IP-S7-LINK

RFC 1006

für

PC – SIMATIC S7

TRAEGER
GmbH Industry Components
Am Steigacker 26, D-92694 Etzenricht
Tel:0961/48230-0 Fax:0961/48230-20
http://www.traeger.de

Version 1.24

Vorausetzungen:

Betriebssysteme:	MS-Windows 95 ,98, NT, 2000, XP,Vista Linux
Programmiersprachen	C, C++, Delphi, VisualBasic, C-Sharp, VB.Net, Access, Excel, PHP
Hardware	PC mit installiertem TCP/IP-Protokoll und Netzwerkkarte
SPS	Simatic S7 200/300/400 mit CP 243, CP343-1, CP-443-1, sowie LeanCP außerdem ProfiNet - CPU und S7-LAN

Lieferumfang:

Folgende Dateien können sich im Lieferumfang befinden:

Hauptverzeichnis	
Ips7lnk.pdf	diese Datei – Dokumentation
Version.htm	Datei über Fehlerbeseitigung

Verzeichnis ´CPP´	**Dateien für Visual CPP C++**
IPS7LNK.H	Header-Datei für C / C++
IPS7LNK.DLL	Treiber DLL
IPS7LNK.LIB	Lib-Datei zum Linken mit C++
IPS7DEMO.DSP	Projektdatei für Visual C++ V 6.00
IPS7DEMO.CPP	Beispielprogramm in ´C++´ einer Konsolenapplikation
IPS7DEMO.EXE	EXE-Datei der CPP-Demo

Verzeichnis ´Delphi´	**Dateien für Delphi**
IPS7LNK.PAS	Delphi-Header TPU im Quellcode
IPS7LNK.DLL	Treiber DLL
IPS7DEMO.exe	EXE-Datei der Delphidemo
IPS7DEMO.cfg IPS7DEMO.dof IPS7DEMO.dpr IPS7DEMO.res IPS7LNK.dcu main.dcu main.dfm main.pas OEM.BMP	Delphi - Projektdateien

Verzeichnis ' VisualBasic'	**Dateien für Visual Basic**
IPS7LNK.DLL	Treiber DLL, **Achtung: Für Visual Basic und Excel diese Datei ins Windowsverzeichnis kopieren !**
IPS7LNK.BAS	Header / Moduldatei für Visual Basic
IPS7DEMO.EXE	EXE-Datei der VB-Demo

IPS7DEMO.FRM IPS7DEMO.FRX IPS7DEMO.VBP IPS7DEMO.VBW	Visual Basic Projektdateien

Verzeichnis Excel	**Dateien für Excel**
IPS7LNK.DLL	Treiber DLL, **Achtung: Für Visual Basic und Excel diese Datei ins Windowsverzeichnis kopieren !**
IPS7LNK.BAS	Header/Moduldatei für Visualbasic
IPS7DEMO.XLS	Excel-Datei mit Makro für Demo

Verzeichnis ´PHP´	**Dateien für PHP**
ips7lnk_php.so	Modul für die PHP-Erweiterung, dieses Modul in das Extensionverzechnis der betreffenden PHP-Installation kopieren

Verzeichnis ´DotNet´	**Dateien für .Net**
DemoCSharp	Verzeichznis Demoprogramm C#
DemoVB.Net	Verzeichnis Demoprogramm VB.Net
NetFiles	Verzeichnis für die Assemblies der verschiedenen Frameworks (2.0/3.0/3.5)
ips7lnk.chm	Hilfedatei / Programmieranleitung für .Net Assemblies

Installation:

Windows: Die DLL ins Verzeichnis des Programms oder ins Systemverzeichnis kopieren.

Linux: Die o-Datei zu Ihrem Programm linken

.Net:

Für .Net entnehmen Sie die Dokumentation der ips7lnk.chm-Datei. Für .Net (C# und VB.Net) wurden alle Funktionen in eine Klasse eingebettet. Es liegen Assemblies für Framwork 2.0/3.0/3.5 bei. Diese Assemblies verwenden die ‚ips7lnk.dll'. Sorgen Sie dafür, daß ‚ips7lnk.dll' enweder im Programmverzeichnis Ihrer Applikation oder im Windowssystemverzeichnis zur Verfügung steht. Fügen Sie das gewünschte Assemblie ips7lnknet.dll einfach als Verweis in Ihre Applikation ein. Schon stehen sämtliche Funktionen zur Verfügung.

PHP:

Die Extionion ips7lnk_php.so in das Extensionverzeichnis der jeweiligen PHP-Installation kopieren

Über PHP.ini oder im Programm selbst dafür sorgen, daß das Modul geladen wird.

php.ini: extension = ips7lnk_php.so

im Programm: ld (´ips7lnk_php.so');

Funktionsweise:

IP-S7-LINK ist eine DLL für MS-Windows (95/98/2000/NT/XP/Vista) bzw. eine Library oder eine Erweiterung für Linux (C,C++,PHP), welche die Anbindung eines PC an Industrial Ethernet der SIMATIC S7 von Siemens ermöglicht. Mit einfachen Funktionen kann der Anwender schnell mit C, C++, Delphi, Visual Basic, Excel

oder auch PHP auf die Daten der SPS'en im Netz zugreifen. Zur Kopplung wird nur die IP-Adresse des CP sowie der Steckplatz der CPU im SPS-Rack benötigt. Sofort kann auf Merker. Eingänge, Ausgänge und auch Datenbausteine der SPS lesend oder auch schreiben zugegriffen werden.

Funktionsbeschreibung im Detail:

Bitte beachten Sie: Die Funktionen werden mit der Standard Socket -Schnittstelle ausgeführt, was zur Folge hat, dass die Funktion erst nach Erfüllung der Aufgabe zum Aufrufer zurückkehrt. Zum Asynchronen Betrieb rufen Sie diese Funktionen einfach von einem separaten Thread aus auf, welcher für die Kommunikation des System zuständig ist.

Folgende Funktionen stehen zur Verfügung:

Funktionen zur Initialisierung:

Funktion	*Funktion (PHP)*	*Beschreibung / Zweck*
IPS7Open	***ips7_open***	zur Initialisierung der Verbindung, dort wird nur Speicher vorbereitet. Beim ersten Aufruf der Lese- oder Schreibfunktionen wird, die TCP/IP-Verbindung automatisch gestartet. **Die Verbindung wird über den OP-Kanal hergestellt.**
IPS7OpenPG	***ips7open_openpg***	Version 1.17 zur Initialisierung der Verbindung, dort wird nur Speicher vorbereitet. Beim ersten Aufruf der Lese- oder Schreibfunktionen wird, die TCP/IP-Verbindung automatisch gestartet. **Die Verbindung wird über den PG-Kanal hergestellt.**
IPS7OpenS7200	***ips7_opens7200***	Version 1.21 zur Initialisierung der Verbindung, dort wird nur Speicher vorbereitet. Beim ersten Aufruf der Lese- oder Schreibfunktionen wird, die TCP/IP-Verbindung automatisch gestartet. **Die Verbindung wird zu einer S7-200 hergestellt.**

Aufrufparameter:

Nr.	***Datentyp***	***Datentyp (PHP)***	***Name***	***Funktion***
1	32-Bit Zeiger auf C-String	string	IPAdr	IP-Adresse der SPS im Format: xxx.xxx.xxx.xxx. Beispiel: “192.169.0.100“
2	32-Bit Wert unsigned	long	Rack	Nummer des Racks ,in dem die SPS-CPU gesteckt ist. Die Zählung beginnt mit „0“. Normalerweise 0 Bei S7-200 egal
3	32-Bit Wert unsigned	long	Slot	Nummer des Steckplatzes der CPU beginnend mit „1“, normalerweise „2“. Bei S7-200 egal
4	32-Bit Wert unsigned	long	RxTimeout	Timeout in Millisekunden für Warten auf TCP/IP-Paket von der SPS 0 bedeutet Standardeinstellung = 500 ms
5	32-Bit Wert unsigned	long	TxTimeout	Timeout in Millisekunden für Senden eines TCP/IP-Paketes an die SPS SPS 0 bedeutet Standardeinstellung = 500 ms

Nr.	***Datentyp***	***Datentyp (PHP)***	***Name***	***Funktion***
6	32-Bit Wert unsigned	long	ConTimeout	Timeout in Millisekunden für Warten auf Verbindungsaufbau mit SPS 0 bedeutet Standardeinstellung = 5000 ms (5sec.) muss bei Bedarf verlängert werden.

Funktion	*Funktion (PHP)*	*Beschreibung / Zweck*
IPS7OpenEx	***ips7_openex***	Version 1.23 zur Initialisierung der Verbindung, dort wird nur Speicher vorbereitet. Beim ersten Aufruf der Lese- oder Schreibfunktionen wird, die TCP/IP-Verbindung automatisch gestartet. **Es können durch die Wahl der Parameter OP/PG S7200 oder auch verbindungen über ein Subbnetz hergestellt werden.**

Aufrufparameter:

Nr.	***Datentyp***	***Datentyp (PHP)***	***Name***	***Funktion***
1	32-Bit Zeiger auf C-String	string	IPAdr	IP-Adresse der SPS im Format: xxx.xxx.xxx.xxx. Beispiel: 192.169.0.100
2	32-Bit Wert unsigned	long	Rack	Nummer des Racks ,in dem die SPS-CPU gesteckt ist. Die Zählung beginnt mit „0“. Normalerweise 0 Bei S7-200 egal
3	32-Bit Wert unsigned	long	Slot	Nummer des Steckplatzes der CPU beginnend mit „1“, normalerweise „2“. Bei S7-200 egal
4	32-Bit Wert unsigned	long	SubNetId	Subnetz-ID, wenn über ein Subnetzu zugegriffen werden soll. In der Step-S7 Software wird die Adresse z,B. so dargestellt: 0035 – 0001 geben Sie dann an 0x00350001 Wird nur bei AccesMode 10 oder 11 verwendet
5	32-Bit Wert unsigned	long	DstMPIAdr	ZielMPI-Adresse, wenn über ein Subnetz die Verbindung aufgebaut werden soll. Siehe AccessMode 10 und 11!

Nr.	*Datentyp*	*Datentyp (PHP)*	*Name*	*Funktion*
6	32-Bit Wert unsigned	long	AccessMode	Art des Zugriffs 0 = OP-Verbindung zur durch Rack und Slot angegebene CPU aufbauen 1 = PG-Verbindung zur durch Rack und Slot angegebene CPU aufbauen 2 = Verbindung zu einer S7-200 über gesteckten TCP/IP-CP 10 = OP-Verbindung über Subnetz, welches an der durch Rack und Slot angegebene CPU angeschlossen ist aufbauen, SubnetId und DstMPI Adresse sind anzugeben. 11 = PG-Verbindung über Subnetz, welches an der durch Rack und Slot angegebene CPU angeschlossen ist aufbauen, SubnetId und DstMPI Adresse sind anzugeben.
7	32-Bit Wert unsigned	long	RxTimeout	Timeout in Millisekunden für Warten auf TCP/IP-Paket von der SPS 0 bedeutet Standardeinstellung = 500 ms
8	32-Bit Wert unsigned	long	TxTimeout	Timeout in Millisekunden für Senden eines TCP/IP-Paketes an die SPS SPS 0 bedeutet Standardeinstellung = 500 ms
9	32-Bit Wert unsigned	long	ConTimeout	Timeout in Millisekunden für Warten auf Verbindungsaufbau mit SPS 0 bedeutet Standardeinstellung = 5000 ms (5sec.) muss bei Bedarf verlängert werden.

Rückgabewerte für die Openfunktionen:

Die Funktionen liefert einen 32-Bit Wert mit Vorzeichen als Rückgabewert mit folgender Bedeutung:

Wert	**Fehlerbeschreibung**	**Bedeutung**
>= 0	Alles OK	Die Rückgabe ist die Referenznummer für diese Verbindung und muss bei allen anderen Funktionen als Eingangsparameter Ref verwendet werden
-2	Keine Resourcen mehr frei.	Maximale Anzahl an verfügbaren Verbindungen erreicht.
-10	AccesMode nicht möglich (ab 1.23)	wenn eine unzulässige Nummer für AccessMode angegeben wird. Siehe IPS7OpenEx

Funktion	*Funktion*	*Beschreibung / Zweck*
IPS7Close	***ips7_close***	Dient zur Deinitialisierung der Verbindung, Speicher wird freigegeben und die TCP/IP-Verbindung wird getrennt.

Aufrufparameter:

Nr.	***Datentyp***	***Datentyp (PHP)***	***Name***	***Funktion***
1	32-Bit Wert unsigned	long	Ref	Die Referenz der Verbindung, welche mit IS7Open generiert wurde. Dient zur internen Identifikation der Verbindung

Rückgabewert:

Die Funktionen liefert einen 32-Bit Wert mit Vorzeichen als Rückgabewert mit folgender Bedeutung:

Wert	**Fehlerbeschreibung**	**Bedeutung / Reaktion**
0	alles OK	Speicher wieder freigegeben und Verbindung, wenn vorhanden geschlossen
-3	Mit der angegebenen Referenznummer wurde kein IPS7Open durchgeführt	Haben Sie IPS7Open aufgerufen ?.
-99 -30 -31	Die Referenznummer ist ungültig nur PHP nur PHP	---------- Die Anzal oder der Typ der übergeben Parameter ist falsch die interne Konvertierung der Daten konnte nicht durchgeführt werden, z.B. wurde ein String übergeben, wo long notwendig ist.

Funktionen zum Lesen und Schreiben

Funktion	*Funktion (PHP)*	*Beschreibung / Zweck*
IPS7RdW	***ips7_rdw***	wortweise lesen aus der SPS (E,A,M, DB)
IPS7RdPlcW	***ips7_rdplcw***	wortweise lesen aus der SPS (E,A,M, DB) jedoch Startadresse nach SPS-Adressierung (ab 1.17) damit ein Zugriff auf ungerade Startadressen möglich
IPS7RdB	***ips7_rdb***	byteweise lesen aus der SPS (E,A,M, DB)
IPS7WrW	***ips7_wrw***	wortweise schreiben in die SPS (E,A,M, DB, Z)
IPS7WrPlcW	***ips7_wrplcw***	wortweise schreiben in die SPS (E,A,M, DB, Z) jedoch Startadresse nach SPS-Adressierung (ab 1.17) damit ein Zugriff auf ungerade Startadressen möglich
IPS7WrB	***ips7_wrb***	byteweise schreiben in die SPS ((E,A,M, DB, Z)
IPS7RdDW	***ips7_rdrw***	doppelwortweise lesen aus der SPS (E,A,M, DB , T)
IPS7WrDW	***ips7_wrdw***	doppelwortweise schreiben in die SPS (E,A,M, DB , T)
IPS7RdReal	***ips7_rdreal***	Real (Fließpunktzahl lesen (E,A,M, DB)
IPS7WrReal	***ips7_wrreal***	Real (Fließpunktzahl schreiben (E,A,M, DB)

Beim Wortweisen Zugriff prüfen Sie bitte, ob Sie die Anfangsadresse nach SPS-Syntax verwenden möchten oder aber die rechnerisch korrekte. Je nachdem müssen Sie IPS7RdPlcW oder IPS7RdW bzw. IPS7WrPlcW oder IPS7RdW verwenden. Näheres finden Sie weiter unten erklärt. Siehe dazu **Neu in Version 1.17 !**

Erläuterungen zu PHP

In PHP sind die Variablen grundsätzlich keinem festem Datentyp zu geordnet. Die Bestimmung des Datentyps übernimmtdes halb das Erweiterungsmodul.

Grundsätzlich gilt: Die Ziel- bzw. Quellvariable für den Lese / Schreibpuffer (=Parameter 6) muß als Referenz übergeben werden. Also das „&“ – Zeichen verwenden.

z.B. $Res = ips7_rdplcw ($Ref, ord (“M“), 0, 0, 2, &$Werte);

Werte[0] ist MW0

Werte[1] ist MW2

Da es in PHP keine 16-Bit-Werte gibt, werden die 16-Bit-Dateien als long gespeichert.

Die Wortfunktionen speichern das Ergebnis in einem long-Array, wird nur eine Einheit gelesen, so wird das Ergebnis als einzelner long gespeichert, wenn die Variable noch kein Array ist. Das Lesen- und Schreiben von Worten (16 Bit) geschieht grundsätzlich mit Vorzeichen. D.h. der Wert wird als 16-Bit integer interpretiert. Sollen die Werte als vorzeichenloser 16-Bit-Wert (unsigned) behandelt werden, bachten Sie dann den optionalen Parameter 7 (bSigned).

Die Bytefunktionen speichern das Ergebnis grundsätzlich als string. Wollen Sie jedoch die Werte einfach als long Array ansprechen, so können Sie mit bLong = 1 das Ergebnis als long Array ablegen lassen.

Aufrufparameter:

Die Lese- und Schreibfunktionen besitzen die selben Eingangparameter:

Nr.	***Datentyp***	***Datentyp (PHP)***	***Name***	***Funktion***
1	32-Bit Wert unsigned	long	Ref	Die Referenz der Verbindung, welche mit IS7Open generiert wurde. Dient zur internen Identifikation der Verbindung
2	32-Bit Wert unsigned	long	Typ	Die Auswahl des Speicherbereichs in der SPS (DB, Eingang, Ausgang, Merker), welcher bearbeitet werden soll: ´D´ = 68 dez. steht für Datenbaustein ´E´ = 69, dez. steht für Eingänge ´A´ = 65 dez. steht für Ausgänge ´M´ = 77 dez. steht für Merker ´T´ = 84 dez. steht für Timer (nur mit Doppelwortfunktionen möglich) Die Timer werden in der SPS mit Zeitbasis und Wert im BCD-Format gespeichert. Um dieses Format sofort im PC verarbeiten zu können, führt der Treiber eine automatische Konvertierung in Millisekunden durch. Das kleinst mögliche Raster ist 10 ms. Beim schreiben in die SPS wählt der Treiber automatisch eine passende Zeitbasis. Dabei kann es zu Rundungen kommen. Der Zeitbereich geht von 0 bis 9990000 ms ´Z´ = 90 dez. steht für Zähler (nur mit Wortfunktionen möglich) Auch die Zähler sind in der SPS BCD-Codiert abgelegt. Die Zählerwerte reichen von 0 – 999.
3	32-Bit Wert unsigned	32-Bit Wert unsigned	DBNr	Datenbausteinnummer, diese wird nur beim Typ ´D´ verwendet. Ansonsten steht dort der Wert „0“
4	32-Bit Wert unsigned	32-Bit Wert unsigned	Ab	Erstes Wort bzw. Byte ab dem gelesen bzw geschrieben werden soll. Bei Wortoperationen Startwort. Bei Byte, Doppelwort und Realfunktionen Startbyte. Bei Timer oder Zähler ist dies die Nummer des ersten Elements, welches gelesen werden soll.
5	32-Bit Wert unsigned	32-Bit Wert unsigned	Anz	Anzahl der Einheiten (Byte, Worte, Doppelworte Real, oder Einheiten, z.B. Timer), die gelesen bzw. geschrieben werden sollen.

Nr.	*Datentyp*	*Datentyp (PHP)*	*Name*	*Funktion*
6	32-Bit Pointer	mixed	Buffer	Die Adresse auf den Quell- bzw. Zielspeicher im PC. Bei den Wortfunktionen ist dies ein Zeiger auf ein Feld von 16-Bit breiten Worten, bei den Bytefunktionen ist das eine Adresse auf ein Feld mit 8-Bit breiten Bytes. Bei Doppelwort Zeiger auf Long. Bei Real Zeiger auf einen Double. **Hinweis für PHP:** Bei PHP geben Sie als hier die Referenz einer Variablen an also: Bemerkung zu ipss7_rdplcw ips7_rdw ips7_rddw ips7_rdreal Aufruf z.B: &Result = ips7_rdplcw ($Ref, ord (“M”), 0, 6,5, &$W); Werden mehr als 1 Element gelesenen, so wird die Variable in ein Array vom Typ long bzw. double umgewandelt. Wird nur ein Wert gelesen und die Variable ist kein Array, wird der Wert als long gespeichert. Ist die Variable bereits ein Array und es wird nur ein Wert gelesen, so wird das Ergebnis im ersten Element des Array abgelegt. Doppelworte (ips7_rddw) werden grundsätzlich mit Vorzeichen verarbeitet (signed).
7	---------	long (optional)	bSigned bei ipss7_rdplcw ips7_rdw ipss7_wrplcw ips7_wrdw bLong bei ips7_rdb	**Hinweis für PHP:** **ipss7_rdplcw** **ips7_rdw** **ipss7_wrplcw** **ips7_wrdw** Optional kann bestimmt werden, ob die Werte als signed oder unsigned 16-Bit-Integer gelsen werden sollen. Wird der Parameter nicht angegeben, wird grundsätzlich mit Vorzeichen gearbeitet (signed). Wir der Parameter übergeben gilt: 0 = ohne Vorzeichen (signed) 1 = mit Vorzeichen Außerdem stehen für eine nachträgliche Konvertierung einzelner Werte die Funktionen **ips7_i2w** und **ips7_w2i** zu Verfügung. Näherer finden Sie dort. **ips7_rdb** Ips7_rdb speichert das Ergebnis grundsätzlich als string. Wollen Sie jedoch die Werte einfach als Array ansprechen, so können Sie mit bLong = 1 das Ergebnis als long.Array ablegen lassen.

Funktionen zum Bit-Lesen, Setzen und Rücksetzten

Funktion	*Funktion (PHP)*	*Beschreibung / Zweck*
IPS7RdBit	***ips7_rdbit***	ein Bit lesen SPS (E,A,M, DB)
IPS7SetBit	***ips7_setbit***	ein Bit setzen lesen aus der SPS (E,A,M,DB)
IPS7ResetBit	***ips7_resetbit***	ein Bit zurücksetzen in der SPS (E,A,M,DB)

Aufrufparameter:

Die Lese- und Schreibfunktionen besitzen die selben Eingangparameter:

Nr.	***Datentyp***	***Datentyp (PHP)***	***Name***	***Funktion***
1	32-Bit Wert unsigned	long	Ref	Die Referenz der Verbindung, welche mit IS7Open generiert wurde. Dient zur internen Identifikation der Verbindung
2	32-Bit Wert unsigned	long	Typ	Die Auswahl des Speicherbereichs in der SPS (DB, Eingang, Ausgang, Merker), welcher bearbeitet werden soll: ´D´ = 68 dez. steht für Datenbaustein ´E´ = 69, dez. steht für Eingänge ´A´ = 65 dez. steht für Ausgänge ´M´ = 77 dez. steht für Merker
3	32-Bit Wert unsigned	long	DBNr	Datenbausteinnummer, diese wird nur beim Typ ´D´ verwendet. Ansonsten steht dort der Wert „0“
4	32-Bit Wert unsigned	long	Ab	Byte Adresse z.B. M 10.0, dann steht hier 10. Beachten Sie hier Unterschied zwischen IPS7RdW und IPS7RdPlcW bzw. IPS7WrW und IPS7WrPlcW. Siehe weiter unten !
5	32-Bit Wert unsigned	long	Bit	BitNr muss zwischen 0 und 7 liegen. z.B. bei M5.4 steht hier 4.
6	32-Bit Adresse	mixed	Buffer	**Dieser Parameter ist nur für IPS7RdBit.** Die Adresse auf den Zielspeicher im PC. Zeiger auf ein Byte. Wenn bBit gestzt ist Inhalt 1 sonst 0. Beispiel: lese M 6.5 BYTE W; IPS7RdBit (Ref, ´M`, 0, 6,5 & W); **Hinweis für PHP:** Bei PHP geben Sie als hier die Referenz einer Variablen an also: ips7_rdbit (Ref, ord (“M”), 0, 6,5, & $W); Diese Variable wird autmatisch in einen “long” konvertiert. Verwenden Sie daher eine Variable, die sonst noch nicht verwendet wurde. Der Zustand des Bit’s (0 oder 1) ist somit als long gespeichert.

Rückgabewert:

Die Funktionen liefert einen 32-Bit Wert mit Vorzeichen als Rückgabewert mit folgender Bedeutung:

Wert	Fehlerbeschreibung	Reaktion
0	Alles OK	Daten auswertren
-1	Zeitüberlauf, gewünschte SPS offensichtlich nicht oder nicht mehr vorhanden	Einfach weitere Schreib- und Leseaufträge absetzten der Treiber baut die Verbindung automatisch auf. Evtl. die Timeoutzeiten insbesondere die Connect-Timeoutzeit verlängern.
2	Baustein oder Datenbereich existiert nicht, z.B. Zugriff auf DB, der nicht vorhanden, oder zu klein ist.	Überprüfen, ob der gewünschte Datenbereich in der SPS vorhanden ist.
-10	Gewünschter Datentyp nicht erlaubt oder wird nicht unterstützt.	Prüfen, ob der Code für Datentyp in Ordnung ist.
-5	Allgemeiner Fehler	Prüfen ob Netzwerk richtig im PC installiert ist: TCP/IP aktiviert ? Winsocket installiert ?
-6	Ziel-CPU nicht gefunden	Rack oder Steckplatznummer falsch Es ist keine Verbindung zu diesem Steckplatz mehr frei. Im CP Konfiguratioin prüfen
-7	Socketfehler aufgetreten	IPS7GetSockErr aufrufen und Fehler auswerten
-8	Speicherfehler	angeforderter Speicher im PC ist nicht verfügbar
-9	Bereichsüberschreitung	z.B Timer > 9990000 ms
-99	Die Referenznummer ist ungültig	Haben Sie IPS7Open aufgerufen ?
4660	Demozeit ist abgelaufen	Vollversion erwerben

Zusatzfunktionen für PHP

```
long ips7_w2i(mixed Buffer,long Count);
```
Konvertiert unsigned 16-Bit - Werte in signed 16-Bit Werte

```
long ips7_i2w(mixed Buffer, long Count);
```
Konvertiert signed 16-Bit - Werte in unsigned 16-Bit Werte

Parameter	*Beschreibung / Zweck*
Buffer	Referenz auf die long Werte (Array) oder den long Wert, der konvertiert werden soll.
Count	Anzahl der Werte

Neu in Version 1.17 !!!!

Unterschied S7RdPlcW <-> **IPS7RdW** und **S7WrPlcW <-> S7WrW**

Der PC und die SPS habenverschiedene Adressierungsarten. In der S7 ist der Speicherbereich byteweise orientiert. So adressieren Sie aus der Sicht des SPS-Programmierers mit MW 0 das MB0 und MB1, mit MW1 aber das MB1 und MB2. Sie sehen, daß sich MW0 und MW1 im MB1 überschneiden.
Vor der Version vor 1.17 war es nur möglich auf gerade Startadressen mit den Wortfunktionen zuzugreifen.
Ab V 1.17 ist mit den Funktionen **S7RdPlcW und S7WrPlcW** ein Zugriff auf ungerade Startadressen möglich. Wollen Sie nun MW 1 lesen, so wie der SPS-Programmierer das sieht rufen Sie auf.
IPS7RdPlcW (Ref, ´M´, 0, 1, 1, WortBuffer);

!!! Beachte bei Wortoperationen mit S7RdW und S7WrW!!!

Beispiel für Merker. Dies gilt auch für Eingänge Ausgänge und Datenworte

Die Wortadressierung in der SPS belegt jeweils folgende Bytes.

Worteadresse	***zugeordnete Bytes***
MW0	MB 0 und MB 1
MW1	MB 1 und MB 2
MW2	MB 2 und MB 3

Sie sehen, dass es bei Verwendung von ungeraden Wortadressen zu einer Doppelbelegung kommen kann. Deshalb unterstützen die Wortfunktionen (IPS7RdW und IPS7WrW) nur den Zugriff auf gerade Wortadressen. Dies bedeutet, dass die Start-Wort-Nr im Treiber immer mit 2 multipliziert wird. Diese Methode erlaubt zu dem ein einfaches Abbild des SPS-Speichers in den PC. Also ein Wortschritt im PC sind 16 Bit im PC und 16 Bit in der SPS

Beispiel:

WORD Buf[64];

Der Aufruf ***IPS7RdW (Ref, Typ, DBNr, 0, 5, Buf)*** hat folgende Wirkung:

PC	***SPS***
Buf[0]	DW 0
Buf[1]	DW 2
Buf[2]	DW 4

Sie müssen also die Start-Wortnummer halbieren, um im PC richtig zugreifen zu können. Dies gilt auch für Datenbausteine !! --> Ungerade Wortadressen der SPS können nicht wortweise gelesen oder geschrieben werden.

Wollen Sie trotzem auf ungerade Startadressen adressieren verwenden Sie

Programmbeispiele:

a) Aufruf von C oder C++ aus:

```
unsigned char ByteBuffer[512];
unsigned short int WordBuffer[512];
// Aufruf der Bytefunktion z.B. Lese DB 10, ab DW0, 10 Worte
IPS7RdW (Ref, ´D´, 10, 0, 10, WordBuffer);
// Aufruf der Bytefunktion z.B. Lese MB 0 , 10 Bytes
IPS7RdB (Ref, ´M´, 0, 0, 10, ByteBuffer);
```

Nach erfolgreichem Aufruf gilt:

PC	=	SPS
WordBuffer[0]	=	DB10.DBW0
WordBuffer[1]	=	DB10.DBW2
WordBuffer[2]	=	DW10.DBW4
ByteBuffer[0]	=	MB 0

ByteBuffer[1]	=	MB 1

b) Aufruf von Delphi aus:

ByteBuffer array [0..511] of Byte;

WordBuffer array [0..511] of Word;

// Aufruf der Wortfunktion z.B. Lese DB 10, ab DW0, 10 Worte

IPS7RdW (Ref, LongWord (´D´), 10, 0, 10, @WordBuffer[0]);

// Aufruf der Bytefunktion z.B. Lese MB 0 , 10 Bytes

IPS7RdB (Ref, ´M´, 0, 0, 10, @ByteBuffer[0]);

c) Aufruf von Visual Basic aus:

Dim ByteBuffer (0 to 511) as Byte;

Dim WordBuffer (0..511) as Word;

// Aufruf der Wortfunktion z.B. Lese DB 10, ab DW0, 10 Worte

IPS7RdW (Ref, 68, 10, 0, 10, WordBuffer(0))

// Aufruf der Bytefunktion z.B. Lese MB 0 , 10 Bytes

IPS7RdB (Ref, 77, 0, 0, 10, ByteBuffer(0));

Nach erfolgreichem Aufruf gilt:

PC	**=**	**SPS**
WordBuffer[0]	=	DB10.DBW0
WordBuffer[1]	=	DB10.DBW2
WordBuffer[2]	=	DW10.DBW4
ByteBuffer[0]	=	MB 0
ByteBuffer[1]	=	MB 1

Funktionen zum Bit-Lesen, Setzen und Rücksetzten

Funktion	*Funktion (PHP)*	*Beschreibung / Zweck*
IPS7RdBit	***ips7_rdbit***	ein Bit lesen SPS (E,A,M, DB)
IPS7SetBit	***ips7_setbit***	ein Bit setzen lesen aus der SPS (E,A,M,DB)
IPS7ResetBit	***ips7_resetbit***	ein Bit zurücksetzen in der SPS (E,A,M,DB)

Funktion	*Funktion (PHP)*	*Beschreibung / Zweck*
IPS7GetSockErr	***ips7_getsockerr***	Liefert den letzten Socket-Fehler zurück

Aufrufparameter:

Nr.	***Datentyp***	***Datentyp (PHP)***	***Name***	***Funktion***

1	32-Bit Wert unsigned	long	Ref	Die Referenz der Verbindung, welche mit IS7Open generiert wurde. Dient zur internen Identifikation der Verbindung

Rückgabewert:

Die Funktionen liefert einen 32-Bit Wert mit Vorzeichen als Rückgabewert mit folgender Bedeutung:

Wert	Fehlerbeschreibung	Bedeutung / Reaktion
0	alles OK	Es liegt kein Fehler an
-3	Mit der angegebenen Referenznummer wurde kein IPS7Open durchgeführt	Haben Sie IPS7Open aufgerufen ?.
-99	Die Referenznummer ist ungültig	----------
Sonstige	Socketerror	Erklärung siehe Liste unterhalb.

Socketfehler: Diese Liste erhebt keinen Anspruch auf Vollständigkeit

Name	Code	Bedeutung
WSAEINTR	10004	Aufruf wurde abgebrochen
WSAEBADF	10009	
WSAEACCES	10013	Zugriffsfehler
WSAEFAULT	10014	Parameter sind falsch
WSAEINVAL	10022	1. Andere Funktion muß vorher aufgerufen werden 2. Socket ist schon an Adresse gebunden 3. Socket noch nicht an Adresse gebunden bzw. schon verbunden
WSAEMFILE	10024	Resourcen fehlen (Dateien, Warteschlangen)
WSAEWOULDBLOCK	10035	Aufruf würde blockieren
WSAEINPROGRESS	10036	Parallele Aufrufe nicht erlaubt
WSAEALREADY	10037	Abgebrochene Routine trotzdem schon fertig
WSAENOTSOCK	10038	Kein gültiger Socket angegeben
WSAEDESTADDRREQ	10039	Zieladresse benötigt
WSAEMSGSIZE	10040	Datagram zu groß, wurde abgeschnitten
WSAEPROTOTYPE	10041	
WSAENOPROTOOPT	10042	Unbekannte Socket-Option
WSAEPROTONOSUPPORT	10043	Protokoll wird nicht unterstützt
WSAESOCKTNOSUPPORT	10044	Sockettyp wird in angegebener Adressfamilie nicht unterstützt
WSAEOPNOTSUPP	10045	Dieser Sockettyp wird nicht unterstützt
WSAEPFNOSUPPORT	10046	Protokollfamilie wird nicht unterstützt
WSAEAFNOSUPPORT	10047	Adressfamilie wird nicht unterstützt
WSAEADDRINUSE	10048	IP-Adresse bzw. Port werden schon/noch benutzt
WSAEADDRNOTAVAIL	10049	Port/Adresse nicht verfügbar
WSAENETDOWN	10050	Netzwerk reagiert nicht
WSAENETUNREACH	10051	Netzwerk kann nicht erreicht werden
WSAENETRESET	10052	Verbindung durch TCP/IP zurückgesetzt
WSAECONNABORTED	10053	Verbindung durch TCP/IP abgebrochen
WSAECONNRESET	10054	Partner hat Verbindung zurückgesetzt
WSAENOBUFS	10055	Resourcen fehlen (Interner Pufferspeicher)
WSAEISCONN	10056	Socket ist schon verbunden
WSAENOTCONN	10057	Socket ist noch nicht verbunden
WSAESHUTDOWN	10058	Andere Seite hat Verbindung einseitig beendet
WSAETOOMANYREFS	10059	

WSAETIMEDOUT	10060	Aufruf dauert zu lange, daher Abbruch
WSAECONNREFUSED	10061	Angerufener möchte keinen Verbindungsaufbau
WSAELOOP	10062	
WSAENAMETOOLONG	10063	
WSAEHOSTDOWN	10064	
WSAEHOSTUNREACH	10065	Host nicht erreicbar
WSAENOTEMPTY	10066	
WSAEPROCLIM	10067	
WSAEUSERS	10068	
WSAEDQUOT	10069	
WSAESTALE	10070	
WSAEREMOTE	10071	
WSASYSNOTREADY	10091	Netzwerk nicht zur Kommunikation bereit
WSAVERNOTSUPPORTED	10092	gewünschte Winsock-Version wird nicht unterstützt
WSANOTINITIALISED	10093	Socket.Initialize muß aufgerufen werden
WSAHOST_NOT_FOUND	11001	DNS-Server nicht gefunden
WSATRY_AGAIN	11002	Gesuchter Rechner nicht gefunden
WSANO_RECOVERY	11003	Nicht behebbarer Fehler
WSANO_DATA	11004	Keine Namensdaten vorhanden
WSANO_ADDRESS	11004	

Funktionsdeklarationen:

C-Header:

```
long WINAPI
IPS7Open (LPCSTR IPAdr, DWORD Rack, DWORD Slot, DWORD RxTimeout, DWORD TxTimeout,
        DWORD ConnectTimeout);
IPS7OpenPG (LPCSTR IPAdr, DWORD Rack, DWORD Slot, DWORD RxTimeout,
            DWORD TxTimeout,    DWORD ConnectTimeout);

//----------------------------
long WINAPI
IPS7Close (long Ref);
//----------------------------
long WINAPI
IPS7RdW (long Ref, DWORD Typ, DWORD DBNr, DWORD AbWort, DWORD WortAnz,
        LPWORD Buffer) ;
long WINAPI
IPS7RdPlcW (long Ref, DWORD Typ, DWORD DBNr, DWORD AbWort, DWORD WortAnz,
        LPWORD Buffer) ;
//----------------------------
long WINAPI
IPS7RdB (long Ref, DWORD Typ, DWORD DBNr, DWORD Ab, DWORD Anz, LPBYTE Buffer);
//----------------------------
long WINAPI
IPS7WrW (long Ref, DWORD Typ, DWORD DBNr, DWORD Ab, DWORD Anz, LPWORD Buffer);
long WINAPI
IPS7WrPlcW (long Ref, DWORD Typ, DWORD DBNr, DWORD Ab, DWORD Anz, LPWORD Buffer);
//----------------------------
long WINAPI
IPS7WrB (long Ref, DWORD Typ, DWORD DBNr, DWORD Ab, DWORD Anz, LPBYTE Buffer);
//----------------------------
long WINAPI
IPS7RdBit (long Ref, DWORD Typ, DWORD DBNr, DWORD Byte, DWORD Bit, LPBYTE Buffer);
//----------------------------
long WINAPI
IPS7SetBit (long Ref, DWORD Typ, DWORD DBNr, DWORD Byte, DWORD Bit);
//----------------------------

long WINAPI
IPS7ResetBit (long Ref, DWORD Typ, DWORD DBNr, DWORD Byte, DWORD Bit);
//----------------------------
long WINAPI
IPS7RdReal (long Ref, DWORD Typ, DWORD DBNr, DWORD Ab, DWORD Anz, double *Buffer);
```

```
//----------------------------
long WINAPI
IPS7WrReal (long Ref, DWORD Typ, DWORD DBNr, DWORD Ab, DWORD Anz, double *Buffer);
//----------------------------
long WINAPI
IPS7RdDW (long Ref, DWORD Typ, DWORD DBNr, DWORD Ab, DWORD Anz, LPDWORD Buffer);
//----------------------------
long WINAPI
IPS7WrDW (long Ref, DWORD Typ, DWORD DBNr, DWORD Ab, DWORD Anz, LPDWORD Buffer);
//----------------------------
long WINAPI
IPS7GetSockErr (long Ref);
```

Visual Basic – Header:

```
Declare Function
IPS7Open& Lib "IPS7LNK.dll" (ByVal IPAdr as String, ByVal Rack&,
ByVal Slot&, ByVal RxTimeout&, ByVal TxTimeout&, ByVal ConnectTimeout&)

IPS7OpenPG& Lib "IPS7LNK.dll" (ByVal IPAdr as String, ByVal Rack&,
ByVal Slot&, ByVal RxTimeout&, ByVal TxTimeout&, ByVal ConnectTimeout&)

‘----------------------------
Declare Function
IPS7Close& Lib "IPS7LNK.dll" (ByVal Ref&)
‘----------------------------
Declare Function
IPS7RdW& Lib "IPS7LNK.dll" (ByVal Ref&, ByVal Typ&, ByVal DBNr&, ByVal AbWort&,
ByVal WortAnz&, Wert As Integer)

Declare Function
IPS7RdPlcW& Lib "IPS7LNK.dll" (ByVal Ref&, ByVal Typ&, ByVal DBNr&, ByVal AbWort&,
ByVal WortAnz&, Wert As Integer)
‘----------------------------
Declare Function
IPS7RdB& Lib "IPS7LNK.dll" (ByVal Ref&, ByVal Typ&, ByVal DBNr&, ByVal AbWort&, ByVal WortAnz&,
Wert As Byte)
‘----------------------------
Declare Function
IPS7WrW& Lib "IPS7LNK.dll" (ByVal Ref&, ByVal Typ&, ByVal DBNr&, ByVal AbWort&, ByVal WortAnz&,
Wert As Integer)
Declare Function
IPS7WrPlcW& Lib "IPS7LNK.dll" (ByVal Ref&, ByVal Typ&, ByVal DBNr&, ByVal AbWort&, ByVal WortAnz&,
Wert As Integer)

‘----------------------------
Declare Function
IPS7WrB& Lib "IPS7LNK.dll" (ByVal Ref&, ByVal Typ&, ByVal DBNr&, ByVal AbWort&, ByVal WortAnz&,
Wert As Byte)
‘----------------------------
Declare Function
IPS7GetSockErr& Lib "IPS7LNK.dll" (ByVal Ref&)
‘----------------------------
Declare Function I
PS7RdBit& Lib "IPS7LNK.dll" (ByVal Ref&, ByVal Typ&, ByVal DBNr&, ByVal ByteNr&, ByVal BitNr&, Wert
As Byte)
```

```
‘----------------------------

Declare Function IPS7SetBit& Lib "IPS7LNK.dll" (ByVal Ref&, ByVal Typ&, ByVal DBNr&, ByVal ByteNr&,
ByVal BitNr&)
‘----------------------------
Declare Function IPS7ResetBit& Lib "IPS7LNK.dll" (ByVal Ref&, ByVal Typ&, ByVal DBNr&, ByVal ByteNr&,
ByVal BitNr&)
‘----------------------------
Declare Function IPS7RdReal& Lib "IPS7LNK.dll" (ByVal Ref&, ByVal Typ&, ByVal DBNr&, ByVal AbWort&,
ByVal WortAnz&, Wert As Double)
‘----------------------------
Declare Function IPS7WrReal& Lib "IPS7LNK.dll" (ByVal Ref&, ByVal Typ&, ByVal DBNr&, ByVal AbWort&,
ByVal WortAnz&, Wert As Double)
‘----------------------------
Declare Function IPS7RdDW& Lib "IPS7LNK.dll" (ByVal Ref&, ByVal Typ&, ByVal DBNr&, ByVal AbWort&,
ByVal WortAnz&, Wert As Long)
‘----------------------------
Declare Function IPS7WrDW& Lib "IPS7LNK.dll" (ByVal Ref&, ByVal Typ&, ByVal DBNr&, ByVal AbWort&,
ByVal WortAnz&, Wert As Long)
‘----------------------------
```

Delphi-Header:

```
unit IPS7LNK;

TYPE PWORD =  ^WORD;
TYPE PBYTE =  ^BYTE;

FUNCTION
IPS7Open (IPAdr : PChar; Rack : LongWord; Slot : LongWord; RxTimeout : LongWord;
TxTimeout : LongWord ; ConnectTimeout : LongWord) : LongInt; stdcall; external 'IPS7LNK.DLL';

FUNCTION
IPS7OpenPG (IPAdr : PChar; Rack : LongWord; Slot : LongWord; RxTimeout : LongWord;
TxTimeout : LongWord ; ConnectTimeout : LongWord) : LongInt; stdcall; external 'IPS7LNK.DLL';

(* ------------------------- *)
FUNCTION
IPS7Close (Ref : LongInt) : LongInt; stdcall; external 'IPS7LNK.DLL';
(* ------------------------- *)
FUNCTION
IPS7RdW (Ref : LongInt; Typ : Longword; DBNr : Longword; AbWort : Longword;
WortAnz : Longword;  Buffer : PWORD)  : LongInt; stdcall; external 'IPS7LNK.DLL';

FUNCTION
IPS7RdPlcW (Ref : LongInt; Typ : Longword; DBNr : Longword; AbWort : Longword;
WortAnz : Longword;  Buffer : PWORD)  : LongInt; stdcall; external 'IPS7LNK.DLL';
(* ------------------------- *)

FUNCTION
IPS7RdrB (Ref : LongInt; Typ : Longword;  DBNr: Longword;
        Ab : Longword;  Anz : Longword; Buffer: PBYTE) : LongInt; stdcall; external 'IPS7LNK.DLL';
(* ------------------------- *)

FUNCTION
IPS7WrW (Ref : LongInt; Typ : Longword; DBNr : Longword; AbWort : Longword;
WortAnz : Longword; Buffer : PWORD)  : LongInt; stdcall; external 'IPS7LNK.DLL';

FUNCTION
IPS7WrPlcW (Ref : LongInt; Typ : Longword; DBNr : Longword; AbWort : Longword;
WortAnz : Longword; Buffer : PWORD)  : LongInt; stdcall; external 'IPS7LNK.DLL';
(* ------------------------- *)
FUNCTION
IPS7WrB (Ref : LongInt; Typ : Longword;  DBNr: Longword;
       Ab : Longword;  Anz : Longword; Buffer: PBYTE) : LongInt; stdcall; external 'IPS7LNK.DLL';
(* ------------------------- *)
FUNCTION
IPS7GetSockErr (Ref : LongInt) : LongInt; stdcall; external 'IPS7LNK.DLL';
(* ------------------------- *)
FUNCTION
IPS7RdBit (Ref : LongInt; Typ : Longword;  DBNr: Longword;
        ByteNr : Longword;  BitNr : Longword; Buffer: PBYTE) : LongInt; stdcall; external 'IPS7LNK.DLL';
(* ------------------------- *)
FUNCTION
IPS7SetBit (Ref : LongInt; Typ : Longword;  DBNr: Longword;
         ByteNr : Longword;  BitNr : Longword) : LongInt; stdcall; external 'IPS7LNK.DLL';
(* ------------------------- *)
FUNCTION
IPS7ResetBit (Ref : LongInt; Typ : Longword;  DBNr: Longword;
         ByteNr : Longword;  BitNr : Longword) : LongInt; stdcall; external 'IPS7LNK.DLL';
(* ------------------------- *)
FUNCTION
IPS7RdDW (Ref : LongInt; Typ : Longword; DBNr : Longword; AbWort : Longword; WortAnz : Longword;
Buffer : PDWORD)  : LongInt; stdcall; external 'IPS7LNK.DLL';
(* ------------------------- *)
```

```
FUNCTION
IPS7WrDW (Ref : LongInt; Typ : Longword; DBNr : Longword; AbWort : Longword; WortAnz : Longword;
Buffer : PDWORD)  : LongInt; stdcall; external 'IPS7LNK.DLL';
(* ------------------------ *)
FUNCTION
IPS7RdReal (Ref : LongInt; Typ : Longword; DBNr : Longword; AbWort : Longword; WortAnz : Longword;
Buffer : PDOUBLE)  : LongInt; stdcall; external 'IPS7LNK.DLL';
(* ------------------------ *)
FUNCTION
IPS7WrReal (Ref : LongInt; Typ : Longword; DBNr : Longword; AbWort : Longword; WortAnz : Longword;
Buffer : PDOUBLE)  : LongInt; stdcall; external 'IPS7LNK.DLL';

implementation

begin
end.
```