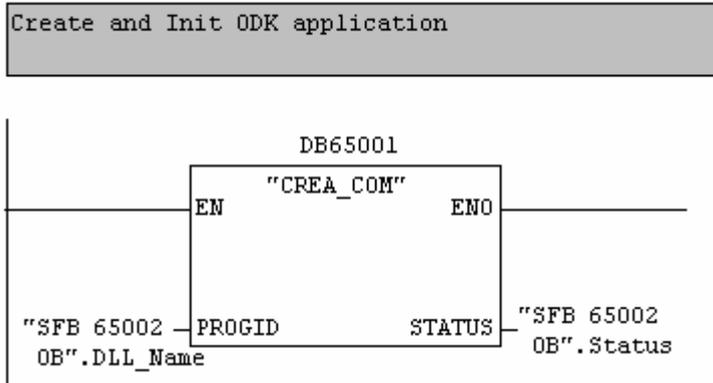
Durch einen Aufruf des speziell dafür vorgesehenen SFB 65001 wird die ODK Applikation erzeugt und initialisiert. Nach erfolgreicher Initialisierung gibt dieser SFB eine eindeutige Kennung (Status) zurück, die gespeichert werden muss für alle folgenden Aufrufe von ODK Funktionen.

### Initialisierung der ODK Applikation: Bild

Folgendes Bild zeigt den Aufruf zur Initialisierung einer ODK Applikation mittels SFB 65001.

Abbildung 4-5

#### Network 1: ODK create



### Allgemeines zum Aufruf von ODK Funktionen

Aus Sicht der WinAC stellt sich der Aufruf einer Funktion in der ODK Applikation genau so dar wie der Aufruf eines SFB:

1. Der SFB 65002 wird zum Beispiel im OB1 mit Eingangsparametern versorgt und aufgerufen
2. Der SFB 65002 initiiert den Aufruf einer Funktion in der ODK Applikation (kurz: ODK Funktion)
3. Die ODK Funktion läuft. Während dieser Zeit ist die Ausführung des OB1 unterbrochen
4. Nach Beendigung der ODK Funktion werden die entsprechende Ausgangsparameter gesetzt und die Kontrolle geht zurück an den SFB 65002
5. Der SFB 65002 empfängt die Rückgabeparameter von der ODK Funktion und gibt sie weiter an den OB 1. Der OB 1 wird fortgesetzt

## SIMATIC Vision Sensoren und WinAC ODK

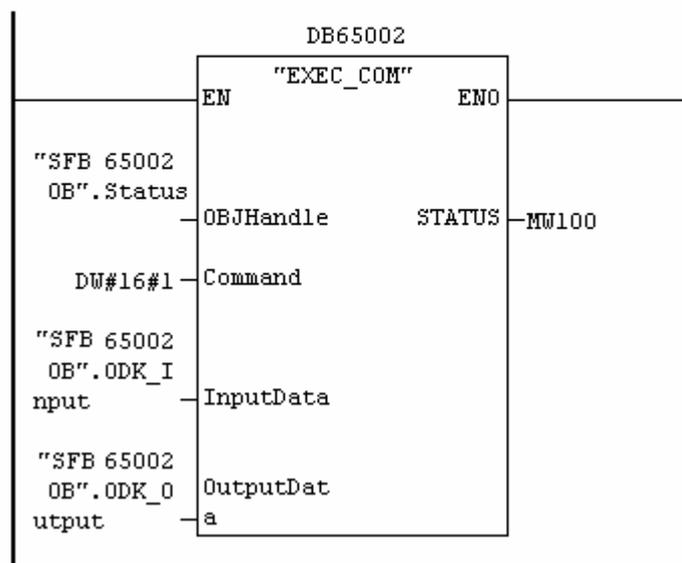
### Aufruf von ODK Funktionen: Bild

Das Bild zeigt den Aufruf des SFB 65002 mit der Funktion 1 (Command = DW#16#1) in der ODK Applikation. Es können beliebig viele Funktionen implementiert werden.

Abbildung 4-6

#### Network 2: Get status

Receives the status from the ODK application



Copyright © Siemens AG 2005 All rights reserved  
21572937\_WinAC\_TO\_V11\_DOKU\_d.doc

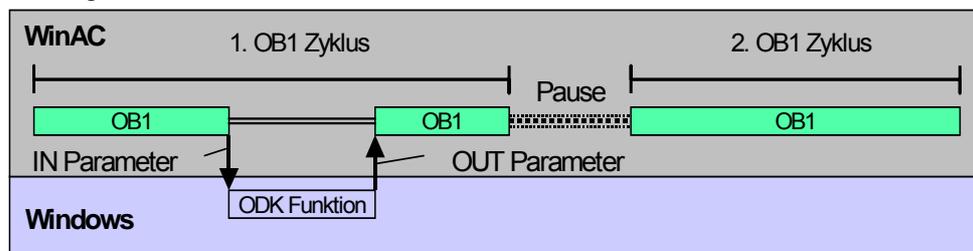
Fehler! Unbekannter Name für Dokument-Einzelblatt

### Der synchrone ODK Aufruf

Der folgende Zeitstrahl zeigt den synchronen Aufruf einer ODK Funktion. Der OB1 ruft den SFB 65002 (mit IN Parametern) auf, der wiederum die ODK Funktion ausführt. Solange diese aktiv ist, wird die Abarbeitung des OB1 unterbrochen. Am Ende gibt die ODK Funktion ihr Ergebnis als OUT Parameter an den OB1 zurück.

Der synchrone Aufruf eignet sich nur für zeitlich kurze ODK Funktionen, da es sonst zu einer Zykluszeitüberschreitung im OB1 kommen kann.

Abbildung 4-7



## SIMATIC Vision Sensoren und WinAC ODK

### Der asynchrone ODK Aufruf

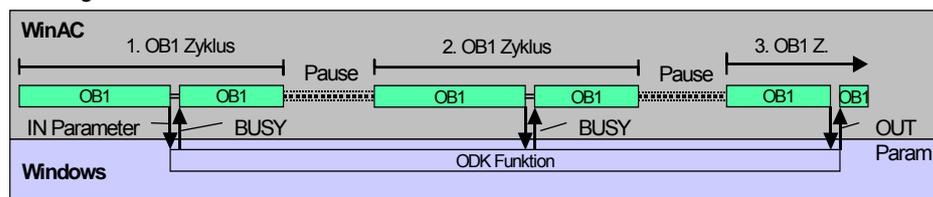
Der folgende Zeitstrahl zeigt den asynchronen Aufruf einer ODK Funktion. Der OB1 ruft den SFB 65002 (mit IN Parametern) auf, der wiederum die ODK Funktion einleitet. Der SFB Aufruf kehrt sofort zurück. Die Abarbeitung des OB1 wird nur für sehr kurze Zeit unterbrochen. Die ODK Funktion läuft „quasi parallel“ zum OB1 im Hintergrund.

Der OB1 fragt zyklisch den Zustand der ODK Funktion ab. Solange diese BUSY meldet, liegt noch kein Ergebnis vor.

Ist die ODK Funktion fertig, so meldet sie beim darauf folgenden Aufruf aus dem OB1 ihr Ergebnis als OUT Parameter an den OB1 zurück.

Der asynchrone Aufruf eignet sich auch für zeitlich lange ODK Funktionen, da es nur zu einer unwesentlichen Verlängerung der OB1 Zyklen kommt.

Abbildung 4-8

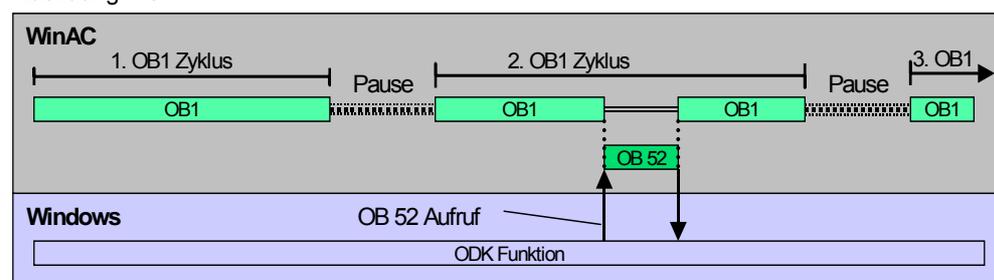


### Ausführung von ODK Funktionen im Hintergrund

Der folgende Zeitstrahl zeigt die kontinuierliche Funktion des ODK. Diese Funktion läuft ständig im Hintergrund der ODK Applikation. Sie läuft also „quasi parallel“ zu allen Programmteilen von WinAC RTX. Gestartet wird die kontinuierliche Funktion bei der Initialisierung des ODK Programms, gestoppt wird sie bei einem Wechsel der Steuerung in den STOP Modus.

Die kontinuierliche Funktion wird typischerweise für Überwachungsfunktionen im Hintergrund verwendet. Sie ähnelt dem OB 90 in einer klassischen Steuerung. Eine typische Anwendung wäre zum Beispiel das Abfragen der seriellen Schnittstelle im PC um festzustellen, ob neue Daten vorliegen.

Abbildung 4-9



## SIMATIC Vision Sensoren und WinAC ODK

### Datenaustausch mit ODK in kontinuierlichen Funktionen

Damit die kontinuierliche Funktion Daten an WinAC RTX liefern kann, kann sie einen asynchronen OB in WinAC RTX (z.B. den speziell dafür vorgesehenen OB52; siehe obiges Bild) anfordern und diesem Daten übergeben. Die Reaktion von WinAC RTX auf einen solchen Aufruf ist frei programmierbar.

Als weitere Alternative kann die ODK Funktion Variablen in WinAC RTX direkt setzen, zum Beispiel Merkerworte. Auch hier muss die Reaktion darauf im S7 Programm programmiert werden.

Um Daten von WinAC RTX an eine kontinuierliche Funktion zu übergeben, kann ein synchroner Aufruf, wie weiter oben beschrieben, genutzt werden.

## 4.2 Grundlagen: Software-Applikationen unter Windows

### Stand-Alone Windows Applikationen

Unter Stand-Alone Windows Applikationen (im folgenden kurz Windows Applikation genannt) versteht man Windows Programme, die als geschlossene Einheit selbständig auf einem Windows Betriebssystem ablauffähig sind. Diese können entweder als Produkt eines Herstellers erworben werden (z.B. Adobe Acrobat Reader) oder selbst entwickelt werden (z.B. eigene Windows Programme, die Regelalgorithmen zur Steuerung von Prozessen enthalten).

Dieser Applikation liegt eine Windows Applikation bei, die mit einem SIMATIC Vision Sensor VS 723 kommuniziert und dessen Bilder visualisiert (siehe Bild Abbildung 4-4). Sie kann auch den Vision Sensor simulieren.

### Windows Prozesse

Eine Windows Applikation, die momentan vom Betriebssystem ausgeführt wird, wird in der Sprache der Windows Programmierung als Prozess bezeichnet. Ein Prozess zeichnet sich dadurch aus, dass er einen eigenen, geschützten Speicherbereich hat, eine festgelegte Priorität und von Windows (genauer: vom Scheduler) zyklisch bearbeitet wird.

### Windows Threads

Als Thread bezeichnet man einen Teil eines Prozesses, der unabhängig von allen anderen Teilen des Prozesses ausgeführt werden kann. Zum Beispiel kann man in einem Thread auf Daten von der seriellen Schnittstelle warten.

Ein Thread ist die kleinste ausführbare Einheit, die das Betriebssystem Windows „quasi parallel“ abarbeiten kann.

### Datenaustausch unter Windows

Zur Erfüllung ihrer Aufgaben müssen Threads, Prozesse und auch verschiedene Computer untereinander und mit dem Betriebssystem Daten austauschen. Datenaustausch zwischen Prozessen wird als

## SIMATIC Vision Sensoren und WinAC ODK

Interprozesskommunikation bezeichnet. Der Datenaustausch mit dem Betriebssystem erfolgt über Betriebssystemaufrufe. Windows stellt dazu eine Reihe von Mechanismen zur Verfügung, die im folgenden beschrieben werden.

Ebenfalls über Windows Mechanismen kann auch auf PC-interne und externe Hardware zugegriffen werden.

Wenn zeitliche Abhängigkeiten bestehen, müssen verschieden Prozesse sich synchronisieren. Auch dazu gibt es eine Reihe von Windows Funktionen.

### Windows Ressourcen

Ein häufig verwendeter Begriff in der Programmierung unter Windows ist die „Ressource“. Ressourcen unter Windows können sein:

- Hardware-Komponenten
- Speicherbereiche (auch Dateien)
- Schnittstellen (z.B. serielle Schnittstelle)
- Virtuelle Geräte (z.B. die Konsole)

#### 4.2.1 Windows Mechanismen zur Synchronisation

##### Überblick

Das Betriebssystem Windows bietet folgende Mechanismen zur **Synchronisation** zwischen Prozessen an:

- Mutexes
- Semaphore
- Events

##### Mutexes

Ein Mutex (Mutual Exclusion) ist ein Objekt, das den exklusiven Zugriff auf eine gemeinsam nutzbare Ressource sichert. Ein Mutex kann belegt und freigegeben werden. Dies ist notwendig, da unter Windows Prozesse und Threads parallel ablaufen. So könnte es vorkommen, dass zwei parallele Prozesse eine Ressource gleichzeitig anfordern. Durch Verwendung von Mutexes lässt sich dies verhindern.

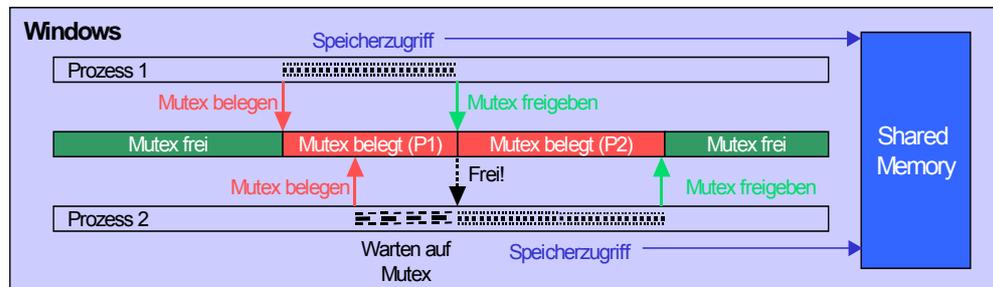
##### Bild: Funktionsweise von Mutexes

Folgende Grafik zeigt, wie zwei Prozesse einen gemeinsamen Speicherbereich (Shared Memory) zum Datenaustausch nutzen. Damit die Datenkonsistenz sicher gestellt ist, muss gewährleistet sein, dass zu einem gegebenen Zeitpunkt immer nur ein Prozess Zugriff auf den Speicher hat. Dies kann durch das Belegen von Mutexes realisiert werden. Ist ein Mutex bereits von Prozess 1 belegt und Prozess 2 versucht ihn ebenfalls zu

SIMATIC Vision Sensoren und WinAC ODK

belegen, so wartet Prozess 2, bis der Mutex von Prozess 1 wieder freigegeben wird. Diesen Mechanismus stellt Windows zur Verfügung.

Abbildung 4-10



## Semaphore

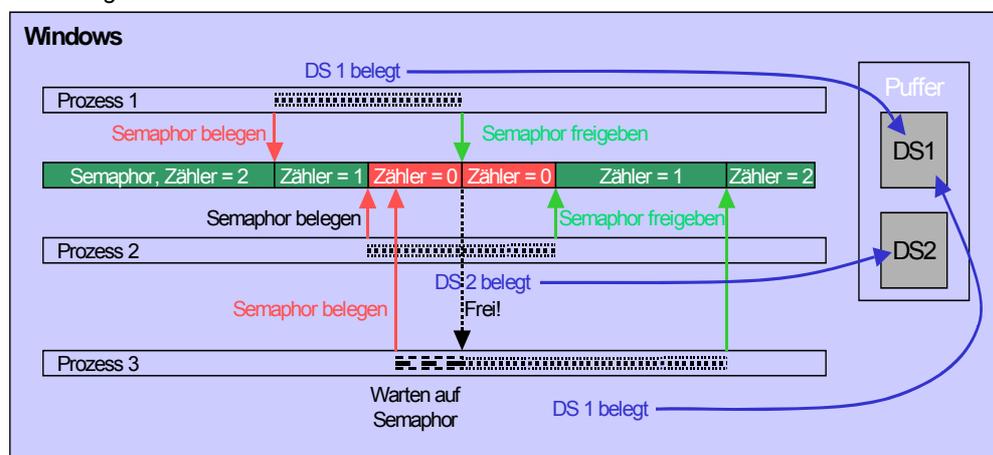
Semaphore funktionieren ähnlich wie Mutexes, haben jedoch einen integrierten Zähler. Damit ist es möglich, die Anzahl der Prozesse festzulegen, die gleichzeitig auf eine Ressource zugreifen können. Wird das Semaphor zum Beispiel mit dem Wert 2 initialisiert, so bedeutet das, dass gleichzeitig zwei Prozesse diese Ressource nutzen können. Fordert ein dritter Prozess das Semaphor an, so muss dieser warten, bis einer der beiden anderen Prozesse das Semaphor wieder frei gibt.

### Funktionsweise von Semaphoren: Bild

Folgende Grafik zeigt einen Puffer, der zwei Datensätze (DS1, 2) beinhaltet. Ein Datensatz kann von jeweils einem Prozess belegt werden. Es können also zwei Prozesse gleichzeitig auf den Puffer zugreifen.

Die Lösung dieses Problems erfolgt über ein Semaphor, das mit dem Wert 2 initialisiert wird. Jede Belegung des Semaphors zählt den Wert um eins herunter, solange bis der Zähler den Stand null erreicht. Prozesse, die dann versuchen, das Semaphor zu belegen, müssen warten, bis ein anderer Prozess das Semaphor wieder frei gibt.

Abbildung 4-11



SIMATIC Vision Sensoren und WinAC ODK

## Events

Ein Event ist ein Windows-Ereignis, das zwei Zustände kennt: Signalisiert und nicht signalisiert. Events dienen zur Benachrichtigung zwischen verschiedenen Prozessen.

### Funktionsweise von Events: Bild

Folgende Grafik zeigt, wie Prozess 1 Daten in einen Puffer schreibt und dem Prozess 2 mitteilt, dass die Daten bereit liegen. Diese Benachrichtigung erfolgt über einen Event. Dazu muss der Prozess 2 auf einen Event warten.

Prozess 2 kann den Event entweder sofort, oder nach Abschluss der Leseoperation wieder zurücksetzen.

Abbildung 4-12



## 4.2.2 Windows Mechanismen zum Datenaustausch

### Überblick

Das Betriebssystem Windows bietet folgende Mechanismen zum Datenaustausch zwischen Prozessen an:

- Windows Nachrichten (Messages)
- Dateien (sowohl auf der Festplatte, als auch im Speicher)
- Über Kommunikationskanäle

### Windows Nachrichten

Zur Übertragung kleiner Datenmengen (einige Bytes) eignen sich Windows Nachrichten. Windows stellt dazu zwei Funktionen zur Verfügung:

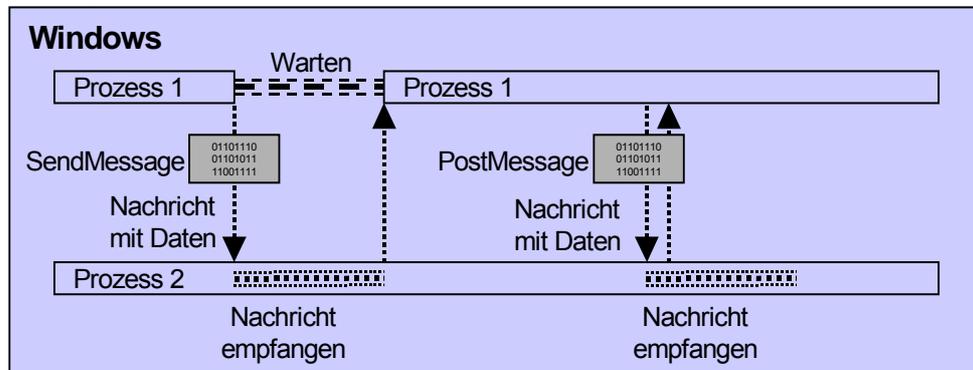
- SendMessage: Sendet eine Nachricht an einen Prozess und wartet, bis die Nachricht empfangen wurde.
- PostMessage: Wie SendMessage, es wird jedoch nicht gewartet.

Beide Funktionen senden eine Nachricht an einen wählbaren Prozess (genauer: an dessen Fenster). Dieser Prozess muss eine Funktion besitzen, die diese Nachricht annimmt und bearbeitet (einen so genannten „Message Handler“).

SIMATIC Vision Sensoren und WinAC ODK

## Bild: Senden von Windows Nachrichten

Abbildung 4-13



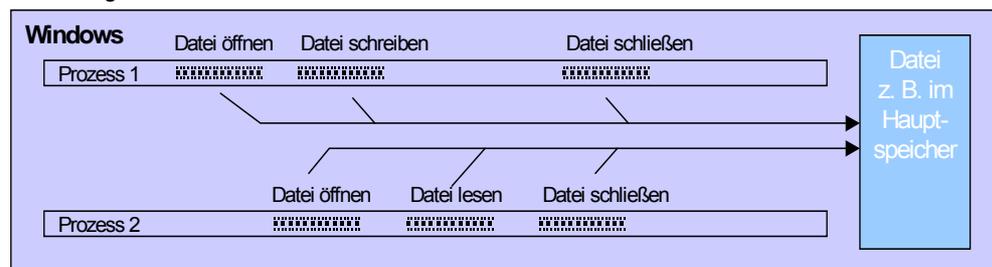
## Dateien

Beliebig große Datenmengen können mittels Dateien übertragen werden. Dabei macht es für Windows keinen Unterschied, ob die Datei auf der Festplatte oder im Hauptspeicher liegt.

### Datenaustausch über Dateien: Bild

Datenaustausch zwischen Dateien funktioniert zwischen beliebig vielen Prozessen. Jeder Prozess, der die Datei nutzen will, muss sie zuerst öffnen und kann dann, je nach Berechtigung, lesen oder schreiben. Um die Datenkonsistenz zu gewährleisten, müssen die oben beschriebenen Synchronisationsmechanismen eingesetzt werden.

Abbildung 4-14



## Teil B: Installation der Beispielapplikation

### Übersicht

#### Inhalt Teil B

Der Teil B führt Sie Schritt für Schritt durch den Aufbau und die Inbetriebnahme der Applikation.

#### Ziel Teil B

Der Teil B dieses Dokuments soll dem Leser

- die Installation des Beispiels mit allen HW-/SW Komponenten erklären
- die Bedienung der Applikation zeigen

#### Behandelte Themen

Kap.	Titel	Seite
5	Installation der Hard- und Software	39
5.1	Vorbereitende Installation	39
5.2	Hardwareinstallation	40
5.3	Installation von WinAC RTX	41
5.4	Installation von SPECTATION	41
5.5	Installation der Applikationssoftware-Komponenten	42
5.6	Konfiguration des PC	44
5.7	Konfiguration und Programmierung des Vision Sensors	45
5.8	Konfiguration und Programmierung der WinAC RTX	48
6	Bedienung der Applikation	51

## 5 Installation der Hard- und Software

### Einleitung

Dieses Kapitel befasst sich mit der Installation der Komponenten für vorliegende Applikation. Es gliedert sich in folgende Abschnitte.

### Behandelte Themen

Kap.	Titel	Seite
5	Installation der Hard- und Software	39
5.1	Vorbereitende Installation	39
5.2	Hardwareinstallation	40
5.3	Installation von WinAC RTX	41
5.4	Installation von SPECTATION	41
5.5	Installation der Applikationssoftware-Komponenten	42
5.6	Konfiguration des PC	44
5.7	Konfiguration und Programmierung des Vision Sensors	45
5.8	Konfiguration und Programmierung der WinAC RTX	48

### Installationsreihenfolge

Für eine fehlerfreie Installation der SIMATIC Komponenten sollte eine bestimmte Installationsreihenfolge eingehalten werden. Nachfolgend sind die Komponenten in ihrer Installationsreihenfolge aufgelistet.

Tabelle 5-1

Nr.	Aktion
1	STEP 7 V5.3 (oder höher), siehe 5.1 Vorbereitende Installation
2	SIMATIC NET PC Software V6.2 (oder höher) , siehe 5.1 Vorbereitende Installation
3	Installation des Vision Sensors VS 723, siehe 5.2 Hardwareinstallation
4	WinAC RTX V4.1 (oder höher) , siehe 5.3 Installation von WinAC RTX Diese Installation teilt sich in folgende Schritte auf: <ol style="list-style-type: none"> <li>1. Installieren und Überprüfen der Ardence RTX-Erweiterungen</li> <li>2. Installieren und Autorisieren der Software WinAC RTX</li> <li>3. Installieren von benötigten Service Packs für WinAC RTX</li> </ol>

### 5.1 Vorbereitende Installation

#### Einleitung

Die Installation von STEP 7 und der SIMATIC NET Software muss unbedingt vor der Installation von WinAC RTX erfolgen.

## SIMATIC Vision Sensoren und WinAC ODK

### STEP 7

Die Installation von STEP 7 erfolgt auf dem PG/PC, welcher für die Projektierung und Programmierung der Automatisierungsstationen vorgesehen ist. Alternativ können Sie STEP 7 auch auf dem PC (WinAC Station) installieren, auf dem WinAC RTX laufen soll.

Auf die Beschreibung der Installation von STEP 7 wird an dieser Stelle verzichtet. Die Installation findet in gewohnter Windows-Umgebung statt und ist selbsterklärend.

### SIMATIC NET

Die SIMATIC NET PC Software wird auf dem PC (WinAC Station) installiert, auf welchem auch WinAC RTX installiert werden soll. Das Softwarepaket enthält alle benötigten Hilfsmittel zur Einrichtung und zum Betrieb einer PC-Station.

Seit der Version 5.2 von STEP 7 wird zur Inbetriebnahme von PC-Stationen das Verfahren „Advanced PC Configuration“ benutzt. Dieses ermöglicht die Projektierung von PC-Stationen direkt in STEP 7. Vor der Benutzung von „Advanced PC Configuration“ empfiehlt es sich unbedingt das Handbuch /2/ „SIMATIC NET, PC-Stationen in Betrieb nehmen – Anleitung und Schnelleinstieg“ zu lesen.

## 5.2 Hardwareinstallation

### Hinweis

Die folgenden Schritte müssen Sie nur ausführen, wenn Sie einen Vision Sensor verwenden wollen. Alternativ können Sie auch im Simulationsmodus arbeiten. Dann können Sie die folgenden Schritte überspringen.

### Herstellen der Datenverbindung über TCP/IP Protokoll

Verbinden Sie den TCP/IP Port der Kamera über das Ethernetkabel mit dem TCP/IP Port Ihres PCs. Falls dieser Port bereits für einen Netzwerkanschluss reserviert ist, können Sie eine zusätzliche Netzwerkkarte in den PC stecken.

### Verbinden der Stromversorgung der Kamera

Trennen Sie die Stromversorgung vom Netz. Schließen Sie das Stromversorgungskabel gemäß Handbuch des Vision Sensors an die Stromversorgung an. Verbinden Sie das andere Ende der Leitung mit dem Vision Sensor.

### Montage des Objektivs

Schrauben Sie das Objektiv auf den Vision Sensor.

SIMATIC Vision Sensoren und WinAC ODK

## Montage der Kamera

Montieren Sie die Kamera Vertikal an einem feststehenden Objekt, so dass das Objektiv ca. 22 cm über der Tischoberkante liegt.

## 5.3 Installation von WinAC RTX

**Hinweis** Zum Installieren von WinAC RTX V4.1 benötigen Sie Administratorrechte („ADMIN“) für Ihr Betriebssystem.

Nachfolgende Tabelle beschreibt die Installation von WinAC RTX.

Tabelle 5-2

Nr.	Aktion
1	Das Setup-Programm startet automatisch, wenn Sie die WinAC RTX CD einlegen. Sollte dies nicht geschehen, führen Sie das Programm „Setup.exe“ auf der CD aus.
2	Nach der Auswahl der Sprache wird ein Dialogfeld angezeigt, welches Sie durch die Installationsaufgaben führt.
3	Betätigen Sie den Button „VenturCom RTX installieren“ und folgen Sie den Anweisungen im Dialogfeld. Es wird Ardence (ehemals VenturCom) RTX auf Ihrem PC installiert. <b>Hinweis</b> Die Lizenznummer (Runtime PAC Number) und die E-Mail Adresse für die Lizenzierung von Ardence RTX finden Sie auf der Rückseite der WinAC RTX CD Hülle. <b>Hinweis</b> Sollte während der Installation der Ardence RTX- Erweiterungen die Fehlermeldung „Your System is using a HAL that is not supported by RTX 5.12“ erscheinen, lesen Sie bitte den FAQ mit der ID <a href="#">17053416</a> auf der A&D Support Homepage ( <a href="http://www.ad.siemens.com/support">www.ad.siemens.com/support</a> ).
4	Nach dem Neustart des PC müssen Sie die Funktionalität der Ardence RTX- Erweiterungen überprüfen. Klicken Sie dazu im „WinAC RTX V4.1 Setup“ Dialog auf den Punkt „Schritt 2“. Sie erhalten Anweisungen zum Testen der RTX-Erweiterungen.
5	Durch Klicken auf den Punkt „Schritt 3“ im Installationsdialog wird die Installation der WinAC RTX Software gestartet.

## 5.4 Installation von SPECTATION

**Hinweis** SPECTATION muss nur installiert werden, wenn Sie den Vision Sensor VS 723 anschließen möchten.

SIMATIC Vision Sensoren und WinAC ODK

## Einsatz der Software Spectation

Spectation dient zur Programmierung und Konfiguration von Vision Sensoren. In der vorliegenden Applikation wurde die Kamera so programmiert, dass Sie zwei verschiedene Flaschentypen erkennen und prüfen kann. Der Kamera wird dazu ein Bild einer „guten“ Flasche gezeigt und abgespeichert. Um Toleranzen und unterschiedliche Lichtverhältnisse auszugleichen, kann die Größe der maximal zulässigen Abweichung konfiguriert werden.

Mit Spectation werden die Systemdaten und Programme in den Vision Sensor geladen.

## Installation der Software Spectation

Spectation ist eine Software, die auf den Betriebssystemen Windows ME, 2000 und XP ablauffähig ist. Installieren Sie die Software, wie Sie dies von anderer Windows Software her gewohnt sind.

## 5.5 Installation der Applikationssoftware-Komponenten

### Kopieren des Quellcodes und der Daten

Die einzelnen Komponenten der Applikation einschließlich des Quellcodes können Sie von der gleichen Webseite laden, von der Sie dieses Dokument geladen haben. Sie können über ein Setup installiert werden.

Öffnen Sie zunächst das ZIP-Archiv „21573047\_WinAC\_TO\_CODE\_v10\_d.zip“ und entpacken Sie die darin enthaltenen Dateien in ein Verzeichnis Ihrer Wahl.

Unter den entpackten Dateien findet sich unter anderem die Datei Setup.exe. Starten Sie diese durch einen Doppelklick.

Es öffnet sich ein Setup Programm. Folgen Sie den Anweisungen dieses Programms. Wenn Sie nach einem Verzeichnis gefragt werden, in das installiert werden soll, so geben Sie bitte C:\WinAC\_ODK\_Sample ein. Damit vermeiden Sie, dass Sie später Änderungen im STEP 7 Programm vornehmen müssen.

### Installieren des ActiveX Controls

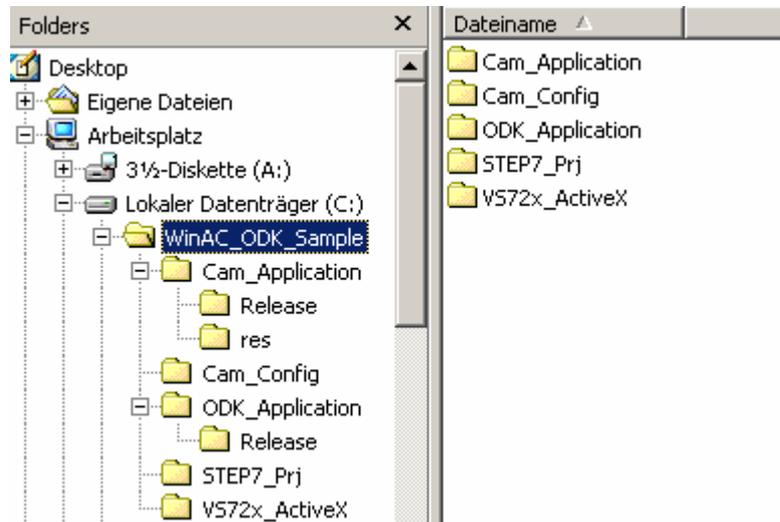
Obiges Setup ruft ein weiteres Setup Programm auf, das das ActiveX Control für den VS 723 installiert. Folgen Sie auch hier den Anweisungen des Programms. Wenn Sie nach einem Pfad gefragt werden, können Sie entweder den vorgeschlagenen Pfad verwenden oder einen beliebigen eigenen Pfad angeben.

### Dateistruktur nach der Installation

Die Dateistruktur nach der Installation sieht wie folgt aus:

## SIMATIC Vision Sensoren und WinAC ODK

Abbildung 5-1



### Inhalt der einzelnen Ordner

Die Ordner unter C:\WinAC\_ODK\_Sample haben folgenden Inhalt:

Tabelle 5-3

Ordner	Inhalt
Cam_Application	Enthält den C++ Quellcode der Kamera Applikation als Visual Studio.net Projekt
Cam_Application\Release	Enthält die ausführbare Kamera Applikation
Cam_Application\res	Enthält Ressourcen Dateien, die zum Quellcode gehören
Cam_Config	Enthält die Konfiguration und das Programm für die Kamera. Außerdem finden Sie dort eine Word Datei mit dem Bild zweier Flaschen zum Ausdrucken.
ODK_Applikation	Enthält den C++ Quellcode der ODK Applikation als Visual Studio.net Projekt
ODK_Applikation\Release	Enthält die fertige Bibliothek der ODK Applikation, die von WinAC RTX geladen werden kann. Dieser Pfad ist als String im Datenbaustein 652 des STEP7 Programms hinterlegt.
STEP7_Prj	Enthält das archivierte STEP 7 Projekt
VS72x_ActiveX	Enthält das Setup Programm des ActiveX Controls für die VS 723. Dieses wurde bereits mit obigem Setup auf Ihrem Rechner installiert.

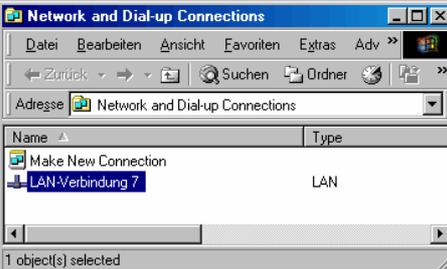
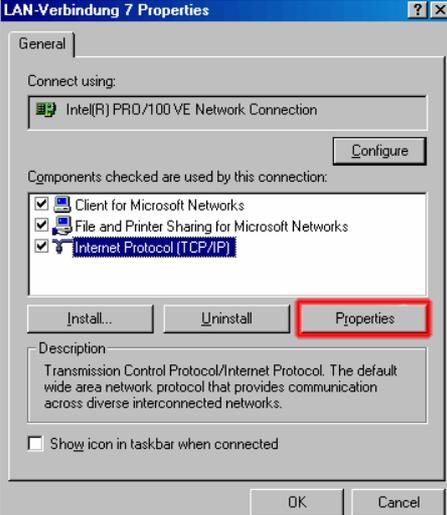
SIMATIC Vision Sensoren und WinAC ODK

## 5.6 Konfiguration des PC

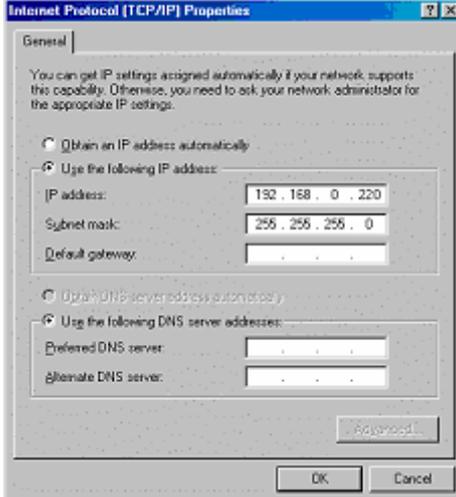
### Konfiguration der Netzwerkverbindung

Damit der Vision Sensor angesprochen werden kann, muss erst die Netzwerkverbindung Ihres PC angepasst werden. Nehmen Sie bitte die Einstellungen wie folgt vor.

Tabelle 5-4

Nr.	Aktion	Anmerkung
1	Klicken Sie mit der rechten Maustaste auf das Symbol „Netzwerkumgebung“ auf Ihrem Desktop und klicken Sie auf „Eigenschaften“. Es öffnet sich ein neues Fenster.	
2	Selektieren Sie in diesem Fenster die LAN Verbindung, über die Sie sich mit dem Vision Sensor verbinden werden. Klicken Sie darauf mit der rechten Maustaste und klicken Sie auf „Eigenschaften“. Es öffnet sich ein neues Fenster	

SIMATIC Vision Sensoren und WinAC ODK

3	<p>Selektieren Sie „Internet Protokoll“ und klicken Sie die Schaltfläche „Eigenschaften“. Es öffnet sich ein neues Fenster.</p> <p>Selektieren Sie dort die Option „Folgende IP-Adresse verwenden“. Geben Sie als IP Adresse 192.168.0.220. Als Subnetzmaske wird automatisch 255.255.255.0 eingetragen.</p>	
4	Schließen Sie alle Dialoge mit OK.	

## 5.7 Konfiguration und Programmierung des Vision Sensors

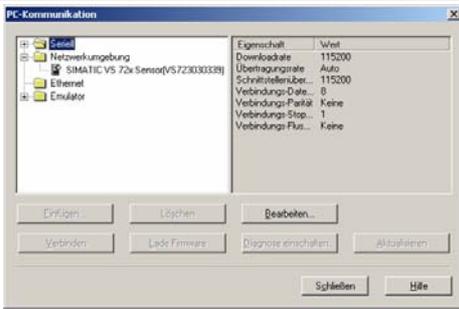
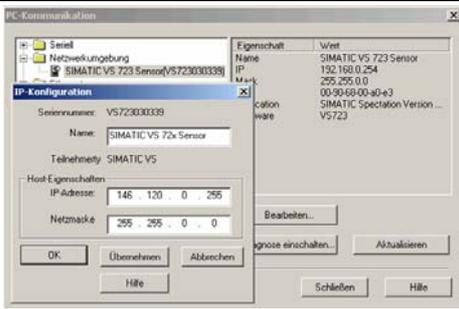
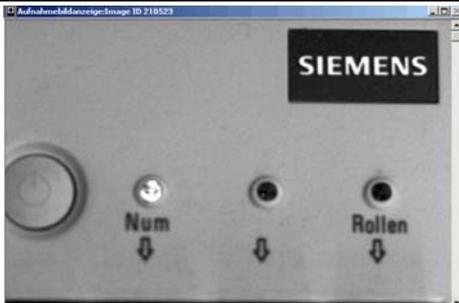
### Herstellen der Verbindung zum Vision Sensor

Damit Sie das Programm in den Vision Sensor laden können, müssen Sie zuerst eine Verbindung zu ihm herstellen.

Tabelle 5-5

Nr.	Aktion	Anmerkung
1	Starten Sie SPECTATION über Start→SIMATIC→Machine Vision→Spectation	Das Spectation Hauptfenster öffnet sich.
2	Öffnen Sie das Menü „Komm.“ und selektieren Sie „PC-Kommunikation ...“ Es öffnet sich ein Fenster.	

SIMATIC Vision Sensoren und WinAC ODK

3	<p>Öffnen Sie den Ordner „Netzwerkumgebung“. Nach einigen Sekunden Suche wird der Vision Sensor angezeigt.</p>	
4	<p>Selektieren Sie den gefundenen Vision Sensor und klicken Sie auf „Bearbeiten...“. Es öffnet sich das Fenster „IP-Konfiguration“</p>	
5	<p>Geben Sie Ihrem Vision Sensor einen Namen. Ändern Sie die IP Adresse auf 192.168.0.224. Ändern Sie die Subnetzmaske zu 255.255.255.0. <b>Hinweis:</b> Vision Sensor und Ethernetschnittstelle des PC müssen sich im gleichen Subnetz befinden. Das bedeutet in unserem Beispiel, dass die ersten drei Ziffernböcke der IP-Adresse identisch sein müssen und der letzte sich unterscheiden muss.</p>	
6	<p>Verlassen Sie den Dialog mit OK. Selektieren Sie den Vision Sensor und Klicken Sie auf „Verbinden“. Wenn Sie alles richtig gemacht haben, erscheint das Fenster „Aufnahmebildanzeige“. Klicken Sie den Knopf „Start“ und sie sehen ein Bild.</p>	

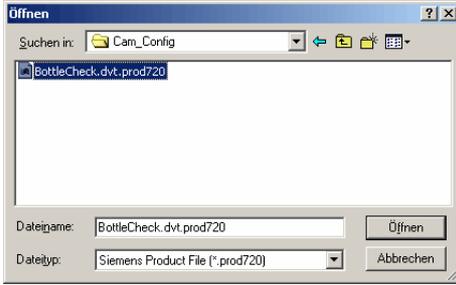
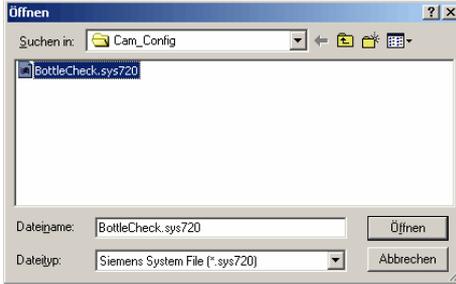
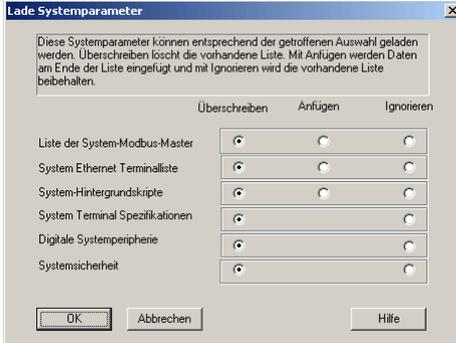
Copyright © Siemens AG 2005 All rights reserved  
 21572937\_WinAC\_TO\_V11\_DOKU\_d.doc  
**Fehler! Unbekannter Name für Dokument-Eigenschaften**

### Laden der Konfiguration in den Vision Sensor

Damit der Vision Sensor seine Testaufgaben erfüllen kann, muss er parametrierung und programmiert werden. Gehen Sie wie folgt vor:

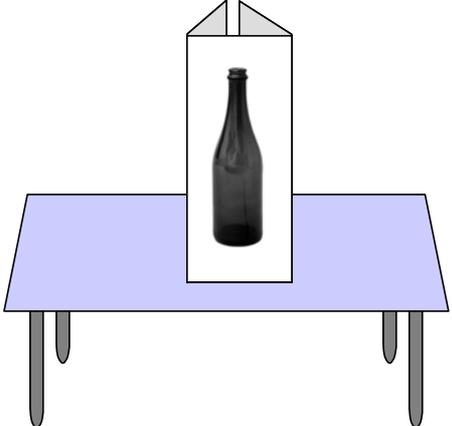
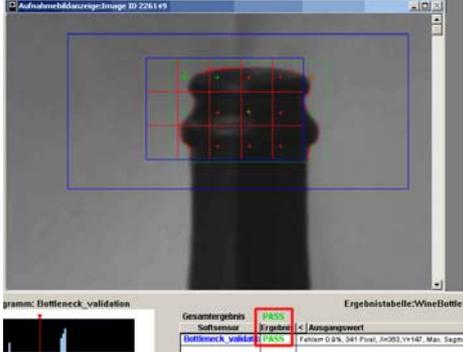
SIMATIC Vision Sensoren und WinAC ODK

Tabelle 5-6

Nr.	Aktion	Anmerkung
1	<p>Sie befinden sich noch immer im Programm SPECTATION. Selektieren Sie im Menü „Prüfprogramme“ den Punkt „Prüfprogramm vom PC wiederherstellen...“. Ein neuer Dialog erscheint. Blättern Sie zum Ordner „Cam_Config“, den Sie im Rahmen dieser Applikation installiert haben.</p> <p>Öffnen Sie die Datei „BottleCheck.dvt.prod720“ Sollte die Datei mit einer älteren Version von Spectation erstellt worden sein dann bestätigen Sie das entsprechende Fenster.</p> <p>Auf die Frage, ob Sie alle Daten im Flash Speicher sichern möchten, antworten Sie bitte mit „Ja“. So wird das Programm direkt in den Vision Sensor übertragen</p>	
2	<p>Selektieren Sie nun im Menü „Prüfprogramme“ den Punkt „Systemdaten vom PC wiederherstellen...“. Ein neuer Dialog erscheint. Blättern Sie wieder zum Ordner „Cam_Config“ und öffnen Sie die Datei „BottleCheck.sys720“</p>	
3	<p>Im nächsten Fenster belassen Sie die vorgegebenen Werte und klicken auf „OK“. Sollte die Datei mit einer älteren Version von Spectation erstellt worden sein dann bestätigen Sie das entsprechende Fenster.</p> <p>Auf die Frage, ob Sie alle Daten im Flash Speicher sichern möchten, antworten Sie bitte mit „Ja“. So werden die Systemdaten direkt in den Vision Sensor übertragen</p>	

Copyright © Siemens AG 2005 All rights reserved  
 21572937\_WinAC\_TO\_V11\_DOKU\_d.doc  
**Fehler! Unbekannter Name für Dokument-Einzelblatt.**

## SIMATIC Vision Sensoren und WinAC ODK

4	<p>Öffnen Sie die Word-Datei „Flaschen.doc“ im Verzeichnis „Cam_Config“ und drucken Sie diese aus. Sie enthält Bilder einer Wein- und einer Wasserflasche, mit denen die Kamera parametrierung wurde. Falten Sie den Ausdruck an beiden Enden nach hinten, so dass Sie es auf einen Tisch stellen können.</p>	
5	<p>Zum Testen können Sie das Bild einer Weinflasche ca. 40 cm vor die Kamera stellen. Stellen Sie das Objektiv scharf. Beobachten Sie in der Ergebnistabelle von SPECTATION das Ergebnis des Softsensors. Wenn Sie das Bild der Flasche im richtigen Abstand und im richtigen Drehwinkel abbilden, erhalten Sie das Ergebnis „Pass“. Die richtige Kameraposition dazu muss durch Probieren ermittelt werden. Der Vision Sensor ist nun einsatzbereit.</p>	

## 5.8 Konfiguration und Programmierung von WinAC RTX

### Wichtige Einstellungen im Komponenten Konfigurator

Öffnen Sie den Komponenten Konfigurator über:

Start → Komponenten Konfigurator

oder über Doppelklick auf das Symbol  in der Taskleiste.

Für die Anwendung des Beispielprojekts für diese Applikation sollten Sie im Komponenten Konfigurator folgende Einstellungen vornehmen:

- Ändern Sie den Stationsnamen über den Button „Stationsname“ auf „PCWinAC“. Dies ist der Name, der im STEP 7 Projekt für die WinAC Station vergeben ist.
- WinAC RTX sollte auf Steckplatz (Index) 2 stehen.

### Dearchivierung des S7 Programms

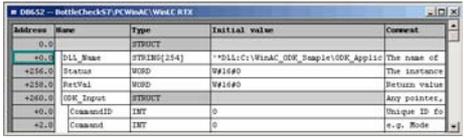
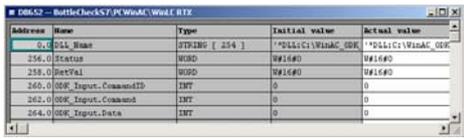
Öffnen Sie STEP 7 und dearchivieren (Datei → Dearchivieren) Sie das Archiv, das Sie mit dieser Applikation installiert haben. Es findet sich im Ordner „WinAC\_ODK\_Sample\STEP7\_Prj\ODK\_xamp.zip“

SIMATIC Vision Sensoren und WinAC ODK

## Anpassung des S7 Programms

Falls Sie bei der Installation dieses Beispiels den vorgeschlagenen Pfad verwendet haben, müssen Sie das S7 Programm nicht anpassen. Ansonsten befolgen Sie bitte folgende Schritte:

Tabelle 5-7

Nr.	Aktion	Anmerkung
1	Öffnen Sie das Projekt BottleCheckS7, das sie vorher dearchiviert haben.	
2	Öffnen Sie den darin enthaltenen DB652.	Der DB652 enthält Daten, die zur Initialisierung des ODK Programms benötigt werden
3	Wechseln Sie in die Deklarationsansicht Ihres Editors mit Ansicht→Deklarationsansicht	
4	An erster Stelle im DB steht die Variable DLL_Name. Sie enthält den Pfad zu der Bibliothek, die das ODK Programm enthält (ODK_BottleChecker.dll). Ändern Sie den Pfad dorthin, wohin Sie die Beispiele installiert haben. Der Prefix „*DLL.“ muss erhalten bleiben. Er gibt an, dass es sich um eine DLL (und nicht um ein COM Objekt) handelt.	
5	Wechseln Sie nun in die Deklarationsansicht Ihres Editors mit Ansicht→Deklarationsansicht	
6	Damit der Aktualwert der geänderten Zeichenkette auch dem Initialwert entspricht, muss der Datenbaustein initialisiert werden. Klicken Sie dazu auf Bearbeiten→Datenbaustein initialisieren. Der geänderte Wert findet sich nun auch im Aktualwert.	
7	Speichern Sie den Datenbaustein.	

Copyright © Siemens AG 2005 All rights reserved  
21572937\_WinAC\_TO\_V11\_DOKU\_d.doc

Fehler! Unbekannter Name für Dokument-Element.

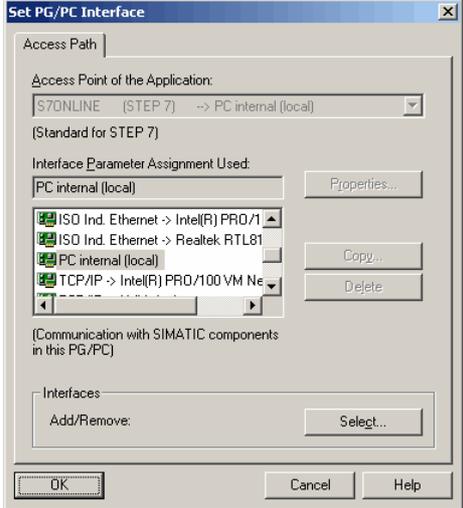
## Laden des Programms in WinAC RTX

Das Programm kann nun in WinAC RTX geladen werden.

Tabelle 5-8

Nr.	Aktion	Anmerkung
1	Starten Sie WinAC RTX über Start→SIMATIC→PC Based Control→WinAC RTX	
2	Führen Sie über die Schaltfläche MRES von WinAC RTX einen Memory Reset durch.	

SIMATIC Vision Sensoren und WinAC ODK

3	Wechseln Sie zu STEP 7 und selektieren Sie dort im geöffneten Projekt die PC Station PCWinAC.	
8	Öffnen Sie den PG/PC Schnittstellen Dialog über Extras→PG/PC Schnittstelle einstellen...	
9	Selektieren Sie dort „PC internal (local)“ und verlassen Sie den Dialog mit OK:	
10	Laden Sie die Bausteine über die Schaltfläche  in die WinAC RTX Steuerung.	

Copyright © Siemens AG 2005 All rights reserved  
 21572937\_WinAC\_TO\_V11\_DOKU\_d.doc  
**Fehler! Unbekannter Name für Dokument-Eigenschaft.**

SIMATIC Vision Sensoren und WinAC ODK

## 6 Bedienung der Applikation

### Einleitung

Hier erfahren Sie, wie Sie die Applikation entweder über die Variablen-tabelle oder über die mitgelieferte Visualisierungsoberfläche bedienen können.

### Starten der Teilapplikationen

Bitte folgen Sie angegebener Reihenfolge, da die korrekte Funktion des Beispiels ansonsten nicht gewährleistet ist.

Tabelle 6-1

Nr.	Aktion	Anmerkung
1	Setzen Sie WinAC RTX in STOP durch anklicken der Schaltfläche STOP	
2	Starten Sie die Variablen-tabelle VAT_1, die sich im Bausteinordner des mitgelieferten Projektes befindet. Klicken Sie das Symbol „Variable beobachten“, um sich die Werte online anzeigen zu lassen.	
3	Starten Sie die Kamera Applikation „VisionSensor_723.exe“ im Ordner C:\WinAC_ODK_Sample\Cam_Application\Release Falls Sie einen Vision Sensor nutzen werden, prüfen Sie, dass die IP Adresse mit der Ihrer Kamera übereinstimmt. Ansonsten selektieren Sie bitte die Option „Simulate camera“	

## SIMATIC Vision Sensoren und WinAC ODK

### Bedienung der Applikation

Sie können die Applikation entweder von der Variablen-tabelle aus oder über die Oberfläche der Kamera Applikation steuern. Wenn Sie auf einer Seite eine Bedienung vornehmen, wird sich diese auf der anderen Seite entsprechend in einer Änderung bemerkbar machen. Die ODK Applikation hat keine eigene Bedienoberfläche.

### Bedienung über die Variablen-tabelle: Bild

Die Variablen-tabelle hat folgende Inhalte:

Abbildung 6-1

	Address	Symbol	Displa	Status value	Modify
1		// Variables for SetMode			
2	MW 54	"CommandID"	HEX		
3	M 80.1	"SetMode_Play"	BOOL		
4	M 80.2	"SetMode_Stop"	BOOL		
5	M 80.3	"SetMode_Pause"	BOOL		
6	M 80.4	"SetMode_Continue"	BOOL		
7	M 80.5	"SetMode_WaterBottle"	BOOL		
8	M 80.6	"SetMode_WineBottle"	BOOL		
9					
10		// Acknowledgement of the above commands			
11	M 80.7	"AcknowledgeBit"	BOOL		
12	MW 76	"AckCommandID"	HEX		
13	MD 90	"BottleCount"	DEC		
14					
15		// Variables for GetMode (called every 20th OB1 cycle)			
16		// CamMode 1 = Play, 2 = Stop, 3 = Pause			
17	MW 70	"CamMode"	DEC		
18		// Bottle type 1 = Water bottle, 2 = wine bottle			
19	MW 72	"BottleType"	DEC		
20		// TestResult 0 = Passed, 1 = warning, 5 = none, -1 = Failed			
21	MW 74	"LastTestResult"	DEC		
22					

Press F1 for help. Offline

SIMATIC Vision Sensoren und WinAC ODK

## Bedienung über die Variablen-tabelle: Bedeutung der Variablen

Die Variablen haben folgende Bedeutung. Der Modus L (Lesen) steht für Variablen, die nur beobachtet werden können, S (Schreiben) für solche, die Sie aktiv verändern können und sollen.

Tabelle 6-2

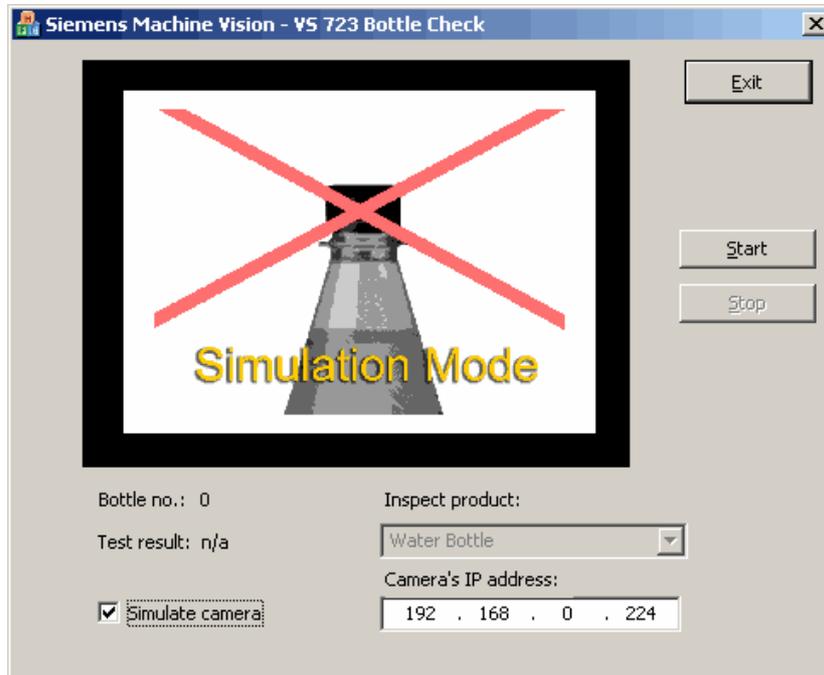
Variable	Modus	Bedeutung
Command_ID	L	Enthält eine eindeutige Kennung des Kommandos, das abgesetzt wird (wird automatisch generiert).
SetMode_Play	S	Kamera Applikation in den Modus Play versetzen. Die laufenden Bilder des Vision Sensors werden angezeigt.
SetMode_Stop	S	Kamera Applikation in den Modus STOP versetzen. Die laufenden Bilder des Vision Sensors werden nicht mehr angezeigt und der Prüfmodus wird beendet.
SetMode_Pause	S	Kamera Applikation in den Modus Pause versetzen. Die laufenden Bilder des Vision Sensors werden angezeigt, aber es werden keine Prüfungen durchgeführt.
SetMode_Continue	S	Pause Modus aufheben und Prüfungen fortsetzen.
SetMode_WaterBottle	S	Wasserflasche als Prüfobjekt auswählen.
SetMode_WineBottle	S	Weinflasche als Prüfobjekt auswählen.
AcknowledgeBit	L	Die ODK Applikation hat auf ein Kommando eine Bestätigung geschickt.
AckCommandID	L	Spezifiziert das Kommando (siehe Command_ID), für das die Bestätigung über AcknowledgeBit geschickt wurde.
BottleCount	L	Ein Zähler, der vom Vision Sensor hochgezählt wird. Spezifiziert die Anzahl geprüfter Flaschen.
CamMode	L	Zeigt den momentanen Modus der Kamera Applikation an.
BottleType	L	Zeigt an, welcher Flaschentyp gerade geprüft wird.
LastTestResult	L	Zeigt das Testergebnis der letzten Flaschenprüfung an.

## Bedienung über die Kamera Applikation: Bild

Anstelle der Variablen-tabelle kann auch die Oberfläche der Kamera Applikation zur Bedienung der Applikation genutzt werden.

SIMATIC Vision Sensoren und WinAC ODK

Abbildung 6-2



## Bedienung über die Kamera Applikation: Bedienelemente

Tabelle 6-3

Element	Bedeutung
Schaltfläche Exit	Beendet die Applikation
Schaltfläche Start	Kamera Applikation in den Modus Play versetzten. Die laufenden Bilder des Vision Sensors werden angezeigt.
Schaltfläche Stop	Kamera Applikation in den Modus STOP versetzten. Die laufenden Bilder des Vision Sensors werden nicht mehr angezeigt und der Prüfmodus wird beendet.
Inspect Product:	Hierüber kann der zu prüfende Flaschentyp ausgewählt werden, oder die Prüfung mit der Option „Pause inspection“ angehalten werden
Camera's IP address	Hier tragen Sie die IP Adresse des Vision Sensors ein, falls Sie mit eine VS 723 angeschlossen haben.
Bottle no	Zeigt an, wie viele Flaschen bereits geprüft wurden
Test result	Zeigt das Testergebnis der letzten Flaschenprüfung an.
Simulate Camera	Simuliert den Vision Sensor. Über diese Option können Sie das Beispiel in Betrieb nehmen, ohne eine VS 723 anzuschließen. Die Daten, die sonst vom Vision Sensor geliefert werden (Flaschenzähler, Testergebnis) werden simuliert.

SIMATIC Vision Sensoren und WinAC ODK

## Teil C: Programmbeschreibung

### Übersicht

#### Inhalt Teil C

Der Teil C ist vor allem dann interessant, wenn Sie auf Basis der vorliegenden Software eine Erweiterung oder Anpassung an Ihre Anlage vornehmen möchten.

#### Ziel Teil C

Dieser Teil der Dokumentation soll

- dem Leser Details aus dem Code einiger Kernprogrammteile erläutern
- Hinweise liefern, wo Erweiterungen sinnvoll sind

#### Voraussetzung

Für diesen Teil ist es hilfreich, wenn Sie sowohl Kenntnisse in der SIMATIC als auch in der Windows Programmierung haben.

Hilfreich ist es, vor der Code-Beschreibung die Kapitel im Teil A1 und A2 zu lesen.

## Einleitung

Nachdem Sie nun die grundlegenden Mechanismen zum WinAC ODK, zu Windows Synchronisations- und Datenaustauschmechanismen verstanden haben, zeigen wir Ihnen nun konkret, wie Sie diese sinnvoll in eigene Automatisierungslösungen integrieren können.

Als praktisches Beispiel dient die Ansteuerung einer intelligenten Kamera (SIMATIC Vision Sensor VS 723) von WinAC RTX aus. Die Mechanismen können jedoch auf jede beliebige andere Lösung übertragen werden.

## Das konkrete Beispiel

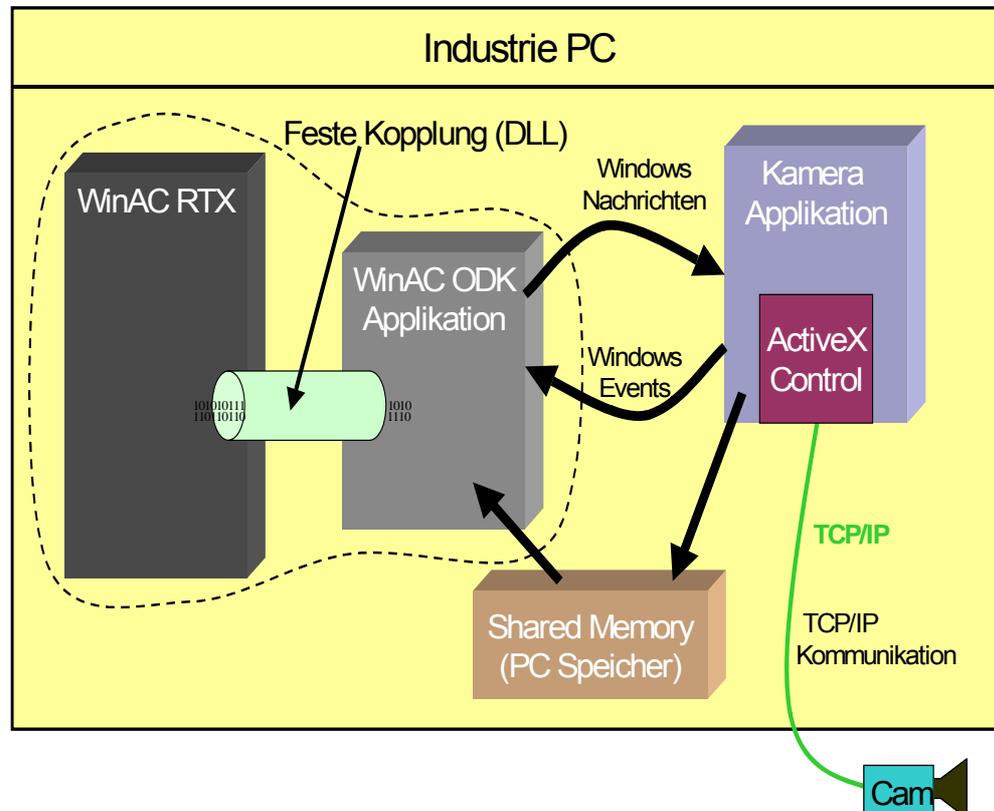
Um Ihnen die Mechanismen des WinAC ODK und den Datenaustausch zwischen Windows Applikationen an einem möglichst konkreten Beispiel erklären zu können, haben wir die Flaschenkontrolle als Teilaufgaben in einer Getränkeabfüllanlage gewählt.

Die Flaschenkontrolle wird mit Hilfe einer Kamera durchgeführt. Sie kann selbständig zwei Flaschentypen unterscheiden, die vorher „angelernt“ wurden. Dazu wird Ihr per Kommando mitgeteilt, welcher Typ aktuell geprüft werden soll. Das Eintreffen einer neuen Flasche wird per Sensor direkt an die Kamera gemeldet. Die Kamera prüft die Flasche und gibt das Prüfergebnis aus, das dann an eine Steuerung übertragen wird.

## Die Beispielapplikation im Überblick

Das folgende Bild gibt einen Überblick über die gewählte Lösung.

Abbildung 6-3



## Elemente der gewählten Lösung

Die Lösung besteht aus folgenden Elementen:

- WinAC RTX: Steuert eine Anlage (z.B. Abfüllanlage); Dazu benötigt sie Daten von der Kamera
- WinAC ODK Programm: Transferiert die Daten zwischen WinAC RTX zur Kamera Applikation und umgekehrt. Das Grundgerüst dieser Applikation wurde mit dem ODK Assistenten erstellt
- Kamera Applikation: Erhält Kommandos von WinAC RTX und steuert den Vision Sensor an; Enthält eine sehr einfache Visualisierung der Kamerabilder; Die Kamera Applikation steht hier stellvertretend für eine beliebige Windows Applikation
- Shared Memory: Dient als Datenübertragungsmedium zwischen den beiden Applikationen
- Kamera (Vision Sensor): Inspiziert Teile in der Anlage selbständig und meldet den Zustand des Teiles (Gut / Warnung / Schlecht) an die Kamera Applikation

## Kommunikation der Elemente untereinander

Damit WinAC RTX Ihre Steuerungsaufgaben erfüllen kann und der Vision Sensor weiß, welcher Flaschentyp zu prüfen ist, müssen alle Elemente in

der Lage sein, miteinander Daten auszutauschen und sich zu synchronisieren. WinAC RTX sendet hierzu Kommandos an den Vision Sensor. Der Vision Sensor wiederum sendet Testergebnisse an WinAC RTX.

Anhand dieser Kommunikation zeigen wir Ihnen auch die Anwendung der unterschiedlichen Modi des CCX im WinAC ODK. Dabei lernen Sie auch die Anwendung von verschiedenen Windows Mechanismen in der Praxis kennen.

## 7 Erläuterung zum Code der einzelnen Elemente

### Einleitung

In diesem Kapitel werden zu den wichtigsten Funktionselementen die wichtigsten Kern-Codesequenzen dargestellt und erläutert. Im Einzelfall wird auf den kommentierten Code in der Applikation verwiesen.

Da der Code in WinAC RTX und der Code der Kamera Applikation mit dem der ODK Applikation sehr eng zusammenhängen, werden diese drei Elemente immer zusammenhängend erklärt.

### Inhalt Kapitel Erläuterungen zum STEP 7-Programm

Tabelle 7-1

Kap.	Titel	Seite
7	Erläuterung zum Code der einzelnen Elemente	60
7.1	Initialisierung des WinAC ODK CCX	60
7.2	Beispiel für einen synchronen Aufruf des WinAC ODK CCX	61
7.3	Beispiel für einen asynchronen Aufruf des WinAC ODK CCX	67
7.4	Beispiel für eine ODK Funktion, die im Hintergrund läuft	73
8	Anpassungen der Programme	79
8.1	Anpassungen im S7-Programm	79
8.2	Anpassung der C++ Applikationen	80

### Einführung

Hier zeigen wir Ihnen, wie die Funktionalitäten in konkreten Code umgesetzt wurden, und zwar

- in WinAC RTX
- in der ODK Applikation
- in der Kamera Applikation

### 7.1 Initialisierung des WinAC ODK CCX

#### Überblick

Bevor die ODK Applikation von WinAC RTX genutzt werden kann, muss sie zunächst gestartet und initialisiert werden. Dies geschieht in unserem Beispiel im OB 100 von WinAC RTX.

#### Beschreibung der Funktion

Die Initialisierung betrifft nur das S7 Programm in WinAC RTX und das ODK Programm.

Tabelle 7-2

Funktionskomponente	Funktionalität
WinAC RTX	Im OB 100 ruft WinAC RTX den SFB 65001 auf. Dieser startet und initialisiert das ODK Programm.
ODK Applikation	In der ODK Applikation wird lediglich die Funktion DLLMain () aufgerufen, die jedoch keinerlei Code beinhaltet.

## Implementierung der Funktion

Hier sehen Sie die relevanten Codestellen für die zwei Funktionskomponenten.

Tabelle 7-3

Funktionskomponente	Code
WinAC RTX	<p><u>OB100</u></p> <pre>CALL SFB65001, DB65001       PROGID:="SFB 65002 DB".DLL_Name       STATUS:="SFB 65002 DB".Status</pre> <ul style="list-style-type: none"> <li>• ProgID enthält den Pfad und Dateinamen (eine DLL) der ODK Applikation</li> <li>• Status enthält entweder einen Fehlercode oder eine Identifikationsnummer, die für spätere Aufrufe gespeichert werden muss</li> </ul>
ODK Applikation	<pre>BOOL WINAPI DllMain (     HINSTANCE hinstDLL, // handle     DWORD fdwReason, // calling     LPVOID lpvReserved // reserved ) {     return TRUE; }</pre> <ul style="list-style-type: none"> <li>• Die Funktion ist leer, wird jedoch vom Betriebssystem erwartet</li> </ul>

## Zusammenfassung

Vor der ersten Benutzung muss die ODK Applikation durch einen Aufruf des SFB 65001 erzeugt und initialisiert werden.

## 7.2 Beispiel für einen synchronen Aufruf des WinAC ODK CCX

### Überblick

Um von WinAC RTX aus den Status der Kamera Applikation und das letzte Testergebnis der Flaschenprüfung zu erfahren, ist eine Funktion zur Statusabfrage implementiert.

Der synchrone Aufruf wird immer dann verwendet, wenn die Funktionen der ODK Applikation sehr schnell (Bereich: wenige Millisekunden) abgearbeitet werden können. Synchrone Aufrufe blockieren den OB1 Zyklus und können zu einer Überschreitung der maximalen Zyklusdauer führen.

### Beschreibung der Funktion

Damit Daten von der Kamera zur WinAC RTX übertragen werden können, muss in allen drei Funktionskomponenten Code implementiert werden.

Tabelle 7-4

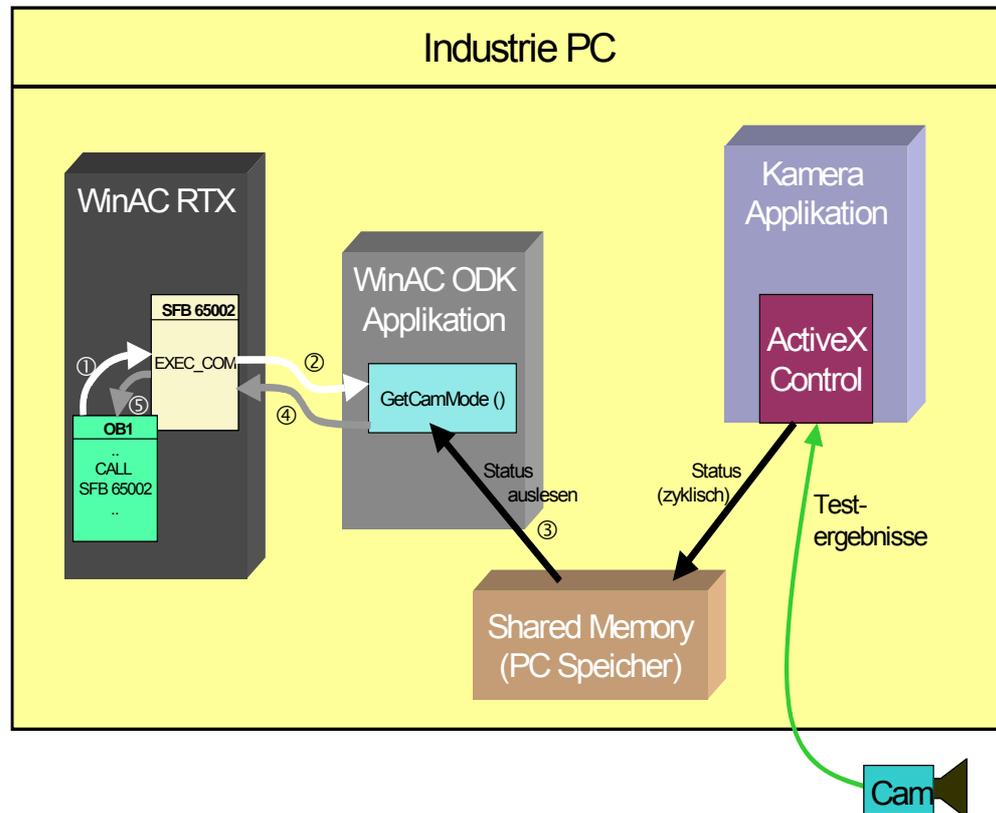
Funktionskomponente	Funktionalität
WinAC RTX	<ol style="list-style-type: none"> <li>1. In WinAC RTX wird zunächst eine eindeutige Kommandokennung generiert. Diese dient dazu, ein gesendetes Kommando eindeutig identifizieren zu können.</li> <li>2. WinAC RTX ruft den SFB 65002 auf. Damit wird eine festgelegte Funktion in der ODK Applikation aufgerufen. Bei diesem Aufruf können Parameter übergeben werden.</li> <li>3. Nach der Rückkehr des SFB 65002 wurden die gewünschten Daten von der ODK Applikation in die Rückgabeparameter des SFB eingetragen.</li> </ol> <p>Das Kommando wird als synchron bezeichnet, da nach Rückkehr des SFB 65002 der Auftrag beendet ist.</p>
Kamera Applikation	<ol style="list-style-type: none"> <li>4. Ein zyklischer Timer ruft in regelmäßigen Abständen eine Timer Funktion auf</li> <li>5. In dieser Timer Funktion sollen die Daten in einen gemeinsam genutzten Bereich des Hauptspeichers geschrieben werden. Dazu muss exklusiver Zugriff auf diesen Speicher gewährleistet sein. Dazu wird ein Mutex belegt.</li> <li>6. Die Daten werden in den Shared Memory geschrieben</li> <li>7. Der Mutex wird wieder frei gegeben.</li> </ol>

ODK Applikation	<ol style="list-style-type: none"> <li>1. Die von WinAC RTX angeforderten Daten stehen im Shared Memory. Die ODK Applikation verschafft sich exklusiven Zugriff auf diesen Speicherbereich durch die Belegung eines Mutex</li> <li>2. Die ODK Applikation liest die gewünschten Daten aus dem Shared Memory</li> <li>3. Der Mutex wird wieder freigegeben</li> </ol>
-----------------	--

## Darstellung des Datenflusses

Folgendes Bild zeigt den Datenfluss zum Auslesen des Kamera Status.

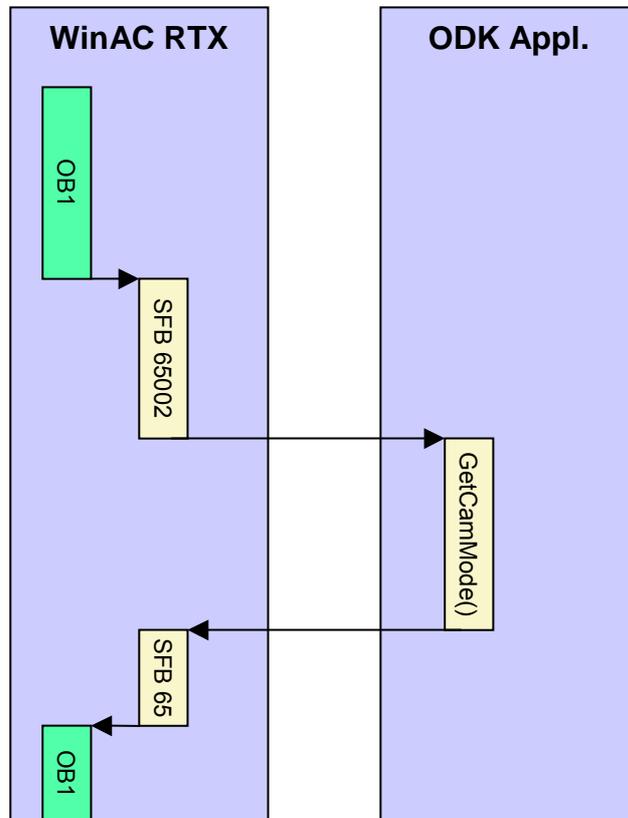
Abbildung 7-1



## Darstellung des Ablaufs als Sequenzdiagramm

Das Bild zeigt den Ablauf eines synchronen Aufrufs.

Abbildung 7-2



## Implementierung der Funktion

Hier sehen Sie die relevanten Codestellen für die drei Funktionskomponenten.

Tabelle 7-5

Funktionskomponente	Code
WinAC RTX	<pre data-bbox="587 461 1300 1086"> // Create a unique command ID L   "CommandID" +   1 T   "CommandID"  // Call ODK CALL "EXEC_COM" , DB65002   OBJHandle := "SFB 65002 OB".Status   Command   := DW#16#2   InputData := "SFB 65002 OB".ODK_Input   OutputData := "SFB 65002 OB".ODK_Output   STATUS    := "SFB 65002 OB".RetVal  L   "SFB 65002 OB".ODK_Output.Data1 T   "CamMode"  L   "SFB 65002 OB".ODK_Output.Data2 T   "BottleType"  L   "SFB 65002 OB".ODK_Output.Data3 T   "LastTestResult" </pre> <ul data-bbox="635 1131 1348 1299" style="list-style-type: none"> <li>• CommandID ist die eindeutige Kennung des Kommandos</li> <li>• Command spezifiziert die auszuführende Funktion in der ODK Applikation (siehe unten)</li> <li>• Input Data sind Eingabedaten für die ODK Applikation</li> <li>• Output Data sind die Ausgabedaten der ODK Applikation</li> </ul>
Kamera Applikation	<pre data-bbox="587 1312 1332 1758"> dwMutexState = WaitForSingleObject                 (m_hSharedMemMutex, 800);  // Set the current status to shared memory if (dwMutexState == WAIT_OBJECT_0) {     SharedMemStruct *smsMode;     smsMode = (SharedMemStruct *) m_pMapPtr;     smsMode-&gt;iBottleType = m_iCurrentBottleType;     smsMode-&gt;iLastTestResult =     m_iTestResult;     smsMode-&gt;iMode = m_iCurrentMode; } ReleaseMutex (m_hSharedMemMutex); </pre> <ul data-bbox="587 1803 1332 1937" style="list-style-type: none"> <li>• WaitForSingleObjekt () belegt den Mutex</li> <li>• smsMode enthält einen Zeiger auf den Shared Memory, in den Daten geschrieben werden</li> <li>• ReleaseMutex () gibt den Mutex wieder frei</li> </ul>

<p>ODK Applikation</p>	<p><u>Funktion Execute:</u></p> <pre> switch(command) {     case 1:         // Code for setting camera         mode ... break;     case 2:         // Example for a synchronous         call         retVal = GetCamMode(Input,         Output);         break; } </pre> <ul style="list-style-type: none"> <li>• Der gelb markierte Wert spezifiziert die auszuführende Funktion und kommt vom SFB 65002 Aufruf (siehe oben)</li> </ul> <p><u>Funktion GetCamMode:</u></p> <pre> dwWaitResult = WaitForSingleObject                 (m_hSharedMemMutex, 5); if (dwWaitResult == WAIT_OBJECT_0) {     Input.ODK_ReadS7INT (0, iCommandID);      SharedMemStruct *pSharedMem =         (SharedMemStruct*) m_pMapPtr;      Output.ODK_WriteS7INT (0, iCommandID);     Output.ODK_WriteS7INT (2, (ODK_SINT16)         pSharedMem-&gt;iMode);     Output.ODK_WriteS7INT (4, (ODK_SINT16)         pSharedMem-&gt;iBottleType);     Output.ODK_WriteS7INT (6, (ODK_SINT16)         pSharedMem-&gt;iLastTestResult);      result = ODK_SUCCESS; } </pre> <ul style="list-style-type: none"> <li>• WaitForSingleObject belegt einen Mutex</li> <li>• pSharedMem enthält einen Zeiger auf den Shared Memory</li> <li>• WriteS7INT schreibt die Ausgangsdaten, die im SFB 65002 als OutputData eingelesen werden</li> </ul>
----------------------------	---

## Zusammenfassung

Mit Hilfe dieses Codes kann der Status der Kamera über den Shared Memory in den Parameter OutputData des SFB65002 übertragen werden. Dieser Mechanismus kann auf jede beliebige Windows Applikation angewendet werden, die Daten zur WinAC RTX übertragen soll.

## 7.3 Beispiel für einen asynchronen Aufruf des WinAC ODK CCX

### Überblick

Um von WinAC RTX aus den Modus der Kamera zu verändern, ist eine Funktion zum Setzen des Kameramodus' implementiert.

Da dies länger dauern kann und somit zu einer Überschreitung der maximalen Zykluszeit in WinAC RTX führen könnte, wird die Funktion als asynchroner Aufruf implementiert. Das bedeutet, dass die Funktion in der ODK Applikation parallel (d. h. gleichzeitig) zum OB1 Zyklus läuft. Dies ist möglich, da Windows ein Multitasking Betriebssystem ist.

Asynchrone Funktionen werden immer dann verwendet, wenn die Abarbeitung der Funktion länger dauert als ein OB1 Zyklus oder wenn die Dauer unbekannt ist.

### Beschreibung der Funktion

Damit die von WinAC RTX gesendeten Daten zur Kamera übertragen werden, muss in allen drei Funktionskomponenten Funktionalität implementiert werden.

Tabelle 7-6

Funktionskomponente	Funktionalität
WinAC RTX	<ol style="list-style-type: none"> <li>1. In WinAC RTX wird zunächst eine eindeutige Kommandokennung generiert. Diese dient dazu, ein gesendetes Kommando eindeutig identifizieren zu können.</li> <li>2. WinAC RTX ruft den SFB 65002 mit entsprechenden Parametern auf. Damit wird eine festgelegte Funktion in der ODK Applikation aufgerufen.</li> <li>3. WinAC RTX stellt den OB 52 zur Verfügung, der direkt aus der ODK Applikation heraus aufgerufen werden kann.</li> </ol> <p>Das Kommando wird als asynchron bezeichnet, da nach Rückkehr des SFB 65002 der Auftrag immer noch von der ODK Applikation bearbeitet wird.</p>

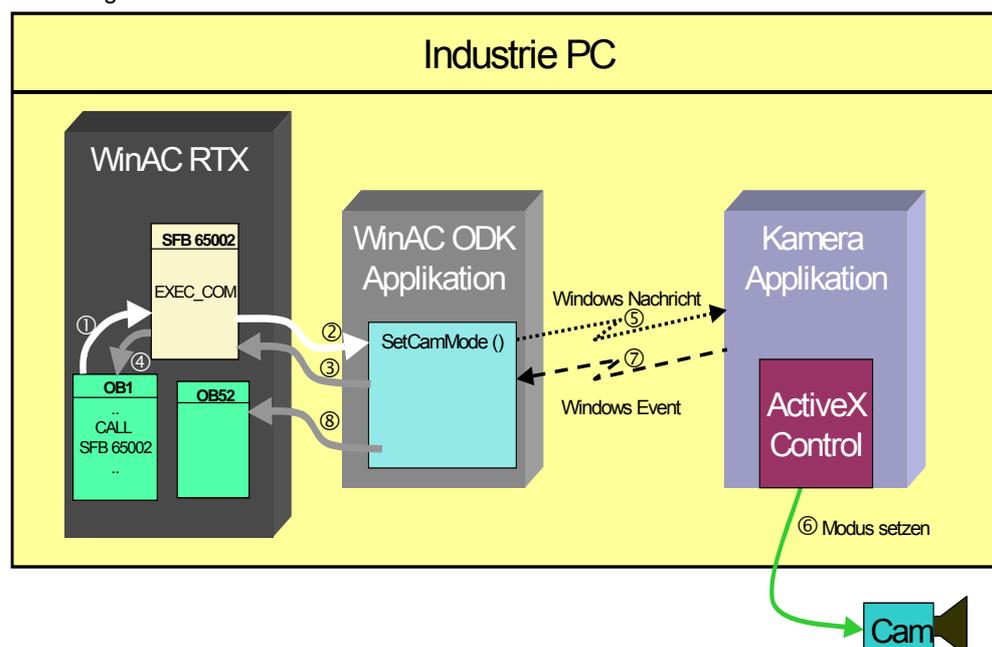
<p>ODK Applikation</p>	<ol style="list-style-type: none"> <li>1. Die ODK Applikation liest und speichert die Eingangsparameter des SFB 65002 Aufrufes. Anhand dieser erkennt sie, dass ein Kommando zum Setzen des Kameramodus aufgerufen wurde. Die bedeutet, dass eine asynchrone Funktion aufgerufen werden muss.</li> <li>2. Ein neuer Thread wird gestartet, der nun parallel zu WinAC RTX läuft.</li> <li>3. Der Thread sendet eine Windows Nachricht an die Kamera Applikation</li> <li>4. Im Thread wird auf ein Windows Event gewartet, das von der Kamera Applikation kommt. Dieser zeigt das erfolgreiche Umschalten des Modus' an.</li> <li>5. Der Thread ruft nun den OB 52 in WinAC RTX auf und teilt dem S7 Programm auf diese Weise mit, dass das Kommando erfolgreich ausgeführt wurde.</li> </ol>
<p>Kamera Applikation</p>	<ol style="list-style-type: none"> <li>1. Die eingehende Windows Nachricht führt zum Aufruf einer zugehörigen Funktion</li> <li>2. In der Funktion wird der Modus der Kamera Applikation gesetzt. Unter Umständen muss dazu ein Kommando an die Kamera gesendet werden.</li> <li>3. Es wird ein Event gesetzt, das der ODK Applikation die Ausführung des Kommandos signalisiert</li> </ol>

Copyright © Siemens AG 2005 All rights reserved  
21572937\_WinAC\_TO\_v11\_DOKU\_d.doc

## Darstellung des Datenflusses

Folgendes Bild zeigt den Datenfluss zum Auslesen des Kamera Status.

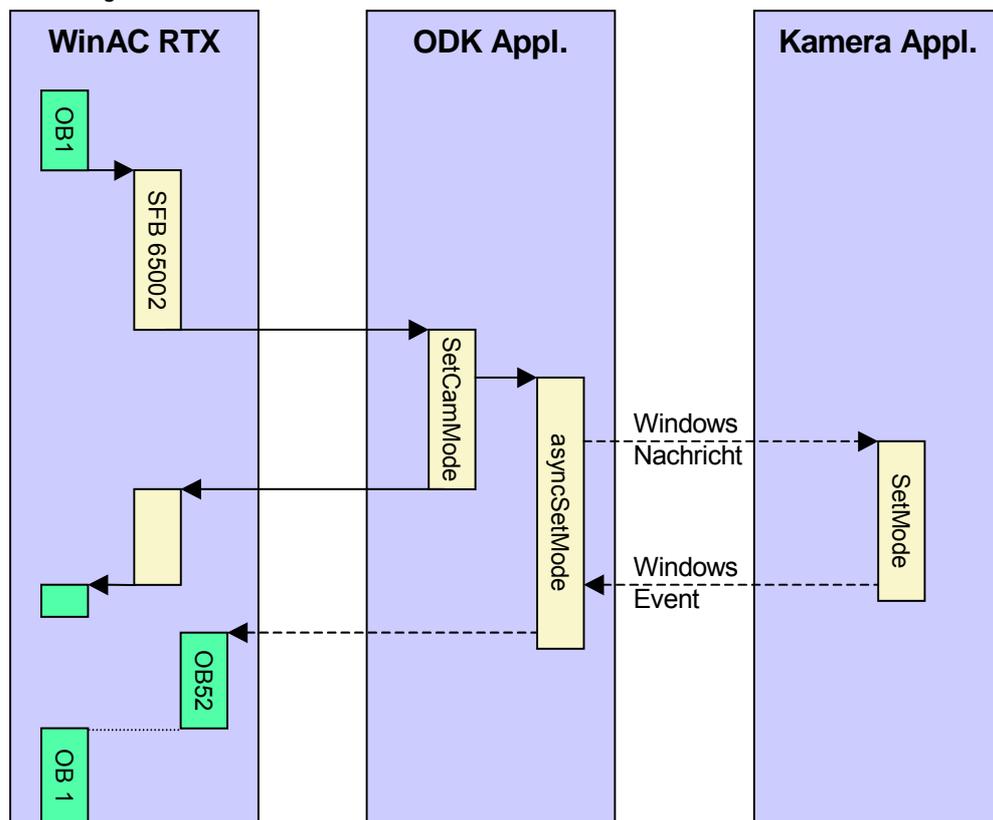
Abbildung 7-3



## Darstellung des Ablaufs als Sequenzdiagramm

Das Bild zeigt den Ablauf eines asynchronen Aufrufs.

Abbildung 7-4



Copyright © Siemens AG 2005 All rights reserved  
21572937\_WinAC\_TO\_v11\_DOKU\_d.doc

## Implementierung der Funktion

Hier sehen Sie die relevanten Codestellen für die drei Funktionskomponenten.

Tabelle 7-7

Funktionskomponente	Code
WinAC RTX	<p><u>Aufruf der asynchronen Funktion</u></p> <pre>// Create a unique command ID go:  L   "CommandID"      +   1      T   "CommandID"      T   "SFB 65002 DB".ODK_Input.CommandID  // Call ODK CALL "EXEC_COM" , DB65002     OBJHandle :="SFB 65002 DB".Status     Command   :=DW#16#1     InputData :="SFB 65002 DB".ODK_Input     OutputData:=P#DB652.DBX266.0 BYTE 4     STATUS    :="SFB 65002 DB".RetVal</pre> <ul style="list-style-type: none"> <li>• CommandID ist die eindeutige Kennung des Kommandos</li> <li>• Command spezifiziert die auszuführende Funktion in der ODK Applikation (siehe unten)</li> <li>• Input Data sind Eingabedaten für die ODK Applikation</li> <li>• Output Data sind die Ausgabedaten der ODK Applikation</li> </ul> <p><u>Code des OB 52</u></p> <pre>// Command = SET_MODE_ACK?? L   #Command L   W#16#20 ==I JCN jer  L   #DW2 T   "AckCommandID"  SET S   "AcknowledgeBit"  L   0 T   "OB52ErrCode" JU  go  // Command = Unknown Command jer: L   #Command      T   "OB52ErrCode"</pre>

	<ul style="list-style-type: none"> <li>• Über die Lokaldaten des OB 52 können die Parameter, die in der ODK Applikation spezifiziert wurden, abgefragt werden. So wurde ganz willkürlich festgelegt, dass der Wert 20h im Lokaldatum „Command“ die Funktion SET_MODE_ACK bedeutet.</li> <li>• Andere Lokaldaten werden in Merkerworte transferiert und stehen zur Auswertung durch andere S7-Programmteile zur Verfügung.</li> <li>• Bei einem unbekanntem Kommando wird in einem Merkerwort ein Fehlercode hinterlegt.</li> </ul>
<p>Kamera Applikation</p>	<p><u>Nachrichten Sprungverteiler:</u></p> <pre>BEGIN_MESSAGE_MAP(CVisionSensor_723Dlg, CDialog)     // ODK messages     ON_MESSAGE(WM_SET_MODE, SetMode) END_MESSAGE_MAP()</pre> <ul style="list-style-type: none"> <li>• Legt fest, dass die Windows Nachricht WM_SET_MODE zum Aufruf der Funktion SetMode führt</li> </ul> <pre>dwMutexState = WaitForSingleObject                 (m_hSharedMemMutex, 800);  // Set the current status to shared memory if (dwMutexState == WAIT_OBJECT_0) {     SharedMemStruct *smsMode;     smsMode = (SharedMemStruct *) m_pMapPtr;     smsMode-&gt;iBottleType = m_iCurrentBottleType;     smsMode-&gt;iLastTestResult = m_iTestResult;     smsMode-&gt;iMode = m_iCurrentMode; } ReleaseMutex (m_hSharedMemMutex);</pre> <ul style="list-style-type: none"> <li>• WaitForSingleObjekt () belegt den Mutex</li> <li>• smsMode enthält einen Zeiger auf den Shared Memory, in den Daten geschrieben werden</li> <li>• ReleaseMutex () gibt den Mutex wieder frei</li> </ul>

<p>ODK Applikation</p>	<p><u>Funktion Execute:</u></p> <pre> switch(command) {     case 1:         // Example for an asynchronous         thread             retVal = SetCamMode(Input, Output,                                 hBottleCheckApp);             break;     case 2:         // Code for getting Cam Mode             break; } </pre> <ul style="list-style-type: none"> <li>• Der gelb markierte Wert spezifiziert die auszuführende Funktion und kommt vom SFB 65002 Aufruf (siehe oben)</li> </ul> <p><u>Funktion SetCamMode:</u></p> <pre> asyncSetMode *SetModeEvent = new (asyncSetMode);  Input.ODK_ReadS7INT (0, SetModeEvent-&gt;                     m_iCommandID); Input.ODK_ReadS7INT (2, SetModeEvent-&gt;m_iMode); Input.ODK_ReadS7INT (4, SetModeEvent-&gt;m_iData);  // Pass the window handle SetModeEvent-&gt;m_hBottleCheckWin =                     hBottleCheckApp; SetModeEvent-&gt;m_hSvcHandle = g_ServiceHandle;  // Create the asynchronous thread by sending an event g_Processor-&gt;ScheduleEvent (SetModeEvent); </pre> <ul style="list-style-type: none"> <li>• ODK_Read... Funktionen lesen die Eingangsparameter vom SFB 65002 Aufruf</li> <li>• SetModeEvent ist eine Klasse, deren Instanz als paralleler Thread ausgeführt werden kann.</li> <li>• ScheduleEvent startet die Ausführung des Threads</li> </ul>
----------------------------	---

## Funktion asyncSetMode::Execute()

```
SendMessage (m_hBottleCheckWin, WM_SET_MODE,
             (WPARAM) m_iMode,
             (LPARAM) m_iData);

// Wait for event that signals acknowledge
if (m_hSetModeEvent != NULL)
// return after timeout = 1 ms
    dwWaitResult = WaitForSingleObject
                  (m_hSetModeEvent, 1);
if (dwWaitResult == WAIT_OBJECT_0)
// Signaled, set mode was successful
// Notify WinAC, that SetMode was successful
  by calling OB 52
    ODK_ScheduleOB (m_hSvcHandle, 0x11, 0x41,
                   0xFE, 52, 0xC4, 0x59, SET_MODE_ACK,
                   m_iCommandID);

ResetEvent (m_hSetModeEvent);
```

- SendMessage sendet die Windows Nachricht an die Kamera Applikation
- WaitForSingleObject wartet auf einen Windows Event von der Kamera Applikation; Dabei wird ein Timeout mit angegeben
- ODK\_ScheduleOB bewirkt, dass in WinAC RTX der spezifizierte OB (hier: OB 52) ausgeführt wird
- ResetEvent setzt den Event wieder zurück

## Zusammenfassung

Hier konnten Sie das Zusammenspiel verschiedener Windows und ODK Mechanismen sehen:

- Asynchroner Aufruf des ODK in WinAC RTX
- Datentransfer von und zur ODK Applikation
- Windows Nachrichten
- Windows Events
- Aufruf von OBs innerhalb von WinAC RTX

## 7.4 Beispiel für eine ODK Funktion, die im Hintergrund läuft

### Überblick

Für manche Aufgabenstellungen ist es notwendig, fortlaufend Prüfungen vorzunehmen, die unabhängig neben den anderen Automatisierungsaufgaben ablaufen (Polling). Dies kann mit kontinuierlichen Funktionen des WinAC ODK CCX gelöst werden.

Kontinuierliche Funktionen werden immer dann angewendet, wenn Aufgaben zyklisch abgearbeitet werden müssen, ähnlich dem OB1. Dazu wird ein eigener Thread erzeugt.

## Beschreibung der Funktion

Das konkrete Beispiel überwacht eine Zählvariable, die im Shared Memory liegt. Der Zähler repräsentiert die Anzahl der geprüften Flaschen. Immer wenn dieser Zähler sich erhöht, soll der Wert in eine Variable in der WinAC RTX geschrieben werden.

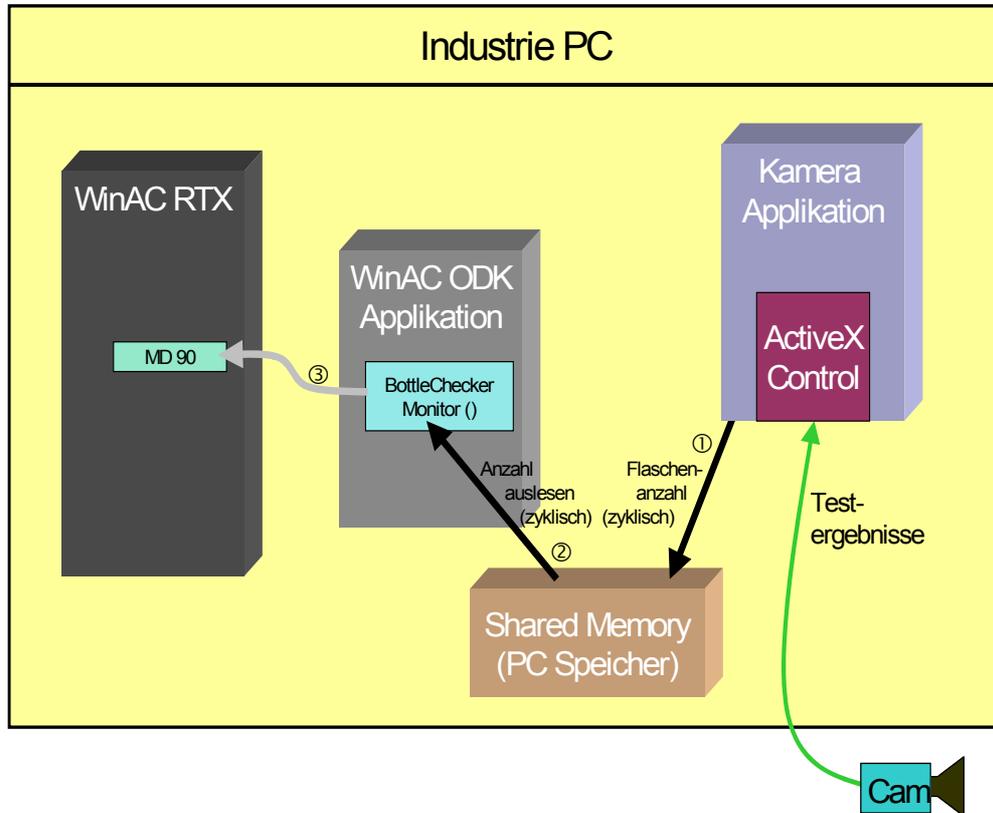
Tabelle 7-8

Funktionskomponente	Funktionalität
WinAC RTX	In WinAC RTX muss hierfür kein Code geschrieben werden.
Kamera Applikation	Die Kamera Applikation schreibt nach jeder erfolgten Prüfung die Anzahl der geprüften Flaschen in das Shared Memory
ODK Applikation	<ol style="list-style-type: none"><li>1. In der kontinuierlichen Funktion (einer Task, die ähnlich dem OB1 zyklisch in der ODK Applikation abläuft), wird eine Zählvariable im Shared Memory überwacht (Polling).</li><li>2. Bei Änderung der Zählvariablen wird die Funktion ODK_WriteData (...) aufgerufen, die den Wert der Zählvariablen direkt in ein Merkerdoppelwort von WinAC RTX schreibt.</li></ol>

## Darstellung des Datenflusses

Folgendes Bild zeigt den Datenfluss für das Schreiben der Anzahl der geprüften Flaschen in WinAC RTX.

Abbildung 7-5

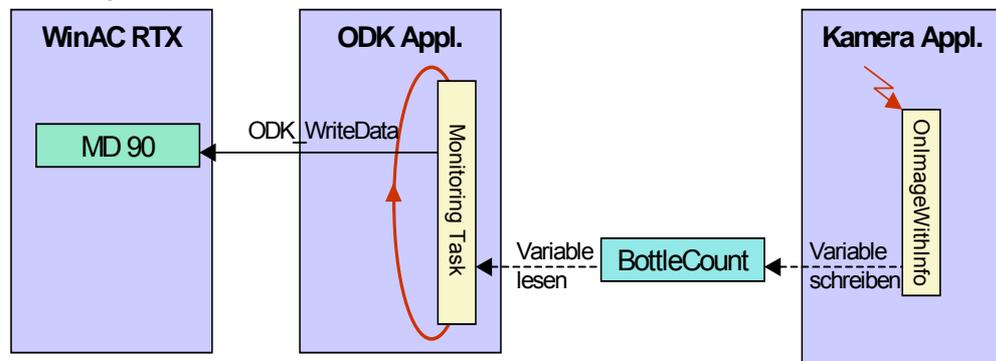


Copyright © Siemens AG 2005 All rights reserved  
21572937\_WinAC\_TO\_v11\_DOKU\_d.doc

## Darstellung des Ablaufs als Sequenzdiagramm

Das Bild zeigt den Ablauf für das Schreiben der Anzahl der geprüften Flaschen in WinAC RTX.

Abbildung 7-6



## Implementierung der Funktion

Hier sehen Sie die relevanten Codestellen für die drei Funktionskomponenten.

Tabelle 7-9

Funktionskomponente	Code
Kamera Applikation	<p><u>Callback Funktion der Kamera:</u></p> <pre data-bbox="584 629 1358 689">// Write bottle number into shared memory pSharedMem-&gt;ulBottleNo = (unsigned long)dImageNo;</pre> <ul data-bbox="584 730 1358 920" style="list-style-type: none"><li>• Die Kamera hat einen internen Zähler, in dem sie die Anzahl der geprüften Flaschen mitzählt. Nach jeder vollendeten Prüfung wird dieser Wert über eine Callback-Funktion in die Variable dImageNo geschrieben</li><li>• Die Anzahl wird mit obiger Funktion in den Shared Memory (pSharedMem) geschrieben</li></ul>

ODK  
Applikation

### Funktion BottleCheckerMonitor::Execute()

```

newBottleNo = (unsigned long) pSharedMem->
                                     ulBottleNo;
if (newBottleNo != m_ulBottleNo)
// New bottle count! Send to WinAC
{
    ODK_DATA_STRUCT      dsStruct;
    ODK_BIT32            WriteVal;
    ODK_RESULT           ret;

    dsStruct.dataType = ODK_DATA_TYPE_DWORD;
    // Access double word
    dsStruct.quantity = 1;
    // Write 1 double word
    dsStruct.dbNumber = 0;
    // DB Number, not relevant
    dsStruct.memoryArea = ODK_MEM_AREA_M;
    // Access flag area
    dsStruct.areaOffset = 90;
    // Write MD 90 in WinAC
    dsStruct.bitNumber = 0;
    // Bit number, not relevant
    dsStruct.pBuff = (unsigned char*)
                    &WriteVal;
    // Set pointer to data
    dsStruct.maxSize = sizeof(WriteVal);
    // Set max size of data = 1 DWORD
    dsStruct.status = 0;
    // Return Value: Status

    // Use S7 access methods for byte swapping
    CWinACReadWriteData readSwap
        (dsStruct.maxSize, dsStruct.pBuff);

    WriteVal = newBottleNo;

    // Use S7 access methods for byte swapping
    ret = readSwap.ODK_WriteS7DWORD
        (0, WriteVal);
    // Write data
    ret = ODK_WriteData(m_WinACSvc, 1,
        &dsStruct, true);
}

```

- Der Anzahl geprüfter Flaschen wird aus dem Shared Memory in die Variable newBottleNo gelesen
- Wenn sich die Anzahl verändert hat, dann muss dies an WinAC RTX gemeldet werden
- Dazu wird eine Struktur ausgefüllt, die Informationen über die zu schreibende Variable in WinAC RTX beinhaltet. Obige Struktur wird das Merkerdoppelwort 90 initialisiert

	<ul style="list-style-type: none"><li>• Da die Anordnung der Bytes in Doppelworten im PC anders ist als in einer SIMATIC Steuerung, müssen diese erst mit Hilfe der Funktion ODS_WriteS7DWORD in die richtige Reihenfolge gebracht werden</li><li>• ODK_WriteData schreibt die Daten direkt in WinAC RTX</li></ul>
WinAC RTX	<ul style="list-style-type: none"><li>• In WinAC RTX ist kein Code notwendig. Es wird direkt in die Variable MD 90 geschrieben</li></ul>

## Zusammenfassung

Mit Hilfe der Funktion ODK\_WriteData können Daten von Windows-Applikationen direkt in die verschiedenen Speicherbereiche von WinAC RTX Steuerung geschrieben werden. Es existieren auch Funktionen für das zyklische Schreiben und Lesen sowie für einfaches Lesen aus WinAC RTX.

## 8 Anpassungen der Programme

### Einleitung

In diesem Kapitel erfahren Sie, wie sie die einzelnen Elemente dieser Applikation an Ihre individuellen Bedürfnisse anpassen können. Vorliegende Applikation stellt nur exemplarisch ein Beispiel dar, wie Windows Applikation in WinAC RTX Steuerprogramme eingebunden werden können.

### Inhalt dieses Kapitels

Tabelle 8-1

Kap.	Titel	Seite
8	Anpassungen der Programme	79
8.1	Anpassungen im S7-Programm	79
8.2	Anpassung der C++ Applikationen	80

### 8.1 Anpassungen im S7-Programm

#### Vorliegendes Beispiel

Das mit dieser Applikation mitgelieferte S7-Programm zeigt lediglich drei wichtige Punkte:

- Initialisierung des ODK Programms
- Aufruf von Funktionen im ODK Programm
- Implementierung des OB 52, der für Aufrufe aus der ODK Applikation zur Verfügung steht

Das S7-Programm enthält keinerlei Steuerungsfunktionalität für die direkte Ansteuerung von externer SIMATIC Peripherie.

#### Anpassungen für individuelle Applikationen

Für Ihre individuelle Applikation implementieren Sie im S7-Programm die Steuerungsfunktionalität für Ihre Anlage. Externe Peripherie kann über PROFIBUS CPs oder über die integrierte Ethernet-Schnittstelle angesprochen werden. Hierfür existieren im Applikationsportal von A&D Service & Support auch die oben aufgeführten Beispiele zum Thema PC-basierte Automatisierung .

Nach den oben gezeigten Codesequenzbeispielen können Sie Ihre individuellen (synchronen, asynchronen oder kontinuierlichen) Funktionen in der ODK Applikation aufrufen.

## 8.2 Anpassung der C++ Applikationen

### Varianten für Ihre Applikation

In den C++ Applikationen implementieren Sie Windows-Funktionalitäten, die in Ihr Steuerungsprogramm mit eingebunden werden sollen. Dabei bieten sich Ihnen zwei Möglichkeiten:

- Implementierung der Funktionalität in einer eigenen Windows Applikation, die mit der ODK Applikation kommuniziert
- Implementierung der Funktionalität direkt in der ODK Applikation

### 8.2.1 Anpassungen in der Windows Applikation

#### Vorliegendes Beispiel

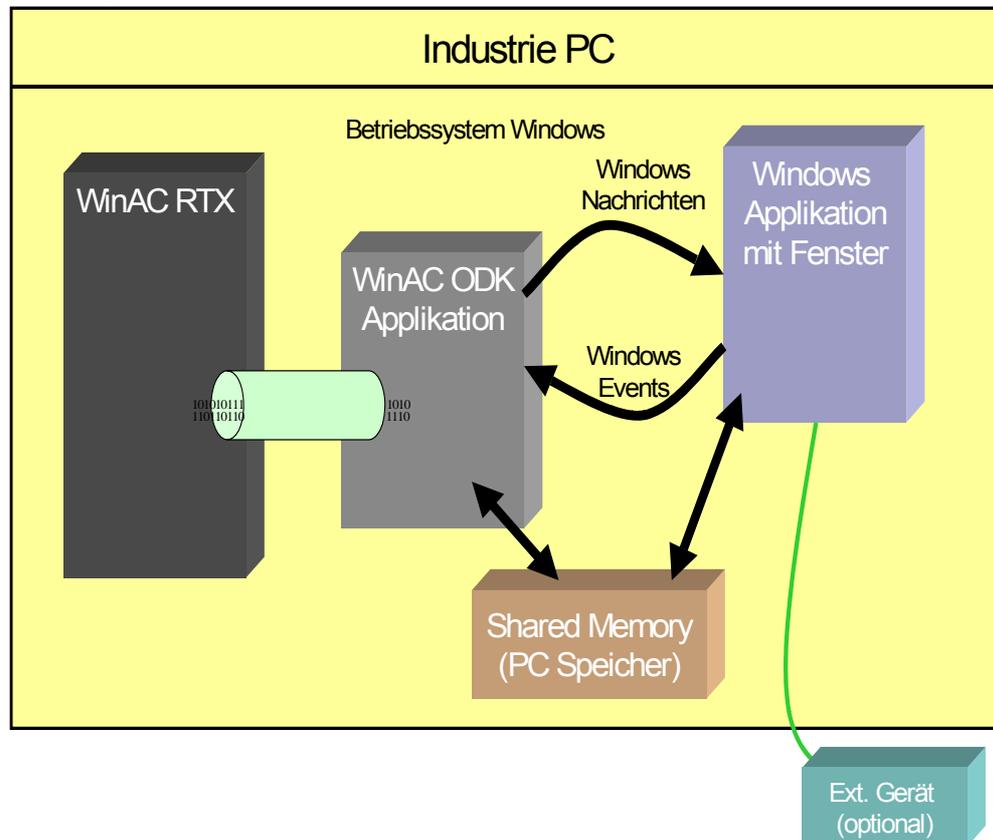
Die mit dieser Applikation mitgelieferte Kamera Applikation soll lediglich verdeutlichen, wie eine Windows Applikation mit der ODK Applikation kommunizieren kann. Folgende Mechanismen werden verwendet:

- Windows Nachrichten
- Shared Memory (Gemeinsam genutzter Hauptspeicher)
- Mutexes (Objekt zur Belegung von Windows Ressourcen)
- Events (Benachrichtigungsobjekt)

Das beiliegende Programm ist sehr speziell auf die am Anfang definierte Aufgabenstellung ausgerichtet und passt vermutlich nicht zu Ihren individuellen Anforderungen. Es dient lediglich als Beispiel für die Kommunikationsmechanismen. Sie können das Programm dazu nutzen, um zu sehen, wie solche Mechanismen in Windows implementiert werden können.

## Grafische Darstellung einer allgemeinen Lösungsvariante

Abbildung 8-1



### Anpassungen für Ihre individuelle Applikation

Sie werden in der Regel das Windows Programm komplett neu erstellen. Dabei kann natürlich bestehender Quellcode verwendet werden. Sie müssen ihn jedoch um die oben genannten Kommunikations- und Synchronisationsmechanismen erweitern, über die Sie die Verbindung zur ODK Applikation herstellen.

### 8.2.2 Anpassungen in der ODK Applikation

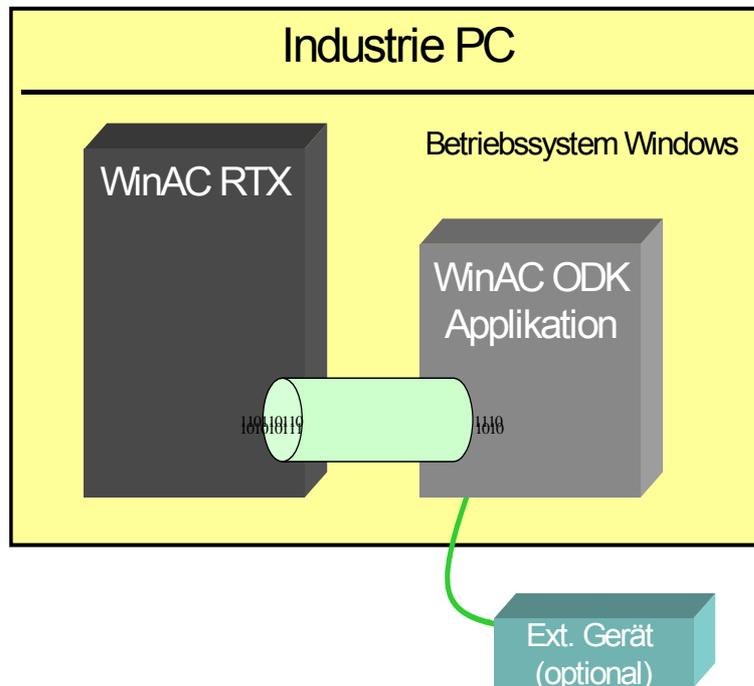
#### Vorliegendes Beispiel

Die mit dieser Applikation mitgelieferte ODK Applikation dient lediglich als Überträger von Daten von WinAC RTX zur Kamera Applikation und umgekehrt. Es enthält keine eigene Steuerungsfunktionalität. Gezeigt werden:

- Synchroner Funktionen der ODK Applikation
- Asynchroner Funktionen der ODK Applikation
- Kontinuierliche Funktionen in der ODK Applikation

## Grafische Darstellung einer allgemeinen Lösungsvariante

Abbildung 8-2



### Anpassungen für Ihre individuelle Applikation

Sie können Ihre Funktionalität auch direkt in der ODK Applikation programmieren. Dies hat den Vorteil, dass Sie die Kommunikation und Synchronisation mit der Windows Applikation nicht berücksichtigen müssen. Dies ist auch dann empfehlenswert, wenn große Datenmengen schnell verarbeitet werden müssen.

Diese Lösung empfiehlt sich nicht, wenn die Applikation ein sichtbares Windows Fenster besitzen soll. Die Implementierung von Fenstern im WinAC ODK ist aufwändiger.

Beispiele, die auf diese Art sehr geschickt realisiert werden können sind:

- Regler oder sonstige mathematische Algorithmen in C++
- Anbindung von Datenbanken
- Einbinden von Treibern für PC-Erweiterungskarten von Fremdherstellern
- Versenden von E-Mails

Der ODK Wizard unterstützt Sie bei der Erstellung des Grundgerüsts der Applikation.

## Teil D: Anhang

### 9 Literaturangaben

#### Literaturangaben

Diese Liste ist keinesfalls vollständig und spiegelt nur eine Auswahl an geeigneter Literatur wieder. Die meisten Handbücher finden Sie, wenn Sie das entsprechende Produkt installiert haben, unter:

Start → Simatic → Dokumentation → <Sprache>

Den Produktsupport finden Sie im Internet unter:

<http://support.automation.siemens.com>

(Geben Sie dort die Beitrags ID in das Suchfeld ein).

Tabelle 9-1

Nr.	Themengebiet	Titel
1	Beschreibung der Funktionen und der Bedienung von WinAC RTX V4.1. Zu finden auf der WinAC RTX V4.1 CD oder direkt vom Bedienpanel aus: Menü Hilfe → Hilfe zum Controller.	SIMATIC WinAC RTX V4.1
2	Beschreibung bzw. Informationen zu: <ul style="list-style-type: none"> <li>Allgemeine Informationen über die PC-Werkzeuge</li> <li>Funktionen von NCM PC</li> </ul> Im Produktsupport unter der Beitrags ID: <a href="#">13542666</a> zu finden.	SIMATIC NET PC-Stationen in Betrieb nehmen - Anleitung und Schnelleinstieg für SIMATIC NCM PC / STEP 7 ab Version V5.2
3	Handbuch zur Industriellen Kommunikation auf PG/PC mit SIMATIC NET. Im Produktsupport unter der Beitrags ID: <a href="#">16923753</a> zu finden.	SIMATIC NET – Industrielle Kommunikation mit PG/PC
4	Vollständiger Überblick über die in den Betriebssystemen der CPUs der S7-300 und S7-400 enthaltenen Organisationsbausteine (OB), Systemfunktionen (SFC), System- und Standardfunktionsbausteine (SFB) sowie IEC-Funktionen. Im Produktsupport unter der Beitrags ID: <a href="#">1214574</a> zu finden.	Systemsoftware für S7-300/400 System- und Standardfunktionen
5	Handbuch für SIMATIC Rack PC IL40S Im Produktsupport unter der Beitrags ID: <a href="#">15317654</a> zu finden.	SIMATIC Rack PC IL40S Handbuch

**Hinweis**

Falls die Beiträge nach dem Klicken auf obige Links nicht sofort angezeigt werden, so klicken Sie auf „Aktualisieren“ in Ihrem Browser.