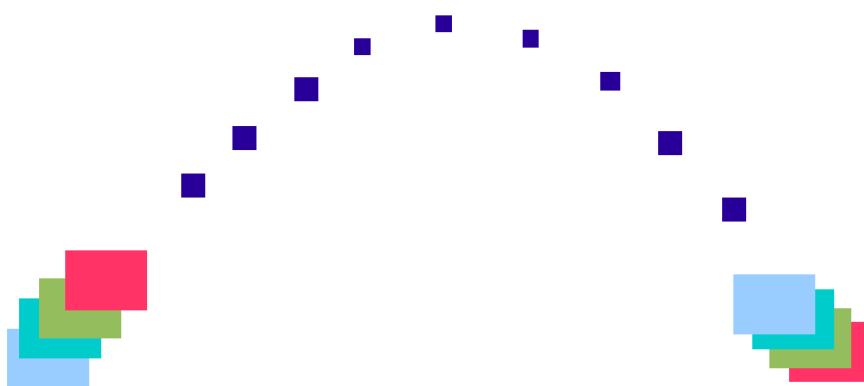


---

# Z M C S

## ZMODEM File Server and Client

DOS, Linux, Windows



## Version 5

[www.adontec.com](http://www.adontec.com)

## LIZENZVEREINBARUNG

ZMODEM FILE SERVER and CLIENT - ZMCS (Software)

© Urheberrechtlich geschütztes Material - Alle Rechte vorbehalten.

Mit der Installation oder Nutzung der SOFTWARE erklären Sie sich mit folgenden Lizenzbestimmungen ausnahmslos und ohne Einschränkung einverstanden. Wenn Sie mit den Lizenzbestimmungen nicht vollständig einverstanden sind, geben Sie die unbenutzte Software sofort an den Verkäufer zurück. Beim Erwerb der SOFTWARE durch Electronic Software Delivery (ESD) z.B. Download oder eMail und bei entsiegelten Datenmedien (z.B. CD, DVD) besteht kein Rückgaberecht.

### LIZENZ

Die Software wird nicht verkauft, sondern lizenziert. Die Software ist geistiges Eigentum und unterliegt somit dem Urheberrecht. Der Lizenzgeber, ADONTEC Computer Systems GmbH ("ADONTEC"), räumt Ihnen als Lizenznehmer ein nicht ausschließliches, eingeschränktes Nutzungsrecht (Lizenz) für die SOFTWARE ein.

### NUTZUNGSUMFANG

Jede Vervielfältigung der SOFTWARE, wird strafrechtlich verfolgt. Nur eine Kopie darf zu Sicherungszwecken angefertigt werden. Entsprechend der Unmöglichkeit, daß ein Buch zugleich an verschiedenen Orten von mehreren Personen gelesen wird, darf die Software nicht gleichzeitig von verschiedenen Personen an verschiedenen Orten und auch nicht auf verschiedenen Geräten benutzt werden.

Jede Vervielfältigung des Handbuches oder der Software, wird strafrechtlich verfolgt. Alle auf den Originaldatenträger der SOFTWARE enthaltenen Dateien, dürfen nicht als Dateien in der Originalform oder in einer abgeänderten Form weiterverkauft werden.

Es ist nicht zulässig die SOFTWARE oder Teile der SOFTWARE zu dekompile, modifizieren, übertragen oder zu disassemblieren. Die zugrundeliegenden Algorithmen und Programmierungen sind und bleiben geschütztes geistiges Eigentum der Entwickler.

Jede Lizenz ermöglicht den Einsatz und die Nutzung der Software auf einen PC. Mehrere Lizenzen ermöglichen die Nutzung auf genauso viele PC, wie die Anzahl der unterschiedlichen Seriennummern die Sie besitzen (Mehrfachlizenz). Bei Mehrfachlizenzen müssen Sie einen Mechanismus zur Verwaltung und Prüfung der installierten Kopien nutzen.

Eine Mehrfachlizenz berechtigt Sie in keinem Fall die Lizenz aufzuteilen und die Teile zu verkaufen.

### VERLEIH, VERKAUF

Verleih oder Unterlizenzierung ist nicht gestattet.

Die permanente und komplette Übertragung der SOFTWARE an einem Dritten ist erlaubt sofern der Dritte mit diesen Lizenzbestimmungen einverstanden ist, keine Kopien zurückbehalten worden sind und ADONTEC schriftlich informiert worden ist.

### DAUER DER LIZENZ

Das Nutzungsrecht an der SOFTWARE beginnt nach Zahlung der Lizenzgebühren. Sie können die Lizenz jederzeit beenden, indem Sie die SOFTWARE zusammen mit allen eventuell angelegten Kopien zerstören. Diese Lizenz endet automatisch, sobald Sie gegen eine Bestimmung dieses Lizenzvertrags verstoßen.

### EINSCHRÄNKUNG DER GEWÄHRLEISTUNG

Wir werden immer bemüht sein, Ihnen einwandfreie Software zu liefern. Wir weisen jedoch darauf hin, dass wir bei dem gegenwärtigen Stand der Technik keine Gewähr dafür übernehmen, dass das Softwareprogramm in allen Kombinationen und Anwendungen unterbrechungs- und fehlerfrei arbeitet. Es wird keine Garantie für die Richtigkeit des Inhaltes dieses Handbuches übernommen, da sich Fehler trotz aller Bemühungen, nie vollständig vermeiden lassen. Hinweise und Verbesserungsvorschläge nehmen wir jederzeit gerne entgegen.

Die Haftung für unmittelbare Schäden, mittelbare Schäden, Folgeschäden und Drittschäden ist, soweit gesetzlich zulässig, ausgeschlossen. Die Haftung bei grober Fahrlässigkeit und Vorsatz bleibt hiervon unberührt, in jedem Fall ist jedoch die Haftung beschränkt auf den Kaufpreis.

## **Support und Registrierung**

Vielen Dank für den Kauf und Nutzung der Software. Durch die Registrierung der Software erhalten Sie kostenlose technische Unterstützung und günstige Updates.

Bitte nutzen Sie die Formulare unter <http://www.adontec.com> für die Registrierung und Ihre Anfragen.

## Inhaltsverzeichnis

LIZENZVEREINBARUNG.....	2
Support und Registrierung.....	3
ZMODEM File Server and Client (ZMCS).....	5
Syntax.....	6
Beispiele zur Anwendung (Seriell).....	7
Beispiele zur Anwendung (ISDN).....	8
Beispiele zur Anwendung (TCP/IP).....	9
Kommandozeilen Parameter.....	10
Der Parameter /C.....	10
Parameter /E.....	10
Der Parameter /F.....	10
Der Parameter /F@ - Auftragsdatei (Job).....	10
Der Parameter /I.....	11
Der Parameter /L.....	11
Der Parameter /M.....	11
Der Parameter /T.....	11
Der Parameter /W.....	12
Der Parameter /X.....	12
Parameter /XI.....	12
Der Parameter /XL.....	13
Manuelle Zugangsüberprüfung.....	13
Der Parameter /XP.....	13
Standardwerte.....	15
Beenden des Servers.....	15
Auftragsdatei (Job-File).....	16
Kommandos und Syntax.....	17
Skript-Kommandos.....	17
Skript - Beispiele.....	26
Rückgabewerte:.....	28
Fehler.....	28
Protokoll Fehlerwerte.....	29
Fragen und Antworten.....	30
Beispiel Sitzung.....	33
Server.....	33
Client.....	34
Beispiel Sitzung 2.....	36
Server.....	36
Client.....	37
Konfiguration.....	38
Benutzung eines Modem.....	38
Leitungssignale.....	38
Black list.....	38
Wahlwiederholung.....	38
Direkte Verbindung.....	38
Serielles Kabel.....	38
Installation.....	39
Windows.....	39
Linux.....	39
DOS.....	39
Lizenzierung.....	39
Installation und Lizenzierung .....	40

## ZMODEM File Server und Client (ZMCS)

### *Was leistet diese Software ?*

ZMCS (ZMODEM Client/Server) ist ein Konsolen-Programm (ohne grafische Oberfläche), daß Dateien per ZMODEM Protokoll oder KERMIT Protokoll überträgt.

ZMCS kann Verbindungen\* über ein *Modem* aufbauen oder eine *direkte Verbindung* über ein serielles *Null-Modem* Kabel oder über ISDN oder TCP/IP und darüber Dateien übertragen.

Das Programm kann als Server oder als Klient arbeiten. Der Arbeitsmodus von ZMCS wird per Kommandozeilenparameter angegeben. Im *Server Modus* wartet es bis ein Klient eine Verbindung aufbaut und Dateien anfordert bzw. ablegt. Im *Klienten Modus* meldet es sich an den Server und sendet oder holt Dateien ab.

Im *Klienten Modus* überträgt das Programm unbeaufsichtigt alle angegebene Dateien und endet selbständig. Im *Server Modus* nimmt es eingehende Verbindungen entgegen, empfängt oder sendet Dateien bis es durch eine Anwendung oder dem Benutzer beendet wird.

ZMCS ist keine Insel Lösung. Sowohl der Klient als auch der Server können nicht nur untereinander sondern auch mit fremden Anwendungen kommunizieren (z.B. Hyperterminal). Der Zugriff ist also nicht zwischen der ZMCS Anwendung beschränkt sondern auch fremde Software können den ZMCS Server anwählen und der ZMCS Klient kann einfach an fremde Server verbinden soweit kein spezielle Spezifikation notwendig ist (z.B. spezielle Anmelde Prozedur).

Bestehende Anwendungen können leicht um eine Dateiübertragungsfunktionalität erweitert werden.

### *Wie starte ich es am besten ?*

ZMCS, kann durch eine Anwendung als externes Programm gestartet werden, über eine Batch-Datei, ein Skript, Windows Autostart-Gruppe, per Verknüpfung u.a. und dabei mit oder ohne ein sichtbares Fenster.

ZMCS arbeitet komplett im Hintergrund und fällt nicht auf. Es kann per Tastendruck ('E', ESC) oder aus einer Anwendung heraus beendet werden.

Die Software kann mit oder ohne Text- und Status-Ausgaben arbeiten. Alternativ können die Ausgaben in eine Log-Datei umgeleitet oder ganz ausgeschaltet werden (Quiet Modus). Dadurch kann es auch unbemerkt, ohne eigenes Fenster, im Hintergrund arbeiten. Eine Low Level Datenaufzeichnung wird ebenfalls unterstützt.

*\*unterschiedliche Anwendung ( Lizenz) unterstützt Seriell oder ISDN oder TCP/IP.*

## Welche Befehle und Optionen bietet es ?

Die ZMCS Software wird per Kommandozeilen-Parameter gesteuert:

## Syntax

```
ZMCS.EXE /M<r|t|s|ux>[H] [/Cnr] [/Tphone] [/Ffilename|/F@filelistname]
[I[r]] [/X[m|c]] [/XP[Z[R|r|..]|K]] /XL[A][U=username,][P=password] [XI[p|c]]
```

## SERIAL

```
ZMCS.EXE /M<r|t|s|ux>[H] [/Cnr] [/Tphone] [/Ffilename|/F@filelistname]
[I[r]] [/X[m|c]] [/XP[Z[R|r|..]|K]] [/XL[A][U=username,P=password] [XI[p|c]]
```

## TCP/IP

```
ZMCS.EXE /M<r|t|s|ux>[H] [/Cnr] [/Tip:port] | [/Eip:port]
[/Ffilename|/F@filelistname] [I[r]] [/X[m|c]] [/XP[Z[R|r|..]|K]] [/XL[A]
[U=username,P=password] [XI[p|c]]
```

## ISDN

```
ZMCS.EXE /M<r|t|s|ux>[H] [/Cnr] [/Tphone] | [/Emsn]
[/Ffilename|/F@filelistname] [I[r]] [/X[m|c]] [/XP[Z[R|r|..]|K]] [/XL[A]
[U=username,P=password] [XI[p|c]]
```

**/M** definiert den Ausführungsmodus, r=receiver, t=transmitter, s=server, u=Server auf Schnittstelle COMx beenden, H=Kein Handshake\* (ermöglicht fremde Anwendungen zu verbinden)

**/C** definiert die serielle Schnittstelle, nr=[1,2,...]

**/C** definiert eine Kommunikationsnummer, nr=[1,2,...]

**/C** definiert eine Kommunikationsnummer, nr=[1,2,...]

**/T** definiert die Telefonnummer für die abgehende Verbindung (Klient)

**/T** definiert die IP und Portadresse für die abgehende Verbindung (Klient)

**/E** definiert die IP und Portadresse auf die der Server reagiert

**/E** definiert die MSN (Rufnummer) auf die der Server Verbindungen beantwortet

**/F** definiert den Dateinamen für die Dateiübertragung

z.B. data.txt, acc\*.txt, \*.doc, \*.A??

**/F@** definiert die Auftragsdatei (Job)

**/I[r]** ignoriere Signale u.a. Konditionen

**/X[C|M]** definiert den Kopiermodus c=copy, m=move

Bei 'm' wird die Datei, nach erfolgreicher Übertragung, gelöscht.

**/XP[Z[R|r|..]|K]** definiert das Protokoll und Optionen (Z=ZMODEM, K=KERMIT)

**/XL[A][U=username,][P=password]** aktiviert die Anmeldeprozedur

**/XI[p|c]** definiert erweiterte Informations Schalter

**/L** stelle eine direkte Verbindung (ohne Modem) her

**/Q** Nachrichten werden unterdrückt (Quiet-Mode)

**/W[Verzeichnis]** Definiert das Arbeitsverzeichnis

weitere Parameter

**/Bbaudrate /Ddatabits /Sstopbits /Pparity**

baudrate=[50-115200] databits=[5-8] stopbits=[1|2] parity=[N|O|E|M|S]

[ ] = optional, | = einer der verfügbaren Werte

\* Diese Option ist verfügbar damit fremde Programme mit dem ZMCS Server kommunizieren können. Bitte beachten Sie dass in diesem Fall der Funktionsumfang auf die Dateiübertragung beschränkt ist.

### **Beispiele zur Anwendung (Seriell)**

#### **Beispiel 1:**

1. Server auf COM1 (ttyS0) starten: ZMCS /C1 /Ms
  2. Klient, senden (copy): ZMCS /C2 /Mt /T123456789 /Fbestellung.dat
  3. Klient, empfangen (copy): ZMCS /C2 /Mr /T123456789 /Fverkauf.dat
  4. Klient, empfangen (copy): ZMCS /C2 /Mr /T123456789 /Fanfrage.dat
2. - 4. Kann auch mittels Auftragsdatei innerhalb einer Verbindung erledigt werden z.B.
2. Klient, Auftragsdatei (copy): ZMCS /C2 /Mt /T123456789 /F@job001.txt

#### **Beispiel 2:**

1. Server auf COM3 (ttyS2) starten: ZMCS /C3 /Ms /L  
Dieser Server nutzt die COM3 (ttyS2) und erwartet eine Direktverbindung über das Kabel. Ein Modem ist nicht angeschlossen.
2. Klient, senden (copy): ZMCS /C2 /Mt /L /Fmessung1239.mdb
3. Beenden des Servers auf COM3 (ttyS2): ZMCS /Mu3

#### **Beispiel 3:**

1. Server auf COM3 (ttyS2) starten: ZMCS /C3 /Ms /XL  
Dieser Server erwartet eine Verbindung auf COM3 (ttyS2) und erwartet eine Benutzeranmeldung mit *Benutzername* und *Passwort*.
2. Klient, senden (move): ZMCS /C2 /Mt /T123456789 /Xm  
/XLU=dorian,P=secret /F\*.pdf  
Der Klient identifiziert sich mit dorian,secret.
3. Beenden des Servers auf COM3 (ttyS2) : ZMCS /Mu3

#### **Beispiel 4:**

1. Server auf COM1 (ttyS0) starten: ZMCS /C1 /MsH /XPZR  
Dieser Server erlaubt fremde Anwendungen, ohne den speziellen ZMCS Verbindungsdialog, eine Verbindung aufzubauen.  
ZMODEM "crash recovery" eingeschaltet.
2. Als Klient kann ein fremdes Programm (z.B. Hyperterminal) genutzt werden und Dateien per ZMODEM zum Server senden.

---

\* Erste serielle /C1 ist COM1 unter DOS und Windows bzw. ttyS0 unter Linux.

**Beispiele zur Anwendung (ISDN)****Beispiel 1:**

1. Server auf Kommunikationsnummer 1 starten: ZMCS /C1 /Ms /E90020  
Dieser Server beobachtet die Rufnummer 90020.
2. Klient, senden (copy): ZMCS /C2 /Mt /T90020 /Fbestellung.dat
3. Klient, empfangen (copy): ZMCS /C2 /Mr / T90020 /Fverkauf.dat

**Beispiel 2:**

1. Beenden des Servers auf Kommunikationsnummer 1: ZMCS /Mu1

**Beispiel 3:**

1. Server auf auf Kommunikationsnummer 1 starten: ZMCS /C1 /Ms /E90020 /XL  
Dieser Server beobachtet die Rufnummer 90020  
und erwartet eine Benutzeranmeldung mit *Benutzername* und *Passwort*.
2. Klient, senden (move): ZMCS /C2 /Mt /T90020 /Xm  
/XLU=dorian,P=secret /F\*.pdf  
Der Klient identifiziert sich mit dorian,secret.
3. Beenden des Servers auf Kommunikationsnummer 1: ZMCS /Mu1

**Beispiel 4:**

1. Server auf Kommunikationsnummer 1 starten: ZMCS /C1 /MsH /XPZR  
Dieser Server erlaubt fremde Anwendungen, ohne den speziellen  
ZMCS Verbindungsdialog, eine Verbindung aufzubauen.  
ZMODEM "crash recovery" eingeschaltet.  
Alle Rufnummer (MSN) werden angenommen.
2. Als Klient kann ein fremdes Programm genutzt werden  
und Dateien per ZMODEM zum Server senden.



**Beispiele zur Anwendung (TCP/IP)****Beispiel 1:**

1. Server auf Kommunikationsnummer 1 starten:

```
ZMCS /C1 /Ms /Ewww.mysite.com:9100
```

Dieser Server erwartet Verbindungen auf „www.mysite.com:9100“.

2. Klient, senden (copy): ZMCS /C2 /Mt /Twww.mysite.com:9100 /Fbestellung.dat
3. Klient, empfangen (copy): ZMCS /C2 /Mr /Twww.mysite.com:9100 /Fverkauf.dat

**Beispiel 2:**

1. Beenden des Servers auf Kommunikationsnummer 1: ZMCS /Mu1

**Beispiel 3:**

1. Server auf auf Kommunikationsnummer 1 starten: ZMCS /C1 /Ms /Ehost:9100 /XL  
Dieser Server erwartet Verbindungen auf „host:9100“  
inkl. Benutzeranmeldung (*Benutzername* und *Passwort*).
2. Klient, senden (move): ZMCS /C2 /Mt /Thost:9100 /Xm  
/XLU=dorian,P=secret /F\*.pdf  
Der Klient identifiziert sich mit dorian,secret.
3. Beenden des Servers auf Kommunikationsnummer 1: ZMCS /Mu1

**Beispiel 4:**

1. Server auf Kommunikationsnummer 1 starten: ZMCS /C1 /MsH /XPZR  
Dieser Server erwartet Verbindungen auf „host:9000“.  
Dieser Server erlaubt fremde Anwendungen, ohne den speziellen  
ZMCS Verbindungsdialog, eine Verbindung aufzubauen.  
ZMODEM "crash recovery" eingeschaltet.
2. Als Klient kann ein fremdes Programm genutzt werden  
und Dateien per ZMODEM zum Server senden.

## **Kommandozeilen Parameter**

Viele der Parameter sind selbsterklärend. Die wichtigsten werden hier detaillierter vorgestellt.

### **Der Parameter /C**

Definiert die Schnittstellennummer (1=erste serielle Schnittstelle z.B. COM1 bzw. ttyS0, 2=zweite serielle Schnittstelle z.B. COM2 bzw. ttyS1, etc.) z.B. /C2.

Bei ISDN und TCP/IP definiert, dieser Parameter eine Kommunikationsnummer (1 bis 255), die z.B. dazu verwendet werden kann um den Server zu beenden.

### **Parameter /E**

Der Parameter /E definiert zusätzliche Konfiguration für den Server:

Seriell: Ohne Funktion.

ISDN: Die Rufnummer (MSN) der Server beachten soll. Wenn keine angegeben wird werden alle MSN\* beachtet.

Beispiel /e90020

TCP/IP: Die IP und Port Adresse über die der Server Verbindungen erwartet.

Beispiel /e192.168.2:9000

Durch setzen von IP auf „host“ wird die PC eigene IP Adresse verwendet.

Beispiel /ehost:9000

Wenn nichts angegeben wird, wird „host:9000“ verwendet.

Hinweis: ISDN und TCP/IP zur Zeit nur unter Windows.

\*MSN = Multiple Subscriber Number.

### **Der Parameter /F**

Der Parameter /F kann außer Dateinamen auch eine Pfadangabe enthalten z.B. /F:\data\db01.txt. Ein Dateiname kann auch so genannte Wildcards (\*,?) enthalten.

Wenn im Dateinamen Leerzeichen existieren muss es in Ausführungszeichen gestellt werden z.B. /F"d:\temp\My Files\\*.txt".

Leerzeichen in Parameter werden in der DOS Version von ZMCS.EXE nicht unterstützt.

### **Der Parameter /F@ - Auftragsdatei (Job)**

Eine *Auftragsdatei* ist ähnlich wie ein *Skript* und definiert verschiedene Aufgaben, die der Klient während seiner Verbindung ausführt. Mittels einer *Auftragsdatei* ist es möglich eine Liste von Dateien zu definieren, welche innerhalb einer Verbindung (Sitzung) übertragen werden. Diese Dateiliste wird in eine Textdatei gespeichert. Mittels dieser *Auftragsdatei* ist es möglich, innerhalb einer Verbindung, sowohl Dateien zu senden als auch zu empfangen.

Die *Auftragsdatei* wird über den Parameter /F@ angegeben. Das Zeichen '@' dient dabei zur Unterscheidung zwischen einer "normalen" Datei und einer *Auftragsdatei*.

z.B. /F@"d:\temp\My Files\Auftragsdateil.txt"

Ohne Pfadangabe, wird die *Auftragsdatei* im ZMCS.EXE Verzeichnis gesucht.

Die *Auftragsdatei* besteht aus einzelnen Zeilen mit jeweils einer Anweisung oder einem Dateinamen. Die *Auftragsdatei* unterstützt zur Zeit die Sektionen "TX" und "RX". Wird keine Sektion angegeben wird "TX" angenommen und die darauf folgende Dateien werden gesendet. "RX" markiert die darauf folgende Dateien für den Empfang.

```

Beispiel Auftrag: job001.txt
# Verkaeufer 001
#RX
verkauf.dat
anfrage.dat
#TX
bestellung.dat

```

Eine *Auftragsdatei* ist eine Textdatei und kann einfach mit jedem Editor erstellt werden. Jede Anweisung oder Dateiname muss in der ersten Spalte beginnen. Eine Anweisung beginnt mit '#'. Ein Dateiname wird ohne besondere Kennzeichnung eingetragen. Kommentare beginnen mit '#' gefolgt von einem Leerzeichen.

Nur Klienten verarbeiten *Auftragsdateien*. Der Klient kann dabei mit /Mr oder /Mt gestartet werden. Der Zusatzparameter **H** (z.B. /Ms**H**) sollte hier nicht genutzt werden, da ohne ZMCS *Synchronisationsprotokoll* die Dateiliste nicht oder nur Teilweise abgearbeitet werden kann (z.B. können keine Dateien angefordert werden, jedoch empfangen, wenn die Gegenstation diese automatisch sendet).

### **Der Parameter /I**

Ignoriere Signale u.a. Konditionen. Zurzeit wird die Option 'r' unterstützt um unerwünschte 'Ringring' Signale vom Modem zu ignorieren (z.B. /**Ir**).

### **Der Parameter /L**

Signalisiert eine lokale Verbindung über serielles Kabel ohne Modem.  
Wird bei ISDN und TCP/IP ignoriert.

### **Der Parameter /M**

Definiert den Ausführungsmodus.

```

/Mr   definiert einen Klienten der versucht Dateien vom Server anzufordern.
/Mt   definiert einen Klienten als Sender um Dateien zum Server zu übertragen.
/Ms   definiert einen Server.
/Mux  beendet den Server auf der Kommunikationsnummer x, die per Parameter /C
      definiert wurde

```

Der Zusatzparameter **H** z.B. /Ms**H** schaltet das ZMCS *Synchronisationsprotokoll* (Handshake) aus. Dies ermöglicht auch fremde Anwendungen Dateien an den Server zu übertragen.

### **Der Parameter /T**

Serial & ISDN:

Definiert die Telefonnummer der Gegenstelle mit der Verbunden werden soll.

Beispiel /T0704390020.

**TCP/IP:**

Definiert die IP und Port Adresse des Servers mit dem Verbunden werden soll.

Beispiel /Twww.mysite.com:9000

Sobald IP auf „host“ gesetzt wird, wird die PC eigene IP Adresse verwendet.

Beispiel /Thost:9000

Bei fehlender Port Adresse, wird 9000 verwendet.

Hinweis: ISDN und TCP/IP zur Zeit nur unter Windows.

**Der Parameter /W**

ZMCS kann auf ein Arbeitsverzeichnis eingeschränkt werden (z.B. /Wd:\data). Die empfangenen Dateien werden nur hier abgelegt. Dateien können nur von hier aus versendet oder abgeholt werden. Andere Verzeichnisse werden somit geschützt. Jede relative Pfadangabe im Parameter /F wird in diesem Arbeitsverzeichnis aufgelöst.

Es wird das lokale Verzeichnis von ZMCS.exe verwendet, wenn kein Arbeitsverzeichnis definiert wird.

*Ein angegebenes Verzeichnis muss existieren und das Dateiaattribut "nur lesen" darf nicht gesetzt sein, da sonst keine neuen Dateien erzeugt werden können. ZMCS erzeugt keine Verzeichnisse. Wenn das angegebene Arbeitsverzeichnis Leerzeichen enthält sollte es in Ausführungszeichen gestellt werden z.B. /W"d:\My Data\My Files".*

Leerzeichen in Parameter werden in der DOS Version von ZMCS.EXE nicht unterstützt.

**Der Parameter /X**

Der Parameter /X in Verbindung mit der Option C oder M kontrolliert den Kopier-Modus.

Mit /Xm werden die original Dateien verschoben. Das bedeutet, dass nach erfolgreicher Übertragung die original Dateien **gelöscht** werden ("Move" Befehl).

Mit /Xc werden die original Dateien kopiert. Das bedeutet, dass auf beiden Seiten eine Kopie jeder Datei existiert ("Copy" Befehl).

**Parameter /XI**

Definiert zusätzliche Optionen für die Informations-Ausgabe. Aktuell definiert die Option P die prozentuelle Ausgabe der Fortschrittsanzeige z.B. [.....], wobei jeder Punkt (1 bis 10) ca. 10% repräsentiert. Aktivierung mit /XIp.

In der Standardeinstellung (ohne XIp) erfolgt die Anzeige der übermittelten Datenblöcken.

Eine weitere Option C aktiviert die Interprocess Kommunikation (IPC) über Mailslots. Für die IPC über Mailslots nutzt ZMCS den Slot „\.\mailslot\zmcs\_c\_xx“ wobei xx=01 für COM1, 02 für COM2 etc.). Ein kleines IPC Beispiel in C/C++ ist erhältlich. Aktivierung mit /XIc.

Um mehrere Optionen zu aktivieren einfach die entsprechende Option an /XI anhängen z.B. /XIpc.

### **Der Parameter /XL**

Der Parameter */XL* aktiviert die Anmeldeprozedur beim Server. Dadurch muss sich jeder Klient anmelden und die *Benutzernamen* und *Passwort* Prüfung bestehen. Hierdurch kann, bei Bedarf, für jeden Benutzer auch ein eigenes Arbeitsverzeichnis definiert werden. Zusätzlich, allerdings optional, kann jeder Zugriff auch durch den Computer Nutzer bestätigt werden (*/XLA*).

Server: */XL*

Aktiviert die Anmeldeprozedur beim Server und lädt die Benutzerdaten aus der Datei *ZMCS\_usr.ini*. Die Datei *ZMCS\_usr.ini* wird im Server Verzeichnis gesucht.

#### **Beispiel *ZMCS\_usr.ini*:**

[dorian]

PASS=secret01

WORKDIR=dorian\_data

Für jeden Benutzer sollte eine Sektion erzeugt werden mit den Schlüsseln 'PASS' und optional 'WORKDIR'. Der Schlüssel 'WORKDIR' definiert das benutzerspezifische Arbeitsverzeichnis. Wenn 'WORKDIR' nicht definiert ist, wird das Arbeitsverzeichnis aus dem Parameter */W* benutzt.

Das obige Beispiel definiert den Benutzer 'dorian' mit Passwort 'secret01' und das relative Arbeitsverzeichnis 'dorian\_data'. Ein Arbeitsverzeichnis ohne absolute Pfadangabe ist immer relativ zum Verzeichnis von ZMCS.EXE.

Der Parameter */XLU=benutzername,P=passwort* aktiviert die Anmeldeprozedur beim Klienten.

Klient: */XLU=dorian,P=secret01*

Der Klient wird versuchen sich mit Benutzernamen 'dorian' und Passwort 'secret01' anzumelden.

### **Manuelle Zugangsüberprüfung**

Jede Klienten Anmeldung kann per *Benutzernamen* und *Passwort* auf Gültigkeit überprüft werden. Darüber hinaus kann jede Klienten Anmeldung auch vom Computer Benutzer einzeln geprüft und bestätigt werden. Diese *Manuelle Zugangsüberprüfung* muss dem Server mit dem Parameter */XLA* mitgeteilt werden. Danach öffnet der Server, bei jedem Anmeldeversuch eines Klienten, einen Bestätigungsdialog. Dieser Dialog sollte innerhalb von ca. 8 Sekunden bestätigt werden, damit die Anmeldeprozedur noch erfolgreich durchgeführt werden kann.

### **Der Parameter /XP**

Der Parameter */XPK* aktiviert das *KERMIT Protokoll*. Der Parameter */XPZ* aktiviert das *ZMODEM Protokoll* (die Standardeinstellung).

Wenn ZMCS auf beiden Seiten genutzt wird, dann bewirkt die Änderung des Protokolls auf der einen Seite die automatische Einstellung auf der anderen Seite. Das funktioniert solange das *ZMCS Synchronisationsprotokoll* aktiv ist (siehe Schalter */MH*).

Ist der Kommunikations-Partner für ZMCS eine fremde Software ist, dann sollte das *ZMCS Synchronisationsprotokoll* abgeschaltet werden (*/MH*), da es unwahrscheinlich ist, dass die fremde Software dieses versteht.

Der Parameter */XPZ* unterstützt auch die optionalen ZMODEM Datei-Optionen */XPZ[<R|r><S><N|O|P|X><->]*. Datei-Optionen sind optionale Anweisungen, vom Sender an den Empfänger, und beschreiben Bedingungen für den Dateiempfang. Wenn der Sender keine Datei-Optionen überträgt, nutzt der Empfänger die eigenen, soweit welche definiert wurden.

Aktuell werden folgende Datei-Optionen unterstützt:

R	Aktiviert die Wiederaufnahme einer unterbrochenen Dateiübertragung.
r	Deaktiviert die Wiederaufnahme einer unterbrochenen Dateiübertragung.
S	Skip if absent: Der Empfänger akzeptiert die Datei, wenn diese bereits existiert. Sonst wird sie nicht übertragen.
N	Der Empfänger akzeptiert die Datei, wenn diese neuer oder länger sind.
O	Overwrite: Die Datei wird immer übertragen (=refresh).
P	Protect: Vorhandene Datei schützen sonst neu übertragen.
X	Extend: Daten an bestehende Datei anhängen sonst neu übertragen.
-	Ignoriere die empfangene Datei-Optionen und nutze die eigenen.

Nur eine Option aus jeder Gruppe kann angegeben werden!

Wenn nichts angegeben, nutzt das ZMODEM Protokoll die **Voreinstellung /XPZr**. Dadurch wird auch eine unterbrochene Dateiübertragung neu übertragen, und Dateien werden übertragen, wenn neu oder neuer (nur das Datum wird überprüft nicht die Länge).

Die optionalen ZMODEM Datei-Optionen funktionieren unabhängig von der Einstellung /MH.

## Beispiele

ZMODEM ohne „crash recovery“.

Auf der Empfängerseite **/XPZr-** oder auf der Senderseite **/XPZr**

ZMODEM mit „crash recovery“.

Auf der Empfängerseite **/XPZR-** oder auf der Senderseite **/XPZR**

ZMODEM ohne „crash recovery“ und vorhandene Datei nicht Überschreiben.

Auf der Empfängerseite **/XPZP-** oder auf der Senderseite **/XPZP**

ZMODEM ohne „crash recovery“ und nur wenn vorhanden und neuer/länger.

Auf der Empfängerseite **/XPZS-** oder auf der Senderseite **/XPZS**

ZMODEM ohne „crash recovery“ und Daten anhängen.

Auf der Empfängerseite **/XPZX-** oder auf der Senderseite **/XPZX**

Damit die Datei auf Empfängerseite, nur mit neuen Daten erweitert wird, sollte die neue Datei nur die neuen Daten enthalten und ein neueres Datum.

ZMODEM ohne „crash recovery“ und Dateien immer übertragen.

Auf der Empfängerseite **/XPZO-** oder auf der Senderseite **/XPZO**  
oder auf der Serverseite **/XPZO-** (auf beiden Seiten aktiv da '-')

ZMODEM mit „crash recovery“ und Dateien nur dann übertragen, wenn auch beim Empfänger vorhanden.

Auf der Empfängerseite **/XPZRSN** oder auf der Senderseite **/XPZRSN**  
oder auf der Serverseite **/XPZRSN-** (auf beiden Seiten aktiv da '-')

## Hinweise

Das *KERMIT Protokoll* und die ZMODEM Datei-Optionen sind nur in der Windows Version verfügbar.

Nicht jeder Klient überträgt die Dateiinformation korrekt. Deshalb sollte /XPZr nur bei geprüften Klienten genutzt werden!

## **Standardwerte**

Wenn über die Parameter nicht anders angegeben gelten folgende Standardwerte:

### Seriell

`/C2 /B38400 /D8 /S1 /Pn /Xc /XPZr` (COM2, 38400,8,1,N, copy files, ZMODEM)

### ISDN

`/C2 /Pn /Xc /XPZr` (alle MSN, copy files, ZMODEM)

### TCP/IP

`/C2 /Pn /Xc /XPZr` („host:9000“, copy files, ZMODEM)

## **Beenden des Servers**

Das Programm kann vom Benutzer mittels Taste 'E' oder ESC beendet werden. Von einer Anwendung heraus, wenn es nochmals per Parameter **/Mux** (x=1, 2, ... COM1, COM2, ...) aufgerufen wird:

Zum Beispiel, beenden des Servers auf COM3: `ZMCS /Mu3`

## **Auftragsdatei (Job-File)**

Eine *Auftragsdatei* ist ähnlich wie ein *Skript* und definiert verschiedene Aufgaben, die der Klient während einer Sitzung / Verbindung ausführt. Die *Auftragsdatei* wird als eine reine Text-Datei im ANSI (8-Bit) Format erwartet.

Die *Auftragsdatei* kann Kommandos für den Versand von Daten und Dateien enthalten, Verzweigungen, logische Abläufe steuern, Verbindungen aufbauen, jederzeit manuelle Anmeldung oder Protokoll-Umschaltung durchführen, Schleifen und Variablen versorgen u.v.a.m.

Die *Auftragsdatei* wird über den Parameter */F@* angegeben. Das Zeichen '@' dient dabei zur Unterscheidung zwischen einer "normalen" Datei und einer *Auftragsdatei*.

z.B. `/F@"d:\temp\My Files\Auftragsdatei.txt"`

Ohne Pfadangabe, wird die *Auftragsdatei* im ZMCS.EXE Verzeichnis gesucht.

Die *Auftragsdatei* besteht aus einzelnen Zeilen mit jeweils einer Anweisung oder einem Dateinamen. Die *Auftragsdatei* unterstützt zur Zeit die Sektionen "TX" und "RX". Wird keine Sektion angegeben wird "TX" angenommen und die darauf folgende Dateien werden gesendet. "RX" markiert die darauf folgende Dateien für den Abruf bzw. Empfang\*.

\* Wenn der optionale Parameter **H** (z.B. */MsH*) gesetzt wird, kann der Klient Dateien empfangen, die die Gegenstation automatisch sendet, es kann jedoch keine bestimmte Dateien anfordern.

### Beispiel Auftrag: job001.txt

```
# Verkaeufer 001
#RX
verkauf.dat
anfrage.dat
#TX
bestellung.dat
```

Eine *Auftragsdatei* ist eine Textdatei und kann einfach mit jedem Editor erstellt werden. Jede Anweisung oder Dateiname muss in der ersten Spalte beginnen. Eine Anweisung beginnt mit '#'. Ein Dateiname wird ohne besondere Kennzeichnung eingetragen. Kommentare beginnen mit '#' gefolgt von einem Leerzeichen.

Eine erweiterte Form mit manueller Anmeldeprozedur wie sie es der ZMCS Server erwartet:

### Beispiel Auftrag: job002.txt

```
# - Manual Login -
#SendStr <13>
#TIMEOUT=3000
#WaitForStr "Username:" # warte auf Frage
#OnFailure L_END
#SendStr dorian<13> # sende Benutzernamen
#
#WaitForStr Password: # warte auf nächste Frage
#OnFailure L_END
#SendStr secret01<13> # sende Passwort
#WaitForStr Welcome # warte auf Begrüßung
#OnFailure L_END
#
# - Verkaeufer 001 -
#RX
verkauf.dat
anfrage.dat
#TX
bestellung.dat
#L_END
```



## **Kommandos und Syntax**

Die einzelnen Kommandos müssen am Anfang stehen und mit dem Zeichen '#' beginnen z.B.  
`#SendStr Hello World`

Beim Kommando wird nicht zwischen Klein- und Großschreibung unterschieden. Wenn ein Kommando Parameter erwartet, werden diese unmittelbar nach einem Leerzeichen oder Gleichheitszeichen erwartet.

Ein Text- bzw. Zeichenparameter kann auch in Anführungszeichen '"' stehen z.B.

```
#WaitForStr "Username:"
```

Anführungszeichen erleichtern die Zeichenkette zu begrenzen, falls der Editor das Zeilenende nicht anzeigt, und um Leerzeichen am Ende der Zeichenkette zu schützen.

Um binäre Daten (also nicht druckbare Zeichen) zu übertragen (z.B. ASCII 1 bis 31 und 125 bis 255) können diese innerhalb der Zeichen '<>' gesetzt werden z.B.

```
#SendStr "Hello World<13>"
```

Werden die Zeichen '<' und '>' als solche benötigt müssen sie verdoppelt werden z.B.

```
#SendStr "Hello World>>"
```

wird die Zeichenkette "Hello World>" an die Gegenstelle übertragen

## **Skript-Kommandos**

Die Skript Kommandos bilden eine kleine Programmiersprache mit einfacher Syntax. Folgende Script-Elemente und Kommandos werden aktuell unterstützt:

```
#.
```

Eine Kommentarzeile beginnt mit dem speziellen Zeichen '#' und es kann ein Leerzeichen oder ein Minus Zeichen '-' folgen. Das Zeichen '#' allein ist auch zulässig.

```
#
#-
#....
# - Login -
```

Ein Kommentar am Ende der Zeile ist also erlaubt:

```
#SendStr "<13>" # sending a wakeup to the server
```

```
Leerzeile
```

Eine leere Zeile ist erlaubt und wird wie die Kommentarzeile ignoriert. Wie jede Zeile ist auch eine leere Zeile mit CR (ASCII 13) und/oder LF (ASCII 10) abgeschlossen.

```
#Lmarke
```

Eine Markierung (Label) markiert eine einzelne Zeile zu der verzweigt werden kann. Eine Markierung beginnt mit der Sequenz '#L' gefolgt von einem einzigartigen Namen. Zur Markierung kann mittels `#Goto`, `#OnSuccess`, `#OnFailure`, `#OnTrue`, `#OnFalse` gesprungen werden.

```
#L_TXFILES
```

```
#TIMEOUT=Wert
```

Definiert den Wert für die Zeitüberschreitung in Millisekunden, welche die Kommandos wie *WaitForStr*, *SendStr*, *ReceiveStr* nutzen. Der voreingestellter Wert ist 1000 (1 Sekunde).

```
#TIMEOUT=5000
```

Definiert 5 Sekunden.

```
#ReceiveStr [Zeichenkette]
```

Das Kommando *ReceiveStr* empfängt Zeichen bis zur angegebenen Zeichenkette oder bis die Anzahl *#RXCOUNT* erreicht wurde oder eine Zeitüberschreitung erfolgt (siehe auch *#TIMEOUT*) oder der interne Puffer komplett aufgefüllt ist (2048 Bytes/Zeichen). Der Parameter *Zeichenkette* ist optional. Wenn keine Zeichenkette angegeben wird, sammelt es Daten bis zur Anzahl *#RXCOUNT* oder bis zur Zeitüberschreitung oder der interne Puffer komplett aufgefüllt ist.

```
#ReceiveStr
```

```
#ShowStr
```

Wartet auf eine Zeichenkette bis zur Zeitüberschreitung und gibt sie aus.

Die Zeichenkette wird ungeachtet auf Groß- oder Kleinschrift geprüft.

```
#SendStr Zeichenkette
```

Das Kommando *SendStr* dient zur Übertragung von Daten.

```
#SendStr "Hello World<13>"
```

Wird die Zeichenkette "Hello World" gefolgt vom Return (ASCII 13) an die Gegenstelle übertragen.

```
#WaitForStr Zeichenkette
```

Das Kommando *WaitForStr* wartet bis die angegebene Zeichenkette empfangen wurde oder eine Zeitüberschreitung erfolgt ist (siehe auch *#TIMEOUT*) oder die in *#RXCOUNT* angegebene Zeichenanzahl empfangen wurde oder der interne Puffer komplett aufgefüllt ist (2048 Bytes/Zeichen).

```
#WaitForStr "Username:"
```

Wartet auf die Zeichenkete "Username:".

Die Zeichenkette wird ungeachtet auf Groß- oder Kleinschrift geprüft.

```
#GOTO Label
```

Dieses Kommando verzweigt zur angegebenen Markierung bzw. Zeile. E ist eine Einweg Übertragung der Kontrolle an eine andere Skript-Zeile. Die Ausführung wird mit der Zeile nach der Markierung fortgesetzt.

```
#GOTO L_END
```

```
#ONFAILURE Label oder auch #ONFALSE Label
```

Dieses Kommando ist eine, an Bedingung geknüpfte, Variante des #GOTO. Es prüft das 'Resultat' der letzten Operation (z.B. *IFINSTR*, *ReceiveStr*, *SendStr*, *WaitForStr*) und springt im Fehlerfall zur angegebenen Marke (Label).

```
#ONFAILURE L_END
```

Übersetzt in: *If LastError<>0 Then Goto L\_END*

```
#ONSUCCESS Label oder auch #ONTRUE Label
```

Dieses Kommando ist eine, an Bedingung geknüpfte, Variante des #GOTO. Es prüft das Resultat der letzten Operation (z.B. *IFINSTR*, *ReceiveStr*, *SendStr*, *WaitForStr*) und springt im Erfolgsfall zur angegebenen Marke (Label).

```
#ONSUCCESS L_TXFILES
```

Übersetzt in: *If LastError==0 Then Goto L\_TXFILES*

```
#PRINTSTR Zeichenkette
```

Gibt die Zeichenkette aus.

```
#PRINTSTR "Line 101"
```

```
#SHOWSTR
```

Gibt die Zeichenkette im internen Puffer aus. Der interne Puffer speichert Daten nach *WaitForStr*, *ReceiveStr* und kann bis zu 2048 Bytes/Zeichen aufnehmen.

```
#SHOWSTR
```

```
#SHOWVAR Variablename [, Variablename, ...] [, Names]
```

Gibt den Wert der Variablen aus (z.B. *Protocol*, *RXCount*, *StrLen*, *Settings*, *VAR\_I*, *VAR\_J*, *VAR\_K*, *VAR\_A*, *VAR\_B*, *VAR\_C*, *LINE*). Der Name der Variablen wird hier als Parameter erwartet und ohne das Zeichen '#'.  
 Mehrere Variablen können angegeben werden und sollten durch ' ' oder ',' getrennt werden. Der spezielle Bezeichner „NAMES“ bewirkt, dass auch der Name der Variablen inkl. „=“ ausgegeben wird.

```
#SHOWVAR RXCount
```

```
#SHOWVAR VAR_I, VAR_A names
```

```
#SHOWVAR Line names # druckt die aktuelle Zeilennummer im Skript
```

```
#IFINSTR Zeichenkette
```

Prüft ob die Zeichenkette sich im internen Puffer befindet. Die Prüfung bewirkt ein Resultat, dass per '#ON..' Kommando ausgewertet werden kann. Der interne Puffer speichert Daten nach *WaitForStr*, *ReceiveStr*.

```
#ReceiveStr
#IFINSTR "Welcome"
#ONSuccess L_TXFILES
```

Empfängt eine Zeichenkette und springt zur Marke L\_TXFILES, wenn die bestimmte Zeichenkette empfangen wurde.

```
#IFNOTINSTR Zeichenkette
```

Prüft ob die Zeichenkette sich im internen Puffer befindet. Die Prüfung bewirkt ein Resultat, dass per '#ON..' Kommando ausgewertet werden kann. Der interne Puffer speichert Daten nach *WaitForStr*, *ReceiveStr*.

```
#ReceiveStr "<13>"
#IFNOTINSTR "Welcome"
#ONSuccess L_END
```

Empfängt eine Zeichenkette bis zum Return und springt zur Marke L\_END, wenn die bestimmte Zeichenkette nicht empfangen wurde.

Übersetzt in: *If Not Found "Welcome" Then Goto L\_END*

```
#DELAY Wert
```

Verzögert die Ausführung um *Wert* Millisekunden.

```
#DELAY 1000
```

Führt die Ausführung nach einer Sekunde Wartezeit fort.

```
#RXCOUNT =|==|>|<|<>|!|=|>=|<= Wert
```

Die Variable *RXCount* steuert die Funktionen *WaitForStr* und *ReceiveStr*. Wird *RXCount* ein Wert größer 0 gesetzt, empfangen die Funktionen bis zur dieser Anzahl Zeichen oder bis eine andere Bedingung erfüllt ist (z.B. Zeitüberschreitung, gesuchte Zeichenkette).

Wird *RXCount* auf 0 gesetzt (Voreinstellung) werden, je nach Funktion, die anderen Bedingungen beachtet (2048 Puffergrenze, Zeitüberschreitung, Such-Zeichenkette) und *RXCount* ignoriert.

Die Bedeutung der Operatoren wird unter #VAR\_x erklärt.

Es können nur positive Werte gesetzt werden.

```
#RXCOUNT=10
#TIMEOUT=2000
#ReceiveStr
#SHOWSTR
#SHOWVAR RXCount
#STRLEN>=10
#ONTRUE L_END
#PRINTSTR „Weniger als erwartet“
```

```
#STRLEN ==|>|<|<>|!=|>|=|<= Wert
```

Die Variable *StrLen* liefert die Anzahl der Zeichen im internen Puffer z.B. nach *ReceiveStr*, *WaitForStr*. Die Variable *StrLen* kann nur abgefragt werden.

Die Bedeutung der Operatoren wird unter *#VAR\_x* erklärt.

```
#RXCOUNT=10
#TIMEOUT=2000
#ReceiveStr
#SHOWSTR
#SHOWVAR StrLen
#STRLEN<10
#ONTRUE L_GO
#PRINTSTR „Weniger als erwartet“
```

```
#VAR_x =|+=|-=|==|>|<|<>|!=|>|=|<= Wert
```

Es werden die folgenden Variablen unterstützt: *A, B, C, I, J, K*. Die Variablen können für Schleifen und als Zähler genutzt werden. Es können sowohl negative als auch positive Werte gesetzt werden.

```
#VAR_I=-10      setzt einen negativen Wert
#VAR_I-=1       vermindert die Variable um den Wert
#VAR_I+=2       erhöht die Variable um den Wert
```

Die Variablen *A, B, C, I, J, K* können verändert und abgefragt werden.

Folgende logische Operationen sind möglich und setzen ein Resultat, dass per *#ON..* Kommando geprüft werden kann:

```
#VAR_A==0      prüft auf Gleichheit
#VAR_B<>0      prüft auf ungleich
#VAR_C!=0      prüft auf ungleich
#VAR_I>0       prüft auf größer als
#VAR_J>=0      prüft auf größer oder gleich
#VAR_K<0       prüft auf kleiner als
#VAR_I<=0      prüft auf kleiner oder gleich
```

Der Wert einer Variablen kann mittels *#SHOWVAR* ausgegeben werden.

## **Beispiele**

### **Inkrementieren und ausgeben**

```
#VAR_A=1      # setze auf 1
#VAR_A+=2     # erhöhe um 2
#PRINTSTR "Variable A="
#SHOWVAR     VAR_A
#PRINTSTR "<13><10>-----<13><10>"
```

### **Schleife mit positiven Werten**

```
#VAR_I=0      # setze auf 0
#L_LOOP1
#Delay 300
#VAR_I+=1     # erhöhe um 1
#SHOWVAR     VAR_I names
#PRINTSTR "<13><10>-----<13><10>"
#VAR_I<>10    # vergleiche mit 10
#ONTRUE L_LOOP1
```

Schleife mit negativen Werten

```
#VAR_I=-1      # setze auf -1
#L_LOOP1
#Delay 300
#VAR_I=-1      # vermindere um -1
#SHOWVAR     VAR_I names
#PRINTSTR    "<13><10>-----<13><10>"
#VAR_I=-10    # vergleiche mit -10
#ONFALSE L_LOOP1
```

```
#FLOW =|== Wert-Zeichenkette
```

Die Variable *Flow* definiert die Flußsteuerung (Handshake) für die serielle Leitung.

Mögliche Werte sind: CTS, DSR, XON auch in jede Kombination. Mehrere werden am besten in Anführungszeichen und durch ' ' oder ',' getrennt angegeben.

Die Bedeutung der Operatoren wird unter #VAR\_x erklärt.

```
#FLOW="CTS,DSR"
#SHOWVAR "Flow names"
```

```
#RTS =|== 0 oder 1
```

Die Variable *RTS* ändert oder ermittelt den Signal-Status der RTS Leitung im seriellen UART.

Die Bedeutung der Operatoren wird unter #VAR\_x erklärt.

```
#DTR =|== 0 oder 1
```

Die Variable *DTR* ändert oder ermittelt den Signal-Status der DTR Leitung im seriellen UART.

Die Bedeutung der Operatoren wird unter #VAR\_x erklärt.

```
#CTS == 0 oder 1
```

Die Variable *CTS* ermittelt den Signal-Status der CTS Leitung im seriellen UART und setzt ein Resultat, dass per #ON.. Kommando geprüft werden kann.

```
#DSR == 0 oder 1
```

Die Variable *DSR* ermittelt den Signal-Status der DSR Leitung im seriellen UART und setzt ein Resultat, dass per #ON.. Kommando geprüft werden kann.

```
#DCD == 0 oder 1 auch #RLSD == 0 oder 1
```

Die Variable *DCD* ermittelt den Signal-Status der DCD Leitung im seriellen UART und setzt ein Resultat, dass per #ON.. Kommando geprüft werden kann.

```
#RI == 0 oder 1
```

Die Variable *RI* ermittelt den Signal-Status der RI Leitung im seriellen UART und setzt ein Resultat, dass per #ON.. Kommando geprüft werden kann.

**#OPENFILE** *Paare-Werte-Zeichenkette*

*OPENFILE* erzeugt bzw. öffnet eine Datei. Die Parameter Zeichenkette enthält "Name=Wert;" Einträge, die durch ';' getrennt sind. Es sollten keine Leerzeichen in der Parameter Zeichenkette stehen.

Die Parameter Zeichenkette liefert den Dateinamen („Name=“) und optionale Optionen („Options="). Der Dateiname sollte den kompletten Pfad enthalten sonst wird es lokal zum ausgeführten Modul vermutet. Die Optionen definieren welche Operationen auf die Datei ausgeführt werden dürfen (lesen, schreiben, hinzufügen) und den Modus (Text oder Binär).

Mögliche Werte für *Options*:

```
Options=[READ | WRITE | APPEND | READ WRITE | READ APPEND] [TEXT | BIN]
```

Wenn keine Optionen angegeben werden, wird Read und Text vorbelegt. Wird WRITE ohne READ auf eine existierende Datei angewendet, wird die Datei zuerst gelöscht.

```
#OpenFile "Name=acc001.txt;Options=WRITE TEXT"
#WriteFile "Eine Zeile.<10>"
#WriteFile "Eine weitere Zeile.<10>"
#WriteFile "Fertig.<10>"
#CloseFile
```

Aktuell kann nur eine Datei gleichzeitig geöffnet werden.

*OPENFILE* setzt ein Resultat, dass per #ON.. Kommando geprüft werden kann.

**#CLOSEFILE**

*CLOSEFILE* schließt die Datei, die per *OPENFILE* geöffnet wurde.

*CLOSEFILE* setzt ein Resultat, dass per #ON.. Kommando geprüft werden kann.

**#READFILE** [*Anzahl*]

*READFILE* liest Daten aus der geöffneten Datei. Die Daten werden im internen Puffer abgelegt und können per #*ShowStr* angezeigt oder per #*SendStr* übertragen werden.

Der Parameter *Anzahl* steuert die *Anzahl* der zu lesende Daten. Wenn der Parameter *Anzahl* nicht versorgt wird, liest die Funktion, bei eine Text-Datei, ganze Zeilen und bei Binäre-Datei, bis der interne Puffer gefüllt ist (2048 Bytes/Zeichen).

Wenn der Parameter *Anzahl* in Verbindung mit einer Text-Datei verwendet wird, liefert die Funktion evtl. weniger Zeichen als angegeben, abhängig von der Zeilenlänge.

*READFILE* setzt ein Resultat, dass per #ON.. Kommando geprüft werden kann.

```
#OPENFILE "Name=acc001.txt;Options=READ BIN;"
#OnFailure L_END
#L_REPEAT
#ReadFile 32 # lese xx Zeichen
#ShowStr
#PRINTSTR "<13><10>"
#OnSuccess L_REPEAT
#CLOSEFILE
```

```
#WRITEFILE [Zeichenkette]
```

*WRITEFILE* schreibt Daten in die geöffnete Datei. Wenn der Parameter *Zeichenkette* nicht versorgt wird, werden die Daten aus dem internen Puffer geschrieben.

```
#ReadStr
#WriteFile           # schreibe den Puffer wie empfangen
#WriteFile "End<10>" # schreibe den Parameter
```

*WRITEFILE* setzt ein Resultat, dass per #ON.. Kommando geprüft werden kann.

```
#SEEKFILE =Anzahl [FromStart | FromEnd]
```

*SEEKFILE* bewegt den Dateizeiger um *Anzahl* Positionen. Die Bewegung erfolgt relativ zur aktuellen Position oder relativ zum Anfang der Datei (*FromStart*) oder relativ zum Ende der Datei (*FromEnd*). Wenn nichts zusätzlichen angegeben wird, ist es relativ zur aktuellen Position.

Bei Text-Dateien sollten nur "=0 Start" oder "=0 End" benutzt werden.

*SEEKFILE* setzt ein Resultat, dass per #ON.. Kommando geprüft werden kann.

```
#SeekFile 20 FromStart
```

```
#CONNECT [=] Paare-Werte-Zeichenkette
```

Erstellt eine Verbindung.

Die Parameter *Zeichenkette* enthält "Name=Wert;" Einträge, die durch ';' getrennt sind. Es sollten keine Leerzeichen in der Parameter *Zeichenkette* stehen.

Die Parameter *Zeichenkette* liefert folgende Einträge:

```
Serial=COM1; | COM2; | , ..
ComType=RS232 | RS422 | RS485 | TCPIP-CLIENT | TAPI | ISDN;
ConnectAddress=192.168.0.1:9100; | 2223334444;
ConnectTimeout=60;
```

und die anderen Werte die unter #*SETTINGS* beschrieben werden für serielle Leitungen.

### Serial

definiert den Namen der seriellen Schnittstelle. Wenn dieser Wert gesetzt ist wird der *ComType* nicht benötigt außer es soll eine RS485 genutzt werden.

### ComType

definiert der Verbindungstyp. Die aktuelle Software-Lizenz muss es unterstützen.

### ConnectAddress

definiert die Verbindungsadresse. Dieser Wert ist vom *ComType* abhängig. Für eine direkt serielle Leitung (nur Kabel kein Modem) wird es nicht benötigt da kein Modem zum wählen.

Für serielle Verbindungen, ISDN wird die Telefonnummer erwartet.

Für TCP/IP wird die IP Adresse und die Port Adresse erwartet z.B. 192.168.0.1:9000 oder  
www.mydomain.com:9100.



ConnectTimeout

definiert die erlaubte Dauer für den Verbindungsversuch. Es kann nicht kürzer als 10 Sekunden sein. Für nationale Anrufe sollten 60 Sekunden ausreichen. Für internationale Anrufe besser 90.

```
#CONNECT "Serial=COM1;BaudRate=115200;Handshaking=CTS;ConnectAddress=2223456;
ConnectTimeout=60;"
#OnFailure L_RETRY

#CONNECT "ComType=TCPIP-CLIENT;ConnectAddress=www.mynet.com:9100;
ConnectTimeout=60;"
#OnFailure L_RETRY
```

#SETTINGS [=] *Paare-Werte-Zeichenkette*

Ändert die aktuellen Kommunikations-Parameter für die serielle Leitung.

Die Parameter Zeichenkette enthält "Name=Wert;" Einträge, die durch ';' getrennt sind. Es sollten keine Leerzeichen vor dem Wert-Namen und zwischen den Sonderzeichen '=' und ';' stehen.

Die Parameter Zeichenkette liefert folgende Einträge:

```
"BaudRate=50..921600" # abhängig was das aktuelle UART erlaubt
"DataBits=5-8"
"StopBits=1 | 2"
"Parity= None | Odd | Even | Mask | Space" # nur ein Wert, der erste Buchstabe genügt
"Handshaking=" None or any combination of "CTS, DSR, XON"
```

Wenn ein Eintrag fehlt, wird der aktuelle Wert genutzt.

```
#Settings "BaudRate=9600;DataBits=8;StopBits=1;Parity=N;Handshaking=CTS;"
#Settings "BaudRate=115200;" # ändert nur die Baudrate
```

#PROTOCOL [=] *Zeichenkette*

Ändert das aktuelle Protokoll für die nächste Dateiübertragung. Vorausgesetzt die aktuelle Software Lizenz beinhaltet das angegebene Protokoll.

Der Parameter *Zeichenkette* liefert den Protokoll-Namen.

```
#Protocol=ZMODEM | KERMIT | XMODEM | XMODEM-CRC | YMODEM | YMODEM-BATCH | ASCII
```

**Skript - Beispiele**

(Wichtig: Nutzen Sie nur reine Text-Dateien im ANSI (8-Bit) Format.)

**#ReceiveStr als #WaitFor Ersatz:**

```
#WaitFor "Username:"
#OnFalse L_END

#ReceiveStr "name:"
#IFINSTR "name:"
#OnFalse L_END
```

**Senden von binären Telegrammen**

```
#SendStr "<2>Data<3>"   wird STX Data ETX   senden
```

**Austausch von Telegrammen**

```
#L_START1
#TIMEOUT=2000
#WaitForStr "Hello<13>"   # warte max 2 Sek auf Zeichenkette
#OnFalse L_START1        # wenn fehlgeschlagen zum Anfang
#SendStr "OK<13>"        # antworte
#L_END
```

**Schleife unter Nutzung der Variablen 'I' als Laufvariable**

```
# Schleife zur Marke LOOP1 10 mal ausführen

#VAR_I=0
#L_LOOP1
#Delay 300
#VAR_I+=1           # erhöhe um eins
#VAR_I<>10          # prüfe
#ONTRUE L_LOOP1    # verzweige wenn zutrifft
#PRINTSTR "Variable I="
#SHOWVAR   VAR_I
#PRINTSTR "<13><10>-----<13><10>"
```

**Manuelles Login (ähnlich wie es der ZMCS Server erwartet)**

```
#SendStr "<13>"
#TIMEOUT=3000
#WaitForStr "Username:"   # warte auf Frage
#IFINSTR "name:"
#OnFalse L_END
#SendStr "dorian<13>"     # sende Benutzernamen
#WaitForStr "Password:"   # warte auf nächste Frage
#OnFailure L_END
#SendStr "secret01<13>"   # sende Passwort
#WaitForStr "Welcome"     # warte auf Begrüßung
#OnFailure L_END

#L_END
```

**Dateibehandlung**

```
#OPENFILE "Options=APPEND;Name=file.txt;"
#OnFailure L_END
#WriteFile "Eine Zeile.<10>"
#WriteFile "Eine weitere Zeile.<10>"
#WriteFile "Fertig.<10>"
```

```
#CLOSEFILE
#L_END
```

### Protokoll-Änderung

```
#TX marks a transmit section
# -----
#Protocol = Kermit      # switching protocol
"atest00.txt"          # use quoted to protect from accidentally added spaces etc.

#Protocol = Zmodem     # switching protocol
"atest01.txt"

#Protocol = Kermit     # switching protocol

#RX marks a receive/retrieve section
# -----
atest11.txt
atest22.txt
atest33.txt
```

## **Rückgabewerte:**

Sobald das ZMCS Programm endet liefert es einen Rückgabewert. Folgende Rückgabewerte definieren ob der Auftrag erfolgreich ausgeführt wurde und ob, im Fehlerfall, eine Wiederholung sinnvoll ist.

- 0 Falsche Parameter oder Benutzerabbruch (Strg-C, Ctrl-C)
- 1 OK, Dateien übertragen**
- 2 Fehler, wiederholen sinnvoll
- 3 Fehler, wiederholen nicht sinnvoll

## **Fehler**

- 301 Ungültige Schnittstellennummer
- 302 Datei nicht gefunden
- 303 Entferntes Modem antwortet nicht oder ist besetzt
- 304 Timeout beim Senden von Daten
- 305 Fehler in der Synchronisation (Handshake)
- 306 Modem ist nicht nicht angeschlossen bzw. ausgeschaltet
- 307 Benutzeranmeldung hat fehlgeschlagen
- 308 Ungültige Seriennummer

Den Rückgabewert kann das aufgerufene Programm abfragen sobald die ZMCS Anwendung endet. Innerhalb einer Batch Datei (.BAT, .CMD) kann dieser Wert per ERRORLEVEL abgefragt werden z.B.:

```
IF ERRORLEVEL 1 GOTO S_OK
ECHO Error Nr: %ERRORLEVEL%
GOTO S_END
:S_OK
ECHO Successful!
:S_END
```

**Protokoll Fehlerwerte**

Die SuperCom Rückgabewerte sind negativ. Rückgabewerte > 0 stammen von Systemfunktionen z.B. 2=Datei nicht gefunden.

Wenn die Kommunikation mit Fehler abbricht liegt es meistens an der schlechten Verbindung oder falsche Protokoll-Einstellungen. Im folgenden einige spezielle Protokoll Fehlerwerte für die Problemsuche:

Eine Zeitüberschreitung (-9999)

Der Partner hat zu spät reagiert. Bei Satellitenstrecken sollten evtl. die Zeiten erhöht werden. Siehe auch *Fragen und Antworten*, *Zeitverhalten*.

Abbruch durch Benutzer (-9998)

Abbruch wegen vielen Wiederholungen (-9997)

Verbindung ist nicht stabil – wahrscheinlich zu viele Wiederholungen. Kommunikation- und Protokoll-Einstellungen des Partners prüfen.

Abbruch durch entfernten Benutzer oder Protokoll (-9996)

Protokoll hat falsche Daten erhalten (-9994)

Kompatibilitäts-Problem. Kommunikation- und Protokoll-Einstellungen des Partners prüfen.

## Fragen und Antworten

### *Kann ich auch binäre Dateien damit übertragen ?*

Ja, ZMCS kann sowohl Textdateien als auch binäre Dateien übertragen.

### *Bietet es Schutz beim Zugriff auf Verzeichnisse und Dateien ?*

Der Server kann auf das aktuelle Arbeitsverzeichnis eingeschränkt werden (siehe Parameter /W). Die empfangenen Dateien werden nur hier abgelegt. Dateien können nur von hier aus versendet bzw. vom Klienten abgeholt werden. Andere Verzeichnisse werden so geschützt. Zusätzlich kann beim Server die *Anmeldeprozedur* per *Benutzernamen* und *Passwort* aktiviert werden (siehe Parameter /XL).

### *Kann ich per fremde Anwendung andocken ?*

Der Server erkennt das Kommando /MsH und schaltet das intern genutzte Synchronisationsprotokoll (Handshake) aus. Dadurch kann eine fremde Anwendung (z.B. Hyperterminal) am Server andocken. Wenn der Server innerhalb von ca. 20 Sekunden keine Daten erhält wird die Verbindung getrennt. Diese Zeitbeschränkung ist bei direkter Verbindung (/L) nicht aktiv. Sofort nach erfolgreicher Verbindung sollte die Fremdanwendung mit der Dateiübertragung anfangen.

### *Was für ein Handshake wird vom Server genutzt ?*

Die Beschreibung des Handshakes, daß der ZMCS Server nutzt, ist für die Kunden kostenlos erhältlich, damit Software-Entwickler auch mit eigenen Funktionen am Server Verbinden können.

### *Warum finde ich die empfangene Datei nicht?*

Es können mehrere Gründe dafür vorliegen:

1. Ein Abbruch der Dateiübertragung wegen Fehler. In der Regel sollte der erfolgreiche Teil der Datei vorhanden sein. Das *ZMODEM Protokoll* erlaubt bei einem späteren Versuch nur noch den fehlenden Teil anzufordern. Wenn aber der Abbruch schon im ersten Datenpaket stattfindet ist die Dateilänge 0 und wird gelöscht.
2. Die angeforderte Datei war auf dem Server nicht vorhanden. Bitte Dateinamen im Parameter /F prüfen. Beim Server das Arbeitsverzeichnis prüfen (siehe Parameter /W).
3. Die zu sendende Datei wurde vom Klienten nicht gefunden. Bitte Pfad und Dateinamen im Parameter /F prüfen. Das Arbeitsverzeichnis prüfen (siehe Parameter /W).

### *Warum funktioniert die Benutzeranmeldung nicht ?*

Der Server sollte mit dem Parameter /XL gestartet werden. Die Benutzerangaben in der Datei *ZMCS\_usr.ini*, im Server Verzeichnis, sollten identisch mit dem Inhalt des Parameters */XLU=benutzer,P=passwort* beim Aufruf des Klienten sein. Z.B.

**Server:** `zmcs.exe /C2 /Ms /XL`

(started den Server auf COM2 (ttyS1) und erwartet, daß die Klienten sich anmelden)

**Client:** `zmcs.exe /C1 /Mt /T123456789 /F@filelist.txt /XLU=dorian,P=secret01`

(stellt eine Wählverbindung zum Server her, führt die Anmeldung und die Anweisungen der Dateiliste durch)

**Wie kann eine Anwendung einen Server am besten beenden ?**

Angenommen der Server arbeitet mit der Schnittstelle COM3 (ttyS2) und soll per externes Programm beendet werden. Dazu wird die ZMCS Anwendung mit den Parametern /Mu gefolgt von der Schnittstellenummer aufgerufen.

Beenden des Servers auf COM3 (ttyS2): ZMCS /Mu3

**Wie kann der ZMCS Server mit starthidden gestartet werden ?**

Einfach "starthidden" vor jeder Kommandozeile eingeben e.g.

```
starthidden zmcs.exe /c3 /Ms
```

**Wie kann das Zeitverhalten im ZMODEM angepasst werden ?**

Es wird eine Konfigurations-Datei mit dem Namen „supercom.ini“ benötigt mit folgendem Eintrag unter der benutzten Schnittstellen z.B. für COM3 (ttyS2).

Die folgenden Wertangaben definieren Millisekunden.

```
[COM3]
; Voreinstellung ist 1000 ms
STD_WAITONECHAR=6000
; Voreinstellung ist 5000 ms
STD_WAITHEADER=12000
```

Die Voreinstellung ist im ZMODEM Protokoll Modul auf folgende Werte gesetzt:

```
STD_WAITONECHAR=1000
STD_WAITHEADER=5000
```

**Wie wird die Verbindungszeit-Überwachung angepasst ?**

Ein Verbindungsversuch wird für die Dauer von maximal 90 Sekunden gestartet. Das ist i.d.R. ausreichend sogar für internationale Anrufe.

Diese Wartezeit kann durch den Schlüssel „CONNWAIT“, in der Konfigurationsdatei „zmcs.ini“, unter der Sektion „CONFIG“, angepasst werden. Die Wertangabe definiert Sekunden.

```
[CONFIG]
; Voreinstellung ist 90 Sekunden
CONNWAIT=120
```

**Wie wird die Überwachungszeit ohne Handshake angepasst ?**

Nach dem Verbindungsaufbau wartet der Server bis zu 20 Sekunden, dass der entfernte Klient mit dem ZMODEM Protokoll beginnt. Dieser Wert wird in der Arbeitsweise ohne Handshake (/MsH) genutzt und kann über den Schlüssel TO\_SYN\_NH, in der Konfigurationsdatei „zmcs.ini“, unter der Sektion „CONFIG“, angepasst werden. Die Wertangabe definiert Sekunden.

```
[CONFIG]
; Voreinstellung ist 20 Sekunden
TO_SYN_NH=60
```

***Kann ZMCS eine Anwendung starten sobald eine Datei empfangen wurde ?***

Ja, es kann eine ausführbare Datei über den Schlüssel `RUN_AFTER_RX`, in der Konfigurationsdatei „zmc.ini“, unter der Sektion „CONFIG“, eingetragen werden. Die auszuführende Datei im selben Verzeichnis wie die Programmdatei „zmc.exe“ erwartet.

```
[CONFIG]
RUN_AFTER_RX=after_rx.cmd
```

***Wo sollten die Konfigurationsdateien gespeichert werden ?***

Die Konfigurations-Dateien z.B. **zmc.ini**, **supercom.ini** werden im selben Verzeichnis wie die Programmdatei „zmc.exe“ erwartet.

Für die Erzeugung und Modifikation von Konfigurationsdateien sollte Notepad oder ein anderer Editor genutzt werden und die Datei im ANSI Format (8-bit) gespeichert werden.

***Ist es möglich die Ausgabe von ZMCS direkt in meiner Anwendungen zu bekommen ?***

Ja. Der einfachste Weg ist die DOS Pipe "<" um die Ausgabe zu einer Datei oder zur eigene Anwendung umzuleiten. IPC (Interprocess Communication) wird von ZMCS auch unterstützt (/XIC). Ein kleines Beispiel in C/C++ Quelltext demonstriert es.

***Kann ZMCS auch unter Windows 64-Bit (x64) laufen ?***

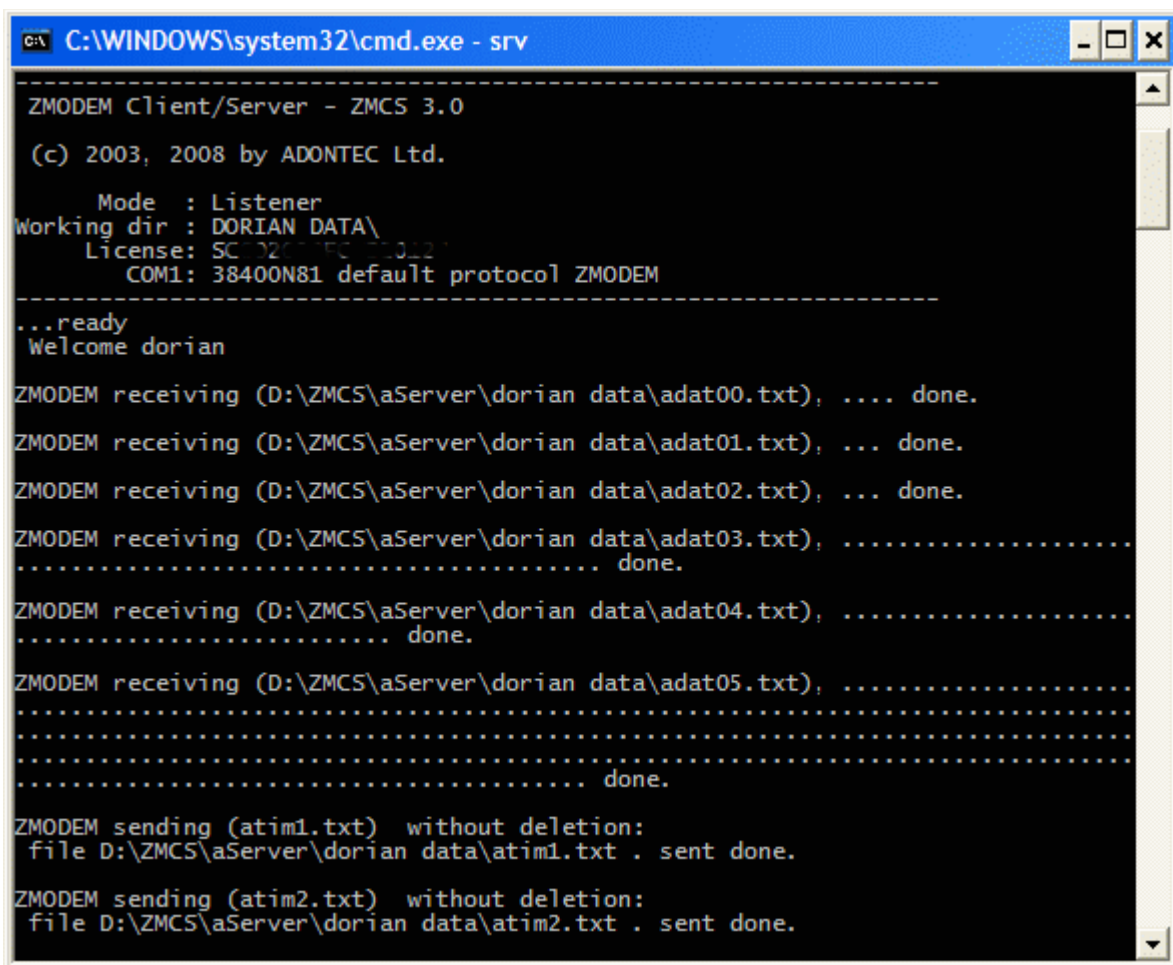
Ja, ZMCS für Windows ist ein echte 32 Bit Windows Anwendung und wurde auch kompatibel zu Windows x64 Systeme entwickelt. Dadurch kann es auch unter Windows x64 ausgeführt werden.



## Beispiel Sitzung

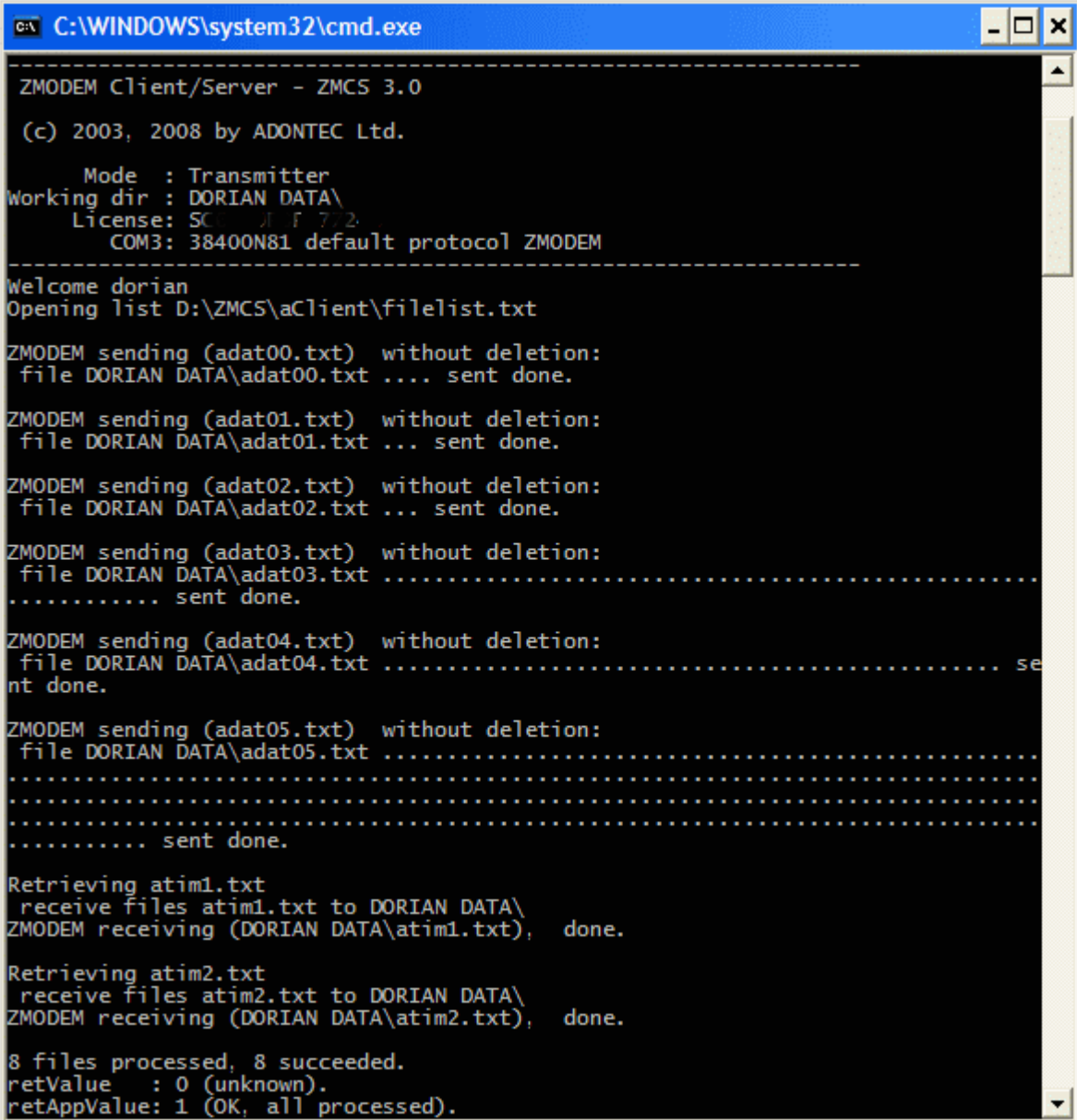
### Server

zmcs.exe /c3 /ms /L /XL



## Client

```
zmcs.exe /c4 /mt /L /w"D:\ZMCS\aClient\dorian data" /f@filelist.txt
/XLU=dorian,P=secret01
```



```
C:\WINDOWS\system32\cmd.exe
-----
ZMODEM Client/Server - ZMCS 3.0

(c) 2003, 2008 by ADONTEC Ltd.

Mode : Transmitter
Working dir : DORIAN DATA\
License: SC...
COM3: 38400N81 default protocol ZMODEM
-----
Welcome dorian
Opening list D:\ZMCS\aClient\filelist.txt

ZMODEM sending (adat00.txt) without deletion:
file DORIAN DATA\adat00.txt .... sent done.

ZMODEM sending (adat01.txt) without deletion:
file DORIAN DATA\adat01.txt ... sent done.

ZMODEM sending (adat02.txt) without deletion:
file DORIAN DATA\adat02.txt ... sent done.

ZMODEM sending (adat03.txt) without deletion:
file DORIAN DATA\adat03.txt ..... sent done.

ZMODEM sending (adat04.txt) without deletion:
file DORIAN DATA\adat04.txt ..... sent done.

ZMODEM sending (adat05.txt) without deletion:
file DORIAN DATA\adat05.txt ..... sent done.

Retrieving atim1.txt
receive files atim1.txt to DORIAN DATA\
ZMODEM receiving (DORIAN DATA\atim1.txt), done.

Retrieving atim2.txt
receive files atim2.txt to DORIAN DATA\
ZMODEM receiving (DORIAN DATA\atim2.txt), done.

8 files processed, 8 succeeded.
retValue : 0 (unknown).
retAppValue: 1 (OK, all processed).
```

### Inhalt der Datei filelist.txt

```
# Sample filelist
# RX and TX section supported to perform within a connection session.
# If no section marked, TX is taken as default
# Any line starting with # is a comment line
# -----
#TX marks a transmit section
adat00.txt
adat01.txt
```

adat02.txt

adat03.txt

adat04.txt

adat05.txt

#RX marks a retrieve section

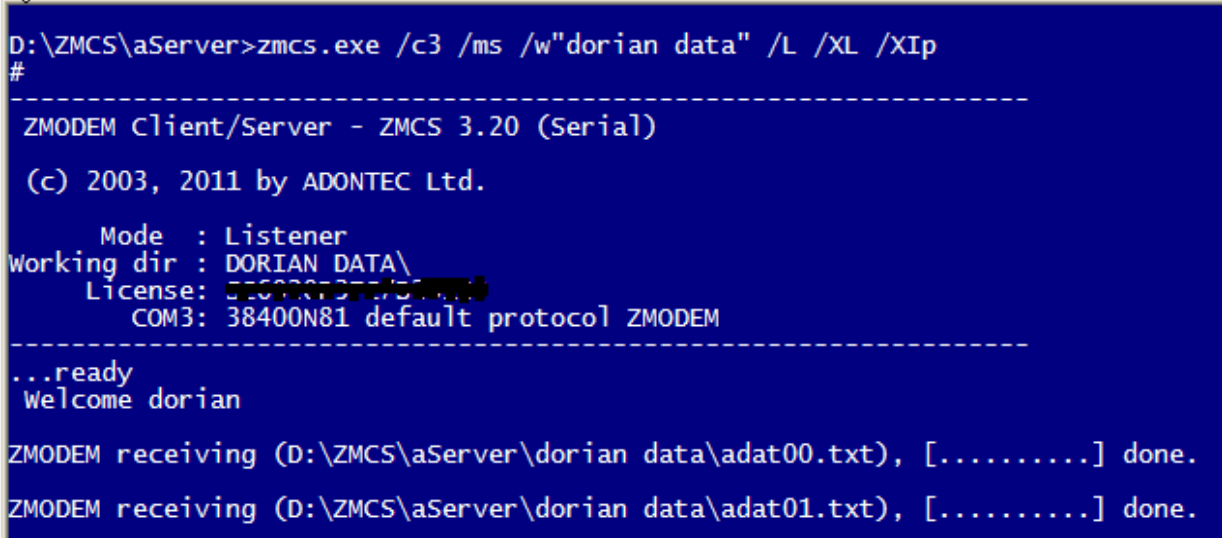
atim1.txt

atim2.txt

## Beispiel Sitzung 2

### Server

```
zmcs.exe /c3 /ms /L /XL /XIp
```



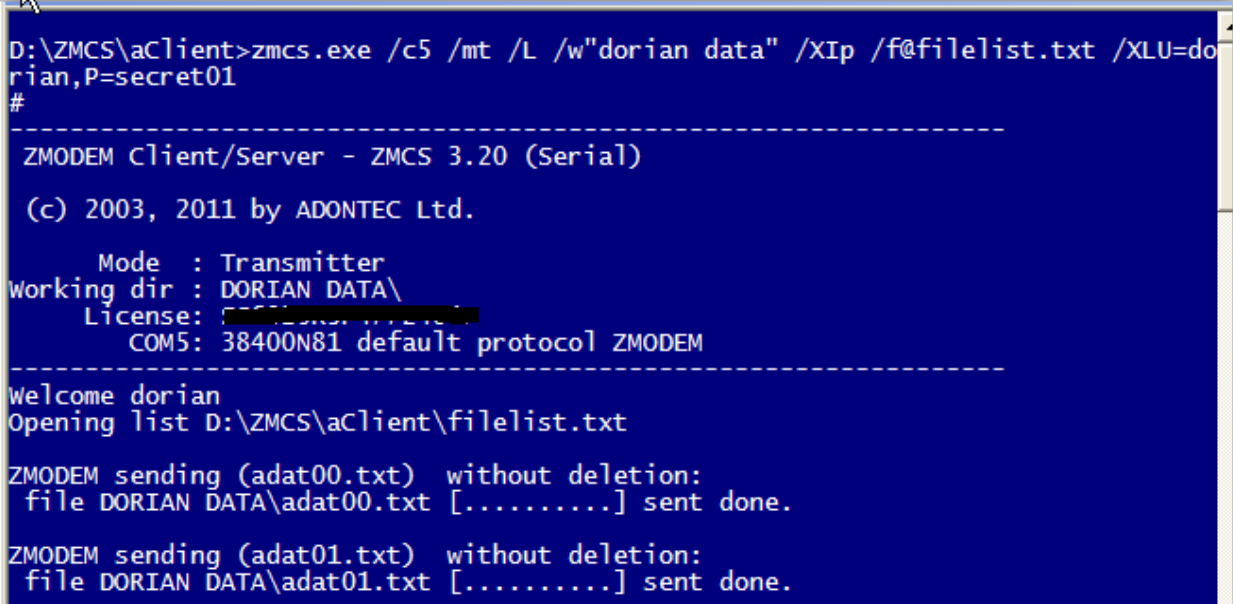
```
D:\ZMCS\AServer>zmcs.exe /c3 /ms /w"dorian data" /L /XL /XIp
#
-----
ZMODEM Client/Server - ZMCS 3.20 (Serial)
(c) 2003, 2011 by ADONTEC Ltd.
Mode : Listener
Working dir : DORIAN DATA\
License: [REDACTED]
COM3: 38400N81 default protocol ZMODEM
-----
...ready
Welcome dorian

ZMODEM receiving (D:\ZMCS\AServer\dorian data\adat00.txt), [.....] done.
ZMODEM receiving (D:\ZMCS\AServer\dorian data\adat01.txt), [.....] done.
```

Prozentuale Fortschrittsanzeige.

## Client

```
zmcs.exe /c4 /mt /L /w"D:\ZMCS\acClient\dorian data" /f@filelist.txt  
/XLU=dorian,P=secret01 /Xip
```



```
D:\ZMCS\acClient>zmcs.exe /c5 /mt /L /w"dorian data" /XIp /f@filelist.txt /XLU=dorian,P=secret01  
#  
-----  
ZMODEM Client/Server - ZMCS 3.20 (Serial)  
  
(c) 2003, 2011 by ADONTEC Ltd.  
  
Mode : Transmitter  
Working dir : DORIAN DATA\  
License: [REDACTED]  
COM5: 38400N81 default protocol ZMODEM  
-----  
welcome dorian  
Opening list D:\ZMCS\acClient\filelist.txt  
  
ZMODEM sending (adat00.txt) without deletion:  
file DORIAN DATA\adat00.txt [.....] sent done.  
  
ZMODEM sending (adat01.txt) without deletion:  
file DORIAN DATA\adat01.txt [.....] sent done.
```

Prozentuale Fortschrittsanzeige.

## Konfiguration

### Benutzung eines Modem

Die Software sollte eine hohe Geschwindigkeit eingestellt werden. Moderne Modem unterstützen 115200 bps und arbeiten zuverlässig mit 57600 bps oder auch 38400bps. Bei sehr niedriger Geschwindigkeit kann es zu Kommunikationsproblemen kommen z.B. Zeitüberschreitungen.

### Leitungssignale

Ein externes Modem sollte über ein gut ausgestattetes Kabel mit dem Computer verbunden werden. Das Kabel sollte dabei folgende Leitungen unterstützen: RX, TX, RTS, CTS, DTR, DSR, DCD (RLSD), RI.

### Black list

Wenn ein Modem, nach einer Serie von Wahlversuchen, einen Teilnehmer nicht erreicht, wird diese Telefonnummer in einer Liste aufgenommen und nicht mehr angewählt. Wenn das passiert hilft die Stromversorgung am Modem für ca. 3 bis 5 Sekunden abzuschalten und wieder einzuschalten.

### Wahlwiederholung

Ein Modem kann so Vorkonfiguriert sein, dass eine Wahlwiederholung erst nach einer gewissen Pause möglich ist. Das ist spezifisch für manche Regionen der Erde. Die Pause kann z.B. 1 Minute betragen.

## Direkte Verbindung

### Seriellles Kabel

Für die direkte Verbindung von Computer zu Computer wird ein seriellles Kabel genutzt. Das Kabel sollte dabei folgende Leitungen unterstützen: RX, TX, RTS, CTS, DTR, DSR. Die Signale CTS und DSR müssen vom entfernten PC kommen. Die Signale RI und DCD sollten auf beiden Seiten kurz geschlossen sein. Ein voll belegtes "NULL-Modem" Kabel ist in der Regel ausreichend.

		<u>DB-9</u>	<u>DB-25</u>
RX	<-----> TX	3	2
TX	<-----> RX	2	3
RTS	<-----> CTS	8	5
DTR	<-----> DSR	6	6
CTS	<-----> RTS	7	4
DSR	<-----> DTR	4	20
GND	<-----> GND	5	7
DCD	<- -> DCD	1	8
RI	<- -> RI	9	22

Die Leitung DCD kann lokal mit DSR verbunden werden. Um falsche Interrupts aus zuschliessen, kann die Leitung RI mit DSR oder DTR lokal verbunden werden.

## Installation

Bitte befolgen Sie die nachfolgende Anweisungen um diese Software zu Installieren.

Sie dürfen nur so viele Instanzen (Kopien) der Software installieren oder ausführen wie die Anzahl der unterschiedlichen Seriennummern ist.

### Windows

In das Anwendungs-Verzeichnis einfach folgende Dateien hineinkopieren:

- zmc.exe
- lic.txt <-- *hier die Seriennummer einfügen*
- supercom.dll (wenn enthalten)
- protocol.dll (wenn enthalten)

### Linux

In das Anwendungs-Verzeichnis einfach folgende Dateien hineinkopieren:

- zmc
- lic.txt <-- *hier die Seriennummer einfügen*

### DOS

In das Anwendungs-Verzeichnis einfach folgende Dateien hineinkopieren:

- zmc.exe
- lic.txt <-- *hier die Seriennummer einfügen*

### Lizenzierung

Bitte achten Sie darauf in jeder installierten ZMCS Anwendung eine unterschiedliche Seriennummer einzustellen.

(Mehr dazu auf nächste Seite)

## Installation und Lizenzierung

Die Datei `lic.txt` muß in Ihrer Installation kopiert und ausgefüllt werden. In die erste Zeile dieser Datei tragen Sie bitte Ihre Seriennummer ein. Jede Seriennummer kann nur einmal genutzt werden. Jeder PC, der diese Software nutzt, benötigt eine eigene unterschiedliche Seriennummer.

Sobald Sie weitere Seriennummern benötigen kontaktieren Sie uns einfach. Gerne bieten wir Ihnen kostengünstige Mehr-Lizenz Pakete.

Nutzen Sie mehrere Lizenzen z.B. seriell und TCP/IP, dann fügen Sie einfach eine weitere Zeile dazu z.B.

```
SN:SC1234seriell1567  
SN:SC1234tcpip567
```

(c) ADONTEC LTD. All rights reserved.  
[www.adontec.com](http://www.adontec.com)

Achten Sie bitte darauf, dass jede Zeile durch drücken der Taste [ENTER] abgeschlossen ist.