TIM Interface

DLL

Funktionsbeschreibung

Version 1.0.0

Inhaltsverzeichnis

Version 1.0.0....1
1 Einleitung....1
2 Voraussetzungen....1
3 Installation....1
4 Funktionen der DLL....2
4.1 Einbinden der DLL in das Anwendungsprogramm....2
4.2 Funktionsergebnisse und Fehlermeldungen....2
4.3 Funktionsaufrufe....3
4.3.1 LOAD_INTERFACE....3
4.3.2 TIM_SET_USS....3
4.3.3 TIM_ADD_ITEM....4
4.3.4 TIM_TRANSACTION....6
4.3.5 TIM_RESET_ITEMS....8
4.3.6 TIM_REPORT....8
4.3.7 TIM_REPORT_SPAN....10
4.3.8 TIM_BOOKED_MONTHS....11
4.3.9 TIM_INFORMATION....11
4.3.10 TIM_CERTIFICATE....12
4.3.11 TIM_ACTIVATE....12
4.3.12 TIM_DEACTIVATE....13
4.3.13 SET_DEBUG_MODE....14
5 Anhang A VBA Beispiele....15
5.1 DLL einbinden....15
5.2 Funktionsaufrufe....18
5.3 Excel Arbeitsmappe....20
6 Anhang B Übersicht der Fehlermeldungen....21
7 Anhang C C++ Interface....22

1 Einleitung

Mit der TIMInterface.dll wird dem Anwender eine einfache Möglichkeit der Kommunikation mit einem TIM (Tax Identification Module) geboten. Das Ansprechen der Hardware der Smartcard sowie die Bedienung des Softwareinterfaces werden in einfachen Funktionen zusammengefasst und abgebildet. Mit der Software erhält man die Möglichkeit alle notwendigen Informationen aus dem TIM auszulesen, Buchungen durchzuführen und Berichte zu erstellen. Damit kann man Kassenanwendungen schnell um eine Schnittstelle für das TIM zu erweitern.

2 Voraussetzungen

Die TIMInterface.dll wurde für den Einsatz unter Windows entwickelt. Ein Kartenleser, welcher den PC/SC-Standard unterstützt, muss installiert sein.

Ob die Systemvoraussetzungen erfüllt sind, kann leicht mit den auf www.inoda.eu zur Verfügung gestellten Demoprogrammen für das TIM, wie TIMi Explorer, überprüft werden.

In dieser Funktionssbeschreibung wird auf die INSIKA TIM Schnittstellendokumenttation V.2.1.0 und das INSIKA Profil Registrierkasse V.2.1.0 Bezug genommen. Deshalb empfiehlt sich ein Registrieren auf www.insika.de, um ein kostenloses, aktuelles Exemplar der TIM-Spezifikationen zu erhalten.

Die Funktionen der TIMInterface.dll sind mit jeder geeigneten Hochsprache nutzbar. In der Dokumentation werden Beispiele für die Verwendung in VBA gebracht. Für andere Hochsprachen sind entsprechende Anpassungen selbst durchzuführen. In Anhang C C++ Interface wird das Vorgehen exemplarisch für die Sprache C++ aufgezeigt.

Es werden TIM der Version T.1.1.0 und V.2.1.0 mit dem Singnaturalgorithmus ECDSA, 192 bit NIST-Kurven, SHA-1 unterstützt.

3 Installation

Das Programmpaket besteht aus folgenden Dateien:

TIMInterface.dll	Basisdatei, um die es in dieser Beschreibung geht
TIMHWcard.dll	Schnittstelle zu einem PC/SC kompatiblen Kartenleser
TIMdllVBA.xlsm	Beispiel für den Einsatz der DLL in VBA und einem Excelarbeitsblatt
TIMInterface.h	Headerdatei für einen Einsatz in C++
TIMInterface.cpp	Interfacedatei für einen Einsatz in C++
DLL-Beschreibung.pdf	Dieses Dokument

Die DLL‘s TIMInterface.dll und TIMHWcard.dll sind in ein Verzeichnis zu kopieren, auf welches auch die Anwendungsprogramme Zugriff haben.

4 Funktionen der DLL

4.1 Einbinden der DLL in das Anwendungsprogramm

Das Einbinden der Funktionen DLL in VBA, wie es z.B. in Excel genutzt wird, ist in Anhang A dargestellt. Es ist wichtig, dass die Anwendung Zugriff auf die DLL hat.

4.2 Funktionsergebnisse und Fehlermeldungen

Die Funktionsaufrufe der DLL sind in der Regel wie folgt aufzubauen:

Funktionsergebnis = Funktionsname (Parameter1 … Parameter n, optional Rückmeldung des TIM)

Jede Funktion wird mit mit einem oder mehreren benötigten Parametern aufgerufen. Ob die Funktion erfolgreich ausgeführt wurde oder ob Fehler aufgetreten sind, kann dem Funktionsergebnis entnommen werden. Eine Übersicht der möglichen Fehlermeldungen ist in Anhang B aufgeführt.

Die optionale Rückmeldung vom TIM enthält die Daten, welche vom TIM generiert wurden. Für diese Variable wird die Bezeichnung DataOut verwendet.

Wichtig!

DataOut ist ein Speicherbereich, welcher vom Anwendungsprogramm zur Verfügung gestellt werden muss und der ausreichend groß sein muss. Es empfiehlt sich einen Bereich von 4 KByte zu reservieren.

Die Rückmeldung des TIM in DataOut ist ein String, der aus einer oder mehreren Zeilen besteht. Jede Zeile wird mit einem Zeilenumbruch, ASCII-Zeichen 10 (hexadezimal 0A), abgeschlossen. Jede Zeile beginnt mit einem Tag, der die entsprechende Variable des TIM kennzeichnet. Nach einem Semikolon als Trennzeichen, folgt der Wert der Variablen.

Tag	;	Wert	LF

Beispiel für das Datenformat:

C0;03
C2;T.1.1.0
C4;INSIKA_TEST_ADM
C5;1
C1;3366001B5B5B5B5B7CFF2720AB151F02
C8;978

CD;03/2025
CB;95
CC;33

Die Bedeutung der Tags und deren Inhalte sind in den einzelnen Funktionsaufrufen beschrieben.

4.3 Funktionsaufrufe

Die Reihenfolge der Beschreibung der Funktionsaufrufe erfolgt nicht alphabetisch, sondern in der Reihenfolge, die für das Abbilden der Funktionalitäten einer Kasse notwendig ist.

4.3.1 LOAD_INTERFACE

Laden einer DLL, welche die Basisfunktionen der Kommunikation mit einer TIM abbildet. Aktuell gibt es nur eine DLL, TIMHWCard.dll, welche mit einem PCC/SC kompatiblen Kartenleser kommuniziert. In Zukunft können hier DLLs entwickelt werden, welche mit einer Cloud oder HSM zusammenarbeiten.

Function LOAD_INTERFACE (ByVal name As String) As Long

Als Parameter „name“ ist der String „TIMHWCard“ zu übergeben.

4.3.2 TIM_SET_USS

Für die internen Funktionen der DLL müssen initial die Umsatzsteuersätze eingetragen werden. Das Datenformat ist ein vierstelliger String ohne Trennzeichen, z.B. ist 7% als „0700“ zu übergeben. Führende Nullen können weggelassen werden. Ein Umsatzsteuersatz von 7% kann auch als String „700“ übergeben werden.
Im letzten Parameter ist anzugeben, ob mit Exklusiv-Steuer gearbeitet („y“ oder „n“) wird.

```
Function TIM_SET_USS    (ByVal USS1 As String, ByVal USS2 As String, _
                         ByVal USS3 As String, ByVal USS4 As String, _
                         ByVal USS5 As String, ByVal USS6 As String, _
                         ByVal VatADDON As String) As Long
```

Wobei

USS1 Umsatzsteuersatz für Container 1

…

USS6 Umsatzsteuersatz für Container 6 ist. Vergleiche auch Punkt 2.6.20 der TIM Schnittstellendokumentation.

VatADDON Merker „Exklusiv Steuer“. Vergleiche auch TIM Schnittstellendokumentation.

Beispiel:

TIM_SET_USS "1900", "0700", "0000", "0000", "1070", "0550", "n"

Der Container 1 arbeitet mit einem Umsatzsteuersatz von 19%, Container 2 mit 7%, Container 3 und 4 mit 0%, Container 5 mit 10,7% und Container 6 mit 5,5%. Die Kasse arbeitet mit „Inklusive Steuern“.

Hinweis

Die einmal getroffene Zuordnung der Umsatzsteuersätze zu den einzelnen Container sollte nur bei einem tatsächlichen Umsatzsteuerwechsel geändert werden, da hierbei automatisch ein Vermerk im TIM erfolgt. Siehe auch TIM Schnittstellendokumentation Punkt 8.2.1:

“Bei der ersten Buchung eines neuen Monats wird der Umsatzsteuersatz in den entsprechenden Container eingetragen. Falls im laufenden Monat ein neuer Wert für den Umsatzsteuersatz gespeichert wird, wird der entsprechende Merker "Änderung Umsatzsteuersatz" gesetzt. Veränderungen des Umsatzsteuersatzes werden nach Aussage der Finanzverwaltung nur zu Monatswechseln durchgeführt. Für eine Prüfung ist jederzeit bekannt, welcher Umsatzsteuersatz in welchem
Monat gültig ist. Das TIM überprüft den übergebenen Umsatzsteuersatz nicht in der Höhe sondern lediglich die darauf basierende Berechnung der Umsatzsteuer.“

4.3.3 TIM_ADD_ITEM

Interne Funktion der DLL, um einzelne Buchungspositionen zu übergeben. Diese Funktion ist für jede Buchungsposition aufzurufen.

Als Strings sind die Menge, die Maßeinheit, der Bezeichner, Rabatt („y“ oder „n“), Gutschein („y“ oder „n“) und der Preis in Cent für jeden Container anzugeben.

Die notwendige Zeichenersetzung im Bezeichner erfolgt innerhalb der DLL.

```
Function TIM_ADD_ITEM  (ByVal quantity As String, ByVal unit As String, _
                        ByVal name As String, ByVal is_discount As String, _
                        ByVal is_voucher As String, _
                        ByVal val1 As String, ByVal val2 As String, _
                        ByVal val3 As String, ByVal val4 As String, _
                        ByVal val5 As String, ByVal val6 As String, _
                        ByVal val7 As String, ByVal val8 As String) As Long
```

Wobei

quantity	Menge/Anzahl entsprechend Punkt 2.3.1 in INSIKA Profil Registrierkasse. Wird ein leerer String übergeben, so ist die Voreinstellung 1.
unit	Mengeneinheit entsprechend Punkt 2.3.2 in INSIKA Profil Registrierkasse.
	In der Funktion erfolgt ein automatisches Begrenzen der Länge des Strings auf maximal 8 Zeichen
name	Handelsübliche Bezeichnung entsprechend Punkt 2.3.3 in INSIKA Profil Registrierkasse.
	In der Funktion erfolgt ein automatisches Begrenzen der Länge des Strings auf maximal 16 Zeichen
is_discount	Rabatt/Aufschlag wird als Flag mit den Zeichen „y“ oder „n“ gesetzt. Die inhaltliche Bedeutung entspricht der Beschreibung in Punkt 2.3.4 in INSIKA Profil Registrierkasse.
is_voucher	Gutschein wird als Flag mit den Zeichen „y“ oder „n“ gesetzt. Die inhaltliche Bedeutung entspricht der Beschreibung in Punkt 2.3.4 in INSIKA Profil Registrierkasse.
val1	Der Preis für die Umsatzsteuerklasse 1 (Container 1). Die Angabe erfolgt in der kleinsten Währungseinheit, z.B. Eurocent. Die inhaltliche Bedeutung entspricht der Beschreibung in Punkt 2.3.6 in INSIKA Profil Registrierkasse. Die dort beschriebenen Anforderungen an das Datenformat werden durch entsprechenden Maßnahmen innerhalb der DLL berücksichtigt.
…	
val6	Der Preis für die Umsatzsteuerklasse 6 (Container 6).
val7	Der Preis bei einem Agenturgeschäft. Siehe auch TIM Schnittstellendokumentation Punkt 2.6.21.

val8	Der Preis bei einem Lieferschein. Siehe auch TIM Schnittstellendokumentation Punkt 2.6.21.

Beispiele:

TIM_ADD_ITEM "1", "kg", "Äpfel", "n", "n", "", "250", "", "", "", "", "", ""

Agenturgeschäft und Lieferschein

Die Implementation des Agenturgeschäftes und des Lieferscheins unterscheiden sich in den TIM Versionen 1.x und 2.x.

In TIM Version 1.x werden Agenturgeschäfte und Lieferschein nicht auf Umsatzsteuerklassen aufgeschlüsselt, sondern die Summe aus allen Umsatzsteuersätzen dieser Buchungsposition als Summenwert in val7 bzw. val8 übergeben. I. d. R. Ist das nur ein Wert.

Für TIM Version 2.x sind die Preise für die einzelnen Umsatzsteuerklassen zu übergeben sowie val7 oder val8 zu setzen. Dabei bedeutet ein beliebiger Wert größer NULL für val7, dass die Buchungsposition ein Agenturgeschäft ist.

Wird bei einer Buchungsposition val8 größer NULL übertragen, dann bedeutet das, dass die gesamte Buchung ein Lieferscheinumsatz darstellt. Die Umsatzsummenzähler von E1 bis E6 des **TIM** werden nicht summiert, Das TIM summiert die Gesamtsumme aus allen übergebenen Umsatzsteuersätzen der Buchung zum Umsatz in E8. Lieferscheinumsätze dürfen demzufolge nicht vom steuerpflichtigen Umsatz aus E1 bis E6 abgezogen werden.

4.3.4 TIM_TRANSACTION

Nachdem alle Buchungspositionen erfasst wurden, ist eine Buchung (TRANSACTION) durchzuführen. Es sind das Datum, die Zeit, der Bediener und die Flags („y“ oder „n“) für Exklusivsteuer und Trainingsmode zu übergeben.

```
Function TIM_TRANSACTION (ByVal year_YYYY As String, ByVal month_MM As String,_
                          ByVal day_DD As String, ByVal hour_hh As String, _
                          ByVal minutes_mm As String, _
                          ByVal operator As String, ByVal VatADDON As String,_
                          ByVal Training As String, ByVal DataOut As String) As Long
```

Wobei

year_YYYY	Aktuelles Datum, Angabe des Jahres in der Form YYYY
month_MM	Aktuelles Datum, Angabe des Monats in der Form MM
day_DD	Aktuelles Datum, Angabe des Tages in der Form DD
hour_hh	Aktuelle Zeit, Angabe der Stunden (24 Stunden Format) in der Form hh
minutes_mm	Aktuelle Zeit, Angabe der Minuten in der Form mm
operator	Bediener, die geforderte Zeichenersetzung und Längenbegrenzung erfolgt innehalb der DLL
VatADDON	Merker „Exklusiv Steuer“. Vergleiche auch TIM Schnittstellendokumentation.
Training	Merker „Trainingsmode“. Vergleiche auch TIM Schnittstellendokumentation.
DataOut	Vom Anwender zur Verfügung zu stellender Speicherbereich in dem das Ergebnis der Buchung gespeichert wird.

Das Ergebnis der TRANSACTION steht als String in DataOut. Es besteht aus mehreren Zeilen. Jede Zeile beginnt mit einem Tag gefolgt von einem „;“ als Trennzeichen und dem eigentlichen Wert.

Ergebnistabelle:

Tag	Bezeichner	Beispiel	Beschreibung
ITL	Item list	oAExoQJrZ6IFI3BmZWyyAiUM	Base64 kodierte Liste der Buchungspositionen. Kann für die Kurzform des Journals verwendet werden.
C7	TIM_HASH_TRANSACTION_ITEMS	5JIUT-BXUWK-TMV2E-B5GX7-TLX7I-PMZM5-V5	Hash der Buchungspositionen Base32 kodiert für den Ausdruck auf Beleg
REQ	TRANSACTION Request	ZQQgFgYBzgIQQMYDb ...	Base64 kodierte TRANSACTION an das TIM. Kann für die Kurzform des Journals verwendet werden.
RES	TRANSACTION Response	XAtERTgxMTI0MDk1Ms..	Base64 kodierte Antwort auf eine TRANSACTION an das TIM. Kann für die Kurzform des Journals verwendet werden.
C4	TIM_TP_ID	DE811240952	
C5	TIM_TP_ID_NO	15	

CB	TIM_SEQ_NO_ TRANSACTION	11546	
9E	TIM_SIGNATU RE	ZSAGW-CQMLR-XAUYI-...	Signatur Base32 kodiert für den Ausdruck auf Beleg
QR		ZQQgEwcEzgIHUcYIaWNobm …4nvjSS5Gw==	Base64 kodierte TRANSACTION Request und Antwort in einem Datensatz. Kann für die Bildung des QR Kodes genutzt werden.

Beispiel:

TIM_TRANSACTION “2016“, “07“, “15“, “16“, “45“,"Max", "n", "n", TIM_Result

Nach dem Aufruf der Funkion wird in jedem Fall, auch im Fehlerfall, die Liste der mit TIM_ADD_ITEM erfassten Buchungspositionen geleert.

4.3.5 TIM_RESET_ITEMS

Sollte die Erfassung der Buchungspositionen abgebrochen werden, so ist der interne Transaction-Puffer der DLL zurückzusetzen. Das geschieht mit der Funktion

Function TIM_RESET_ITEMS () As Long

Nach dem Aufruf der Funktion TIM_TRANSACTION erfolgt automatisch das Rücksetzen des internen Transaction-Puffers der DLL. Die Funktion TIM_RESET_ITEMS muss nicht aufgerufen werden.

4.3.6 TIM_REPORT

Einen Tagesabschluss kann man mit der Funktionen TIM_REPORT erhalten. Dieser kann mit oder ohne Signatur erfolgen. Als Parameter sind das Datum, die Zeit und das Flag („y“ oder „n“), ob signiert werden soll, zu übergeben.

```
Function TIM_REPORT     (ByVal year_YYYY As String, ByVal month_MM As String, _
                         ByVal day_DD As String, ByVal hour_hh As String, _
                         ByVal minutes_mm As String, ByVal signed As String, _
                         ByVal DataOut As String) As Long
```

Wobei

year_YYYY	Aktuelles Datum, Angabe des Jahres in der Form YYYY
month_MM	Aktuelles Datum, Angabe des Monats in der Form MM
day_DD	Aktuelles Datum, Angabe des Tages in der Form DD
hour_hh	Aktuelle Zeit, Angabe der Stunden (24 Stunden Format) in der Form hh
minutes_mm	Aktuelle Zeit, Angabe der Minuten in der Form mm
signed	„y“ Tagesabschluss mit Signatur „n“ Tagesabschluss ohne Signatur
DataOut	Vom Anwender zur Verfügung zu stellender Speicherbereich, in dem das Ergebnis des Tagesabschlusses gespeichert wird.

Das Ergebnis des Tagesabschlusses steht als String in DataOut. Das Ergebnis besteht aus mehreren Zeilen. Jede Zeile beginnt mit einem Tag gefolgt von einem „;“ als Trennzeichen und dem eigentlichen Wert.

Ergebnistabelle:

Tag	Bezeichner	Beispiel	Beschreibung
REQ	REPORT Request	ZQQgFgYBzgIQQMYDb ...	Base64 kodierter REPORT-Anforderung an das TIM. Kann für die Kurzform des Journals verwendet werden.
RES	REPORT Response	XAtERTgxMTI0MDk1Ms..	Base64 kodierte Antwort auf eine REPORT-Anforderung an das TIM. Kann für die Kurzform des Journals verwendet werden.
C0	TIM_LIFECYCLE	3	Lebenszyklus des TIM
C4	TIM_TP_ID	DE811240952	Steuerpflichtigen-ID
C5	TIM_TP_ID_NO	15	lfd. Nummer des TIM
CC	TIM_SEQ_NO_ REPORT	8	TIM Sequenznummer eines signierten Tagesabschlusses
D2	TIM_SEQ_NO_ TRANSACTION_ FIRST	1	TIM Sequenznummer der ersten Buchung einer Umsatzperiode
D3	TIM_SEQ_NO_ TRANSACTION_ LAST	19418	TIM Sequenznummer der letzten Buchung einer Umsatzperiode

E1D8	TIM_TURNOVER	859407558	E1 steht für Container E1 D8 ist der Parameter im Container E1
Weitere Elemente und Container entsprechend „INSIKA TIM Schnittstellendokumentation V.2.1.0“ Tabelle 3-14 ….			
9E	TIM_SIGNATURE	ZSAGW-CQMLR-XAUYI-...	Signatur Base32 kodiert für den Ausdruck auf Beleg

4.3.7 TIM_REPORT_SPAN

Mit dieser Funktion kann ein Bericht über einen Zeitraum angefordert werden. Dieser enthält keine Signatur.

Function TIM_REPORT_SPAN (ByVal year_YYYY As String, ByVal month_MM As String,_
ByVal day_DD As String, _
ByVal hour_hh As String, ByVal minutes_mm As String, _
ByVal first_month_YYYYMM As String, _
ByVal last_month_YYYYMM As String, _
ByVal DataOut As String) As Long

Wobei

year_YYYY	Aktuelles Datum, Angabe des Jahres in der Form YYYY
month_MM	Aktuelles Datum, Angabe des Monats in der Form MM
day_DD	Aktuelles Datum, Angabe des Tages in der Form DD
hour_hh	Aktuelle Zeit, Angabe der Stunden (24 Stunden Format) in der Form hh
minutes_mm	Aktuelle Zeit, Angabe der Minuten in der Form mm
first_month_YYYYMM	Der Startmonat des Berichts in einer Kombination von Monats- und Jahresangabe der Form YYYYMM
last_month_YYYYMM	Der letzter Monat des Berichts in einer Kombination von Monats- und Jahresangabe der Form YYYYMM
DataOut	Vom Anwender zur Verfügung zu stellender Speicherbereich, in dem das Ergebnis des Berichtes gespeichert wird.

Das Ergebnis des Berichtes steht als String in DataOut. Es besteht aus mehreren Zeilen. Jede Zeile beginnt mit einem Tag gefolgt von einem „;“ als Trennzeichen und dem eigentlichen Wert.

Das Ergebnis entspricht dem eines Berichtes 4.3.6 ohne Signatur

4.3.8 TIM_BOOKED_MONTHS

Diese Funktion erlaubt es die auf dem TIM gebuchten Monate auszulesen.

Die Funktion entspricht inhaltlich „TIM Booked Months“ wie in Punkt 3.2.3 der TIM Schnittstellendokumentation beschrieben.

Function TIM_BOOKED_MONTHS (ByVal DataOut As String) As Long

Die gebuchten Monate stehen als String in DataOut. Es besteht aus mehreren Zeilen. Jede Zeile beginnt mit einem Tag gefolgt von einem „;“ als Trennzeichen und dem eigentlichen Wert.

Ergebnistabelle:

Tag	Bezeichner	Beispiel	Beschreibung
CD	TIM_DATE	03/2015	Zeitpunkt des ersten Monats, in denm gebucht werden kann
CF	TIM_MONTH	0 1 4 5 7 8 10 11	Liste der Monate mit Umsatz. Die einzelnen Werte sind durch ein Leerzeichen getrennt, ein Monat mit Umsatz wird als Offset zum ersten Monat, in dem gebucht werden kann, übertragen

4.3.9 TIM_INFORMATION

Die Basisinformationen des TIM können mit der Funktion TIM_INFORMATION ausgelesen werden. Die Funktion entspricht inhaltlich dem „TIM Status extended“ wie in Punkt 3.2.2 der TIM Schnittstellendokumentation beschrieben.

Function TIM_INFORMATION (ByVal DataOut As String) As Long

Das Ergebnis steht als String in DataOut. Es besteht aus mehreren Zeilen. Jede Zeile beginnt mit einem Tag gefolgt von einem „;“ als Trennzeichen und dem eigentlichen Wert.

Ergebnistabelle:

Tag	Bezeichner	Beispiel	Beschreibung
C0	TIM_LIFECYCLE	3	Lebenszyklus des TIM
C2	TIM_VERSION	T.1.1.0	TIM-Version
C4	TIM_TP_ID	DE811240952	Steuerpflichtigen-ID
C5	TIM_TP_ID_NO	15	lfd. Nummer des TIM
C1	TIM_TP_ID	DE811240952	Seriennummer des TIM
C8	TIM_CURRENCY	978	Währungscode
CD	TIM_DATE	09/2023	Datum (des letzten buchbaren Monats)
CB	TIM_SEQ_NO_ TRANSACTION	30	Sequenznummer für Buchungen (aktuell)
CC	TIM_SEQ_NO_ REPORT	8	Sequenznummer für Tagesabschlüsse (aktuell)

4.3.10 TIM_CERTIFICATE

Erlaubt es das auf dem TIM gespeicherte Zertifikat auszulesen.

Function TIM_CERTIFICATE (ByVal DataOut As String) As Long

Das Ergebnis des Reports steht als String in DataOut.

Tag	Bezeichner	Beispiel	Beschreibung
CRT	Zertifikat	MIIDGjCCAgKgAwI ...	Base64 kodiertes Zertifikat des TIM. Kann im Journal verwendet werden.

4.3.11 TIM_ACTIVATE

Erstmaliges Aktivieren des TIM mit Hilfe der Transport-PIN, siehe auch Punkt 3.5.4 der TIM Schnittstellendokumentation.

Function TIM_ACTIVATE (ByVal year_YYYY As String, ByVal month_MM As String,_
ByVal day_DD As String, _

ByVal PIN As String, _

ByVal DataOut As String) As Long

Wobei

year_YYYY	Aktuelles Datum, Angabe des Jahres in der Form YYYY
month_MM	Aktuelles Datum, Angabe des Monats in der Form MM
day_DD	Aktuelles Datum, Angabe des Tages in der Form DD
PIN	Transport-PIN der Karte
DataOut	Vom Anwender zur Verfügung zu stellender Speicherbereich, in dem der, in Folge einer erfolgreichen Aktivierung, generierte Tagesabschluss gespeichert wird. Das Format entspricht dem eines signierten Tagesabschusses 4.3.6.

4.3.12 TIM_DEACTIVATE

Achtung!

Dieser Befehl deaktiviert das TIM dauerhaft! Eine erneute Aktivierung ist nicht möglich.

Siehe auch Punkt 3.5.5 der TIM Schnittstellendokumentation.

Function TIM_DEACTIVATE (ByVal year_YYYY As String, ByVal month_MM As String,_
ByVal day_DD As String, _
ByVal DataOut As String) As Long

Wobei

year_YYYY	Aktuelles Datum, Angabe des Jahres in der Form YYYY
month_MM	Aktuelles Datum, Angabe des Monats in der Form MM
day_DD	Aktuelles Datum, Angabe des Tages in der Form DD
DataOut	Vom Anwender zur Verfügung zu stellender Speicherbereich, in dem der, in Folge einer erfolgreichen Deaktivierung, generierte Tagesabschluss gespeichert wird. Das Format entspricht dem eines signierten Tagesabschusses 4.3.6.

4.3.13 SET_DEBUG_MODE

Zuschalten eines Monitorfensters, welches die Kommunikation mit dem TIM anzeigt.

Function SET_DEBUG_MODE (ByVal mode As Long) As Long

Wobei

mode = 1 das Monitorfenster einschaltet

mode = 2 das Monitorfenster ausschaltet

mode = 3 die Anzeige des Monitorfensters löscht

5 Anhang A VBA Beispiele

5.1 DLL einbinden

```
Declare Function LOAD_INTERFACE Lib "TIMinterface.dll" (ByVal name As String) As Long

Declare Function SET_DEBUG_MODE Lib "TIMinterface.dll" (ByVal mode As Long) As Long

Declare Function TIM_INFORMATION Lib "TIMinterface.dll" (ByVal DataOut As String) As Long

Declare Function TIM_BOOKED_MONTHS Lib "TIMinterface.dll" (ByVal DataOut As String) As Long

Declare Function TIM_CERTIFICATE Lib "TIMinterface.dll" (ByVal DataOut As String) As Long

Declare Function TIM_REPORT Lib "TIMinterface.dll"    (ByVal year_YYYY As String, ByVal month_MM As String, _
                                                       ByVal day_DD As String, _
                                                       ByVal hour_hh As String, ByVal minutes_mm As String, _
                                                       ByVal signed As String, _
                                                       ByVal DataOut As String) As Long

Declare Function TIM_REPORT_SPAN Lib "TIMinterface.dll"    (ByVal year_YYYY As String, ByVal month_MM As String, _
                                                            ByVal day_DD As String, _
                                                            ByVal hour_hh As String, ByVal minutes_mm As String, _
```

```
                                        ByVal first_month_YYYYMM As String, ByVal last_month_YYYYMM As String, _
                                        ByVal DataOut As String) As Long

Declare Function TIM_ACTIVATE Lib "TIMinterface.dll"(ByVal year_YYYY As String, ByVal month_MM As String, ByVal day_DD As String, _
                                        ByVal PIN As String, _
                                        ByVal DataOut As String) As Long

Declare Function TIM_DEACTIVATE Lib "TIMinterface.dll"      (ByVal year_YYYY As String, ByVal month_MM As String, _
                                        ByVal day_DD As String, _
                                        ByVal DataOut As String) As Long

Declare Function TIM_SET_USS Lib "TIMinterface.dll"   (ByVal USS1 As String, ByVal USS2 As String, ByVal USS3 As String, _
                                        ByVal USS4 As String, ByVal USS5 As String, ByVal USS6 As String, _
                                        ByVal VatADDON As String) As Long

Declare Function TIM_ADD_ITEM Lib "TIMinterface.dll"        (ByVal quantity As String, ByVal unit As String, ByVal name As String, _
                                        ByVal is_discount As String, ByVal is_voucher As String, _
                                        ByVal val1 As String, ByVal val2 As String, ByVal val3 As String, _
                                        ByVal val4 As String, ByVal val5 As String, ByVal val6 As String, _
                                        ByVal val7 As String, ByVal val8 As String) As Long
```

```
Declare Function TIM_RESET_ITEMS Lib "TIMinterface.dll" () As Long

Declare Function TIM_TRANSACTION Lib "TIMinterface.dll" (ByVal year_YYYY As String, ByVal month_MM As String, _
                                                         ByVal day_DD As String, ByVal hour_hh As String, ByVal minutes_mm As String, _
                                                         ByVal operator As String, ByVal VatADDON As String, ByVal Training As String, _
                                                         ByVal DataOut As String) As Long
```

5.2 Funktionsaufrufe

Deklarieren der globalen Variablen TIM_Result mit der in 4.2 geforderten Größe für den Rückgabewert des TIM:

Dim TIM_Result As String * 4000

Laden einer zusätzlichen DLL, welche die Hardwareverbindung zu einer physisch vorhandenen TIM herstellt. Es ist zu beachten, dass nur der Name der DLL, ohne die Dateierweiterung „.dll“ anzugeben ist.

LOAD_INTERFACE ("TIMHWCard")

siehe 4.3.1

Einzelne Buchungspositionen erfassen

TIM_ADD_ITEM "1", "kg", "Äpfel", "n", "n", "", "250", "", "", "", "", "", ""

siehe 4.3.3

TIM mit Transport PIN aktivieren

TIM_ACTIVATE “2016“, “07“, “15“, “16“, “45“, “372984“, TIM_Result

siehe 4.3.11

TIM dauerhaft deaktivieren

TIM_ACTIVATE “2016“, “07“, “15“, “16“, “45“, TIM_Result

siehe 4.3.12

Die auf dem TIM gebuchten Monate auslesen

TIM_BOOKED_MONTHS (TIM_Result)

siehe 4.3.8

Das Zertifikat des TIM auslesen

TIM_CERTIFICATE (TIM_Result)

siehe 4.3.10

Zuschalten eines Monitorfensters, welches die Kommunikation mit dem TIM anzeigt.

SET_DEBUG_MODE (1)

siehe 4.3.13

Die Informationen bzgl. des Inhabers des TIM und andere Informationen auslesen

TIM_INFORMATION (TIM_Result)

siehe 4.3.9

Bericht (mit oder ohne Signatur) erstellen

TIM_REPORT “2016“, “07“, “15“, “16“, “45“, “y“, TIM_Result

siehe 4.3.6

Speicher der erfassten Buchungspositionen zurücksetzen

TIM_RESET_ITEMS

siehe 4.3.5

Bericht über einen Zeitraum z.B. von zwei Jahre abrufen

TIM_REPORT_SPAN “2016“, “07“, “15“, “16“, “45“, “201406“, “201606“, TIM_Result

siehe 4.3.7

Setzen der Umsatzsteuersätze. Der Container 1 arbeitet mit einem Umsatzsteuersatz von 19%, Container 2 mit 7%, Container 3 und 4 mit 0%, Container 5 mit 10,7% und Container 6 mit 5,5%. Die Kasse arbeitet mit „Inklusive Steuern“.

TIM_SET_USS "1900", "0700", "0000", "0000", "1070", "0550", "n"

siehe 4.3.2

Buchung durchführen

TIM_TRANSACTION “2016“, “07“, “15“, “16“, “45“,"Max", "n", "n", TIM_Result

siehe 4.3.4

5.3 Excel Arbeitsmappe

Die Datei TIMdllVBA.xlsm kann als Beispielanwendung für den Einsatz der Funktionen der DLL dienen. Zum Ausführen der Datei ist das Programm Excel notwendig. Es empfiehlt sich die Dateien TIMdllVBA.xlsm, TIMInterface.dll und TIMHWcard.dll in ein Verzeichnis zu kopieren.

Mit den einzelnen Schaltflächen können verschiedene Funktionen oder deren Kombination ausgelöst werden. Die Rückgabewerte werden nach ihren Tags sortiert dargestellt. Die von dem TIM generierte Antwort wird in einem extra Feld dargestellt.

Eine nähere Beschreibung kann den Kommentaren im VB Quelltext entnommen werden.

6 Anhang B Übersicht der Fehlermeldungen

In der nachfolgenden Tabelle sind die allgemeingültigen Funktionsergebnisse angegeben.

Fehlernummer als Hexadezimalwert	Bedeutung
0	Die Funktion wurde fehlerfrei ausgeführt
1	Das TIM ist durch einen anderen Prozess beschäftigt. Die Funktion kann z.Z. nicht ausgeführt werden
2	Falscher Parameter beim Ein/Ausschalten der Debugfunktion
3	Keinen Kartenleser gefunden
4	Kein TIM gefunden
5	Falsches Datenformat bei der Kommunikation mit dem TIM
6	Fehler beim Ausführen eines Kartenkommandos (APDU). Die Meldung des TIM ist ungleich 9000h. Details zum Fehler sind der Rückmeldung des TIM (DataOUT) und der Spezifikation des TIM zu entnehmen.
101	Library zur Kommunikation mit dem TIM nicht geladen
102	Library zur Kommunikation mit dem TIM nicht gefunden
103	Geforderte Funktion zur Kommunikation mit dem TIM nicht gefunden
104	Die Anwendung hat nicht genug Speicher in DataOut für die Daten des TIM zur Verfügung gestellt.
105	Fehler im Datumsformat
106	Fehler im Zeitformat
107	Fehler im Format des Bedieners
108	Fehler im Format des Mengenbezeichners
109	Fehler bei der Auswertung der Daten vom TIM
110	Fehler im Format des Umsatzsteuersatzes
111	Die Buchung (TIM_TRANSACTION) konnte nicht ausgeführt werden da der Umsatzsteuersatz noch nicht mit TIM_SET_USS gesetzt wurde
112	Allgemeiner Parameterfehler

7 Anhang C
C++ Interface

Um das Einbinden der DLL in eine C++ Umgebung zu erleichtern werden im Verzeichnis C++ die Dateien Interface.h und Interface.cpp zur Verfügung gestellt.

Es ist wichtig, dass die Anwendung Zugriff auf die DLL hat. Das erreicht man in dem sich die Datei TIMInterface.dll im Suchpfad befindet oder in der Datei TIMInterface.h in der Zeilenumbruch

LPCSTR DLLName = "TIMinterface.dll";

der korrekte Pfad zur DLL angegeben wird.

In der Datei TIMInterface.cpp sind die folgenden Funktionen definiert, die für den Umgang mit der DLL genutzt werden können.

unsigned TTIMInterface::Call_LOAD_INTERFACE(const char * Name)

unsigned TTIMInterface::Call_TIM_SET_USS(const char * USS1, const char * USS2,
const char * USS3, const char * USS4,
const char * USS5, const char * USS6,
const char * VatADDON)

unsigned TTIMInterface::CALL_TIM_ADD_ITEM(const char * quantity, const char * unit,
const char * name, const char * is_discount,
const char * is_voucher, const char * val1,
const char * val2, const char * val3,
const char * val4, const char * val5,
const char * val6, const char * val7,
const char * val8)

unsigned TTIMInterface::CALL_TIM_TRANSACTION(const char * year_YYYY,
const char * month_MM,
const char * day_DD,
const char * hour_hh,
const char * minutes_mm,
const char * operator_name,
const char * VatADDON,
const char * Training, char * DataOut)

unsigned TTIMInterface::CALL_TIM_RESET_ITEMS()

unsigned TTIMInterface::CALL_TIM_REPORT(const char * year_YYYY,
const char * month_MM, const char * day_DD,
const char * hour_hh, const char * minutes_mm,
const char * is_signed, char * DataOut)

```
unsigned TTIMInterface::CALL_TIM_REPORT_SPAN(const char * year_YYYY,
                                             const char * month_MM,
                                             const char * day_DD,
                                             const char * hour_hh,
                                             const char * minutes_mm,
                                             const char * first_month_YYYYMM,
                                             const char * last_month_YYYYMM,
                                             char * DataOut)

unsigned TTIMInterface::CALL_TIM_BOOKED_MONTHS(char * DataOut)

unsigned TTIMInterface::CALL_TIM_INFORMATION(char * DataOut)

unsigned TTIMInterface::CALL_TIM_CERTIFICATE(char * DataOut)

unsigned TTIMInterface::CALL_TIM_ACTIVATE(const char * year_YYYY,
                                          const char * month_MM,
                                          const char * day_DD, const char * PIN,
                                          char * DataOut)

unsigned TTIMInterface::CALL_TIM_DEACTIVATE(const char * year_YYYY,
                                            const char * month_MM,
                                            const char * day_DD,
                                            char * DataOut)

unsigned TTIMInterface::CALL_TIM_DEBUG_MODE(const unsigned mode)
```

Die Funktionalität der Funktionen, die Beschreibung der Parameter und der Funktionsergebnisse entspricht denen der analogen Funktionen wie sie für VBA in den vorherigen Kapiteln beschrieben wurden.