

REA JET

KENNZEICHNUNGSLÖSUNGEN
FÜR DIE INDUSTRIE –
MADE IN GERMANY

Benutzerhandbuch

Schnittstellenprotokoll REA-PLC

REA JET HR, HR_{pro}, CL

für Steuergeräte Firmware: 3.4 · Dokumentenversion: 1.31 · Stand vom: 13.08.2017



REA Elektronik GmbH

Teichwiesenstraße 1

64367 Mühlthal

Tel.: +49 6154 638 0

© 2015 REA Elektronik GmbH, alle Rechte vorbehalten

Ohne schriftliche Erlaubnis von REA Elektronik GmbH darf kein Teil dieser Bedienungsanleitung in irgendeiner Form reproduziert oder mit elektronischen Systemen verarbeitet, vervielfältigt oder verbreitet werden.

Die in diesem Dokument enthaltenen Informationen können ohne vorherige Ankündigung geändert werden.

Inhaltsverzeichnis

1	Aufbau der Dokumentation	4
1.1	Verwendete Fachbegriffe	4
1.2	Was ist ein Byte	5
1.2.1	Zulässige Inhalte eines Byte in diesem PLC Protokoll.....	6
1.2.2	Darstellung der Stringlängen in Byte(s).....	7
2	REA-PLC	9
2.1	Verbindungsaufbau zwischen Client und Server	9
2.1.1	Schnittstellenprotokoll auf Port: 22169 (EOT), ab Firmware 1.83	10
2.1.2	Schnittstellenprotokoll auf Port: 22170, ab Firmware 1.70.....	11
2.2	Schnittstellenprotokoll Beschreibung	12
2.3	Befehle.....	12
2.3.1	Druckjob zuweisen.....	12
2.3.2	Druckjob starten.....	13
2.3.3	Druckjob stoppen.....	14
2.3.4	Objekthinhalte überschreiben	14
2.3.5	Objekteigenschaften überschreiben.....	19
2.4	Antworten vom Server (Response).....	23
2.4.1	Befehl	23
2.4.2	Jobstatus	23
2.4.3	Statusfeld.....	23
2.4.4	Fehlercodes.....	24
3	Software zu Test und Evaluierungszwecken.....	29
3.1	REA-PLC Tester, ab Version: 1.14.00	29
3.1.1	Verbindungsaufbau (Connect)	29
3.1.2	Druckjob Befehle (Set Job, Start Job, Stop Job)	31
3.1.3	Objekthinhalte überschreiben (Set Label Content).....	32
3.1.4	Erweiterung (Advanced)	33
3.1.5	Proxy	35
3.1.6	Einstellungen speichern (Save settings)	39
3.1.7	Einstellungen löschen (Clean settings)	39
4	titan-add-on-serial2plc.....	40
4.1	serial_parameter.ini	40
4.1.1	9600/8/N/1	40
4.1.2	19200/8/E/1	40
4.1.3	115200/8/O/1	41
5	Notizen	42

1 Aufbau der Dokumentation

Das Handbuch ist in Kapiteln unterteilt, die thematisch abgegrenzt sind. Falls weitere Informationen zu einer Vorgehensweise oder einem Arbeitsschritt verfügbar sind, finden Sie an der entsprechenden Stelle einen Querverweis auf das Kapitel mit Seitenzahl um ergänzenden Informationen schnell finden zu können.

Dieses Handbuch wurde sorgfältig ausgearbeitet und überprüft. Sollten Sie Hinweise oder Verbesserungsvorschläge haben, sind wir für Ihre geschätzte Meinung dankbar. Wir bitten Sie, Ihre Vorschläge per E-Mail mit Hinweis auf dieses Handbuch an info@rea.de weiterzuleiten. Vielen Dank.

1.1 Verwendete Fachbegriffe

Die Fachbegriffe sind in der Tabelle Alphanumerisch sortiert.

Fachbegriff	Beschreibung
ASCII	Die Initialen stehen für „ <u>A</u> merican <u>S</u> tandard <u>C</u> ode for <u>I</u> nformation <u>I</u> nterchange“. In dieser Spezifikation sind einzelne Zeichen zu einem Zahlenwert von 0 (0x00) bis 255 (0xFF) definiert.
Byte	Ein Byte besteht aus 8 Bit, wobei ein Bit die kleinste Speichereinheit in einem Rechner darstellt. Ein Bit kann den Zustand 0 (false) oder 1 (true) annehmen. Ein Byte kann somit einen Zahlenwert von 0 bis 255 darstellen.
7-Bit	Von einem Byte das 8-Bit hat werden nur die ersten 7-Bit ausgewertet. Ein Byte mit 7-Bit kann somit einen Zahlenwert von 0 bis 127 darstellen.
DHCP	Die Initialen stehen für „ <u>D</u> ynamic <u>H</u> ost <u>C</u> onfiguration <u>P</u> rotocol“. Das bedeutet das die IP-Adresse im Steuergerät dynamisch vom „Host“ vergeben wird und sich bei einem Neustart (Ein- und Ausschalten) des Steuergerätes ändern kann.
IP-Adresse	Die IP-Adresse beinhaltet die Adresse unter der das Steuergerät im Netzwerk zu erreichen ist. Jedes Gerät hat seine eigene, eindeutige IP-Adresse im Netzwerk.
Kennzeichnungssystem	Ein Kennzeichnungssystem besteht mindestens aus einem Steuergerät und einem Schreibkopf.
Proxy	Ein Vermittler in Netzwerken

Fachbegriff	Beschreibung
RS-232	Ein sehr verbreitetes serielles Schnittstellenprotokoll. Die max. Kabellänge ist abhängig von der Übertragungsgeschwindigkeit, die in Baud (bit/s) angegeben wird.
RS-422	Ein serielles Schnittstellenprotokoll das im Gegensatz zu dem RS-232 deutlich störunempfindlicher und somit für größere Distanzen geeignet ist.
Samba Protokoll	Ein Schnittstellenprotokoll für den Dateitransfer in TCP/IP- Netzwerken.
String	Ein String ist ein Container der eine beliebige Zeichenkette enthalten kann. Er besteht i.d.R. aus mehreren zusammengesetzten Bytes. Ein String enthält die Befehle oder Dateninhalte die zur Kommunikation dienen.
TCP/IP	Steht für „Transmission Control Protocol / Internet Protocol“ und hat sich als Standard durchgesetzt. Hierüber laufen die DHCP, FTP und Samba Protokoll, aber auch vieles mehr.

Tabelle 1: Fachbegriffe

1.2 Was ist ein Byte

Da in dieser Dokumentation an vielen Stellen von Bytes gesprochen wird, hier einige Erklärungen.

Ein Byte besteht aus 8Bits, d.h. ein Byte kann die Zahlenwerte von 0 bis 255 beinhalten, wenn als Basis das Dezimale (DEZ) Zahlensystem verwendet wird. Das Dezimale Zahlensystem findet bei Schnittstellenprotokollen keine Verwendung, hier wird üblicherweise das Hexadezimale (HEX) Zahlensystem verwendet. Hier kann ein Byte die Zahlenwerte 0x00 bis 0xFF annehmen, wobei 0x eine Kennung für das Hexadezimale Zahlensystem ist und hier in der weiteren Dokumentation, durch die Überschrift „Hex-Format“ gekennzeichnet ist.

In den folgenden Beschreibungen werden die Befehle und Dateninhalte nicht in der Hexadezimalen Darstellung wiedergegeben, sondern im ASCII-Format, denn dieses Format lässt sich für uns Menschen am Besten lesen. Um bei der Fehlersuche nicht umrechnen zu müssen, wird zusätzlich die Hexadezimale Darstellung mit angegeben, somit sollten sich Fehler bei der Datenkommunikation leichter aufspüren lassen.

Drei verschiedene Schreibweisen für ein und denselben Sachverhalt und jede Schreibweise hat ihre Vor- und Nachteile.

Beispiel:

Ein String mit neun Bytes im Hex-Format mit folgendem Dateninhalt:

00 02 03 0A 20 30 39 41 7A

Der gleiche String im ASCII-Format, natürlich auch mit neun Zeichen:

[NUL] [STX] [ETX] [LF] [SP] 0 9 A z

Und das Ganze sieht im Dezimalen-Format wie folgt aus:

0 2 3 10 32 48 57 64 122

Im Hex-Format können die Alphanumerischen Zeichen (A bis F) sowohl in Groß- als auch in Kleinbuchstaben angegeben werden.

1.2.1 ***Zulässige Inhalte eines Byte in diesem PLC Protokoll***

Auch wenn ein Byte 255 (0xFF) verschiedene Werte annehmen kann, sind nicht alle Dateninhalte in diesem Protokoll erlaubt. Von einem Byte können nur die folgenden Zeichen für die Datenübertragung genutzt werden:

ASCII-Format:

[SP] bis ~

Hex-Format:

20 bis 7E

Das Zeichen Semikolon (;) dient in diesem Protokoll als Trennzeichen und darf somit nicht anderweitig verwendet werden, mit Ausnahme von Objekthinhalten.

ASCII-Format:

;

Hex-Format:

3B

Der Hintergrund für diese Einschränkung ist, dass die zu übermittelten Zeichen stets lesbar sind und mögliche Datenkommunikationsfehler schnell gefunden werden können.

Im Hexadezimalsystem werden die Ziffern A bis F i.d.R. stets großgeschrieben. In der Praxis können die Ziffern auch kleingeschrieben werden, das wird von vielen Protokollen, wie auch vom REA-PLC

Protokoll unterstützt. Das heißt, auch in diesem Dokument kann im ASCII-Format ein „E“ angegeben werden, während im HEX-Format hierfür eine „45“ oder eine „65“ stehen kann.

1.2.2 *Darstellung der Stringlängen in Byte(s)*

In diesem Protokoll müssen die Längen von den Nutzdaten berechnet werden, als Beispiel siehe hierzu bitte Kap. 2.3.4 auf Seite 14.

In diesem Beispiel beträgt die Länge des Strings 26 Zeichen. Die Zahl 26 entspricht der 0x1A in Hex. Wie im Kapitel 1.2.1 auf Seite 6 beschreiben, darf ein Byte nicht 0x1A beinhalten. Daher werden diese Größenangaben, wie folgt in (zwei) Bytes aufgesplittet:

ASCII-Format:

1A

Hex-Format:

31 61

Der Vorteil dieses Verfahrens ist, das die Daten im ASCII-Format immer lesbar bleiben. Das hat auch Auswirkungen auf Zähler z.B. für die „ID“ der wie folgt zählt:

ASCII-Format	Hex-Format
00000001	30 30 30 30 30 30 30 31
00000002	30 30 30 30 30 30 30 32
00000003	30 30 30 30 30 30 30 33
00000004	30 30 30 30 30 30 30 34
00000005	30 30 30 30 30 30 30 35
00000006	30 30 30 30 30 30 30 36
00000007	30 30 30 30 30 30 30 37
00000008	30 30 30 30 30 30 30 38
00000009	30 30 30 30 30 30 30 39
0000000a	30 30 30 30 30 30 30 61
0000000b	30 30 30 30 30 30 30 62
0000000c	30 30 30 30 30 30 30 63
0000000d	30 30 30 30 30 30 30 64
0000000e	30 30 30 30 30 30 30 65
0000000f	30 30 30 30 30 30 30 66
00000010	30 30 30 30 30 30 31 30
00000011	30 30 30 30 30 30 31 31

ASCII-Format	Hex-Format
00000012	30 30 30 30 30 30 31 32
00000013	30 30 30 30 30 30 31 33

2 REA-PLC

REA-PLC ist der Name für ein Schnittstellenprotokoll zwischen einem Server (Kennzeichnungssystem) und einem Client (Bspw. eine SPS-Steuerung) das speziell für SPS-Anlagen konzipiert wurde. Das von der Firma **REA-Elektronik GmbH** entwickelte Protokoll wird von folgenden Kennzeichnungssystemen unterstützt:

- **REA JET HR**
- **REA JET HR *pro***
- **REA JET CL**

Alle Kennzeichnungssysteme verwenden eine TCP/IP Kommunikation zum Client. Durch einen Schnittstellenwandler (Interfaces) können auch die Schnittstellen RS-422, RS-485 oder RS-232 angebunden werden.

Dieses Protokoll wurde speziell für die Kommunikation für die oben genannten Kennzeichnungssysteme und SPS-Anlagen entwickelt. Es zeichnet sich durch folgende Punkte aus:

- Unterstützung der fünf wichtigsten Kommandos.
- Keine automatische, unaufgeforderte Datenkommunikation.
- Immer die gleiche Antwortlänge auf die fünf implementierten Kommandos. Die Standardantwort enthält auch die Statusinformationen zum Steuergerät und dessen Schreibkopf bzw. Schreibköpfen.



Aus Sicht der Kommunikationsstruktur ist das Kennzeichnungssystem der Server und die SPS-Anlage bzw. der PC der Client. Ein Server kann gleichzeitig mehrere Clients haben, aber nicht umgedreht!

2.1 Verbindungsaufbau zwischen Client und Server

Um Befehle an den Server senden zu können, muss zuvor eine Verbindung vom Client zum Server aufgebaut werden. Auch wenn es sich hierbei auf Ebene von TCP/IP um eine 1:1 Verbindung handelt, sind mehrere gleichzeitige Verbindungen vorstellbar, da ein Server von unterschiedlichen Clients TCP/IP Pakete empfangen kann.

Zu einem Kommunikationsaufbau benötigen Sie eine

- IP-Adresse und eine
- Portnummer

Die IP-Adresse bekommen Sie von ihrem Kennzeichnungssystem, eine Beschreibung wo Sie diese Adresse finden können, entnehmen Sie bitte aus Ihrer Betriebsanleitung vom Kennzeichnungssystem. Üblicherweise über die Funktionstaste <F5> in der Basisanzeige.

Das Kennzeichnungssystem ist vorkonfiguriert (Werkseinstellungen) und verwendet DHCP, das bedeutet die IP-Adresse Ihres Kennzeichnungssystems ändert sich dynamisch mit dem Ein- und Ausschalten des Kennzeichnungssystems.



Die Portnummer hängt von ihrer Anwendung ab. **REA-PLC** unterstützt zwei unterschiedliche Varianten, die über die jeweilige Portnummer ausgewählt werden.

2.1.1 Schnittstellenprotokoll auf Port: 22169 (EOT), ab Firmware 1.83

Dieses Schnittstellenprotokoll verwendet ein Datenendzeichen nach jedem Befehl. Hierfür wird das Steuerzeichen [EOT] (0x04) verwendet, dass sowohl auf Server- und Client Seite immer angehängt wird. Dies ermöglicht auch eine Kommunikation die SPS ähnlich arbeitet, aber zu Erkennung des String-Endes ein Terminierungszeichen benötigt wie z.B. verschiedene Kamerasysteme.

Beispiel für „Start Job“, siehe Kapitel 2.3.2 auf Seite 13:

ASCII-Format:

0002000000020000010 [EOT]

Hex-Format:

30 30 30 32 30 30 30 30 30 30 30 32 30 30 30 30 30
31 30 04

Antwort vom Server:

ASCII-Format:

00020000000200000066000000030000000000260000000000
1000000010000 [EOT]

Hex-Format:

```
30 30 30 32 30 30 30 30 30 30 30 32 30 30 30 30 30
30 36 36 30 30 30 30 30 30 30 33 30 30 30 30 30 30
30 30 30 30 32 36 30 30 30 30 30 30 30 30 30 30 30
31 30 30 30 30 30 30 30 30 31 30 30 30 30 04
```

2.1.2 Schnittstellenprotokoll auf Port: 22170, ab Firmware 1.70

Das Schnittstellenprotokoll ist lngenbezogen, d.h. Strings haben immer eine feste Lnge, unter Vernachlssigung der zu bergebenen Parameter, die ihre Stringlngen als Parametersatz mit bergeben.

2.2 Schnittstellenprotokoll Beschreibung

Ein Befehl ist grundsätzlich wie folgt aufgebaut und beinhaltet mindestens 18 Bytes.

<Befehl><ID><Länge> [<Parameter>]

	Länge	Beschreibung
Befehl	4 Bytes	Siehe Kap. 2.3 auf Seite 12
ID	8 Bytes	Ein beliebiger Wert im Wertebereich: 1 bis FFFFFFFF, siehe Kap. 1.2 auf Seite 5.
Länge	6 Bytes	Gesamtanzahl der Zeichen der folgenden Parameter.
Parameter	{ } Bytes	Befehlsabhängig

Wird ein Befehl erfolgreich an den Server gesendet, gibt der Server automatisch eine Antwort (Response) mit dieser ID und den Befehlscode an den Client unverändert zurück, siehe Kap. 2.4 auf Seite 19.

2.3 Befehle

Befehle sind Kommandos die vom Client an den Server gesendet werden, damit der Server eine Aktion ausführt.

Eine genaue Beschreibung des Befehls und unter welchen Voraussetzungen der Befehl ausgeführt werden kann, entnehmen Sie bitte der Dokumentation Ihres Kennzeichnungssystems. Grundsätzlich werden mit dem Befehl auch die zugehörigen spezifischen Parameter mit übertragen. Diese werden in den folgenden Abschnitten ausführlich beschreiben.

2.3.1 Druckjob zuweisen

Mit diesem Befehl kann ein Druckjob an das Kennzeichnungssystem zugewiesen werden. Ein Druckauftrag besteht üblicherweise aus den Anlageeinstellungen und dem zu druckenden Inhalt.

Prinzipieller Aufbau des Befehls:

<Befehl><ID><Länge><JobId><Dateiname>

Aufbau	Länge	Beschreibung
Befehl:	4 Bytes	0001 ≡ Der Befehl für „Set Job“
ID:	8 Bytes	Ein beliebiger Wert, siehe Kap. 2.2 auf Seite 12
Länge:	6 Bytes	Gesamtanzahl der folgenden Parameter. Hier setzt sich die Länge aus JobId und Dateiname zusammen (Länge = JobId + Dateiname)
JobId:	1 Byte	0 ≡ Job 1 aktuell wird nur Job 1 unterstützt
Dateiname:	{ } Bytes	Name der Jobdatei die bereits im Server vorhanden sein muss. Bitte Groß- und Kleinschreibung beachten.

Ein Beispiel um einen Druckauftrag mit dem Namen „demo-job_1ph.job“ an den Job Nr. 1 zuzuweisen:

ASCII-Format:

0001000000010000100demojob_1ph.job

Hex-Format:

30 30 30 31 30 30 30 30 30 30 30 31 30 30 30 30 31
30 30 64 65 6D 6F 6A 6F 62 5F 31 70 68 2E 6A 6F 62

2.3.2 Druckjob starten

Erst nach dem der Druckjob gestartet wurde ist das Steuergerät bereit den Druckauftrag auszuführen. Der Befehl ist identisch mit der Taste <Start> am Steuergerät.

Prinzipieller Aufbau des Befehls:

<Befehl><ID><Länge><JobId>

Aufbau	Länge	Beschreibung
Befehl:	4 Bytes	0002 ≡ Der Befehl für „Start Job“
ID:	8 Bytes	Ein beliebiger Wert, siehe Kap. 2.2 auf Seite 12
Länge:	6 Bytes	Gesamtanzahl der Zeichen der folgenden Parameter. Hier ist das nur die JobId , die Länge ist somit immer gleich eins.
JobId:	1 Byte	0 ≡ Job 1 aktuell wird nur Job 1 unterstützt

Ein Beispiel um den Druckjob Nr. 1 zu starten:

ASCII-Format:

00020000000010000010

Hex-Format:

30 30 30 32 30 30 30 30 30 30 30 31 30 30 30 30 30
31 30

2.3.3 **Druckjob stoppen**

Im Gegensatz zum Druckjob starten, stoppt dieser Befehl den Druckjob. Der Befehl ist identisch mit der Taste <Stop> am Steuergerät.

Prinzipieller Aufbau des Befehls:

<Befehl><ID><Länge><JobId>

Aufbau	Länge	Beschreibung
Befehl:	4 Bytes	3 ≡ Der Befehl für „Stopp Job“
ID:	8 Bytes	Ein beliebiger Wert, siehe Kap. 2.2 auf Seite 12
Länge:	6 Bytes	Gesamtanzahl der Zeichen der folgenden Parameter. Hier JobId , die Länge ist somit immer gleich eins
JobId:	1 Byte	0 ≡ Job 1 aktuell wird nur Job 1 unterstützt

Ein Beispiel um den Druckjob Nr. 1 zu stoppen:

ASCII-Format:

00030000000010000010

Hex-Format:

30 30 30 33 30 30 30 30 30 30 30 31 30 30 30 30 30
31 30

2.3.4 **Zeichensatzumschaltung**

Dieser Befehl ist nur nötig, wenn Sie Dateninhalte in einem anderen Format wie ASCII (Standard) übertragen möchten. Das betrifft nur den eigentlichen „**Inhalt**“ und nicht den übrigen Aufbau eines Befehls wie Bspw. „**Befehl**“, „**ID**“, „**Länge**“ etc.

Prinzipieller Aufbau des Befehls:

<Befehl><ID><Länge><Encoding>

Aufbau	Länge	Beschreibung
Befehl:	4 Bytes	CODE (Achtung: Das zweite Zeichen ist eine 0 (Null) und kein O.)
ID:	8 Bytes	Ein beliebiger Wert, siehe Kap. 2.2 auf Seite 12
Länge:	6 Bytes	Gesamtanzahl der Zeichen der folgenden Parameter.
Encoding:	{ } Byte	Folgende Zeichen-Formate werden unterstützt, wobei ASCII voreingestellt (Standard) ist: ASCII, GB18030, GB2312, GBK, UCS-2, UCS-2BE, UCS-2LE, UCS-4, UCS-4BE, UCS-4LE, UTF-8, UTF-16, UTF-16BE, UTF-16LE, UTF-32, UTF-32BE, UTF-32LE

Ein Beispiel um auf den Code UTF-16BE umzuschalten:

ASCII-Format:

CODE00000001000008UTF-16BE

Hex-Format:

43 30 44 45 30 30 30 30 30 30 30 32 30 30 30 30 30
38 55 54 46 2D 31 36 42 45

Hinweis: Der Befehl wird erst ab Firmware: 3.61.66034~4633 unterstützt.

2.3.5 Objektinhalte überschreiben

Mit diesem Befehl können Objektinhalte überschrieben oder geändert werden.

Prinzipieller Aufbau des Befehls:

```
<Befehl><ID><Länge>
<ID Länge>
<JobId>;<Gruppenname>;<Objektname>;<Inhaltsname>
<Inhaltlänge>
<Inhalt>
...
<ID Länge>
<JobId>;<Gruppenname>;<Objektname>;<Inhaltsname>
<Inhaltlänge>
<Inhalt>
```

Der verwendete Gruppenname und die verwendeten Objektnamen und Inhaltsnamen sind durch die Anlageneinstellungen oder bei der Erstellung des Labels bereits vorgesehen, siehe hierzu Handbuch „**REA JET LabelCreator**“.

Aufbau	Länge	Beschreibung
Befehl:	4 Bytes	0004 ≡ Der Befehl für „Set Label Contents“
ID:	8 Bytes	Ein beliebiger Wert, siehe Kap. 2.2 auf Seite 12
Länge:	6 Bytes	Gesamtanzahl der Zeichen der folgenden Parameter.
ID Länge:	4 Bytes	Anzahl der Zeichen von: JobId + Gruppenname + Objektname + Inhaltsname + 3 (Semikolon als Trennzeichen)
JobId:	1 Byte	0 ≡ Job 1 aktuell wird nur Job 1 unterstützt
Trennzeichen:	1 Byte	Immer Semikolon
Gruppenname:	{ } Bytes	Name der Gruppe Bitte Groß- und Kleinschreibung beachten.
Trennzeichen:	1 Byte	Immer Semikolon
Objektname:	{ } Bytes	Name des Objekts Bitte Groß- und Kleinschreibung beachten.
Trennzeichen:	1 Byte	Immer Semikolon
Inhaltsname:	{ } Bytes	Name des Inhaltes Bitte Groß- und Kleinschreibung beachten.
Inhaltslänge:	4 Bytes	Anzahl der Zeichen des Inhaltes
Inhalt:	{ } Bytes	Der eigentliche neue Inhalt
...		
ID Länge:	4 Bytes	Anzahl der Zeichen von: JobId + Gruppenname + Objektname + Inhaltsname + 3 (Semikolon als Trennzeichen)
JobId:	1 Byte	0 ≡ Job 1 aktuell wird nur Job 1 unterstützt
Trennzeichen:	1 Byte	Immer Semikolon
Gruppenname:	{ } Bytes	Name der Gruppe Bitte Groß- und Kleinschreibung beachten.
Trennzeichen:	1 Byte	Immer Semikolon
Objektname:	{ } Bytes	Name des Objekts Bitte Groß- und Kleinschreibung beachten.
Trennzeichen:	1 Byte	Immer Semikolon
Inhaltsname:	{ } Bytes	Name des Inhaltes Bitte Groß- und Kleinschreibung beachten.

Aufbau	Länge	Beschreibung
Inhaltslänge:	4 Bytes	Anzahl der Zeichen des Inhaltes
Inhalt:	{ } Bytes	Der eigentliche neue Inhalt

Ein Beispiel um zwei Objekthinhalte mit „REA Elektronik GmbH“ und „?“ zu überschreiben.

ASCII-Format:

```
0004000000001000062
001a0;Front;Test-Text_1;Text_1
0013REA Elektronik GmbH
00240;Front;Test-Text_2;Exclamation-mark
0001?
```

Hex-Format:

```
30 30 30 34 30 30 30 30 30 30 30 31 30 30 30 30 36
32 30 30 31 61 30 3B 46 72 6F 6E 74 3B 54 65 73 74
2D 54 65 78 74 5F 31 3B 54 65 78 74 5F 31 30 30 31
33 52 45 41 2D 45 6C 65 6B 74 72 6F 6E 69 6B 20 47
6D 62 48 30 30 32 34 30 3B 46 72 6F 6E 74 3B 54 65
73 74 2D 54 65 78 74 5F 32 3B 45 78 63 6C 61 6D 61
74 69 6F 6E 2D 6D 61 72 6B 30 30 30 31 21
```

Nebenrechnung für die ID Länge:

JobId:	0	1 Zeichen
Separator	Semikolon	1 Zeichen
Gruppenname:	Front	5 Zeichen
Separator	Semikolon	1 Zeichen
Objektname:	Test-Text_1	11 Zeichen
Separator	Semikolon	1 Zeichen
Inhaltsname:	Text_1	6 Zeichen
=====		
Summe:		26 Zeichen ≡ (HEX: <u>1A</u>)

2.3.6 Objekteigenschaften überschreiben

Ab Firmware 3.20

Mit diesem Befehl können Eigenschaften von Objekten überschrieben oder geändert werden. Folgende Objekteigenschaften sind möglich, aber stehen nicht für alle Objekte zur Verfügung:

Objekteigenschaften	Einheit	Beschreibung
Position/X@value	mm	Numerische Kommazahl, z.B. "1.5", "6"
Position/Y@value	mm	Numerische Kommazahl, z.B. 1.5", "2.346"
Position/Z@transparency	[bool]	Boolescher Wert: - Wahr := „True“ oder „true“ - Falsch := {}
Size/Width@value	mm	Breite des Objektes Numerische Kommazahl, z.B. "20", "40.123"
Size/Height@value	mm	Numerische Kommazahl, z.B. "12", "6.34"
HiddenCount@value	[]	Numerischer, ganzzahliger positiver Wert mit Anzahl der Drucke, in denen das Objekt ausgeblendet ist: -1 := Das Objekt ist immer ausgeblendet 0 := Objekt ist immer eingeblendet N := Anzahl der Drucke bis das Objekt wieder eingeblendet wird, z.B. „14“, „1“, „2“
Inverted@value	[bool]	Boolescher Wert: - Wahr := „True“ oder „true“ - Falsch := {}
Rotation@value	°	Numerische Kommazahl mit dem Drehungswinkel im Uhrzeigersinn von 0 bis 360 Grad (für HR nur in 90 Grad Schritten), z.B. "90", "37.57"
Font/NameEmphasis@value	[string]	Eine Zeichenkette mit gültigen Namen und Schriftschnitt einer installierten Schriftart, getrennt durch einen Schrägstrich, z.B. "FreeSans/Medium"
Content@value	[string]	Eine Zeichenkette mit beliebiger Länge und Inhalt, ASCII/ANSI (7 Bit Zeichen) kodiert, z.B. "Hello World"

Bei den Objekteigenschaften ist auf die Schreibweise zu achten, es wird zwischen Groß- und Kleinbuchstaben unterschieden.

Prinzipieller Aufbau des Befehls:

```

<Befehl><ID><Länge>
<ID Länge>
<JobId>;
<Gruppenname>;<Objektname>;<Inhaltsname>;<Objekteigenschaft>
<Inhaltslänge>
<Inhalt>
...
<ID Länge>
<JobId>;<Gruppenname>;<Objektname>;<Inhaltsname>;<Objekteigenschaft>
<Inhaltslänge>
<Inhalt>

```

Bei allen Objekteigenschaften, mit Ausnahme von „Content@value“ kann die Angabe des Inhaltsnamens entfallen.

Der verwendete Gruppenname und die verwendeten Objektnamen und Inhaltsnamen sind durch die Anlageneinstellungen bzw. bei der Erstellung des Labels bereits vorgesehen, siehe hierzu Handbuch „**REA JET LabelCreator**“.

Aufbau	Länge	Beschreibung
Befehl:	4 Bytes	5 ≡ Der Befehl für „Set Label Object“ (HEX: 0005)
ID:	8 Bytes	Ein beliebiger Wert, siehe Kap. 2.2 auf Seite 12
Länge:	6 Bytes	Gesamtlänge des Datenstrings, ohne Befehl , ID und der Länge selbst, aber inklusive des Semikolons.
ID Länge:	4 Bytes	Anzahl der Zeichen von: JobId + Gruppenname + Objektname + Inhaltsname + Eigenschaft + 4 (Semikolon als Trennzeichen)
JobId:	1 Byte	0 ≡ Job 1 aktuell wird nur Job 1 unterstützt
Trennzeichen:	1Byte	Immer Semikolon
Gruppenname:	{ } Bytes	Name der Gruppe Bitte Groß- und Kleinschreibung beachten.
Trennzeichen:	1Byte	Immer Semikolon
Objektname:	{ } Bytes	Name des Objekts Bitte Groß- und Kleinschreibung beachten.
Trennzeichen:	1Byte	Immer Semikolon
Inhaltsname:	{ } Bytes	Name des Inhaltes Bitte Groß- und Kleinschreibung beachten.

Aufbau	Länge	Beschreibung
Trennzeichen:	1Byte	Immer Semikolon
Objekteigenschaft:	{ Bytes	Bezeichnung der Eigenschaft, siehe oben Bitte Groß- und Kleinschreibung beachten.
Inhaltslänge:	4 Bytes	Anzahl der Zeichen des Inhaltes
Inhalt:	{ Bytes	Der eigentliche neue Inhalt
...		
ID Länge:	4 Bytes	Anzahl der Zeichen von: JobId + Gruppenname + Objektname + Inhaltsname + Eigenschaft + 4 (Semikolon als Trennzeichen)
JobId:	1 Byte	0 ≡ Job 1 aktuell wird nur Job 1 unterstützt
Trennzeichen	1Byte	Immer Semikolon
Gruppenname:	{ Bytes	Name der Gruppe Bitte Groß- und Kleinschreibung beachten.
Trennzeichen:	1Byte	Immer Semikolon
Objektname:	{ Bytes	Name des Objekts Bitte Groß- und Kleinschreibung beachten.
Trennzeichen:	1Byte	Immer Semikolon
Inhaltsname:	{ Bytes	Name des Inhaltes Bitte Groß- und Kleinschreibung beachten.
Trennzeichen:	1Byte	Immer Semikolon
Objekteigenschaft:	{ Bytes	Bezeichnung der Eigenschaft, siehe oben Bitte Groß- und Kleinschreibung beachten.
Inhaltslänge:	4 Bytes	Anzahl der Zeichen des Inhaltes
Inhalt:	{ Bytes	Der eigentliche neue Inhalt

Ein Beispiel um eine Objekteigenschaft („Position/X@value“) auf 10mm zu ändern.

ASCII-Format:

```
00050000000100002f00250;Front;Test-Text_1;;
Position/X@value000220
```

Hex-Format:

```
30 30 30 35 30 30 30 30 30 30 30 31 30 30 30 30 32
66 30 30 32 35 30 3B 46 72 6F 6E 74 3B 54 65 73 74
2D 54 65 78 74 5F 31 3B 3B 50 6F 73 69 74 69 6F 6E
2F 58 40 76 61 6C 75 65 30 30 30 32 32 30
```

Ein Beispiel um ein Objektinhalte („Content@value“) mit REA-Elektronik GmbH zu überschreiben.

ASCII-Format:

```
000500000000100004300280;Front;Test-  
Text_1;Text_1;Content@value0013REA-Elektronik GmbH
```

Hex-Format:

```
30 30 30 35 30 30 30 30 30 30 30 31 30 30 30 30 34  
33 30 30 32 38 30 3B 46 72 6F 6E 74 3B 54 65 73 74  
2D 54 65 78 74 5F 31 3B 54 65 78 74 5F 31 3B 43 6F  
6E 74 65 6E 74 40 76 61 6C 75 65 30 30 31 33 52 45  
41 2D 45 6C 65 6B 74 72 6F 6E 69 6B 20 47 6D 62 48
```


2.4 Antworten vom Server (Response)

Das Schnittstellenprotokolle auf Port: 22170 hat eine Antwortlänge von 64Bytes. Bei dem Schnittstellenprotokoll auf Port: 22169 wird das Datenendzeichen [EOT] (0x04) hinzugefügt, somit ist die Antwort insgesamt 65Bytes lang.

Die ersten 64Bytes in allen Protokollen haben folgende Bedeutung:

Aufbau	Länge	Beschreibung
Befehl	4 Bytes	Der Befehl der ausgeführt wurde
ID	8 Bytes	Die gleiche ID die mit dem Befehl zum Server gesendet wurde
Fehlercode	8 Bytes	siehe Kap. 2.4.4 auf Seite 24
Gerätestatus	4 Bytes	Der Wert beträgt z.Z. immer Null (HEX: 0000)
Jobstatus	8 Bytes	Druckauftragstatus, siehe Kap. 2.4.1 auf Seite 23
Statusfeld	32 Bytes	Statusinformationen, siehe Kap. 2.4.3 auf Seite 23

2.4.1 *Befehl*

Der Empfangene Befehl (Druckjob zuweisen/starten/stoppen etc.) siehe Kapitel 2.3 ab Seite 12 wird vom Server an dieser Stelle wieder zurückgegeben. Wenn der Server den empfangenen Befehl nicht kennt, dann sendet der Server nicht den empfangenden Befehlscode, sondern „FFFF“ zurück.

2.4.2 *Jobstatus*

Die folgende Tabelle gibt den jeweiligen Jobstatus zurück.

Jobstatus	Beschreibung
0000xxxx	Kein Job zugewiesen und keine Druckfreigabe
0001xxxx	Ein Job ist zugewiesen, aber keine Druckfreigabe
0002xxxx	Fehler! Druckfreigabe aber kein Job zugewiesen, dieser Zustand dürfte niemals eintreffen!
0003xxxx	Ein Job ist zugewiesen und der Druck ist freigegeben

2.4.3 *Statusfeld*

Die Interpretation des Statusfelds ist vom jeweiligen Server (Steuergerät) abhängig.

2.4.3.1 Statusfeld: REA JET HR und REA JET HR pro

Für den **REA JET HR** und den **REA JET HR pro** werden die jeweiligen Tintenfüllstände und Kartuschenstatusinformationen im Statusfeld zurückgegeben. Die Informationen für Tintenfüllstände können nur von Kartuschen ermittelt werden, die über einen integrierten Chip verfügen. Unabhängig vom Kennzeichnungssystem und der verwendeten Bestückung wird immer das volle Datenwort zurückgegeben. Hiermit wird die Einheitlichkeit konsequent gewährleistet.

Aufbau	Länge	Beschreibung
Kartusche 1	4 Bytes	Siehe Tabelle Kartuschen Status
Kartusche 1	4 Bytes	Beinhaltet den Tintenfüllstand in ml (Milliliter)
Kartusche 2	4 Bytes	Siehe Tabelle Kartuschen Status
Kartusche 2	4 Bytes	Beinhaltet den Tintenfüllstand in ml (Milliliter)
Kartusche 3	4 Bytes	Siehe Tabelle Kartuschen Status
Kartusche 3	4 Bytes	Beinhaltet den Tintenfüllstand in ml (Milliliter)
Kartusche 4	4 Bytes	Siehe Tabelle Kartuschen Status
Kartusche 4	4 Bytes	Beinhaltet den Tintenfüllstand in ml (Milliliter)

Kartuschen Status	Beschreibung
Bit 0:	Kartusche ist eingelegt
Bit 1:	Kartusche ist leer
Bit 2:	Kartuschentemperatur zu hoch
Bit 3:	Tintenfüllstand unter der Sollmarke
Bit 4:	Kartuschentemperatur über der Sollmarke

2.4.3.2 Statusfeld: REA JET CL

Das Statusfeld beim **REA JET CL** ist zurzeit nicht spezifiziert.

2.4.4 Fehlercodes

Die Fehlercodes sind z.Z. nur in Englischer Sprache verfügbar.

Fehlercode	Beschreibung
0000xxxx	No Error
0001xxxx	print job not started
0002xxxx	unknown severe error

Fehlercode	Beschreibung
0003xxxx	unknown error
0004xxxx	invalid parameters
0005xxxx	fatal error, device will restart
0006xxxx	a memory exception occurred
0007xxxx	Not supported
000A01F4	file not found
000B01F4	cannot open file
000A0065	invalid label tag
00140065	label used resource missing
00190065	label used bitmap missing
001E0065	label used font missing
005A0065	access to label or label tag not granted
00620065	object content set without any changes
00630065	object content set produces an stop condition/request for printing
000A0066	you cannot do some actions while running a job
000B0066	you cannot do some actions while NOT running a job
000C0066	There is no active/assigned job
000D0066	job contains no group
000E0066	job does not exist
000F0066	group contains no label
00100066	tried action on not assigned group
00110066	invalid ink information
00120066	print head not ready
00130066	shaft encoder not configured
00140066	prerendering at activation failed (probably a broken label)
00150066	a entity of the printing system is active and cannot be (re)parameterized
00160066	rendering failed, no label-image from renderer available
00170066	you cannot do some actions while job is purging
00180066	not all printheads/cartridges found are included in a job
00190066	can not purge with double existing printheads/cartridges
001A0066	can not purge without any printheads/cartridges
001B0066	can not purge printhead/cartridge <id> in a running job
001C0066	can not purge not locked or not connected cartridge <id>

Fehlercode	Beschreibung
001D0066	group does not exist
001E0066	group must not be active to perform
001F0066	group has to be active to perform
00200066	no buffer to store the image data for group
00210066	group has to contain at least one printhead
00220066	desired hardware not supported
00230066	group must not contain printheads with identical IDs
00240066	group must not be executing purge to perform
00250066	group/job can not be started within external stopped condition
00320066	type of ink for spitting doesn't match with settings
000A0067	common error in parsing xml
000B0067	error in xml-syntax
000C0067	xml-document doesn't contain valid root node
000F0067	xml-document invalid or missing job node
00100067	xml-document job version not supported
00130067	file of xml-document for label not found
00140067	xml-document invalid or missing label node
00150067	xml-document Label version not supported
00280067	logic error in xml-tag
00290067	error in xml-tag label
002A0067	error in xml-tag layout (of label)
002C0067	error in xml-tag object (of label)
002D0067	error in xml-tag renderer (of label)
002E0067	xml-tag renderer type (of label) not supported
002F0067	error in xml-tag content (of label object)
00300067	xml-tag content type (of label object) not supported
00310067	xml-tag renderer type needs content
00360067	xml-tag referenced content can not be found
00460067	unknown feature
000A0068	rendering result image failed
000B0068	cannot create or start render thread
00140068	invalid size of destination image
001E0068	cannot create renderer

Fehlercode	Beschreibung
001F0068	cannot create renderer for text out
00230068	cannot create renderer for images
00240068	cannot create renderer for barcodes
00280068	cannot create renderer for graphical objects
00320068	error in initialize Freetype library
003C0068	error in initialize TBarcode library
003E0068	cannot render barcode object
00500068	render failed on invalid format of source image
00520068	cannot render bitmap object
00530068	cannot render image because of expired print count
005A0068	render failed on invalid shiftcode
006E0068	code list for is containing insufficient entries
000A00C8	label data for FPGA too late
000B00C8	pulses from shaft encoder too fast
000C00C8	unsteady pulses from shaft encoder
001400C8	invalid shaft encoder parameter(s)
001E00C8	hardware subsystem missing
002800C8	number of layers exceeded / not supported
003200C8	invalid FPGA configuration
003C00C8	failed to copy image to FPGA
004600C8	logical AND-operation with edges not allowed
005000C8	if command updatefonts fails
000B012C	print head not connected
000C012C	print head is not locked
000D012C	initialization of print head failed
000E012C	print head contains no cartridge
000F012C	no ink left in cartridge
0010012C	print head temperature too high
000A0136	print cartridge has no chip
000B0136	cartridge-chip communication failure
000A0190	event subscription failed

Fehlercode	Beschreibung
00140190	xml-document REA-PI version not supported
00150190	version already selected

3 Software zu Test und Evaluierungszwecken

Die Firma **REA-Elektronik GmbH** stellt Software für den PC zur Verfügung, um die PLC-Schnittstelle testen zu können. Dies gibt Benutzern die Möglichkeit sich schneller mit dem Schnittstellenprotokoll vertraut zu machen.

3.1 REA-PLC Tester, ab Version: 1.14.00

Der **REA-PLC** Tester ist eine Software die im Standardlieferungsumfang des Steuergerätes mit ausgeliefert wird und befindet sich in komprimierter Form auf dem beigelegten Datenträger.

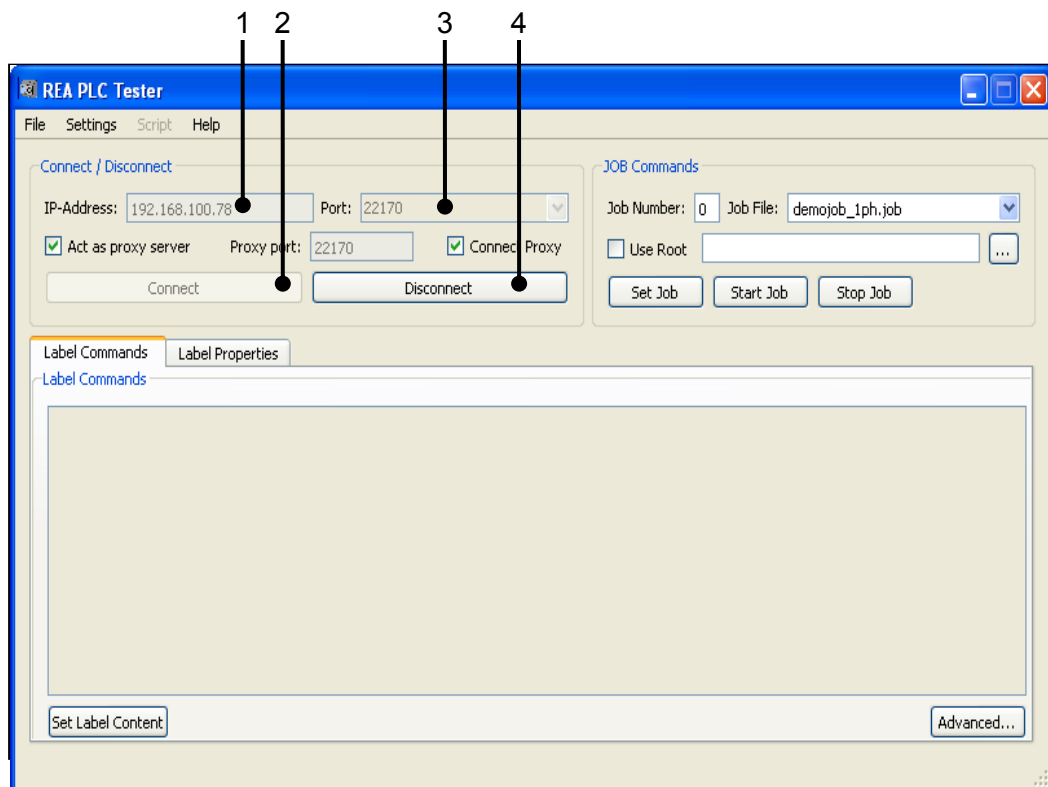
.....

Dieses Programm ist ausschließlich für Testzwecke gedacht und ist nicht für eine Produktionsumgebung ausgelegt.

3.1.1 *Verbindungsaufbau (Connect)*

Nach dem Aufruf des Programms (EXE- Datei) erscheint das Hauptfenster, siehe Abbildung 1 auf Seite 30. Hier können die Informationen die für einen Verbindungsaufbau notwendig sind eingetragen werden, siehe Kap. 2.1 auf Seite 9. Mit der Schaltfläche (Button) „Connect“ wird versucht eine Verbindung zum Server herzustellen. Ist eine Verbindung hergestellt wird die Schaltfläche „Connect“ inaktiv und die Schaltfläche „Disconnect“ dafür aktiv, andernfalls folgt nach einem Timeout eine Fehlermeldung.

Falls Sie die Kommunikation Ihres Clients (PLC) testen möchten, können Sie alternativ auch eine Verbindung zu dem integrieren Proxy Server im **REA-PLC Tester** aufbauen, siehe Kap. 0 auf Seite 35.



Nr.	Beschreibung
1	IP-Adresse des Servers (Format: xxx.xxx.xxx.xxx)
2	Schaltfläche um eine Verbindung mit dem Server aufzubauen
3	Portnummer, siehe Kap. 2.1.1 auf Seite 10
4	Verbindung zum Server trennen

3.1.2 Druckjob Befehle (Set Job, Start Job, Stop Job)

Erst wenn ein Verbindungsaufbau erfolgreich stattgefunden hat werden die Symbole im Fenster „JOB Commands“ aktiv. In der Abbildung 2 ist zu sehen, dass sich auf dem Server eine Job-Datei mit dem Namen „demojob_1ph.job“ befindet.

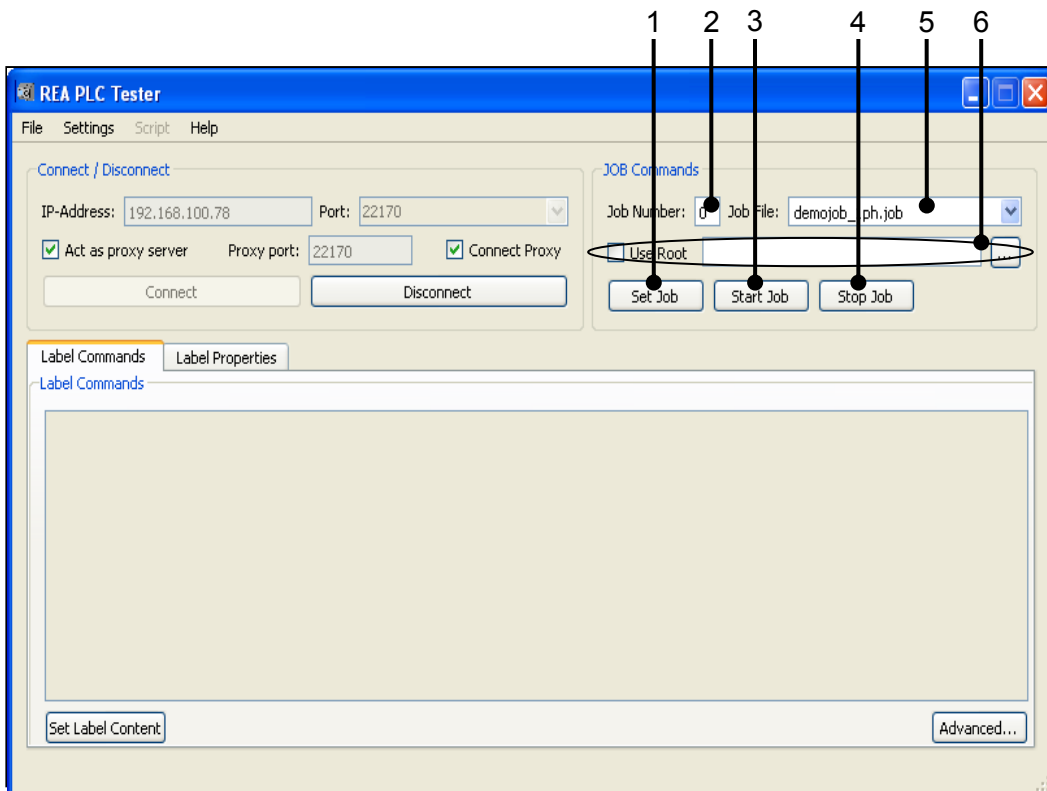


Abbildung 2: REA-PLC Tester, Job Command mit einem REA-JET HK

Nr.	Beschreibung
1	Einen Druckauftrag (Job) zuweisen, siehe Kap. 2.3.1 auf Seite 12
2	Zuweisen an die Job Nr. 0 ≡ Job 1 aktuell wird nur ein Job unterstützt
3	Einen Text zum Druck freigeben, siehe Kap. 2.3.2 auf Seite 13
4	Eine Druckfreigabe zurücknehmen, siehe Kap. 2.3.3 auf Seite 14
5	Auswahl des Druck-Jobs (Dateiname), siehe Kap. 2.3.1 auf Seite 12
6	Druckjobs von einer externen Quelle (Netzwerk oder ein lokales Laufwerk) auswählen. Siehe auch Kap. 0 auf Seite 35

3.1.3 Objektinhalte überschreiben (Set Label Content)

Erst nach dem die Schaltfläche „Set Job“ ausgewählt wurde, wird der angegebene Job im Feld „Job File“ an den Server zugewiesen und änderbare Inhalte werden im Fenster „Label Commands“ aufgelistet.

Hier können jetzt die einzelnen Felder abgeändert werden und über die Schaltfläche „Set Label Content“ werden diese an den Server übermittelt und gesetzt. Siehe hierzu Kap. 2.3.4 auf Seite 14

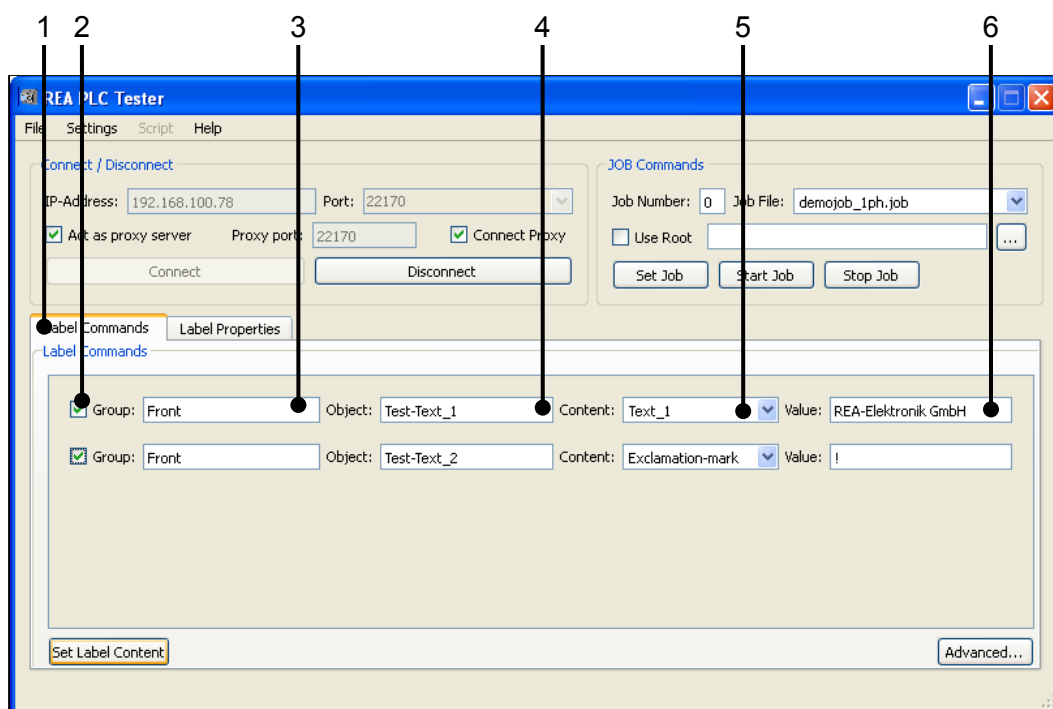


Abbildung 3: REA-PLC Tester, Set Label Content mit einem REA-JET HR

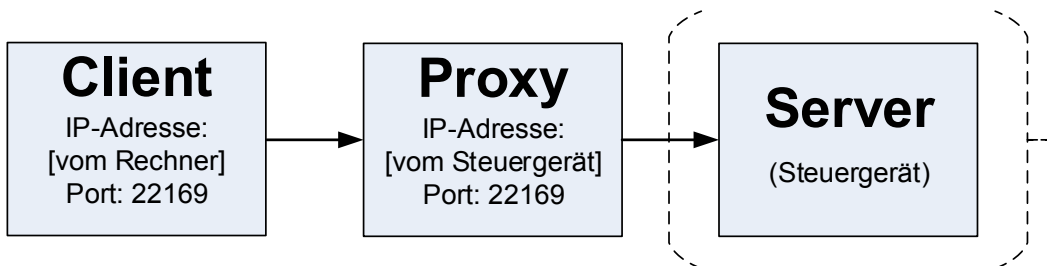
Nr.	Beschreibung
1	Nur ein Inhalt der ausgewählt wurde, wird mit der Schaltfläche „Set Label Content“ an den Server übertragen
2	Schaltfläche „Set Label Content“ zum Übertragen der Textinhalte an den Server
3	Name der Gruppe
4	Name des Objekts
5	Name des Inhalts
6	Inhalt der neu geschrieben werden soll

Nr.	Beschreibung
1	Löscht den Inhalt des Log-Fensters, siehe Punkt 2
2	Ein Log-Fenster um den Datenaustausch zwischen dem Server und dem Client verfolgen zu können.
3	Eingabefeld für eine Antwort (64Byte lang) , die über die Schaltfläche „Respond“ abgeschickt werden kann.
4	Angabe einer Datei, in die alle Einträge die im Log-Fenster zu sehen sind abgespeichert werden
5	Eingabefeld für einen Befehl der an den Server gesendet werden soll, siehe hierzu Kap. 2.3 auf Seite 12.
6	Hierüber wird der Datenstring der im Feld 5 eingetragen wurde abgesendet.
7	Öffnet das Advanced Fenster
8	Hierüber wird der Datenstring der im Feld 3 eingetragen wurde abgesendet.

3.1.5 Proxy

Um sich mit dem **REA-PLC** Schnittstellenprotokoll vertraut zu machen, oder Kommunikationsfehler aufspüren zu können ist kein Steuergerät (Server) notwendig. Sie können eine Verbindung über einen sogenannten Proxy aufbauen. In diesem Fall übernehmen Sie manuell die Funktionen vom Server und vom Client. Wie Sie das tun können, entnehmen Sie bitte dem Kap. 2.19 ab Seite 12.

Prinzipieller Aufbau einer Proxy Verbindung:



3.1.5.1 Beispielkonfiguration

3.1.5.1.1 Beispielkonfiguration für den Client:

- IP-Address: 192.168.101.139
IP-Adresse des Rechners
- Port: [siehe Kap.2.1 auf Seite 9, oder beliebig]

3.1.5.1.2 Beispielkonfiguration für den Proxy:

- IP-Address: 192.168.100.78
IP-Adresse des Servers (Steuergeräts), ist aber für eine reine Proxy Verbindung nicht erforderlich!
- Port: [siehe Kap.2.1 auf Seite 9]
- Act as proxy server: ✓
- Proxy port: [Siehe „Einstellungen am Client“]
- Connect Proxy: [beliebig, empfohlen wird: ✓]

In der Praxis kann das dann wie folgt aussehen:



3.1.5.2 Beschreibung der GUI für die Proxy Komponenten

Die folgenden Punkte werden für eine Proxykonfiguration benötigt:

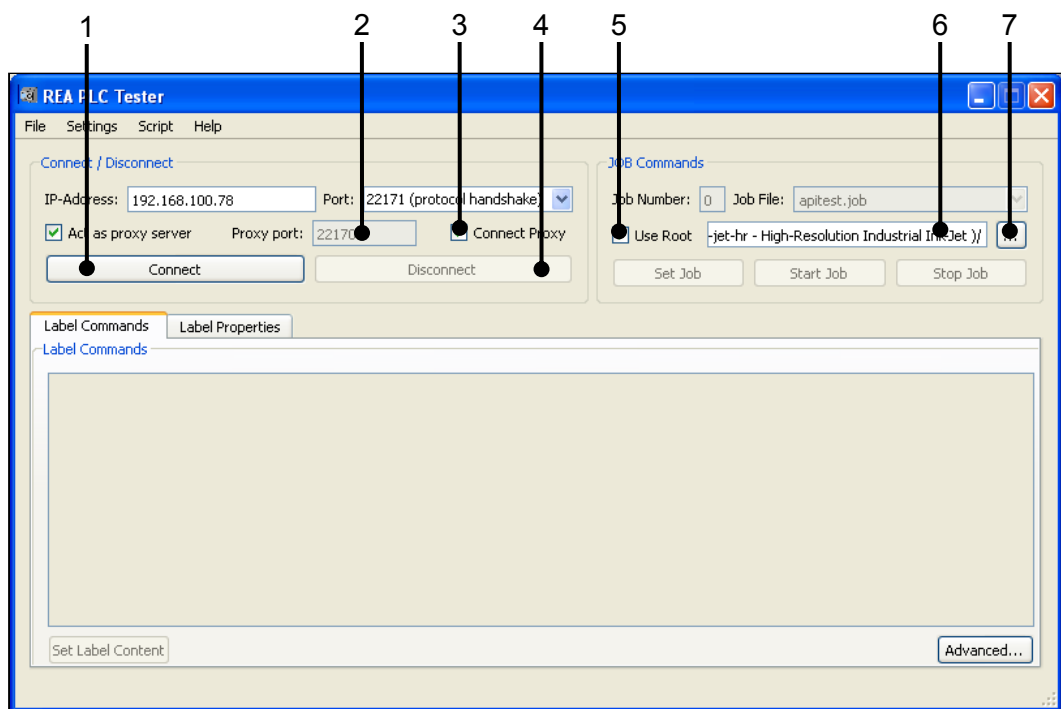


Abbildung 5: REA-PLC Tester, Proxy Verbindung mit einem REA JET HR

Nr.	Beschreibung
1	Eine Verbindung zum Proxy Server aufbauen, auf der Clientseite Eine Verbindung zum Server aufbauen, auf der Proxy Seite
2	Für eine Proxy Verbindung muss ein Port angegeben werden, dieser kann identisch sein mit dem Port auf der Serverseite
3	Auf der Proxy Seite wird in Abhängigkeit des Clients eine Verbindung automatisch auf- bzw. abgebaut, siehe „Nr. 1“ und „Nr. 4“.
4	Die Verbindung, siehe „Nr. 1“ beenden
5	Muss angewählt werden, wenn die Verzeichnisstruktur von einem Server genommen und nur auf den Root Ordner (siehe Punkt 6) verwiesen wird.
6	Pfad des Datenquellordners wo unter anderem die Job-Dateien abgelegt sind
7	Fenster zur Auswahl des Datenquellordners öffnen.

3.1.5.3 Kochrezept für eine Proxyverbindung

Sie sollten folgende Punkt für eine Proxy Verbindung beachten:

Für dieses Beispiel wurde über das „Samba Protokoll“ eine Datenkopie vom Server (Steuergerät (**REA-JET HR**)) auf einem lokalen Laufwerk angelegt um einen bereits definierten Job zuweisen zu können.

3.1.5.3.1 Proxykonfiguration

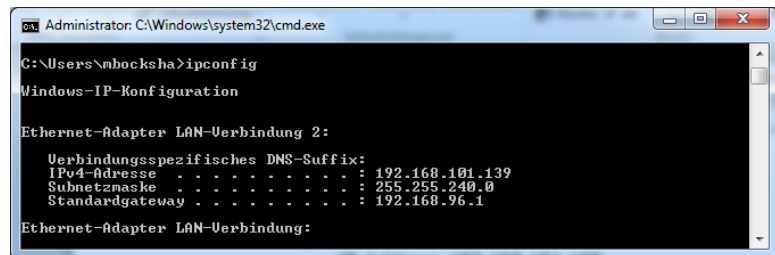
Starten Sie das Programm „REA PLC Tester“.

- Wählen Sie ein Verzeichnis aus, siehe bitte Punkt 7 in der nachfolgenden Abbildung.
- Aktivieren Sie das Kontrollfeld (siehe Punkt 5) „Use Root“. Danach sollten Sie, sofern Sie Daten im Verzeichnis haben im Feld „Job File:“ ein Job angezeigt bekommen, siehe folgende Abbildung mit dem „Job file: apitest.job“.
- Ein „Connect“ ist für eine reine Proxy Verbindung nicht nötig, insofern müssen Sie hier auch keine IP-Adresse vom Server (Steuergerät) eintragen!

3.1.5.3.2 Clientkonfiguration

Starten Sie ein weitere REA PLC Tester (Client) Instanz

- Als erstes sollten Sie die IP-Adresse des Rechners ermitteln. Dazu können Sie bspw. über das Programm „cmd.exe“ auf Ihrem Rechner (OS: Windows) aufrufen und das Kommando „ipconfig“ absetzen. Sie sollten dann folgende Informationen über Ihren Rechner erhalten.



```
Administrator: C:\Windows\system32\cmd.exe

C:\Users\nbocksha>ipconfig

Windows-IP-Konfiguration

Ethernet-Adapter LAN-Verbindung 2:

    Verbindungsspezifisches DNS-Suffix:
    IPv4-Adresse . . . . . : 192.168.101.139
    Subnetzmaske . . . . . : 255.255.240.0
    Standardgateway . . . . . : 192.168.96.1

Ethernet-Adapter LAN-Verbindung:
```

In diesem konkreten Beispiel lautet die IP-Adresse:
192.168.101.139, siehe Beispielkonfiguration.

- Tragen Sie diese IP-Adresse in das Feld „IP-Adresse“ ein
- Wählen Sie den gleichen Port aus wie bei der Proxykonfiguration
- Verbinden Sie sich jetzt mit dem Proxy, indem Sie die Schaltfläche „Connect“ anklicken.

3.1.6 *Einstellungen speichern (Save settings)*

Um Einstellungen nicht immer erneut eingeben zu müssen, können diese über das Menü `File, Save settings` abgespeichert werden.

Hierbei werden unter anderem die folgenden Daten wie IP-Adresse, Port, Act as proxy server, Proxy port gespeichert.

Die Daten werden in der Registrierungsdatenbank von MS- Windows im Pfad: `HKEY_USERS, [], Software, REA-Elektronik GmbH, REA-PLC Tester` Benutzerbezogen abgespeichert.

3.1.7 *Einstellungen löschen (Clean settings)*

Hierüber werden die unter Kap. 3.1.6 auf Seite 37 vorgenommen Einstellungen gelöscht bzw. zurückgesetzt. Die Auswirkungen werden erst nach einem erneuten Programmstart ersichtlich.

4 titan-add-on-serial2plc

Die Firma **REA Elektronik GmbH** bietet eine Softwarelösung an, mit der die Daten über die serielle Schnittstelle gesendet werden kann. Zu beachten ist hierbei das der Datenstring mit 0x0D abgeschlossen sein muss.

Die Antworten vom Server (Response) werden auch mit einem zusätzlichen 0x0D 0x0A abgeschlossen.

Beispieldatensätze für das Terminalprogramm „SerMoni.exe“ das Sie kostenlos von der Firma **REA-Elektronik GmbH** beziehen können.

“00040000000300001f00160;1;Test-Text_1;Text_10001A” 0D

“00040000000300001f00160;1;Test-Text_1;Text_10001B” 0D

Wenn das Steuergerät bereit ist für die Annahme der Telegramme wird auf die serielle Schnittstelle der Text: „**REA-JET is READY** 0x0D 0x0A“ ausgegeben.

4.1 serial_parameter.ini

Im Verzeichnis [IP-Adresse des Steuergeräts]\rea-jet\service\“ befindet sich die Datei „serial_parameter.ini“. Hierrüber können die serielle Schnittstellenparameter nachträglich geändert werden. Bei jeder Änderung muss ein Neustart des Steuergeräts durchgeführt werden.

Die erste serielle Schnittstelle kann wie folgt heißen, je nach Gerätetyp:

- /dev/ttyS0
- /usr/reajet/dev/ttyS0

Die Konfiguration der seriellen Schnittstelle kann wie in den folgenden Unterkapiteln beschrieben aussehen:

4.1.1 9600/8/N/1

```
stty -F /usr/reajet/dev/ttyS0 9600 intr 00 quit 00 erase 00 kill 00 ixany  
-parenb -parodd cread -icanon min 0 time 0
```

4.1.2 19200/8/E/1

```
stty -F /usr/reajet/dev/ttyS0 19200 intr 00 quit 00 erase 00 kill 00  
ixany parenb -parodd cread -icanon min 0 time 0
```

4.1.3 115200/8/O/1

```
stty -F /usr/reajet/dev/ttyS0 115200 intr 00 quit 00 erase 00 kill 00  
ixany parenb parodd cread -icanon min 0 time 0
```



5 Notizen