MQTT-Client für SIMATIC S7-1500 und S7-1200

Bausteine für S7-1500 und S7-1200

https://support.industry.siemens.com/cs/ww/de/view/109748872

Siemens Industry Online Support

Rechtliche Hinweise

Nutzung der Anwendungsbeispiele

In den Anwendungsbeispielen wird die Lösung von Automatisierungsaufgaben im Zusammenspiel mehrerer Komponenten in Form von Text, Grafiken und/oder Software-Bausteinen beispielhaft dargestellt. Die Anwendungsbeispiele sind ein kostenloser Service der Siemens AG und/oder einer Tochtergesellschaft der Siemens AG („Siemens“). Sie sind unverbindlich und erheben keinen Anspruch auf Vollständigkeit und Funktionsfähigkeit hinsichtlich Konfiguration und Ausstattung. Die Anwendungsbeispiele stellen keine kundenspezifischen Lösungen dar, sondern bieten lediglich Hilfestellung bei typischen Aufgabenstellungen. Sie sind selbst für den sachgemäßen und sicheren Betrieb der Produkte innerhalb der geltenden Vorschriften verantwortlich und müssen dazu die Funktion des jeweiligen Anwendungsbeispiels überprüfen und auf Ihre Anlage individuell anpassen.
Sie erhalten von Siemens das nicht ausschließliche, nicht unterlizenzierbare und nicht übertragbare Recht, die Anwendungsbeispiele durch fachlich geschultes Personal zu nutzen. Jede Änderung an den Anwendungsbeispielen erfolgt auf Ihre Verantwortung. Die Weitergabe an Dritte oder Vervielfältigung der Anwendungsbeispiele oder von Auszügen daraus ist nur in Kombination mit Ihren eigenen Produkten gestattet. Die Anwendungsbeispiele unterliegen nicht zwingend den üblichen Tests und Qualitätsprüfungen eines kostenpflichtigen Produkts, können Funktions- und Leistungsmängel enthalten und mit Fehlern behaftet sein. Sie sind verpflichtet, die Nutzung so zu gestalten, dass eventuelle Fehlfunktionen nicht zu Sachschäden oder der Verletzung von Personen führen.

Haftungsausschluss

Siemens schließt seine Haftung, gleich aus welchem Rechtsgrund, insbesondere für die Verwendbarkeit, Verfügbarkeit, Vollständigkeit und Mangelfreiheit der Anwendungsbeispiele, sowie dazugehöriger Hinweise, Projektierungs- und Leistungsdaten und dadurch verursachte Schäden aus. Dies gilt nicht, soweit Siemens zwingend haftet, z.B. nach dem Produkthaftungsgesetz, in Fällen des Vorsatzes, der groben Fahrlässigkeit, wegen der schuldhaften Verletzung des Lebens, des Körpers oder der Gesundheit, bei Nichteinhaltung einer übernommenen Garantie, wegen des arglistigen Verschweigens eines Mangels oder wegen der schuldhaften Verletzung wesentlicher Vertragspflichten. Der Schadensersatzanspruch für die Verletzung wesentlicher Vertragspflichten ist jedoch auf den vertragstypischen, vorhersehbaren Schaden begrenzt, soweit nicht Vorsatz oder grobe Fahrlässigkeit vorliegen oder wegen der Verletzung des Lebens, des Körpers oder der Gesundheit gehaftet wird. Eine Änderung der Beweislast zu Ihrem Nachteil ist mit den vorstehenden Regelungen nicht verbunden. Von in diesem Zusammenhang bestehenden oder entstehenden Ansprüchen Dritter stellen Sie Siemens frei, soweit Siemens nicht gesetzlich zwingend haftet.
Durch Nutzung der Anwendungsbeispiele erkennen Sie an, dass Siemens über die beschriebene Haftungsregelung hinaus nicht für etwaige Schäden haftbar gemacht werden kann.

Weitere Hinweise

Siemens behält sich das Recht vor, Änderungen an den Anwendungsbeispielen jederzeit ohne Ankündigung durchzuführen. Bei Abweichungen zwischen den Vorschlägen in den Anwendungsbeispielen und anderen Siemens Publikationen, wie z. B. Katalogen, hat der Inhalt der anderen Dokumentation Vorrang.
Ergänzend gelten die Siemens Nutzungsbedingungen (https://support.industry.siemens.com).

Securityhinweise

Siemens bietet Produkte und Lösungen mit Industrial Security-Funktionen an, die den sicheren Betrieb von Anlagen, Systemen, Maschinen und Netzwerken unterstützen.
Um Anlagen, Systeme, Maschinen und Netzwerke gegen Cyber-Bedrohungen zu sichern, ist es erforderlich, ein ganzheitliches Industrial Security-Konzept zu implementieren (und kontinuierlich aufrechtzuerhalten), das dem aktuellen Stand der Technik entspricht. Die Produkte und Lösungen von Siemens formen einen Bestandteil eines solchen Konzepts.
Die Kunden sind dafür verantwortlich, unbefugten Zugriff auf ihre Anlagen, Systeme, Maschinen und Netzwerke zu verhindern. Diese Systeme, Maschinen und Komponenten sollten nur mit dem Unternehmensnetzwerk oder dem Internet verbunden werden, wenn und soweit dies notwendig ist und nur wenn entsprechende Schutzmaßnahmen (z.B. Firewalls und/oder Netzwerksegmentierung) ergriffen wurden.
Weiterführende Informationen zu möglichen Schutzmaßnahmen im Bereich Industrial Security finden Sie unter https://www.siemens.com/industrialsecurity.
Die Produkte und Lösungen von Siemens werden ständig weiterentwickelt, um sie noch sicherer zu machen. Siemens empfiehlt ausdrücklich, Produkt-Updates anzuwenden, sobald sie zur Verfügung stehen und immer nur die aktuellen Produktversionen zu verwenden. Die Verwendung veralteter oder nicht mehr unterstützter Versionen kann das Risiko von Cyber-Bedrohungen erhöhen.
Um stets über Produkt-Updates informiert zu sein, abonnieren Sie den Siemens Industrial Security RSS Feed unter https://www.siemens.com/cert.

Inhaltsverzeichnis

1 Einführung

1.1 Überblick

Motivation

Die Digitalisierung hat einen großen Einfluss auf die Wirtschaft und die Gesellschaft und schreitet unaufhaltsam voran. Das "Internet of Things" (Internet der Dinge, Kurzform: IoT) ist einer der Haupttreiber der Digitalisierung. Der Begriff "Internet of Things" steht synonym für eine der größten aktuellen Veränderungsdynamiken: die zunehmende Vernetzung und Automatisierung von Geräten, Maschinen und Produkten.

Das Protokoll "Message Queue Telemetry Transport" (Kurzform: MQTT) wird im "Internet of Things" als Kommunikationsprotokoll eingesetzt. Durch seinen leichtgewichtigen Ansatz eröffnet es ganz neue Möglichkeiten in der Automatisierung.

Schlank und schnell: MQTT

Das MQTT ist ein einfach aufgebautes binäres Publish- und Subscribe-Protokoll auf TCP/IP-Ebene. Es eignet sich für den Nachrichtenaustausch zwischen Geräten mit geringer Funktionalität und für die Übertragung über unzuverlässige Netze mit geringer Bandbreite und hoher Latenz. Mit diesen Charakteristiken spielt MQTT eine wichtige Rolle für das IoT und in der M2M-Kommunikation.

Merkmale von MQTT

Das MQTT-Protokoll hebt sich durch folgende Merkmale hervor:

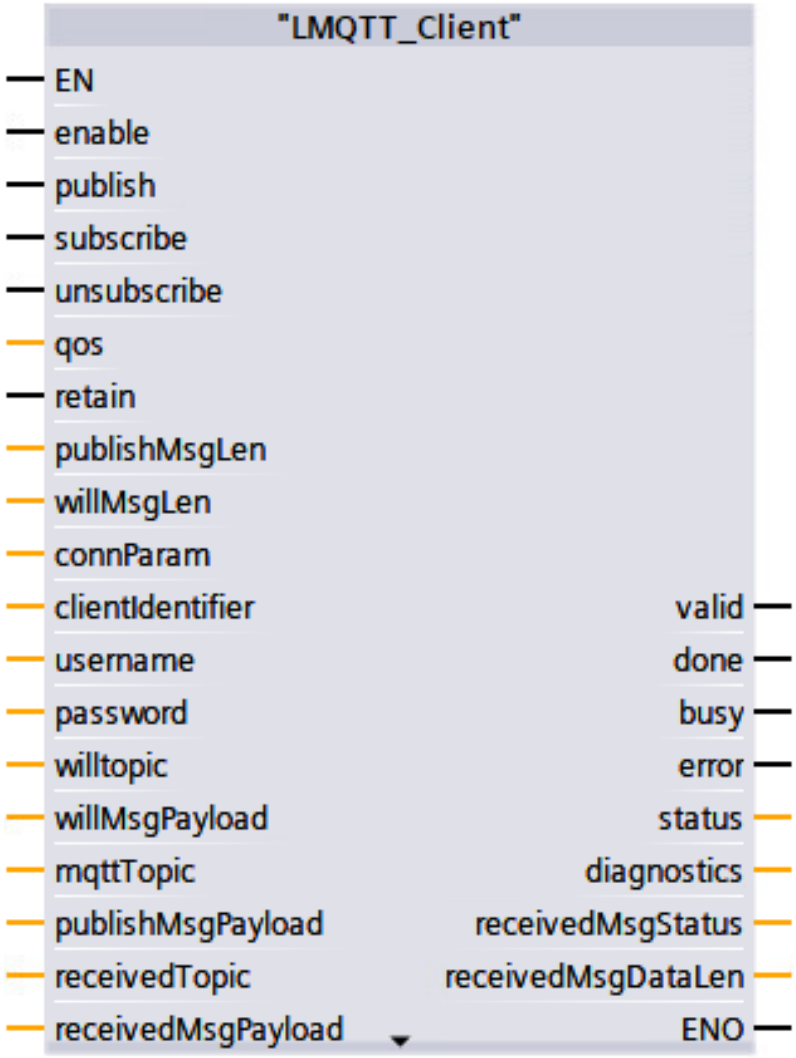
- Leichtgewichtiges Protokoll mit geringem Transport-Overhead
- Minimaler Bedarf an Netzwerk-Bandbreite durch Push-Mechanismus
- Funktion zum Wiederverbinden nach Abbruch der Verbindung
- Erneutes Ausliefern von Nachrichten nach Verbindungsabbruch
- Mechanismus zur Benachrichtigung von Interessenten nach einem unvorhergesehenen Verbindungsabbruch eines Clients
- Einfache Nutzung und Implementierung durch einen geringen Satz an Befehl-Kommandos
- Qualitätssicherung (QoS-Level) mit verschiedenen Zuverlässigkeitsstufen für die Nachrichten-Zustellung
- Optionale Verschlüsselung der Nachrichten mit TLS
- Authentifizierung der Publisher und der Subscriber mit Benutzername und Passwort

Applikative Umsetzung

Um das MQTT-Protokoll in eine SIMATIC S7-Steuerung zu implementieren, wird Ihnen mit der Bibliothek "LMQTT" eine adäquate Lösung angeboten.

Die Bibliothek "LMQTT" stellt Ihnen einen Funktionsbaustein für die SIMATIC S7-1500 und SIMATIC S7-1200 zur Verfügung. Der Funktionsbaustein "LMQTT_Client" integriert die MQTT-Client-Funktion und ermöglicht es Ihnen, MQTT-Nachrichten an einen Broker zu übermitteln (Publisher-Rolle) und Abonnements zu erstellen (Subscriber-Rolle). Die Kommunikation kann dabei über eine TLS-Verbindung gesichert werden.

Abbildung 1-1



Hinweis Der MQTT-Client unterstützt die MQTT-Protokollversion 3.1.1.

1.2 Funktionsweise

Schematische Darstellung

Die folgende Abbildung zeigt die wichtigsten Zusammenhänge zwischen den beteiligten Komponenten und Schritte, die für eine gesicherte MQTT-Kommunikation (MQTT over TLS) notwendig sind.

Abbildung 1-2

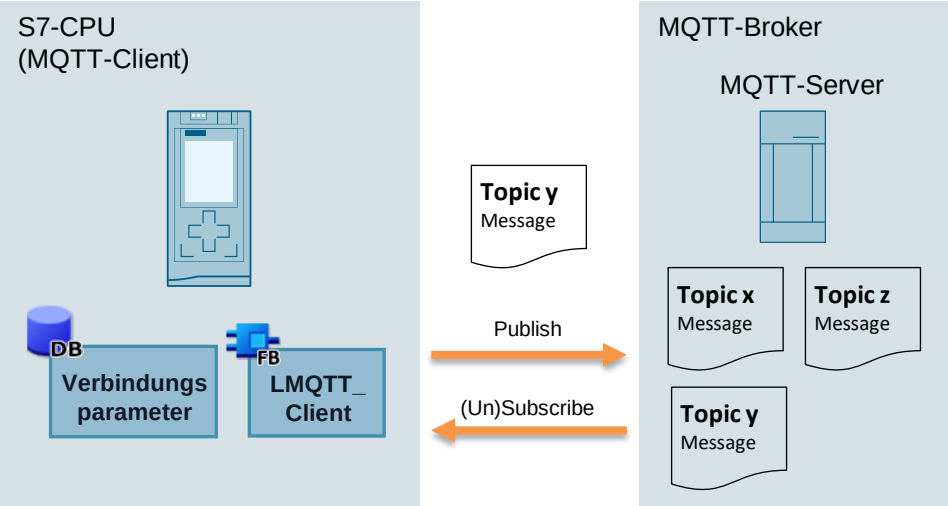


Tabelle 1-1

Schritt	Beschreibung
1	Ermitteln Sie das CA-Zertifikat des MQTT-Brokers.
2	Importieren Sie das Fremdzertifikat in STEP 7 (TIA Portal). Das Zertifikat befindet sich nun im globalen Zertifikatsmanager von STEP 7.
3	Das importierte Zertifikat müssen Sie der S7-CPU zuordnen. Um das Zertifikat als gültig anzuerkennen, muss die Uhrzeit der S7-CPU aktuell sein.
4	Der Funktionsbaustein "LMQTT_Client" übernimmt folgende Rollen: • Publisher, um MQTT-Nachrichten an den MQTT-Broker zu senden • Subscriber, um MQTT-Nachrichten zu abonnieren oder Abonnements zu beenden Die MQTT-Nachricht ist über eine gesicherte Verbindung (MQTT over TLS) verschlüsselt.

Hinweis Eine detailliertere Funktionsbeschreibung des Funktionsbausteins "LMQTT_Client" und Informationen zum MQTT-Protokoll finden Sie im Kapitel 3.

1.3 Verwendete Komponenten

Dieses Anwendungsbeispiel wurde mit diesen Hard- und Softwarekomponenten erstellt:

Tabelle 1-2

Komponente	Anzahl	Artikelnummer	Hinweis
CPU 1513-1 PN	1	6ES7513-1AL01-0AB0	• Alternativ können Sie auch eine andere S7 1500 CPU oder ET 200 CPU (ET 200SP, ET 200pro) verwenden. Für eine gesicherte MQTT-Kommunikation über TLS ist mindestens die Firmware-Version 2.0 erforderlich. • Alternativ können Sie auch eine S7-1200 CPU mit Firmware V4.4 oder höher verwenden. • Alternativ können Sie auch folgende Komponenten verwenden: - CP 1543-1 (6GK7543-1AX00-0XE0) mit Firmware V2.0 oder höher - CP 1545-1 (6GK7545-1GX00-0XE0) - CP 1543SP-1 (6GK7543-6WX00-0XE0) - CP 1243-1 (6GK7243-1BX30-0XE0) mit Firmware V3.2 oder höher - CP 1243-8 IRC (6GK7243-8RX30-0XE0) mit Firmware V3.2 oder höher - CP 1243-7 LTE (6GK7243-7KX30-0XE0 / 6GK7243-7SX30-0XE0) mit Firmware V3.2 oder höher
TIA Portal V16	-	DVD: 6ES7822-1AA06-0YA5 Download: 6ES7822-1AE06-0YA5	-
MQTT-Broker	-	-	Wenn Sie die Kommunikation verschlüsseln möchten, muss der MQTT-Broker das TLS-Verfahren unterstützen.

Dieses Anwendungsbeispiel besteht aus den folgenden Komponenten:

Tabelle 1-3

Komponente	Dateiname
Bibliothek "LMQTT"	109780503_Libraries_Comm_Controller_LIB_V1_0_0.zip
Dieses Dokument	109748872_MQTT_Client_DOKU_V3-0_de.pdf

Hinweis

Mit S7-1500 CPUs (Firmware V2.0 oder höher) oder S7-1200 CPUs (Firmware V4.4 oder höher) können Sie den MQTT-Broker über eine statische IP-Adresse oder einen Domänennamen ("Qualified Domain Name", kurz: QDN) erreichen, wenn Sie die Bibliothek "LMQTT" einsetzen.

2 Engineering

Hinweis

Das Engineering in diesem Kapitel fokussiert die MQTT-Client-Funktion, die dieses Anwendungsbeispiel realisiert.
Es wird vorausgesetzt, dass Sie den MQTT-Broker bereits installiert und konfiguriert haben.

2.1 Bausteinbeschreibung

Die Bausteinbeschreibung finden Sie in folgendem Beitrag:
https://support.industry.siemens.com/cs/ww/de/view/109780503

2.2 Projektierung

Das im Beitrag 109748872 mitgelieferte Anwendungsbeispiel zeigt die Projektierung.

2.2.1 TIA Portal Projekt erstellen

1. Erstellen Sie ein TIA Portal Projekt mit der CPU, die Sie für das Anwendungsbeispiel verwenden möchten.
2. Parametrieren Sie die Ethernet-Schnittstelle der CPU mit einer IP-Adresse, die im gleichen Subnetz wie der MQTT-Broker liegt.
3. Wenn Sie einen Cloud-Service wie AWS verwenden, dann parametrieren Sie einen Router und einen DNS-Sever.
4. Verbinden Sie die CPU und den MQTT-Broker über Ethernet.

Hinweis Für die gesicherte MQTT-Kommunikation über TLS benötigen Sie eine S7-1500 CPU ab Firmware-Version 2.0 oder eine S7-1200 CPU ab Firmware V4.4.

2.3 Integration des Funktionsbausteins in das Anwenderprogramm

Der Baustein "LMQTT_Client" und die benötigten Datentypen stehen Ihnen in der Bibliothek "LMQTT" zur Verfügung.

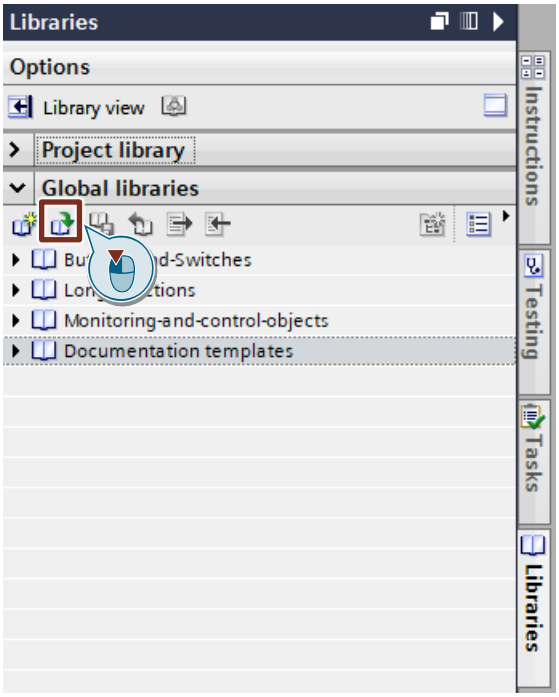
2.3.1 Globale Bibliothek "LMQTT" öffnen

Hinweis

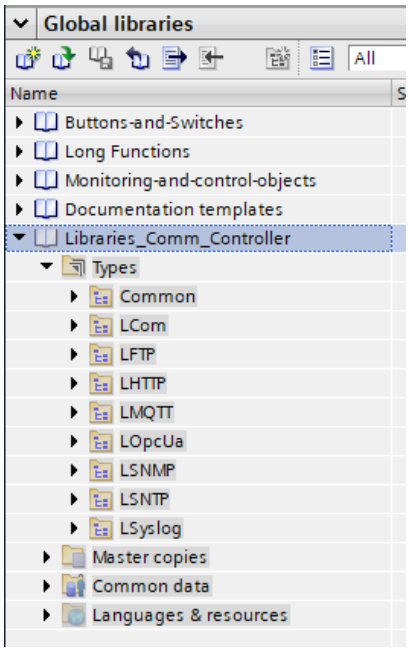
Für dieses Kapitel müssen Sie die Bibliothek "109780503_Libraries_Comm_Controller_LIB_V1_0_0.zip" heruntergeladen und in ein Verzeichnis Ihrer Wahl entpackt haben.

Die Bibliothek 109780503_Libraries_Comm_Controller_LIB_V1_0_0.zip finden Sie in folgendem Beitrag:
https://support.industry.siemens.com/cs/ww/de/view/109780503

1. Klicken Sie im TIA Portal Projekt auf die Task Card "Bibliotheken" ("Libraries") und öffnen Sie die Palette "Globale Bibliotheken" ("Global libraries").
2. Klicken Sie auf die Schaltfläche "Globale Bibliothek öffnen" ("Open global library").
   Der Dialog "Globale Bibliothek öffnen" ("Open global library") wird geöffnet.

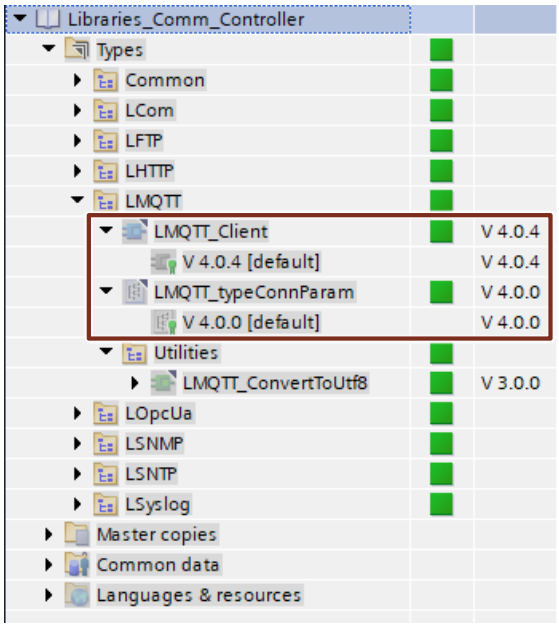


3. Gehen Sie zu dem Verzeichnis, in welches Sie die heruntergeladene Datei entpackt haben. Das Verzeichnis enthält die globable Bibliothek "Libraries_Comm_Controller.al17" ("al" steht für "application library"). Markieren Sie diese und klicken Sie auf die Schaltfläche "Öffnen" ("Open").
4. Die Bibliothek "Libraries_Comm_Controller" wird geöffnet und unter der Palette "Globale Bibliotheken" ("Global libraries") angezeigt.

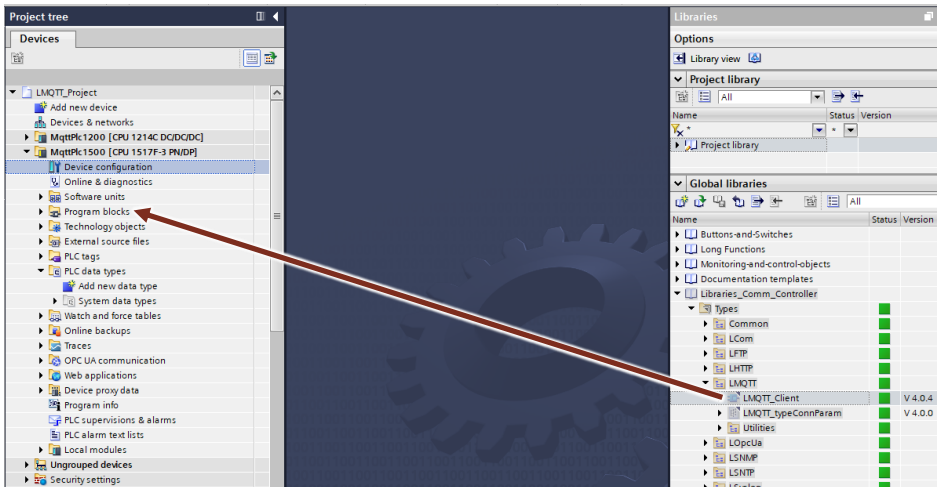


2.3.2 Funktionsbaustein und Datentyp ins Anwenderprogramm kopieren

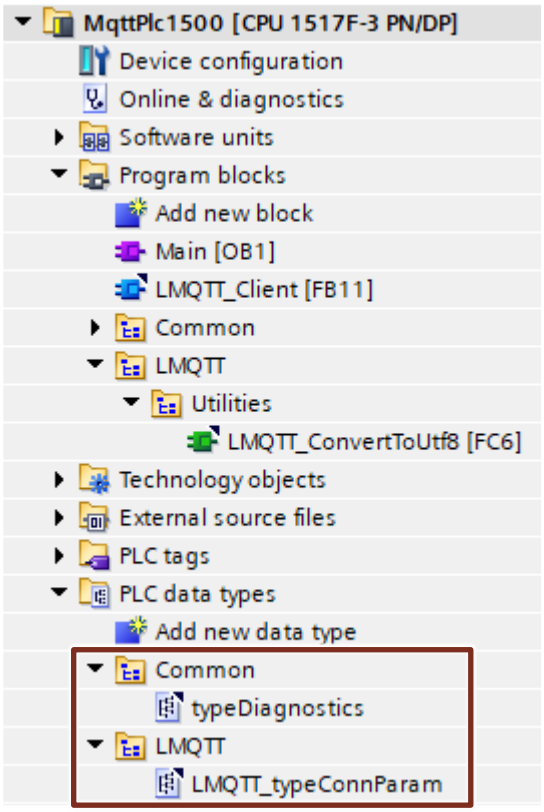
1. In der Bibliothek "Libraries_Comm_Controller" finden Sie den FB "LMQTT_Client" und die zugehörigen PLC-Datentypen unter "Typen > LMQTT" ("Types > LMQTT").



2. Fügen Sie den Funktionsbaustein für Ihre CPU per Drag & Drop in den Ordner "Programmbausteine" ("Program blocks") Ihres Geräts, z. B. S7-1500 CPU, ein.



3. Die Datentypen, die vom FB "LMQTT_Client" verwendet werden, werden automatisch in den Ordner "PLC-Datentypen" ("PLC data types") Ihres Geräts, z. B. S7-1500 CPU, eingefügt.

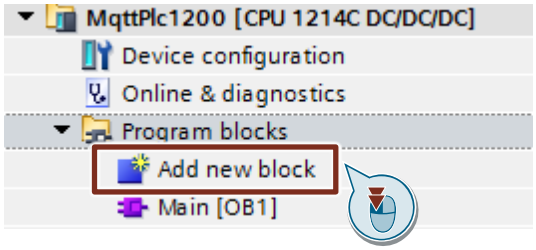


2.3.3 Globalen Datenbaustein erstellen

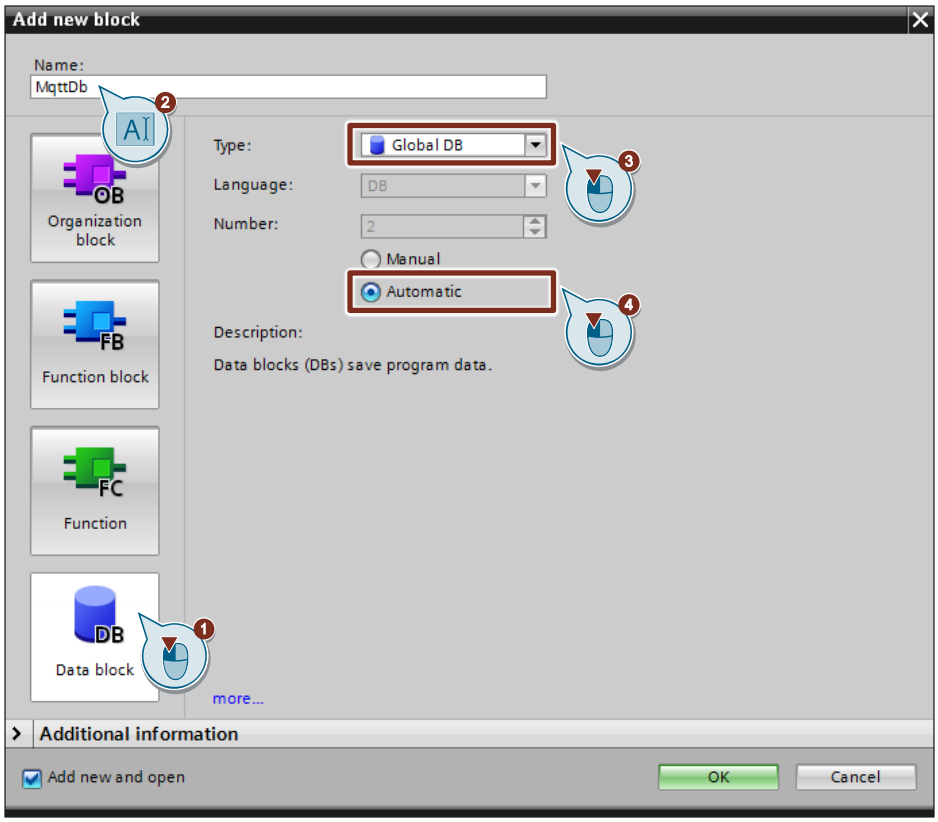
Dieses Kapitel zeigt Ihnen, wie Sie einen globalen Datenbaustein (DB) erstellen. Dieser DB dient zum Speichern der folgenden Daten:

- TCP-Verbindungsparameter
- MQTT-Verbindungsparameter
- Topic und Nachricht, die an den MQTT-Broker gesendet werden soll (publish)
- Empfangsdaten, d. h. Nachricht und Name des abonnierten Topic (subscribe)

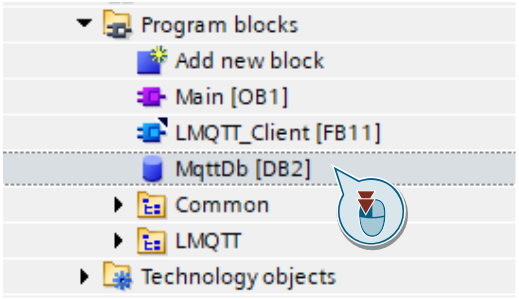
1. Navigieren Sie in der "Projektnavigation" ("Project tree") in den Geräteordner der CPU.
2. Öffnen Sie den Ordner "Programmbausteine" ("Program blocks") und doppelklicken Sie auf den Befehl "Neuen Baustein hinzufügen" ("Add new block").
   Der Dialog "Neuen Baustein hinzufügen" ("Add new block") öffnet sich.



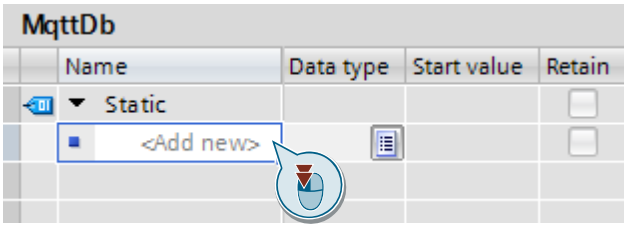
3. Nehmen Sie die nachfolgenden Einstellungen vor und bestätigen Sie anschließend die Eingaben über die Schaltfläche "OK".
   - Wählen Sie das Symbol "Datenbaustein" ("Data block") aus.
   - Wählen Sie als Typ "Global-DB" aus.
   - Tragen Sie den Namen des DB ein.
   - Aktivieren Sie das Optionsfeld "Automatisch" ("Automatic") für die automatische Nummernvergabe. Die Nummer des globalen DB wird durch das TIA Portal vergeben.



4. Doppelklicken Sie auf den neu eingefügten globalen Datenbaustein, um diesen zu öffnen.



5. Doppelklicken Sie auf "<Hinzufügen>" ("<Add new>"), um die entsprechenden Variablen hinzuzufügen.



Ergebnis

Folgende Abbildung zeigt die Variablen im DB "MqttDb", um die Ein- und Ausgänge des FB "MQTT_Client" zu verschalten.

Abbildung 2-1

MqttDb

Name	Data type	Start value
Static		
control	Struct	
enable	Bool	false
publish	Bool	false
subscribe	Bool	false
unsubscribe	Bool	false
output	Struct	
valid	Bool	false
done	Bool	false
busy	Bool	false
error	Bool	false
status	Word	16#0
diagnostics	"typeDiagnostics"	
connParam	"LMQTT_typeConnParam"	
clientIdentifier	WString[20]	WSTRING#''
username	WString[20]	WSTRING#''
password	WString[20]	WSTRING#''
willTopic	WString[20]	WSTRING#''
willMsgPayload	Array[0..99] of Byte	
willMsgLen	UInt	0
qos	USInt	0
mqttTopic	WString[100]	WSTRING#''
publishMsgPayload	Array[0..999] of Byte	
publishMsgLen	UDInt	0
retain	Bool	false
receivedTopic	WString[200]	WSTRING#''
receivedMsgPayload	Array[0..999] of Byte	
receivedMsgDataLen	UDInt	0
receivedMsgStatus	USInt	0

Folgende Abbildung zeigt die Parameter der Variable "connParam".

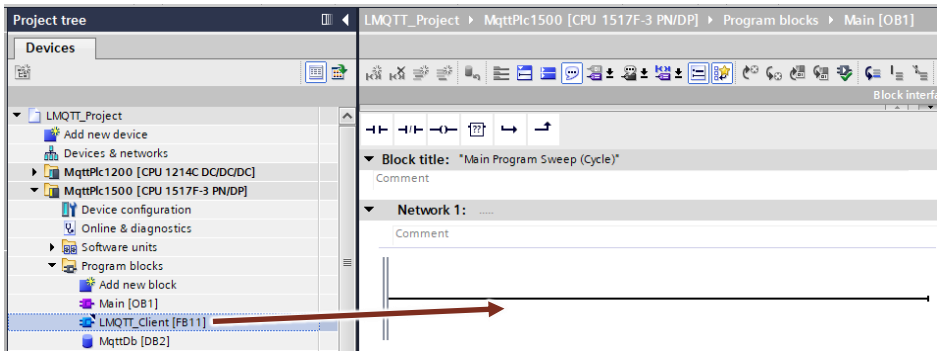
Abbildung 2-2

MqttDb

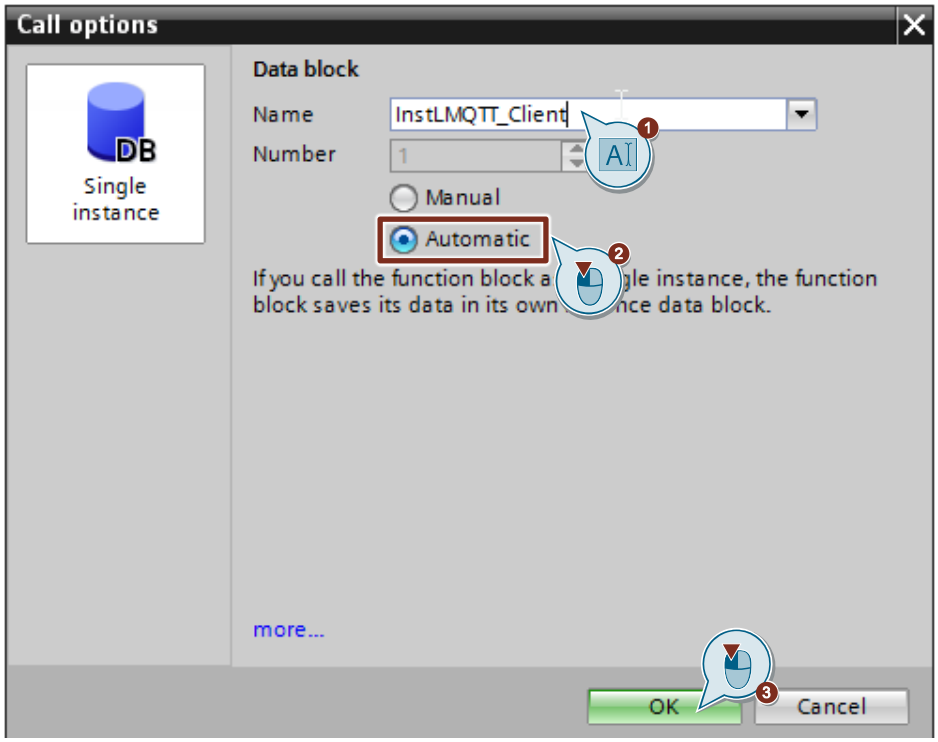
	Name	Data type	Start value
	connParam	"LMQTT_typeConnParam"	
	hwId	HW_ANY	0
	connId	CONN_OUC	0
	broker	String	''
	port	UInt	0
	tls	Struct	
	enableTls	Bool	false
	validateServerIdentity	Bool	false
	brokerCert	UDInt	0
	clientCert	UDInt	0
	keepAlive	UInt	0

2.3.4 Funktionsbaustein im Anwenderprogramm aufrufen

1. Öffnen Sie in der "Projektnavigation" ("Project tree") den Ordner "Programmbausteine" ("Program blocks") Ihrer CPU.
2. Doppelklicken Sie auf den Baustein "Main [OB1]", um den zugehörigen Programmeditor zu öffnen.
3. Ziehen Sie per Drag & Drop den FB "LMQTT_Client" aus der Projektnavigation in ein beliebiges Netzwerk des OB 1.



4. Der Dialog "Aufrufoptionen" ("Call options") zur Generierung des Instanz-DB des FB "LMQTT_Client" öffnet sich automatisch.
5. Nehmen Sie folgende Einstellungen vor und bestätigen Sie anschließend die Eingaben über die Schaltfläche "OK".
   - Tragen Sie den Namen des Instanz-DB ein.
   - Aktivieren Sie das Optionsfeld "Automatisch" ("Automatic") für die automatische Nummernvergabe. Die Nummer des Instanz-DB wird durch das TIA Portal vergeben.
   - Übernehmen Sie die Einstellungen mit "OK".

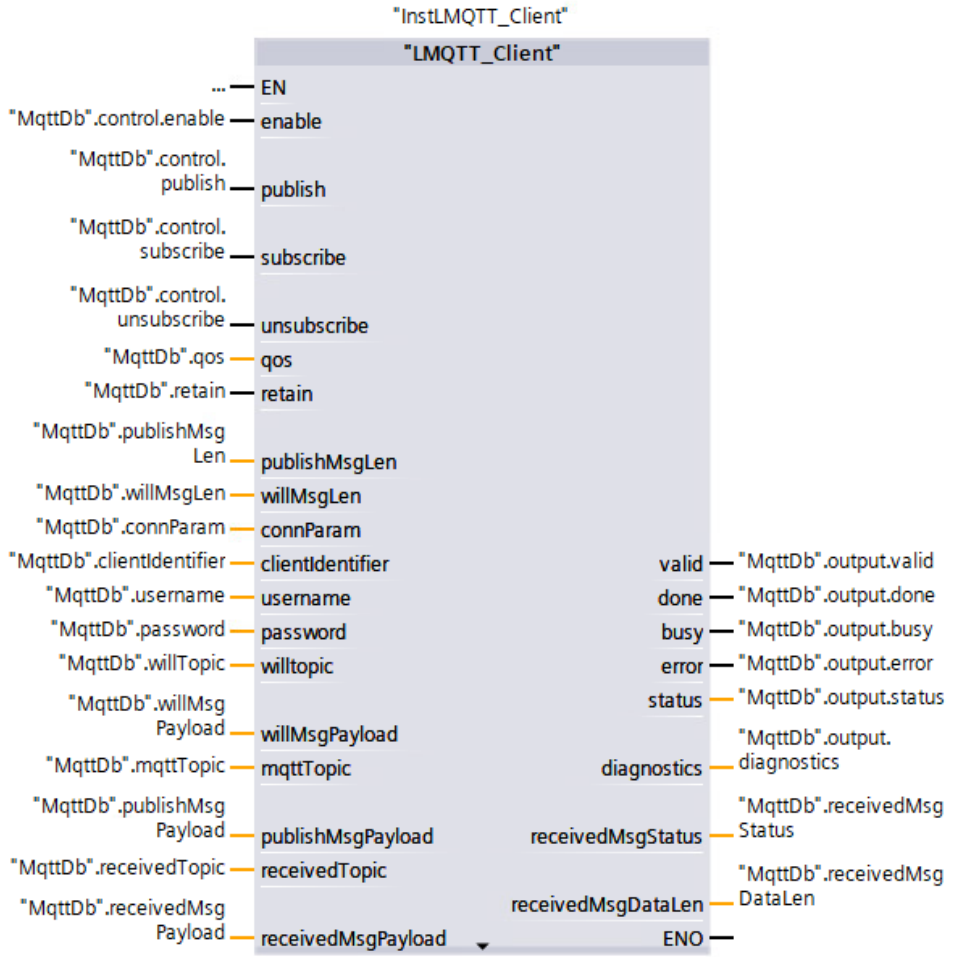


6. Belegen Sie die Ein- und Ausgänge des FBs mit den Variablen, die Sie im globalen Datenbaustein angelegt haben (siehe Kapitel 2.3.3).

Ergebnis

Die folgende Abbildung zeigt die Verschaltung der Variablen des DB "MqttDb" am FB "MQTT_Client".

Abbildung 2-3



2.4 Konfiguration der Security-Funktion

Hinweis

Die folgenden Security-Einstellungen sind nur zu tätigen, wenn Sie die MQTT-Verbindung per TLS absichern wollen.
In dem Anwendungsbeispiel authentifiziert nur der MQTT-Client den MQTT-Broker, aber nicht umgekehrt.

Die Verschlüsselung per TLS funktioniert über Zertifikate. Der MQTT-Client benötigt daher zur Authentifizierung des MQTT-Brokers dessen Zertifikat. Es gilt zwei Fälle zu unterscheiden:

- Falls der Broker ein selbst-signiertes Zertifikat verwendet, wird dieses in der Konfigurationsphase in den Client importiert. Der Broker schickt dann beim Verbindungsaufbau sein Zertifikat an den Client, der dieses mit dem hinterlegten Zertifikat vergleicht und die Übereinstimmung feststellt.
- Falls der Broker ein von einer Zertifizierungsstelle (Certification Authority, CA) abgeleitetes Zertifikat verwendet, wird das CA-Stammzertifikat in der Konfigurationsphase in den Client geladen. Der Broker schickt dann beim Verbindungsaufbau sein abgleitetes Zertifikat an den Client, der mithilfe des CA-Zertifikats feststellt, dass das erhaltene Zertifikat vertrauenswürdig ist.

Da die Menüs und Klickbefehle in beiden Fällen identisch sind, wird im Folgenden nur allgemein der Import eines "Zertifikats" in den MQTT-Client dargestellt.

Voraussetzung für eine TLS-Verschlüsselung

Um eine gesicherte MQTT-Kommunikation zwischen der SIMATIC S7-CPU (MQTT-Client) und einem MQTT-Broker in Ihrem Netzwerk einzurichten, müssen die folgende Punkte erfüllt sein:

- Der MQTT-Broker ist installiert und für das TLS-Verfahren vorkonfiguriert.
- Das Zertifikat des Brokers liegt zum Import in den Client vor.
- Die Uhrzeit der CPU ist auf die aktuelle Zeit eingestellt.
  Ein Zertifikat enthält immer eine Zeitspanne, in der es gültig ist. Um mit dem Zertifikat verschlüsseln zu können, muss die Uhrzeit der S7-CPU auch in dieser Zeitspanne liegen. Bei einer fabrikneuen S7-CPU oder nach Urlöschen der S7-CPU steht die interne Uhr auf einem Default-Wert, der außerhalb der Zertifikatslaufzeit liegt. Das Zertifikat wird dann als ungültig markiert.

2.4.1 Globalen Zertifikatsmanager im TIA Portal nutzen

Sie müssen das Zertifikat des MQTT-Brokers in STEP 7 (TIA Portal) importieren.

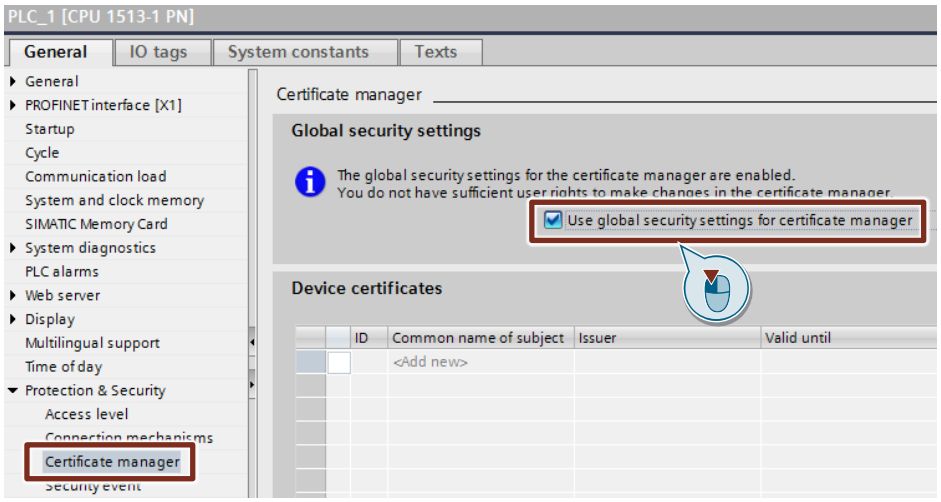
Im TIA Portal werden die Zertifikate im globalen Zertifikatsmanager verwaltet. Der Zertifikatsmanager enthält eine Übersicht aller im Projekt verwendeten Zertifikate. Im Zertifikatsmanager können Sie z. B. neue Zertifikate importieren sowie bestehende Zertifikate exportieren, erneuern oder ersetzen. Jedem Zertifikat ist eine ID zugeordnet, über die das Zertifikat in den Programmbausteinen referenziert werden kann.

Globalen Zertifikatsmanager aktivieren

Wenn Sie den Zertifikatsmanager in den Security-Einstellungen nicht verwenden, haben Sie nur Zugriff auf den lokalen Zertifikatsspeicher der CPU. Sie haben dann keinen Zugriff auf importierte Zertifikate von Fremdgeräten.

Um das Zertifikat des MQTT-Brokers zu importieren und nutzen zu können, müssen Sie den globalen Zertifikatsmanager aktivieren.

1. In der Geräte- oder Netzsicht markieren Sie die CPU. Die Eigenschaften der CPU werden im Inspektorfenster angezeigt.
2. Wählen Sie in der Bereichsnavigation des Registers "Eigenschaften" ("Properties") den Eintrag "Schutz & Security > Zertifikatsmanager" ("Protection & Security > Certificate manager") aus. Aktivieren Sie die Option "Globale Security-Einstellungen für den Zertifikatsmanager verwenden" ("Use global security settings for certificate manager").



Ergebnis

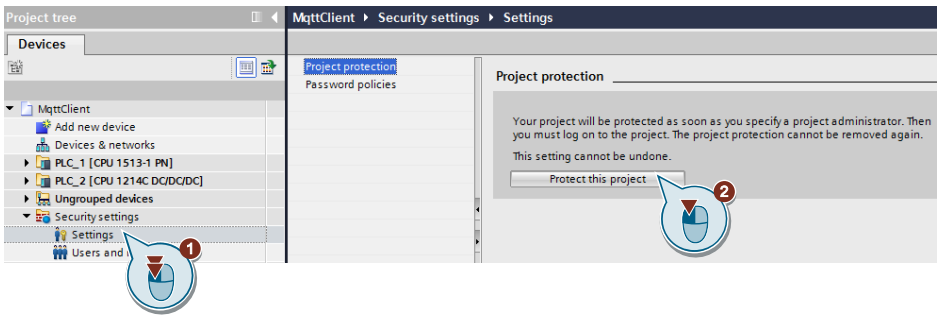
In der Projektnavigation erscheint der neue Eintrag "Security-Einstellungen" ("Security settings").

Benutzer anmelden

Nachdem Sie die globalen Security-Einstellungen für den Zertifikatsmanager aktiviert haben, müssen Sie sich bei den Security-Einstellungen anmelden. Ohne Anmeldung können Sie nicht auf den globalen Zertifikatsmanager zugreifen.

Melden Sie sich als Security-Benutzer für die Security-Einstellungen wie folgend beschrieben an:

1. Doppelklicken Sie in der Projektnavigation unter "Security-Einstellungen" ("Security settings") auf den Eintrag "Einstellungen" ("Settings").
2. Der Editor für die Benutzerverwaltung wird geöffnet und der Bereich für den Projektschutz wird angezeigt.
   Klicken Sie auf die Schaltfläche "Dieses Projekt schützen" ("Protect this project").

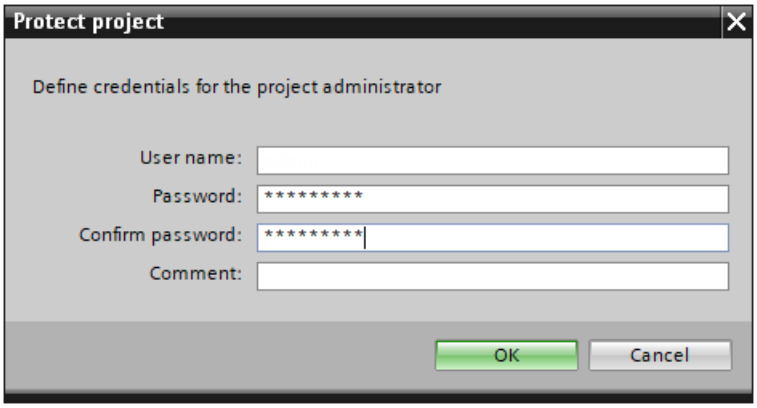


3. Der Dialog "Projektschutz" ("Protect project") wird geöffnet.
   Geben Sie einen Benutzernamen und ein Passwort ein.

   Das Passwort muss den folgenden Passwortrichtlinien entsprechen:
   - Passwortlänge: mindestens acht Zeichen, maximal 128 Zeichen
   - Mindestens ein Großbuchstabe
   - Mindestens ein Sonderzeichen (Sonderzeichen § und ß sind nicht erlaubt)
   - Mindestens eine Zahl

   Geben Sie zur Bestätigung das Passwort erneut ein.

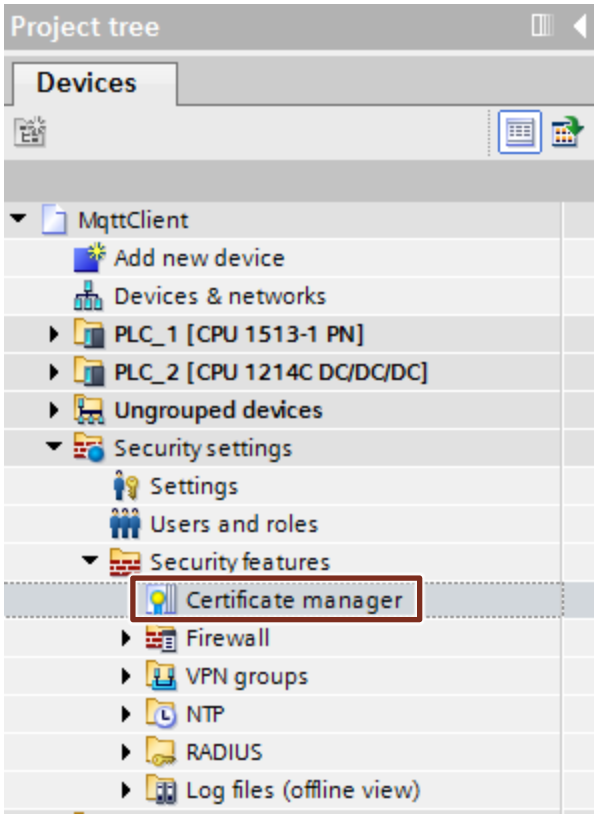
4. Geben Sie bei Bedarf einen Kommentar ein.
   Bestätigen Sie Ihre Eingaben mit "OK".



Ergebnis

Sie haben die Benutzerverwaltung aktiviert. Sie sind als Projektadministrator angemeldet und können die Security-Einstellungen nutzen.
Wenn Sie sich angemeldet haben, erscheint unter dem Eintrag "Security-Einstellungen > Sicherheitsfunktionen" ("Security settings > Security features") unter anderem eine Zeile "Zertifikatsmanager" ("Certificate manager").

Abbildung 2-4

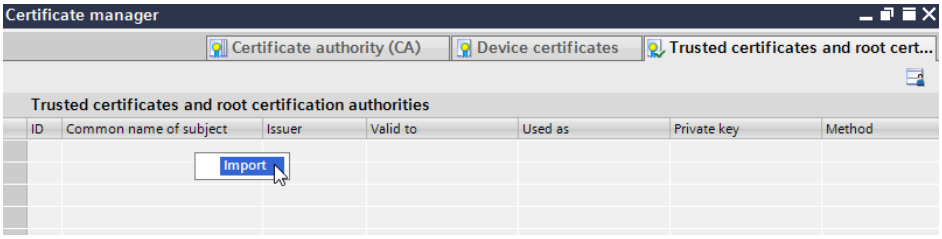


Globalen Zertifikatsmanager verwenden

Mit dem globalen Zertifikatsmanager haben Sie nun die Möglichkeit, Fremdzertifikate in das TIA Portal zu importieren. Mit einem Doppelklick auf die Zeile "Zertifikatsmanager" erhalten Sie Zugang zu allen Zertifikaten im Projekt, aufgeteilt in folgende Register:

- "Zertifizierungsstelle (CA)"
- "Gerätezertifikate"
- "Vertrauenswürdige Zertifikate und Stammzertifizierungsstellen"

1. Doppelklicken Sie in der Projektnavigation unter "Security-Einstellungen > Security-Funktionen" ("Security settings > Security features") auf den Eintrag "Zertifikatsmanager" ("Certificate manager").
2. Wählen Sie für das zu importierende Zertifikat das passende Register aus, z. B. "Vertrauenswürdige Zertifikate und Stammzertifizierungsstellen".
3. Rechts-klicken Sie in das Register und wählen Sie im Kontextmenü "Importieren" ("Import").

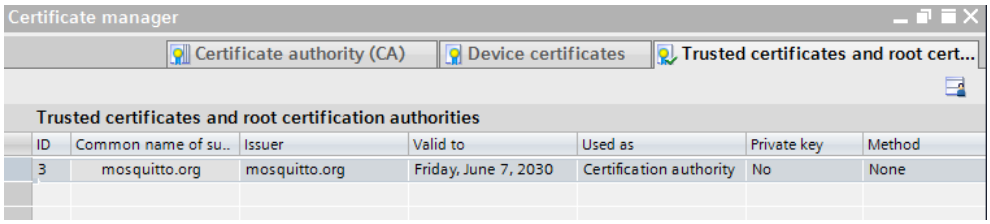


4. Navigieren Sie im Dateiexplorer zum Ablageort des Zertifikats und importieren Sie dieses. Beachten Sie dabei das Importformat des Zertifikats:
   - CER, DER, CRT oder PEM für Zertifikate ohne privaten Schlüssel
   - P12 (PKCS12-Archiv) für Zertifikate mit privatem Schlüssel

Ergebnis

Das Zertifikat des MQTT-Brokers befindet sich nun im globalen Zertifikatsmanager.

Abbildung 2-5



Hinweis

Wenn zusätzlich der MQTT-Broker den MQTT-Client authentifizieren soll, müssen Sie entweder das selbst-signierte Zertifikat des Clients oder dessen CA-Zertifikat in den Broker importieren. Beachten Sie dabei:

- Falls das Client-Zertifikat CA-abgeleitet ist, muss es von derselben CA wie das Broker-Zertifikat signiert sein.

Das Client-Zertifikat muss natürlich auch in den Client selbst geladen werden:

- Das Client-Zertifikat muss als PK12-Container (mit Zertifikat und privatem Schlüssel) in den globalen Zertifikatsmanager des Clients importiert werden.
- Das Client-Zertifikat muss in die Tabelle "Gerätezertifikate" ("Device certificates") importiert werden.

2.4.2 Lokalen Zertifikatsmanager der CPU nutzen

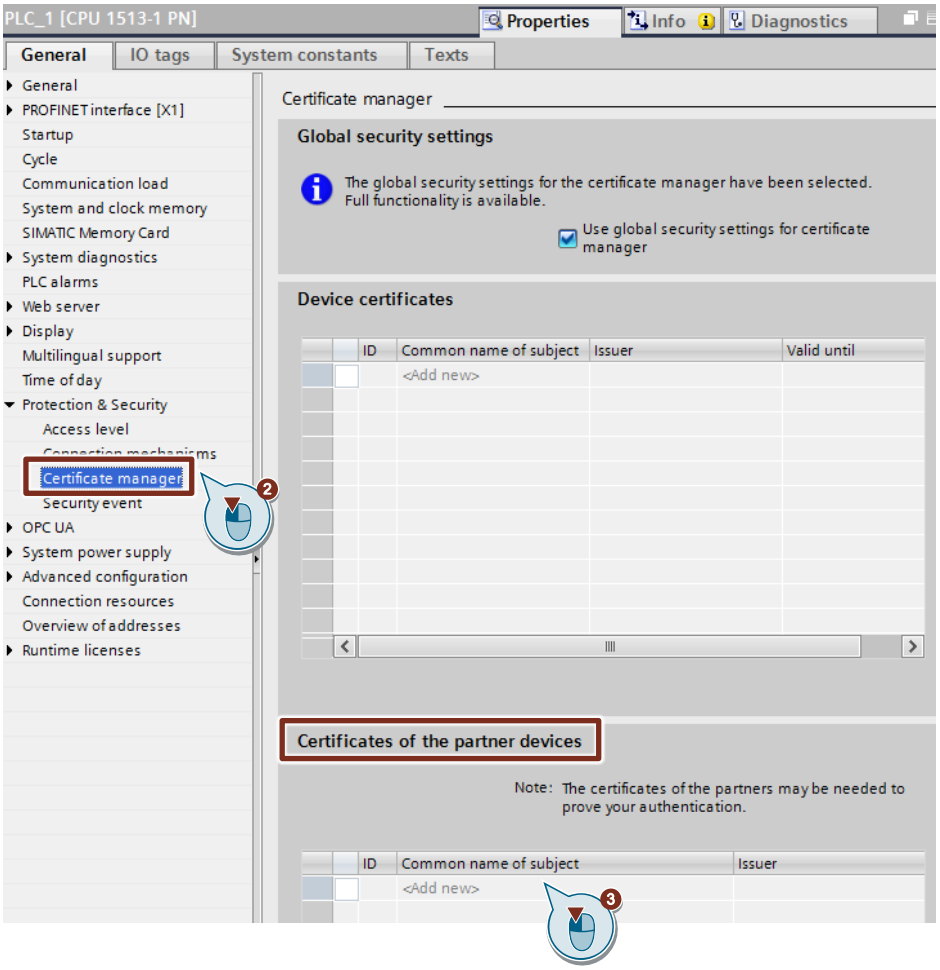
Das Zertifikat befindet sich vorerst nur im globalen Zertifikatsmanager des TIA Portal. Zertifikate, die über den Zertifikatsmanager in den globalen Security-Einstellungen importiert wurden, werden nicht automatisch den zugehörigen Modulen zugeordnet.

Um den MQTT-Broker zu authentifizieren, müssen Sie das Zertifikat in die CPU laden. Es werden nur diejenigen Gerätezertifikate auf die Baugruppe geladen, die Sie der Baugruppe über den lokalen Zertifikatsmanager als Gerätezertifikate zugeordnet haben.

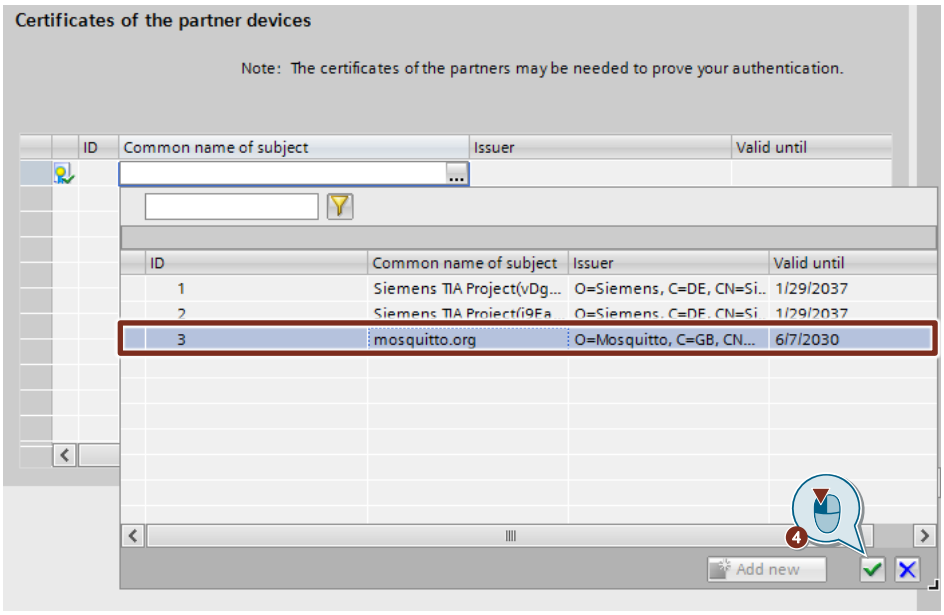
Diese Zuordnung erfolgt in den lokalen Security-Einstellungen der Baugruppe im Eintrag "Zertifikatsmanager" ("Certificate manager") über den Tabelleneditor "Gerätezertifikate" ("Device certificates"). Bei der Zertifikatszuordnung stehen die Zertifikate des globalen Zertifikatsmanagers zur Verfügung.

Die folgenden Schritte zeigen Ihnen, wie Sie das Zertifikat aus dem globalen Zertifikatsmanager der CPU zuordnen.

1. In der Geräte- oder Netzsicht markieren Sie Ihre CPU. Die Eigenschaften der CPU werden im Inspektorfenster angezeigt.
2. Um das CA-Zertifikate hinzuzufügen, wählen Sie in der Bereichsnavigation des Registers "Eigenschaften" ("Properties") unter "Schutz & Security" ("Protection & Security") den Eintrag "Zertifikatsmanager" ("Certificate manager") aus.
3. Gehen Sie zum Abschnitt "Zertifikate der Partner-Geräte" ("Certificates of the partner devices"). Klicken Sie in der Tabelle der Zertifikate auf "Hinzufügen" ("Add new"). Eine neue Zeile wird in die Tabelle eingetragen.



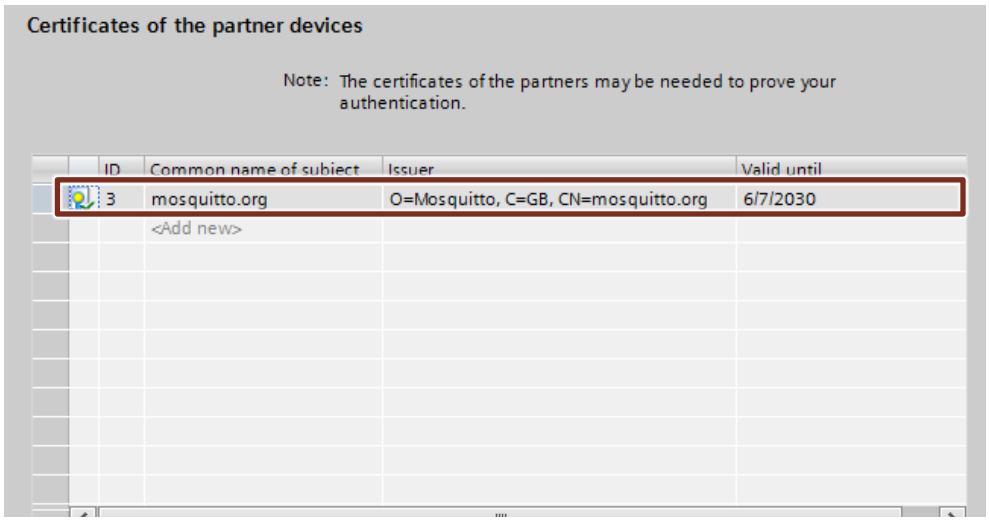
4. Klicken Sie in die neue Zeile. Die Auswahl für neue Zertifikate öffnet sich. Wählen Sie das zuvor importierte Zertifikat aus dem globalen Zertifikatsmanager aus und klicken Sie auf den grünen Haken.



Ergebnis

Das ausgewählte Zertifikat wurde der CPU zugewiesen und mit einer ID versehen. Die ID ist die Nummer des Zertifikats. Diesen Wert tragen Sie in den Verbindungsparametern für den Parameter "brokerCert" ein (siehe Abbildung 2-2).

Abbildung 2-6



Hinweis

Wenn zusätzlich der MQTT-Broker den MQTT-Client authentifizieren soll, muss das Client-Zertifikat ebenso dem Client zugewiesen werden und im Abschnitt "Gerätezertifikate" ("Device certificates")) vorhanden sein. Die ID des Zertifikats tragen Sie in den Verbindungsparametern für den Parameter "clientCert" ein (siehe Abbildung 2-2).

2.5 Parametrierung und Bedienung

Einstellen der Parameter

Bevor Sie das Anwendungsbeispiel testen können, müssen Sie die Parameter für die gesicherte oder ungesicherte TCP-Verbindung und für MQTT nach Ihren Vorgaben einstellen.

Alle Parameter, die Sie selbst definieren können, befinden sich im globalen Datenbaustein "MqttDb". Definieren Sie vor allem die Werte der folgenden Parameter in der Spalte "Startwert" ("Start value"):

- Indentifikator der Verbindung
- Indentifikator des Clients
- IPv4-Addresse oder Domainname des MQTT-Brokers. Der Domainname muss mit einem "." abgeschlossen werden.
- Port, auf dem der MQTT-Broker die Nachrichten empfängt
  - ungesicherte Verbindung: remote Port 1883
  - gesicherte Verbindung: remote Port 8883
- Parameter für die gesicherte Kommunikation
  - Gesicherte Verbindung (TLS) aktiviert
  - ID des Broker-Zertifikats (nur bei einer gesicherten Verbindung relevant); das Zertifikat ist entweder selbst-signiert oder CA-abgeleitet
  - ID des eigenen Client-Zertifikats, falls der MQTT-Broker den Client ebenfalls authentifiziert (nur bei einer gesicherten Verbindung relevant)
- MQTT-Parameter, z. B.
  - Anmeldeinformationen am MQTT-Broker
  - Topic
  - Nachrichtentext

Laden Sie anschließend das Projekt in Ihre CPU.

Folgende Tabelle zeigt die Variablen im DB "MqttDb", um die Ein- und Ausgänge des FB "MQTT_Client" zu verschalten.

Tabelle 2-1

Variable		Datentyp	Hinweis
control		Struct	Diese Datenstruktur enthält die Variablen zur Steuerung der Aufträge des FB "LMQTT_Client".
	enable	Bool	Diese Variable steuert den Verbindungsaufbau. • Setzen Sie die Variable auf den Wert "1", um die TCP- und MQTT-Verbindung aufzubauen. • Wenn der Wert der Variablen von"1" nach "0" wechselt (negative Flanke), wird die TCP- und MQTT-Verbindung abgebaut.
	publish	Bool	Über diese Variable starten Sie einen Auftrag zum Senden eines PUBLISH-Paket.
	subscribe	Bool	Über diese Variable starten Sie einen Auftrag zum Senden eines SUBSCRIBE-Paket.
	unsubscribe	Bool	Über diese Variable starten Sie einen Auftrag zum Senden eines UNSUBSCRIBE-Paket.
output		Struct	Diese Datenstruktur enthält die Variablen zur Auswertung der Ausgänge des FB "LMQTT_Client".
	valid	Bool	Statusanzeige True: Die Werte an den Ausgängen des FB "LMQTT_Client" sind gültig.
	done	Bool	Statusanzeige • True: Auftrag fehlerfrei ausgeführt. • False: Auftrag noch nicht gestartet oder noch in Bearbeitung.
	busy	Bool	Statusanzeige • True: Auftrag noch nicht beendet. Ein neuer Auftrag kann nicht gestartet werden. • False: Auftrag noch nicht gestartet oder bereits beendet.
	error	Bool	Statusanzeige • True: Fehler aufgetreten • False: kein Fehler
	status	Word	Status des FB "LMQTT_Client" Detaillierte Informationen finden Sie in der Bibliotheksbeschreibung im Beitrag 109780503.

Variable	Datentyp	Hinweis
diagnostics	"typeDiagnostics"	Diagnoseinformationen des FB "LMQTT_Client" Detaillierte Informationen finden Sie in der Bibliotheksbeschreibung im Beitrag 109780503.
connParam	"LMQTT_typeConnParam"	Tragen Sie die Parameter ein, um eine Verbindung zum MQTT-Broker aufzubauen. Detaillierte Informationen finden Sie in der Tabelle 2-2.
clientIdentifier	WString[20]	Tragen Sie den Client-Identifier ein, der beim Verbindungsaufbau genutzt wird, z. B. "SiemensClient".
username	WString[20]	Optional ist es möglich einen Benutzername für den Verbindungsaufbau einzutragen. Wenn kein Benutzername eingetragen wird, wird der Parameter nicht ausgewertet.
password	WString[20]	Optional ist es möglich ein Passwort für den Verbindungsaufbau einzutragen. Wenn kein Passwort eingetragen wird, wird der Parameter nicht ausgewertet.
willTopic	WString[20]	Optional ist es möglich ein Topic einzutragen, an den die "Last Will" Nachricht gesendet wird.
willMsgPayload	Array[*] of Byte	Optional ist es möglich eine Nachricht einzutragen, die als "Last Will" versendet wird. Die Länge des Array kann beliebig gewählt werden. Die wirkliche Nachrichtenlänge, die aus diesem Array gesendet wird, wird mit der Variable "willmesscnt" eingestellt.
willMsgLen	UInt	Aktuelle Länge an gültigen Daten im Array "willMessage".
qos	USInt	Tragen Sie Quality of Service ein, mit der die Nachrichten gesendet werden. Mögliche Werte sind 0, 1 oder 2.
mqttTopic	WString[100]	Tragen Sie das MQTT Topic ein, das beim publish-, subscribe- oder unsubscribe-Auftrag verwendet wird.
publishMsgPayload	Array[*] of Byte	Tragen Sie die MQTT Nachricht ein, die als Nutzdaten beim publish-Auftrag übertragen wird. Die Länge des Array kann beliebig gewählt werden. Die wirkliche Nachrichtenlänge, die aus diesem Array gesendet wird, wird mit der Variable "pubMessageCnt" eingestellt.

Variable	Datentyp	Hinweis
publishMsgLen	UDInt	Aktuelle Länge an gültigen Daten im Array "message".
retain	Bool	Tragen Sie ein, ob die Daten mit oder ohne dem "retain" flag gesendet werden. • True: Die Daten werden mit dem "retain" flag gesendet. • False: Die Daten werden ohne "retain" flag gesendet.
receivedTopic	WString[200]	In dieser Variable wird das MQTT Topic, auf dem eine Nachricht über die Subscription empfangen wurde, gespeichert.
receivedMsgPayload	Array[*] of Byte	In dieser Variablen werden die Nutzdaten, die in der Nachricht über eine Subscription empfangen wurden, gespeichert. Die Länge des Array kann beliebig gewählt werden. Die wirkliche Nachrichtenlänge, die in diesem Array empfangen wird, wird in der Variable "receivedMsgLen" angezeigt.
receivedMsgDataLen	UDInt	Anzahl an gültigen Daten im Array "receivedmessage".
receivedMsgStatus	USInt	Diese Variable zeigt jeweils für einen Zyklus an, wenn eine neue Nachricht empfangen wurde (Subscription). • 0: Keine neue Nachricht empfangen. • 1: Neue gültige Nachricht empfangen. • 2: Neue Nachricht empfangen, aber Nachricht ungültig oder empfangene Daten sind größer als der Speicherbereich des Empfangstopic oder der Empfangsnachricht.

Die folgende Tabelle zeigt die Parameter der Variable "connParam".

Tabelle 2-2

Parameter	Datentyp	Hinweis
hwId	HW_ANY	Tragen Sie die Hardware-Kennung der PN/IE-Schnittstelle für den Verbindungsaufbau ein. Wenn die Variable den Wert "0" hat, wird automatisch eine passende Hardware-Kennung gewählt.
connId	CONN_OUC	Verbindungs-ID für den Verbindungsaufbau.
broker	String	Die IP-Adresse oder der Host-name des Brokers
port	UInt	Tragen Sie den MQTT-Port ein: • Port 1883: ungesicherte Verbindung • Port 8883: gesicherte Verbindung
tls	Struct	Diese Datenstruktur enthält die Parameter für eine gesicherte Verbindung.
enableTls	Bool	Tragen Sie den Wert "True" ein, wenn die Verbindung mit TLS gesichert werden soll.
validateServerIdentity	Bool	Tragen Sie den Wert "True" ein, wenn das Zertifikat des MQTT-Brokers beim Verbindungs-aufbau validiert werden soll.
brokerCert	UDInt	Zertifikats-ID des MQTT-Broker-Zertifikats.
clientCert	UDInt	Zertifikats-ID des MQTT-Client-Zertifikats.
keepAlive	UInt	Tragen Sie für die Aktivierung des Keep-Alive Mechanismus von MQTT einen Wert in Sekunden ein. Wenn die Variable den Wert "0" hat, ist kein Keep-Alive aktiv.

Hinweis

Wenn die TCP-Verbindung wird über den vollqualifizierten Domainnamen aufgebaut werden soll, dann müssen Sie in der CPU einen DNS-Server konfigurieren.

Anwendungsbeispiel bedienen

Wenn Sie alle Parameter eingestellt haben und das CA-Zertifikat des MQTT-Brokers in den lokalen Zertifikatsmanager der CPU hinzugefügt haben, können Sie das Anwendungsbeispiel testen.

Bevor Sie das Anwendungsbeispiel testen, überprüfen Sie folgende Punkte:

1. Das Projekt ist in die CPU geladen.
2. Die CPU und der MQTT-Broker sind miteinander über Ethernet verbunden und erreichbar.
3. Der MQTT-Broker ist ordnungsgemäß konfiguriert und gestartet.
4. Das Logging am MQTT-Broker ist bei Bedarf gestartet, um die Anmeldung des MQTT-Clients und den Publish-Mechanismus mitzuversorgen.

Wenn die genannten Punkte erfüllt sind, können Sie die MQTT-Kommunikation zwischen CPU und MQTT-Broker anstoßen. Triggern Sie dafür den Eingang "enable" des Funktionsbausteins "LMQTT_Client“. Solange der Eingang "enable" auf "True" gesetzt ist, bleibt die Verbindung bestehen. Wird der Eingang "enable" auf "False" zurückgesetzt, wird die Verbindung abgebaut.

Im positiven Fall werden die internen Zustandsautomaten durchlaufen und stellen eine TCP- und MQTT-Verbindung zum MQTT-Broker her. Die Ausgangsvariable "status" wird auf den Wert "16#7004" gesetzt und signalisiert eine bestehende TCP- und MQTT-Verbindung.

Nun können Sie folgende Funktionen ausführen:

- MQTT-Nachricht senden: Triggern Sie die Eingangsvariable "publish".
- MQTT-Nachricht zu einem abonnierten Topic empfangen: Triggern Sie die Eingangsvariable "subscribe".
  Wenn die Verbindung zum MQTT-Broker unterbrochen wird (status = 16#9000), wird die Verbindung automatisch wieder aufgebaut. Nach einer Verbindungsunterbrechung ist es notwendig einen "subscribe"-Auftrag für die abonnierten Topics durchzuführen.
- Sich selbst von abonnierten Topics abmelden: Triggern Sie die Variable "unsubscribe".

Wenn die Verbindung zum MQTT-Broker nicht aufgebaut wird, überprüfen Sie die Ausgangsvariable "status" und "diagnostics", um den Fehler zu diagnostizieren.

2.6 Fehlerhandling

Informationen zur Bedeutung der Werte der Variablen "status" und "diagnostics" finden Sie in der Bibliotheksbeschreibung in folgendem Beitrag:

https://support.industry.siemens.com/cs/ww/de/view/109780503

3 Wissenswertes

3.1 Grundlagen zu MQTT

Hinweis Eine genaue Beschreibung zu MQTT finden Sie in der MQTT-Spezifikationsbeschreibung (siehe \3\ im Kapitel 4.2).

3.1.1 Terminologie

Im Folgenden werden die wichtigsten Begriffe bei dem Telemetrie-Protokoll MQTT erläutert.

MQTT-Nachricht

Eine Nachricht bei MQTT besteht aus mehreren Teilen:

- Einem definierten Betreff ("Topic")
- Ein zugewiesenes Merkmal zur Qualitätssicherung ("Quality of Service")
- Dem Nachrichtentext

MQTT-Client

Ein MQTT-Client ist ein Programm oder ein Gerät, das MQTT nutzt. Ein Client baut immer aktiv die Verbindung zum Broker auf. Ein Client kann folgende Funktionen ausführen:

- Nachrichten mit einem definierten Betreff ("Topic") an den MQTT-Broker senden, an denen andere Clients interessiert sein könnten (Publish-Mechanismus)
- Nachrichten beim MQTT-Broker abonnieren, die einem bestimmten Betreff ("Topic") folgen (Subscriber-Mechanismus)
- Sich selbst von abonnierten Nachrichten abmelden
- Die Verbindung zum Broker trennen

Hinweis Der Funktionsbaustein "LMQTT_Client" in diesem Anwendungsbeispiel unterstützt folgende Funktionen:

- Anmeldung am MQTT-Broker
- Publish-Mechanismus
- Subscribe- und Unsubscribe-Mechanismus
- Ping-Mechanismus
- Abmeldung am MQTT-Broker.

MQTT-Broker

Ein MQTT-Broker ist die zentrale Komponente von MQTT und kann ein Programm oder ein Gerät sein. Der MQTT-Broker agiert als Vermittler zwischen dem sendenden MQTT-Client und dem abonnierenden MQTT-Client. Der MQTT-Broker verwaltet die Topics inklusive der darin enthaltenen Nachrichten und regelt den Zugriff auf die Topics. Der MQTT-Broker hat folgende Funktionen:

- Netzwerkverbindungen von den MQTT-Clients akzeptieren
- Nachrichten eines MQTT-Client entgegennehmen
- Abonnement-Anfragen von MQTT-Clients bearbeiten
- Nachrichten an die MQTT-Clients weiterleiten, die mit Ihrem Abonnement übereinstimmen

Hinweis Der MQTT-Broker ist nicht Teil dieses Anwendungsbeispiels und wird als gegeben vorausgesetzt.

Topics

MQTT-Nachrichten sind in Topics organisiert. Ein Topic "beschreibt" ein Themengebiet. Die Topics können von den MQTT-Clients abonniert werden (Subscriber-Mechanismus). Dem Absender einer Nachricht (Publisher-Mechanismus) obliegt die Aufgabe, Inhalt und Topic beim Versand der Nachricht festzulegen. Der Broker kümmert sich dann darum, dass die Subscriber die Nachrichten von den abonnierten Topics bekommen. Die Topics folgen einem definierten Schema. Sie ähneln einem Verzeichnispfad und bilden eine Hierarchie ab.

3.1.2 Standard und Architektur

ISO Standard

MQTT definiert einen OASIS- bzw. ISO-Standard (ISO/IEC PRF 20922).

Je nach verwendeten Sicherheitsprotokollen läuft MQTT auf unterschiedliche Zugriffs-Ports. Angebotene Ports sind:

- 1883: MQTT, unverschlüsselt
- 8883: MQTT, verschlüsselt
- 8884: MQTT, verschlüsselt, Client Zertifikat notwendig
- 8080: MQTT über WebSockets, unverschlüsselt
- 8081: MQTT über WebSockets, verschlüsselt

Architektur

MQTT ist ein Publish- und Subscribe-Protokoll. Dieser Mechanismus entkoppelt einen Client, der Nachrichten sendet (Publisher) von einen oder mehreren Clients, welche die Nachrichten empfangen (Subscriber). Das bedeutet auch, dass die "Publisher" nichts von der Existenz der "Subscriber" wissen (und umgekehrt).
Es gibt eine dritte Komponente in der MQTT-Architektur, der MQTT-Broker. Der MQTT-Broker befindet sich zwischen "Publisher" und "Subscriber". Der MQTT-Broker sorgt für die Kommunikationssteuerung.

3.1.3 Features

MQTT bietet durchaus nützliche Features an.

Quality of Service

Für die Qualitätssicherung bei der Nachrichtenübermittlung sieht die MQTT-Spezifikation drei Servicequalitäten vor:

- QoS "0": Bei der niedrigsten Stufe 0 handelt es sich um ein "fire and forget"-Verfahren. Es gibt also keine Garantie, dass die Nachricht überhaupt ankommt.
- QoS "1": Bei dem QoS-Level 1 ist sichergestellt, dass die Nachricht mindestens einmal in der Topic-Queue landet. Der MQTT-Broker quittiert den Erhalt der Nachricht.
- QoS "2": In der höchsten Stufe 2 garantiert der MQTT-Broker durch mehrfachen Handshake mit dem MQTT-Client, dass die Nachricht genau einmal abgelegt wird.

Last will

MQTT unterstützt die Funktion "Last Will and Testament". Diese Funktion wird verwendet, um andere MQTT-Clients zu unterrichten, wenn die Verbindung zu einem MQTT-Client unvorhergesehen getrennt wurde.

Jeder MQTT-Client kann während des Verbindungsaufbaus zum MQTT-Broker seinen letzten Willen spezifizieren und dem MQTT-Broker mitteilen. Dieser letzte Wille ist wie eine normale MQTT-Nachricht aufgebaut, inklusive Topic, QoS und Payload. Der MQTT-Broker speichert den letzten Willen. Sobald der MQTT-Broker merkt, dass die Verbindung mit dem betreffenden MQTT-Client unvorhergesehen abgebrochen wurde, schickt der MQTT-Broker an alle Abonnenten, die sich für das Topic registriert haben, den letzten Willen als MQTT-Nachricht heraus. Auf diese Weise erfahren auch die Abonnenten, dass der MQTT-Client getrennt wurde.

KeepAlive

MQTT unterstützt die Funktion "KeepAlive". Diese gewährleistet, dass die Verbindung noch offen ist sowie MQTT-Client und MQTT-Broker miteinander verbunden sind.

Für das KeepAlive definieren die MQTT-Clients ein Zeitintervall und teilen es dem MQTT-Broker während ihres Verbindungsaufbaus mit. Dieses Intervall ist die größtmögliche, geduldete Zeitperiode, in welcher der MQTT-Client und der MQTT-Broker ohne Kontakt verharren dürfen. Wird die Zeit überschritten, muss der MQTT-Broker die Verbindung trennen.

Das bedeutet, solange der MQTT-Client regelmäßig Nachrichten innerhalb des KeepAlive-Intervalls an den Broker schickt, muss der MQTT-Client keine besondere Aktion ausführen, um die Verbindung aufrecht zu erhalten. Wenn der MQTT-Client aber keine Nachrichten innerhalb des KeepAlive-Intervalls sendet, muss er vor Ablauf der Frist ein Ping-Paket an den MQTT-Broker absetzen. Mit diesem Ping signalisiert der MQTT-Client dem MQTT-Broker, dass er weiterhin verfügbar ist.

Wenn eine Nachricht oder ein Ping-Paket an den MQTT-Broker geschickt wurde, beginnt die Zeitmessung für das KeepAlive-Intervall von vorne.

Hinweis

- Der Client bestimmt das KeepAlive-Intervall. So kann er das Intervall seiner Umgebung anpassen, z. B. wegen einer langsamen Bandbreite.
- Der größtmögliche Wert für das KeepAlive-Intervall ist 18 h 12 m 15 s.
- Wenn der Client das KeepAlive-Intervall auf den Wert "0" setzt, wird der KeepAlive-Mechanismus deaktiviert.

Message Persistence

Wird die Verbindung zu einem MQTT-Client unterbrochen, so kann der Broker neue Nachrichten für diesen Client für eine spätere Zustellung zwischenspeichern.

Retained Messages

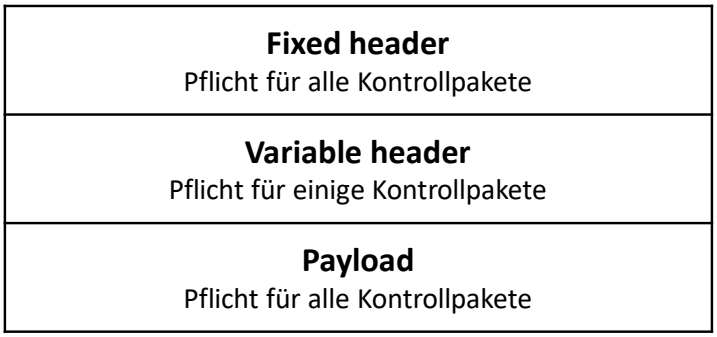
Abonniert ein MQTT-Client zum ersten Mal ein Topic, bekommt er normalerweise erst dann eine Nachricht, wenn ein anderer MQTT-Client das nächste Mal eine Nachricht mit dem abonnierten Topic sendet. Mit "Retained Messages" bekommt der Abonnent den letzten Wert, der vor seiner Abonnementsanfrage an das Topic gesendet wurde, sofort zugestellt.

3.1.4 Aufbau der MQTT-Kontrollpakete

Die meisten MQTT-Kontrollpakete arbeiten nach dem Handshake-Verfahren. Der MQTT-Client ist immer das aktive Element und setzt einen Auftrag an den MQTT-Broker ab. Der Broker bestätigt je nach Auftrag die Anfrage.

Die Struktur eines MQTT-Kontrollpakets ist fest vorgegeben. Die folgende Grafik zeigt die Struktur:

Abbildung 3-1



Der "Fixed Header" besteht immer aus den folgenden Elementen:

- Eine Kennungsnummer für den MQTT-Kontrollpakettyp
- Ein Bereich für mögliche Flags; falls keine Flags für das Kontrollpaket vorgesehen sind, werden die Bits als "reserved" gekennzeichnet
- Die Anzahl der folgenden Bytes nach dem "Fixed Header"

Der "Variable Header" ist nur bei einigen Kontrollpaketen erforderlich. Der Inhalt des "Variable Header" ist abhängig vom Kontrollpakettyp.

Der Payload ist bei den meisten Kontrollpaketen Pflicht. Auch hier ist der Inhalt vom Kontrollpakettyp abhängig. Für jedem Kontrollpakettyp gibt es klare Regelungen, mit was und in welcher Reihenfolge der Payload befüllt werden kann.

Hinweis

Eine genaue Beschreibung zu den MQTT-Kontrollpaketen finden Sie in der MQTT-Spezifikationsbeschreibung (siehe \3\ im Kapitel 4.2).

Die MQTT-Kontrollpakete aus diesem Anwendungsbeispiel werden im Folgenden kurz erläutert.

3.1.5 MQTT-Verbindungsaufbau

Eine MQTT-Verbindung wird immer zwischen einem MQTT-Client und dem MQTT-Broker hergestellt. Eine direkte Client-Client-Verbindung ist nicht möglich.

Die Verbindung wird von einem MQTT-Client initiiert, sobald der MQTT-Client ein "CONNECT"-Paket an den MQTT-Broker schickt. Im positiven Fall antwortet der MQTT-Broker mit einem "CONNACK"-Paket und einem Statuscode.

In folgenden Fällen schließt der MQTT-Broker sofort die Verbindung:

- Wenn das "CONNECT"-Paket schadhaft ist
- Wenn der Aufbau des "CONNECT"-Pakets nicht der Spezifikation entspricht
- Wenn der Verbindungsaufbau zu lange dauert

MQTT-Kontrollpaket "CONNECT"

Tabelle 3-1 zeigt den Aufbau des "Fixed Header" des "CONNECT"-Pakets.

Tabelle 3-1

Fixed Header								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Byte 1	Kennungsnummer für MQTT-Kontrollpakettyp = 1 (dez)				Reserve			
Byte 2	Remaining Length: Die Anzahl der folgenden Bytes nach dem "Fixed Header" = "Variable Header" + "Payload"							

Ein "CONNECT"-Paket enthält im "Variable Header" folgende Bereiche:

1. Protokollname: Der Protokollname "MQTT" wird als UTF-8 Zeichenkette übertragen.
2. Protokolllevel: 4 (dez)
3. Connect Flags: Das Byte "Connect Flags" beinhaltet eine Reihe an Parametern, die das Verhalten der MQTT-Verbindung spezifizieren. Zudem zeigt das Byte "Connect Flags" auch, welche optionalen Felder im "Payload" vorhanden sind oder nicht. Mit dem Flag "Clean Session" kann die Verbindungsart geregelt werden.
4. Keep Alive: Mit der KeepAlive-Zeit wird das Zeitintervall bestimmt, in der sich der MQTT-Client verpflichtend beim MQTT-Broker melden muss. Das kann entweder durch das Senden einer Nachricht oder einem PING-Kommando erfolgen. Meldet sich der Client nicht in dem Zeitintervall, baut der MQTT-Broker die Verbindung zum Client ab.

Tabelle 3-2 zeigt den Aufbau des "Variable Header" des "CONNECT"-Pakets.

Tabelle 3-2

Variable Header								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Protokollname								
Byte 1	Länge MSB = 0 (dez)							
Byte 2	Länge LSB = 4 (dez)							
Byte 3	'M'							
	0	1	0	0	1	1	0	1
Byte 4	'Q'							
	0	1	0	1	0	0	0	1
Byte 5	'T'							
	0	1	0	1	0	1	0	0
Byte 6	'T'							
	0	1	0	1	0	1	0	0
Protokolllevel								
Byte 7	Protokolllevel = 4 (dez)							
Connect Flags								
Byte 8	Benutzer-name-Flag	Passwort-Flag	Will Retain-Flag	Will QoS-Flag		Will-Flag	Clean Session-Flag	Reserve
Keep Alive								
Byte 9	Keep Alive MSB							
Byte 10	Keep Alive LSB							

Im "Payload" erscheinen die vorhandenen Felder in folgender Reihenfolge:

- Client-ID: Die Client-ID dient zur Identifikation des Clients am MQTT-Broker. Die Client-ID muss als erstes Feld im "Payload" erscheinen.
- Will-Topic: Das Feld erscheint optional, wenn das Flag "Will" auf "TRUE" gesetzt wurde.
- Will-Message: Das Feld erscheint optional, wenn das Flag "Will" auf "TRUE" gesetzt wurde.
- Benutzername: Das Feld erscheint optional, wenn das Flag "Benutzername" auf "TRUE" gesetzt wurde.
- Passwort: Das Feld erscheint optional, wenn das Flag "Passwort" auf "TRUE" gesetzt wurde.

MQTT-Kontrollpaket "CONNACK"

Tabelle 3-3 zeigt den Aufbau des "Fixed Header" des "CONNACK"-Pakets:

Tabelle 3-3

Fixed Header								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Byte 1	Kennungsnummer für MQTT-Kontrollpakettyp = 2 (dez)				Reserve			
Byte 2	Remaining Length: Die Anzahl der folgenden Bytes nach dem "Fixed Header" = "Variable Header" = 2 Byte							

Tabelle 3-4 zeigt den Aufbau des "Variable Header" des "CONNACK"-Pakets.

Tabelle 3-4

Variable Header								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Connect Acknowledge Flags								
Byte 1	Reserve							Session Present
Connect Return Code								
Byte 2	• 0x00 = Der MQTT-Broker akzeptiert die Verbindung. Der MQTT-Broker unterstützt das Level des MQTT-Protokolls, das vom Client angefordert wird, nicht. • 0x01 = Der MQTT-Broker unterstützt das Level des MQTT-Protokolls, das vom MQTT-Client angefordert wird, nicht. • 0x02: Der MQTT-Broker lässt die Client-ID nicht zu. • 0x03: Der MQTT-Service ist nicht verfügbar. • 0x04: Die Daten im Benutzernamen und Passwort sind inkorrekt. • 0x05: Der MQTT-Client ist nicht berechtigt sich zu verbinden.							

3.1.6 MQTT-Push-Mechanismus

Sobald sich ein MQTT-Client mit dem MQTT-Broker verbunden hat, kann er Nachrichten an den MQTT-Broker schicken. Dazu nutzt der Client das "PUBLISH"-Paket. Da die Nachrichten bei MQTT topic-basiert gefiltert und verwaltet werden, muss jede MQTT-Nachricht ein Topic beinhalten. Das Topic ist Teil des "Variable Header". Der eigentliche Nachrichtentext ist im "Payload" untergebracht.

"PUBLISH"-Paket

Tabelle 3-5 zeigt den Aufbau des "Fixed Header" des "PUBLISH"-Pakets.

Tabelle 3-5

Fixed Header								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Byte 1	Kennungsnummer für MQTT-Kontrollpakettyp = 3 (dez)				DUP-Flag	QoS-Level		Retain-Flag
	0	0	1	1	X	X	X	X
Byte 2	Remaining Length: Die Anzahl der folgenden Bytes nach dem "Fixed Header" = "Variable Header" + Payload							

Abhängig von der Einstellung der Qualitätssicherung ("QoS") endet an dieser Stelle der Push-Mechanismus oder es werden weitere Kontrollpakete ausgetauscht:

- QoS = 0 (dez): Die Nachricht wird höchstens einmal gesendet. Der Sendeauftrag endet an dieser Stelle.
- QoS = 1 (dez): Die Nachricht wird mindestens einmal gesendet. Der MQTT-Broker quittiert das "PUBLISH"-Paket mit einem "PUBACK"-Paket.
- QoS = 2 (dez): Die Nachricht wird genau einmal gesendet. Der MQTT-Broker quittiert das "PUBLISH"-Paket mit einem "PUBREC"-Paket. Daraufhin erfolgt ein weiterer Handshake zwischen MQTT-Client und MQTT-Broker. Der Client beantwortet das "PUBREC"-Paket mit einem "PUBREL"-Paket. Der MQTT-Broker komplettiert den zweifachen Handshake mit einem "PUBCOM"-Paket.

Hinweis Nähere Informationen zu der Qualitätssicherung QoS finden Sie in Kapitel 3.1.3.

Der "Variable Header" des "Publish"-Pakets enthält folgende Felder:

- Name des Topic
- Paket-ID

Im "Payload" ist der Nachrichtentext enthalten.

"PUBACK"-Paket (Publish Acknowledgement)

Der MQTT-Broker beantwortet das "PUBLISH"-Paket mit QoS=1 mit dem "PUBACK"-Paket.

Tabelle 3-6 zeigt den Aufbau des "Fixed Header" des "PUBACK"-Pakets.

Tabelle 3-6

Fixed Header								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Byte 1	Kennungsnummer für MQTT-Kontrollpakettyp = 4 (dez)				Reserve			
	0	1	0	0	0	0	0	0
Byte 2	Remaining Length: Die Anzahl der folgenden Bytes nach dem "Fixed Header" = "Variable Header" = 2 Byte							

Der "Variable Header" des "PUBACK"-Pakets enthält die Paket-ID.

Das "PUBACK"-Paket hat keinen "Payload".

"PUBREC"-Paket (Publish Received)

Der MQTT-Brocker beantwortet das "PUBLISH"-Paket mit QoS=2 mit dem "PUBREC"-Paket.

Tabelle 3-7 zeigt den Aufbau des "Fixed Header" des "PUBREC"-Pakets.

Tabelle 3-7

Fixed Header								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Byte 1	Kennungsnummer für MQTT-Kontrollpakettyp = 5 (dez)				Reserve			
	0	1	0	1	0	0	0	0
Byte 2	Remaining Length: Die Anzahl der folgenden Bytes nach dem "Fixed Header" = "Variable Header" = 2 Byte							

Der "Variable Header" des "PUBREC"-Pakets enthält die Paket-ID.

Das "PUBREC"-Paket hat keinen "Payload".

"PUBREL"-Paket (Publish Release)

Der MQTT-Client antwortet auf das "PUBREC"-Paket mit dem "PUBREL"-Paket.

Tabelle 3-8 zeigt den Aufbau des "Fixed Header" des "PUBREL"-Pakets.

Tabelle 3-8

Fixed Header								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Byte 1	Kennungsnummer für MQTT-Kontrollpakettyp = 6 (dez)				Reserve			
	0	1	1	0	0	0	1	0
Byte 2	Remaining Length: Die Anzahl der folgenden Bytes nach dem "Fixed Header" = "Variable Header" = 2 Byte							

Hinweis

Die Reserve-Bits im "Fixed Header" müssen wie folgt gesetzt werden:

- Bit 3 = 0
- Bit 2 = 0
- Bit 1 = 1
- Bit 0 = 0

Das "Variable Header" des "PUBREL"-Pakets enthält die Paket-ID.

Das "PUBREL"-Paket hat kein "Payload".

"PUBCOMP"-Paket (Publish Complete)

Der MQTT-Broker beantwortet das "PUBREL"-Paket mit dem "PUBCOMP"-Paket.

Tabelle 3-9 zeigt den Aufbau des "Fixed Header" des "PUBCOMP"-Paket.

Tabelle 3-9

Fixed Header								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Byte 1	Kennungsnummer für MQTT-Kontrollpakettyp = 7 (dez)				Reserve			
	0	1	1	1	0	0	0	0
Byte 2	Remaining Length: Die Anzahl der folgenden Bytes nach dem "Fixed Header" = "Variable Header" = 2 Byte							

Der "Variable Header" des "PUBCOMP"-Pakets enthält die Paket-ID.

Das "PUBCOMP"-Paket hat keinen "Payload".

3.1.7 MQTT-Sub-Mechanismus

Sobald sich ein MQTT-Client mit dem MQTT-Broker verbunden hat, kann er Abonnements erstellen oder abmelden.

"SUBSCRIBE"-Paket

Um ein Abonnement zu erstellen, nutzt der MQTT-Client das "SUBSCRIBE"-Paket. Eine Liste der Topics, die der MQTT-Client abonnieren möchte, ist im "Payload" untergebracht.

Tabelle 3-10 zeigt den Aufbau des "Fixed Header" des "SUBSCRIBE"-Pakets.

Tabelle 3-10

Fixed Header								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Byte 1	Kennungsnummer für MQTT-Kontrollpakettyp = 8 (dez)				Reserve			
	1	0	0	0	0	0	1	0
Byte 2	Remaining Length: Die Anzahl der folgenden Bytes nach dem "Fixed Header" = "Variable Header" + "Payload"							

Hinweis

Die Reserve-Bits im "Fixed Header" müssen wie folgt gesetzt werden:

- Bit 3 = 0
- Bit 2 = 0
- Bit 1 = 1
- Bit 0 = 0

Der "Variable Header" des "SUBSCRIBE"-Pakets enthält die Paket-ID.

Tabelle 3-11 zeigt den Aufbau des "Payload" des "SUBSCRIBE"-Pakets.

Tabelle 3-11

Payload								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Topic-Name								
Byte 1	Länge MSB							
Byte 2	Länge LSB							
Byte 3…n	Topic-Name							
Angeforderte Servicequalität QoS								
Byte n+1	Reserve							QoS-Level Mögliche Werte: • 0 (dez) • 1 (dez) • 2 (dez)

"SUBACK"-Paket (Subscribe Acknowledgement)

Der MQTT-Broker beantwortet das "SUBSCRIBE"-Paket mit dem "SUBACK"-Paket.

Tabelle 3-12 zeigt den Aufbau des "Fixed Header" des "SUBACK"-Pakets.

Tabelle 3-12

Fixed Header								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Byte 1	Kennungsnummer für MQTT-Kontrollpakettyp = 9 (dez)				Reserve			
	1	0	0	1	0	0	0	0
Byte 2	Remaining Length: Die Anzahl der folgenden Bytes nach dem "Fixed Header" = "Variable Header" + "Payload"							

Der "Variable Header" des "SUBACK"-Pakets enthält die Paket-ID.

Tabelle 3-13 zeigt den Aufbau des "Payload" des "SUBACK"-Paket.

Tabelle 3-13

Payload								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Return Code								
Byte 1	• 0x00: Erfolgreich: Servicequalität maximal QoS 0 • 0x01: Erfolgreich: Servicequalität maximal QoS 1 • 0x02: Erfolgreich: Servicequalität maximal QoS 2 • 0x80: Fehler							

"UNSUBSCRIBE"-Paket

Um ein Abonnement abzumelden, nutzt der MQTT-Client das "UNSUBSCRIBE"-Paket. Eine Liste der Topics, die der MQTT-Client abmelden möchte, ist im "Payload" untergebracht.

Tabelle 3-14 zeigt den Aufbau des "Fixed Header" des "UNSUBSCRIBE"-Pakets.

Tabelle 3-14

Fixed Header								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Byte 1	Kennungsnummer für MQTT-Kontrollpakettyp = 10 (dez)				Reserve			
	1	0	1	0	0	0	1	0
Byte 2	Remaining Length: Die Anzahl der folgenden Bytes nach dem "Fixed Header" = "Variable Header" + "Payload"							

Hinweis

Die Reserve-Bits im "Fixed Header" müssen wie folgt gesetzt werden:

- Bit 3 = 0
- Bit 2 = 0
- Bit 1 = 1
- Bit 0 = 0

Der "Variable Header" des "UNSUBSCRIBE"-Pakets enthält die Paket-ID.

Tabelle 3-15 zeigt den Aufbau des "Payload" des "UNSUBSCRIBE"-Pakets.

Tabelle 3-15

Payload								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Topic-Name								
Byte 1	Länge MSB							
Byte 2	Länge LSB							
Byte 3…n	Topic-Name							

"UNSUBACK"-Paket

Der MQTT-Broker beantwortet das "UNSUBSCRIBE"-Paket mit dem "UNSUBACK"-Paket.

Tabelle 3-16 zeigt den Aufbau des "Fixed Header" des "UNSUBACK"-Pakets.

Tabelle 3-16

Fixed Header								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Byte 1	Kennungsnummer für MQTT-Kontrollpakettyp = 11 (dez)				Reserve			
	1	0	1	1	0	0	0	0
Byte 2	Remaining Length: Die Anzahl der folgenden Bytes nach dem "Fixed Header" = "Variable Header" = 2 Byte.							

Der "Variable Header" des "UNSUBACK"-Pakets enthält die Paket-ID.

Das "UNSUBACK"-Paket hat keinen "Payload".

3.1.8 MQTT-Ping-Mechanismus

Wenn das KeepAlive-Intervall größer als "0" ist, ist die KeepAlive-Funktion aktiv. Wenn die KeepAlive-Funktion aktiv ist, muss der MQTT-Client innerhalb des KeepAlive-Intervalls mindestens eine Nachricht an den MQTT-Broker schicken. Ist das nicht der Fall, muss der MQTT-Broker die Verbindung zum MQTT-Client beenden. Um solch einen Zwangsabbruch zu verhindern, muss der MQTT-Client vor Ablauf der KeepAlive-Zeit ein Ping-Request an den MQTT-Broker absetzen. Dafür dient das Kontrollpaket "PINGREQ".

"PINGREQ"-Paket

Tabelle 3-17 zeigt den Aufbau des "Fixed Header" des "PINGREQ"-Pakets

Tabelle 3-17

Fixed Header								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Byte 1	Kennungsnummer für MQTT-Kontrollpakettyp = 12 (dez)				Reserve			
	1	1	0	0	0	0	0	0
Byte 2	Remaining Length: Die Anzahl der folgenden Bytes nach dem "Fixed Header" = 0 Byte.							

Das "PINGREQ"-Paket hat keinen "Variable Header" und keinen "Payload".

"PINGRESP"-Paket

Der MQTT-Broker beantwortet das "PINGREQ"-Paket mit dem "PINGRESP"-Paket und signalisiert dem MQTT-Client damit seine Verfügbarkeit.

Hinweis

Dieses Anwendungsbeispiel setzt eine aktive KeepAlive-Funktion voraus. Das KeepAlive-Intervall muss größer als zwei Sekunden eingestellt sein.

Tabelle 3-18 zeigt den Aufbau des "Fixed Header" des "PINGRESP"-Pakets.

Tabelle 3-18

Fixed Header								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Byte 1	Kennungsnummer für MQTT-Kontrollpakettyp = 13 (dez)				Reserve			
	1	1	0	1	0	0	0	0
Byte 2	Remaining Length: Die Anzahl der folgenden Bytes nach dem "Fixed Header" = 0 Byte.							

Das "PINGRESP"-Paket hat keinen "Variable Header" und keinen "Payload".

3.1.9 MQTT-Verbindungsabbau

Ein MQTT-Client kann die Verbindung zu einem MQTT-Broker schließen, indem er ein "DISCONNECT"-Paket an den MQTT-Broker schickt. Nachdem der MQTT-Client das "DISCONNECT"-Paket gesendet und die Verbindung geschlossen hat, muss er keine weiteren MQTT-Kontrollpakete senden. Wenn der MQTT-Broker ein "DISCONNECT"-Paket empfängt, löscht er alle "Last Will and Testament"-Informationen. Da der MQTT-Client aktiv und aus freiem Willen die Verbindung abgebaut hat, verschickt der MQTT-Broker nicht seinen letzten Willen an die registrierten Abonnenten.

"DISCONNECT"-Paket

Tabelle 3-19 zeigt den Aufbau des "Fixed Header" des "DISCONNECT"-Pakets.

Tabelle 3-19

Fixed Header								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Byte 1	Kennungsnummer für MQTT-Kontrollpakettyp = 14 (dez)				Reserve			
	1	1	1	0	0	0	0	0
Byte 2	Remaining Length: Die Anzahl der folgenden Bytes nach dem "Fixed Header" = 0 Byte.							

Das "DISCONNECT"-Paket hat keinen "Variable Header" und keinen "Payload".

3.2 Details zur Funktionsweise des FB "LMQTT_Client"

3.2.1 Voraussetzungen und Umsetzung

Für eine Kommunikationsbeziehung zwischen einem MQTT-Client und einem MQTT-Broker müssen folgende Voraussetzungen erfüllt sein:

1. Die TCP-Verbindung zum MQTT-Broker ist erfolgreich aufgebaut (Ausgang "status" = 16#7003 "STATUS_MQTT_CONNECTING").
2. Der Funktionsbaustein "LMQTT_Client" hat sich über die bestehende TCP-Verbindung als MQTT-Client am Broker angemeldet und sich mit diesem verbunden (Ausgang "status" = 16#7004 "STATUS_MQTT_CONNECTED").
3. Der Trigger zum Senden der Nachricht oder zum Erhalt der MQTT-Verbindung ("KeepAlive") ist aktiv. Abhängig von der gewünschten Qualitätssicherung wird die Nachricht an den Broker über die bestehende MQTT-Verbindung gesendet.

Hinweis

Ein MQTT-Verbindungsaufbau ist nur möglich, wenn die TCP-Verbindung zum MQTT-Broker erfolgreich aufgebaut ist und anschließend gehalten wird.

Eine MQTT-Nachricht oder ein KeepAlive kann nur gesendet werden, wenn eine TCP- und MQTT-Verbindung zum MQTT-Broker besteht.

Übersicht

Um die genannten Voraussetzungen zu erfüllen, wurden im Programm mehrere Zustandsautomaten realisiert:

- Zustandsautomat "STATE_MACHINE_FUNCTION_BLOCK_TCP": Verwaltung der TCP-Verbindung
- Zustandsautomat "MQTT_STATE_MACHINE": Verwaltung der MQTT-Verbindung, des Sende- und Empfangsvorgangs
- Zustandsautomat "MQTT_COMMANDS": Verwaltung der MQTT-Kontrollpakte

3.2.2 Zustandsautomat "STATE_MACHINE_FUNCION_BLOCK_TCP"

Der Zustandsautomat "STATE_MACHINE_FUNCTION_BLOCK_TCP" wird gestartet, wenn am Eingangsparameter "enable" eine positive Flanke erkannt wurde. Dieser Zustandsautomat hat folgende Funktionen:

- Er regelt den Aufbau der TCP-Verbindung.
- Er überwacht die bestehende TCP-Verbindung auf Verbindungsfehler, z. B. Kabelbruch.
- Er regelt den Abbau der TCP- und MQTT-Verbindung.
- Wenn ein Fehler aufgetreten ist oder am Eingangsparameter "enable" keine positive Flanke erkannt wurde, setzt er alle statischen Variablen und die anderen Zustandsautomaten in einen definierten Zustand.

Der Zustandsautomat "STATE_MACHINE_FUNCTION_BLOCK_TCP" beinhaltet folgende Zustände:

- FB_STATE_NO_PROCESSING
- FB_STATE_VALIDATE_INPUT
- FB_STATE_TCP_CONNECTING
- FB_STATE_OPERATING_MONITOR_TCP
- FB_STATE_RECONNECTING
- FB_STATE_DISABLING

Die Bedeutung der Zustände listet die folgende Tabelle.

Tabelle 3-20

Zustand	Beschreibung
FB_STATE_ NO_PROCCESSING	Der Zustandsautomat wartet solange in diesem Zustand, bis er eine positive Flanke am Eingangsparameter "enable" erkennt. Sobald eine positive Flanke am Eingangsparameter "enable" erkannt wird, wird der Zustandsautomat in den Zustand "FB_STATE_VALIDATE_INPUT" versetzt und der Ausgang "valid" wird auf den Wert "TRUE" gesetzt.
FB_STATE_ VALIDATE_INPUT	In diesem Zustand werden alle Verbindungsparameter eingelesen und ausgewertet. Der FB wechselt ohne Weiterschaltbedingung in den Zustand "FB_STATE_TCP_CONNECTING".
FB_STATE_ TCP_CONNECTING	In diesem Zustand wird die TCP-Verbindung zum MQTT-Broker aufgebaut. Wenn die TCP-Verbindung mit "TSEND_C" erfolgreich aufgebaut ist, wechselt der FB in den Zustand "FB_STATE_OPERATING_MONITOR_TCP" und am Ausgang "status" wird der Wert "16#7003" ausgegeben. Die TCP-Verbindung bleibt solange bestehen, bis sie mit "TSEND_C" abgebaut wird. Wenn beim Verbindungsaufbau ein Fehler auftritt, werden die Statusmeldungen zum Fehler an den Ausgängen "status" und "diagnostics" ausgegeben und der FB wechselt in den Zustand "FB_STATE_NO_PROCESSING".
FB_STATE_OPERATING_ MONITOR_TCP	In diesem Zustand werden folgende Aktionen ausgeführt: • MQTT-Kontrollpaket empfangen und auswerten • MQTT-Kontrollpakte senden • Der Zustand der TCP-Verbindung wird überwacht. Wenn die TCP-Verbindung abbricht, z. B. aufgrund eines Kabelbruchs, wechselt der FB in den Zustand FB_STATE_RECONENCTING". • Zustandsautomat "MQTT_STATE_MACHINE" verwalten (siehe Kapitel 3.2.3)
FB_STATE_RECONNECTING	Wenn die TCP-Verbindung wieder aufgebaut ist, wechselt der FB zurück in den Zustand "FB_STATE_OPERATING_MONITORING_TCP".
FB_STATE_DISABLING	In diesem Zustand wird die MQTT- und TCP-Verbindung abgebaut.

3.2.3 Zustandsautomat "MQTT_STATE_MACHINE"

Der Zustandsautomat "MQTT_STATE_MACHINE" wird automatisch gestartet, wenn die TCP-Verbindung aufgebaut ist und am Ausgang "status" der Wert "7003" ausgegeben wird. Dieser Zustandsautomat hat folgende Funktionen:

- Handshake-Verfahren zum Aufbau der MQTT-Verbindung regeln.
- Die Eingänge "publish", "subscribe", "unsubscribe" werden ausgewertet, um das zum Auftrag passende MQTT-Kontrollpaket zu senden.
- Zustandsautomat "MQTT_COMMANDS" verwalten.
- PING-Paket vor Ablauf des KeepAlive-Intervalls verschicken.

Der Zustandsautomat "MQTT_STATE_MACHINE" beinhaltet folgende Zustände:

- MQTT_CONNECT_STATE_NO_PROCESSING
- MQTT_CONNECT_STATE_BUILD_PAKET
- MQTT_CONNECT_STATE_SEND_PAKET_WAIT_FOR_CONNACK
- MQTT_CONNECT_STATE_CONNECTED

Die Bedeutung der Zustände listet die folgende Tabelle:

Tabelle 3-21

Zustand	Beschreibung
MQTT_CONNECT_ STATE_NO_PROCESSING	Wenn die TCP-Verbindung nicht aufgebaut ist oder wenn ein Fehler auftritt, ist der Zustandsautomat im Zustand "MQTT_CONNECT_NO_PROCESSING". Erst wenn eine TCP-Verbindung aufgebaut ist, wird automatisch die Weiterschaltbedingung auf den Zustand "MQTT_CONNECT_STATE_BUILD_PAKET" aktiviert.
MQTT_CONNECT_ STATE_BUILD_PAKET	In diesem Zustand wird die MQTT-Verbindung zum MQTT-Broker aufgebaut. Dafür wird ein "CONNECT"-Paket zusammengebaut und anschließend mit dem Baustein "TSEND_C" an den MQTT-Broker geschickt. Der Zustandsautomat wechselt in den Zustand "MQTT_CONNECT_STATE_SEND_PAKET_WAIT_FOR_ CONNACK".
MQTT_CONNECT_ STATE_SEND_PAKET_ WAIT_FOR_CONNACK	Der MQTT-Client erwartet ein "CONACK"-Paket" vom MQTT-Broker als Quittierung auf das "CONNECT"-Paket. Wenn der MQTT-Broker den Erhalt des "CONNECT"-Paket mit einem "CONNACK"-Paket bestätigt hat, wird der Ausgang "status" auf den Wert "16#7004" gesetzt. Die Statusanzeige "16#7004" signalisiert, dass die MQTT-Verbindung aufgebaut ist.
MQTT_CONNECT_ STATE_CONNECTED	In diesem Zustand werden folgende Aktionen ausgeführt: • Es wird geprüft, ob ein Sendeauftrag für eines der folgenden MQTT-Kontrollpakete vorliegt: - PUBLISH - SUBSCRIBE - UNSUBSCRIBE • Zustandsautomat "MQTT_COMMANDS" verwalten (siehe Kapitel 3.2.4) • KeepAlive-Intervall überwachen. Wenn das KeepAlive-Intervall bald endet, muss das MQTT-Kontrollpaket "PINGREQ" gesendet und das KeepAlive-Intervall wieder gestartet werden.

3.2.4 Zustandsautomat "MQTT_COMMANDS"

Der Zustandsautomat "MQTT_COMMANDS" wird nur durchlaufen, wenn sich der Zustandsautomat "MQTT_STATE_MACHINE" im Zustand "MQTT_CONNECT_STATE_CONNECTED" befindet. Denn hier wird entschieden, von welcher Stelle aus der Zustandsautomat "MQTT_COMMANDS" gestartet wird. Wenn ein Sendeanstoß für eine MQTT-Nachricht vorliegt, dann wird die Senderoutine aktiv. Nach jedem Sendeauftrag "publish", "subscribe" oder "unsubscribe" wird der Ausgang "done" des FB "LMQTT_Client" auf den Wert "True" gesetzt und am Ausgang "status" wird der Wert "16#0000" für 1 Zyklus angezeigt. Anschließend wechselt der Wert des Ausgangs "status" auf den Wert "16#7004". Erst wenn der Eingang "publish", "subscribe" oder "unsubscribe" auf den Wert "false" zurückgesetzt wird, wird auch der Ausgang "done" des FB "LMQTT_Client" wieder auf den Wert "false" zurückgesetzt.

Wenn die KeepAlive-Zeit in Kürze endet, wird die PING-Routine gestartet.

Der Zustandsautomat "MQTT_COMMANDS" beinhaltet folgende Zustände:

- MQTT_COMMAND_NO_PROCESSING
- MQTT_COMMAND_STATE_BUILD_PUBLISH
- MQTT_COMMAND_STATE_SEND_PUBLISH
- MQTT_COMMAND_STATE_BUILD_SUBSCRIBE
- MQTT_COMMAND_STATE_SEND_SUBSCRIBE
- MQTT_COMMAND_STATE_BUILD_UNSUBSCRIBE
- MQTT_COMMAND_STATE_SEND_UNSUBSCRIBE
- MQTT_COMMAND_STATE_SEND_PING
- MQTT_COMMAND_STATE_PING_RESP

Tabelle 3-22

Zustand	Beschreibung
MQTT_COMMAND_ NO_PROCESSING	Solange kein Sendeanstoß vorliegt oder das KeepAlive-Intervall nicht abläuft, ist der Zustand immer "MQTT_COMMAND_NO_PROCESSING".
MQTT_COMMAND_STATE_ BUILD_PUBLISH	Wenn im Zustand "MQTT_CONNECT_STATE_CONNECTED" am Eingang "publish" eine positive Flanke erkannt wird, wird der interne Zustandsautomat "MQTT_COMMANDS" in den Zustand "MQTT_COMMAND_STATE_BUILD_PUBLISH" versetzt. Wenn kein anderer Sendeauftrag läuft, wird ein "PUBLISH"-Paket oder ein "PUBREL"-Paket zusammengebaut und anschließend mit dem Baustein "TSEND_C" an den MQTT-Broker geschickt. Der Ausgang "status" wird auf den Wert "16#7006" gesetzt, um zu signalisieren, dass der MQTT-Push-Mechanismus läuft. Der Zustandsautomat wechselt in den Zustand "MQTT_COMMAND_STATE_SEND_PUBLISH".
MQTT_COMMAND_STATE_ SEND_PUBLISH	Abhängig vom QoS-Level erwartet der MQTT-Client eines der folgenden MQTT-Kontrollpakte als Quittierung auf das PUBLISH-Paket. • QoS = 0 (dez): Der Sendeauftrag endet an dieser Stelle. • QoS = 1 (dez): "PUBACK"-Paket". Wenn der MQTT-Broker den Erhalt des "PUBLISH"-Paket mit einem "PUBACK"-Paket bestätigt hat, ist der Sendeauftrag beendet. • QoS = 2 (dez): "PUBREC"-Paket. Wenn der MQTT-Broker den Erhalt des "PUBLISH"-Paket mit einem "PUBREC"-Paket bestätigt hat, sendet der MQTT-Client ein "PUBREL"-Paket als Bestätigung. Der MQTT-Client erwartet ein "PUBCOMP"-Paket" vom MQTT-Broker als Quittierung auf das "PUBREL"-Paket. Wenn der MQTT-Broker den Erhalt des "PUBREL"-Paket mit einem "PUBCOMP"-Paket bestätigt hat, ist der Sendeauftrag beendet. Wenn der Sendeauftrag beendet ist, werden folgende Aktionen ausgeführt: • Der Ausgang "status" für 1 Zyklus auf den Wert "16#0000" bevor er wieder auf den Wert "16#7004" gesetzt wird. Die Statusanzeige "16#7004" signalisiert, dass die MQTT-Verbindung aufgebaut ist und kein Auftrag aktiv ist. • Der Ausgang "done" wird auf den Wert "true" gesetzt. Erst wenn der Eingang "publish" auf den Wert "false" zurückgesetzt wird, wird auch der Ausgang "done" des FB "LMQTT_Client" wieder auf den Wert "false" zurückgesetzt. • Der Zustandsautomat wechselt in den Zustand "MQTT_COMMAND_STATE_NO_ PROCESSING".

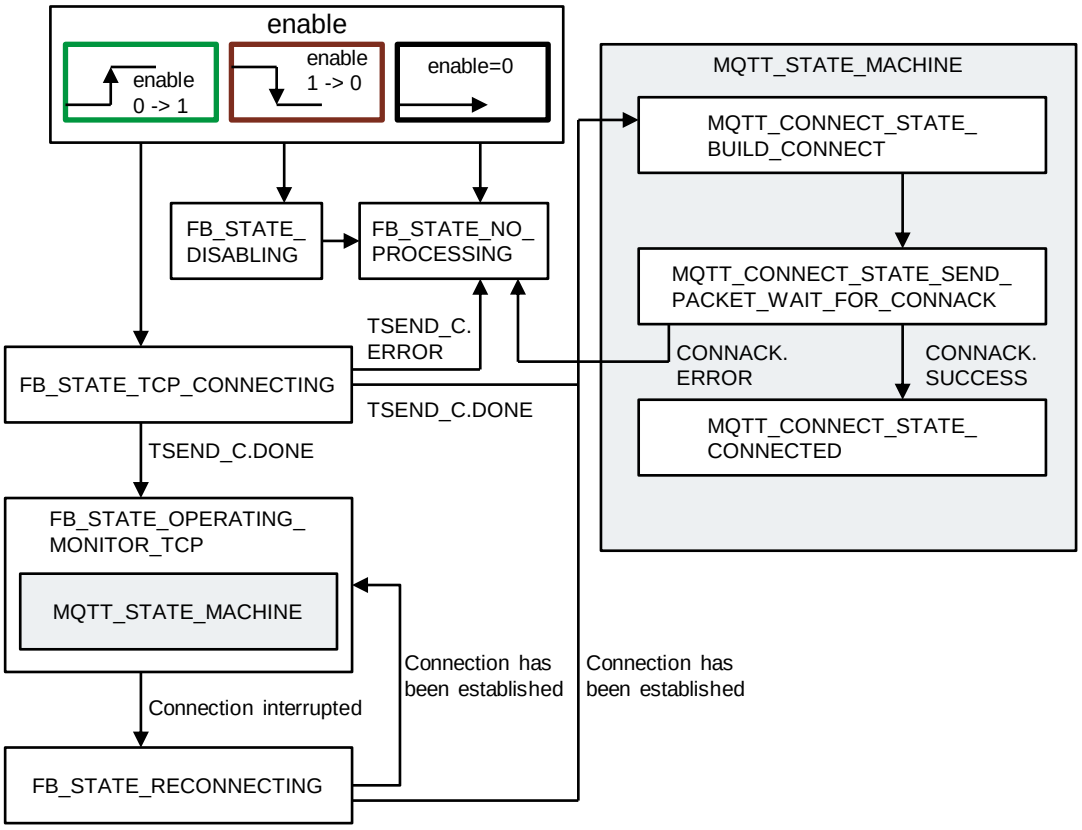
Zustand	Beschreibung
MQTT_COMMAND_STATE_ BUILD_SUBSCRIBE	Wenn im Zustand "MQTT_CONNECT_STATE_CONNECTED" am Eingang "subscribe" eine positive Flanke erkannt wird, wird der interne Zustandsautomat "MQTT_COMMANDS" in den Zustand "MQTT_COMMAND_STATE_BUILD_SUBSCRIBE" versetzt. Wenn kein anderer Sendeauftrag läuft, wird ein "SUBSCRIBE"-Paket zusammengebaut und anschließend mit dem Baustein "TSEND_C" an den MQTT-Broker geschickt. Der Ausgang "status" wird auf den Wert "16#7008" gesetzt, um zu signalisieren, dass der MQTT-Sub-Mechanismus läuft. Der Zustandsautomat wechselt in den Zustand "MQTT_COMMAND_STATE_SEND_SUBSCRIBE".
MQTT_COMMAND_STATE_ SEND_SUBSCRIBE	Der MQTT-Client erwartet ein "SUBACK"-Paket" vom MQTT-Broker als Quittierung auf das "SUBSCRIBE"-Paket. Wenn der MQTT-Broker den Erhalt des "SUBSCRIBE"-Paket mit einem "SUBACK"-Paket bestätigt hat, werden folgende Aktionen ausgeführt: • Der Ausgang "status" wird für 1 Zyklus auf den Wert "16#0000" bevor er wieder auf den Wert "16#7004" gesetzt wird. Die Statusanzeige "16#7004" signalisiert, dass die MQTT-Verbindung aufgebaut ist und kein Auftrag aktiv ist. • Der Ausgang "done" wird auf den Wert "true" gesetzt. Erst wenn der Eingang "subscribe" auf den Wert "false" zurückgesetzt wird, wird auch der Ausgang "done" des FB "LMQTT_Client" wieder auf den Wert "false" zurückgesetzt. • Der Zustandsautomat wechselt in den Zustand "MQTT_COMMAND_STATE_NO_ PROCESSING".
MQTT_COMMAND_STATE_ BUILD_UNSUBSCRIBE	Wenn im Zustand "MQTT_CONNECT_STATE_CONNECTED" am Eingang "unsubscribe" eine positive Flanke erkannt wird, wird der interne Zustandsautomat "MQTT_COMMANDS" in den Zustand "MQTT_COMMAND_STATE_BUILD_UNSUBSCRIBE" versetzt. Wenn kein anderer Sendeauftrag läuft, wird ein "UNSUBSCRIBE"-Paket zusammengebaut und anschließend mit dem Baustein "TSEND_C" an den MQTT-Broker geschickt. Der Ausgang "status" wird auf den Wert "16#7010" gesetzt, um zu signalisieren, dass der MQTT-Sub-Mechanismus läuft. Der Zustandsautomat wechselt in den Zustand "MQTT_COMMAND_STATE_SEND_UNSUBSCRIBE".

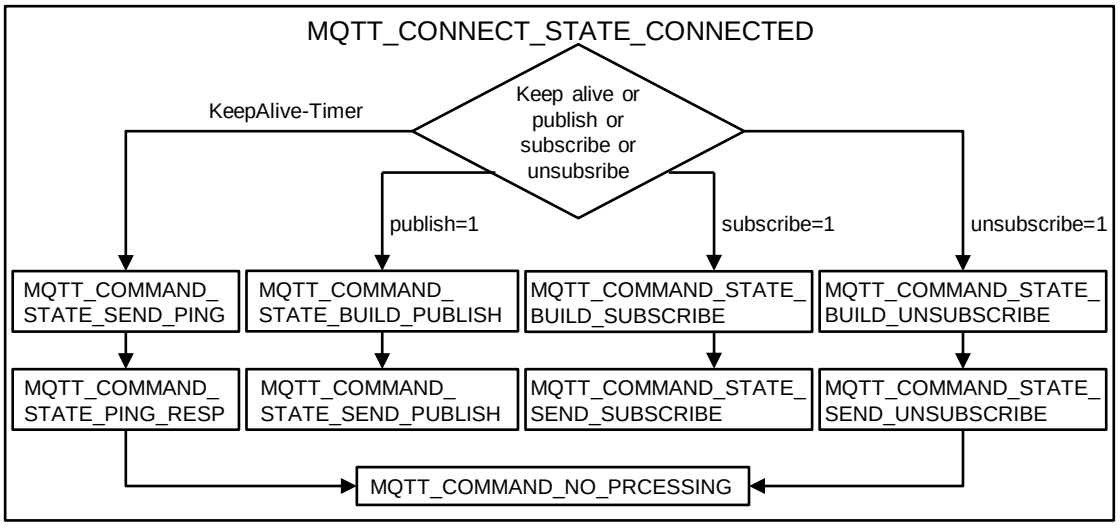
Zustand	Beschreibung
MQTT_COMMAND_STATE_ SEND_UNSUBSCRIBE	Der MQTT-Client erwartet ein "UNSUBACK"-Paket" vom MQTT-Broker als Quittierung auf das "UNSUBSCRIBE"-Paket. Wenn der MQTT-Broker den Erhalt des "UNSUBSCRIBE"-Paket mit einem "UNSUBACK"-Paket bestätigt hat, werden folgende Aktionen ausgeführt: • Der Ausgang "status" wird für 1 Zyklus auf den Wert "16#0000" bevor er wieder auf den Wert "16#7004" gesetzt wird. Die Statusanzeige "16#7004" signalisiert, dass die MQTT-Verbindung aufgebaut ist und kein Auftrag aktiv ist. • Der Ausgang "done" wird auf den Wert "true" gesetzt. Erst wenn der Eingang "unsubscribe" auf den Wert "false" zurückgesetzt wird, wird auch der Ausgang "done" des FB "LMQTT_Client" wieder auf den Wert "false" zurückgesetzt. • Der Zustandsautomat wechselt in den Zustand "MQTT_COMMAND_STATE_NO_ PROCESSING".
MQTT_COMMAND_STATE_ SEND_PING	Wenn im Zustand "MQTT_CONNECT_STATE_CONNECTED" das KeepAlive-Intervall abgelaufen ist, wird der interne Zustandsautomat "MQTT_COMMANDS" in den Zustand "MQTT_COMMAND_STATE_SEND_PING" versetzt und das KeepAlive-Intervall wird neu gestartet. Wenn kein anderer Sendeauftrag läuft, wird ein "PING"-Paket zusammengebaut und anschließend mit dem Baustein "TSEND_C" an den MQTT-Broker geschickt. Der Ausgang "status" wird auf den Wert "16#7005" gesetzt, um zu signalisieren, dass der MQTT-Ping-Mechanismus läuft. Der Zustandsautomat wechselt in den Zustand "MQTT_COMMAND_STATE_PING_RESP".
MQTT_COMMAND_STATE_ PING_RESP	Der MQTT-Client erwartet ein "PINGRESP"-Paket" vom MQTT-Broker als Quittierung auf das "PING"-Paket. Wenn der MQTT-Broker den Erhalt des "PING"-Paket mit einem "PINGRESP"-Paket bestätigt hat, wechselt der Zustandsautomat in den Zustand "MQTT_COMMAND_STATE_NO_PROCESSING" und der Ausgang "status" wird wieder auf den Wert "16#7004" gesetzt. Die Statusanzeige "16#7004" signalisiert, dass die MQTT-Verbindung aufgebaut ist.

3.2.5 Funktionsdiagramm

Folgende Abbildung zeigt das Diagramm der Funktionsweise mit den drei Zustandsautomaten.

Abbildung 3-2





4 Anhang

4.1 Service und Support

SiePortal

Die integrierte Plattform für Produktauswahl, Einkauf und Support – und Verbindung von Industry Mall und Online Support. Die neue Startseite ersetzt die bisherigen Startseiten der Industry Mall sowie des Online Support Portals (SIOS) und fasst diese zusammen.

- Produkte & Services
  Unter Produkte & Services finden Sie alle unsere Angebote, die bisher im Mall Katalog verfügbar waren.
- Support
  Im Bereich Support finden Sie alle Informationen, die für die Lösung technischer Probleme mit unseren Produkten hilfreich sind.
- mySieportal
  mySiePortal ist Ihr persönlicher Bereich, der Funktionen, wie z.B. die Warenkorbverwaltung oder die Bestellübersicht anzeigt. Den vollen Funktionsumfang sehen Sie hier erst nach erfolgtem Login.

Das SiePortal rufen Sie über diese Adresse auf:
sieportal.siemens.com

Technical Support

Der Technical Support von Siemens Industry unterstützt Sie schnell und kompetent bei allen technischen Anfragen mit einer Vielzahl maßgeschneiderter Angebote – von der Basisunterstützung bis hin zu individuellen Supportverträgen.

Anfragen an den Technical Support stellen Sie per Web-Formular:
support.industry.siemens.com/cs/my/src

SITRAIN – Digital Industry Academy

Mit unseren weltweit verfügbaren Trainings für unsere Produkte und Lösungen unterstützen wir Sie praxisnah, mit innovativen Lernmethoden und mit einem kundenspezifisch abgestimmten Konzept.
Mehr zu den angebotenen Trainings und Kursen sowie deren Standorte und Termine erfahren Sie unter:
siemens.de/sitrain

Industry Online Support App

Mit der App "Industry Online Support" erhalten Sie auch unterwegs die optimale Unterstützung.
Die App ist für iOS und Android verfügbar:

Erhältlich im App Store

ANDROID APP ON Google play

4.2 Links und Literatur

Tabelle 4-1

Nr.	Thema
\1\	Siemens Industry Online Support https://support.industry.siemens.com
\2\	Link auf die Beitragsseite des Anwendungsbeispiels https://support.industry.siemens.com/cs/ww/de/view/109748872
\3\	MQTT Spezifikation http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html

4.3 Änderungsdokumentation

Tabelle 4-2

Version	Datum	Änderung
V1.0	07/2017	Erste Ausgabe
V1.1	08/2018	Bibliothek "LMqttQdn" eingefügt.
V2.0	08/2019	Subscribe-Mechanismus ergänzt
V2.1	12/2019	Update auf TIA Portal V16
V3.0	03/2021	FB "LMQTT_Client" V3.0 in die Bibliotheken für Kommunikation für SIMATIC Controller integriert und Dokumentation für FB "LMQTT_Client" V3.0 angepasst
V3.1	04/2024	Update auf FB "LMQTT_Client" V4.0.4 und Update auf TIA Portal V17