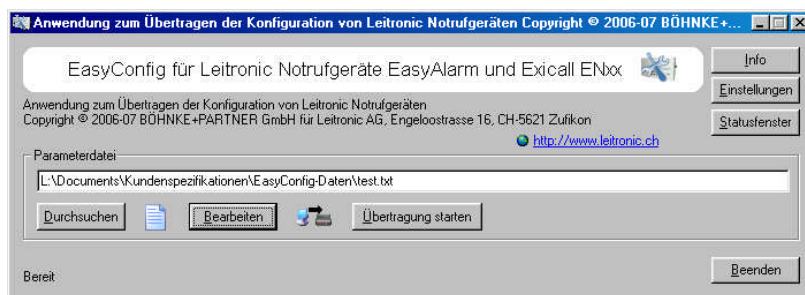


REMOTE-ACCESS-LÖSUNG

EasyConfig



Inhaltsverzeichnis

1. Voraussetzung	3
1.1 Hardware	3
1.2 Software	3
2. Funktion	4
2.1 Text-File bearbeiten	4
2.2 Übertragung starten	5
Neuerungen in der Anwendung EasyConfig	6
3. Einführung in Neuerungen.....	7
3.1 Beschreibung	7
3.2 Anrufe entgegennehmen	7
3.3 Übertragen einer Batchdatei an mehrere Empfänger (EasyBat)	7
3.4 Auswahl der Empfänger aus einer Liste.....	7
3.5 Definieren von Bedingungen in einer Konfigurationsdatei.	8
3.6 Befehl für Lesezugriff:	8
3.7 Befehl für Schreibzugriffe:	8
3.8 Variablen, Operanden und Strings:.....	8
3.9 Sprungbefehle (GOTO, GOSUB und RETURN)	9
3.10 Bedingungen (IF, THEN, ELSE und ENDIF)	10
4. Beispiele.....	12
4.1 Rufnummern bearbeiten (3 oder 9 Rufnummern).....	12
4.2 Ansage-Sprache auswählen (Sprachchip: ein-, zwei- oder vier-sprachig).....	12
4.3 Kontrollanruf auf WinMOS®300 aktivieren.....	13
4.4 Batch-File Protokoll P100 deaktivieren	13

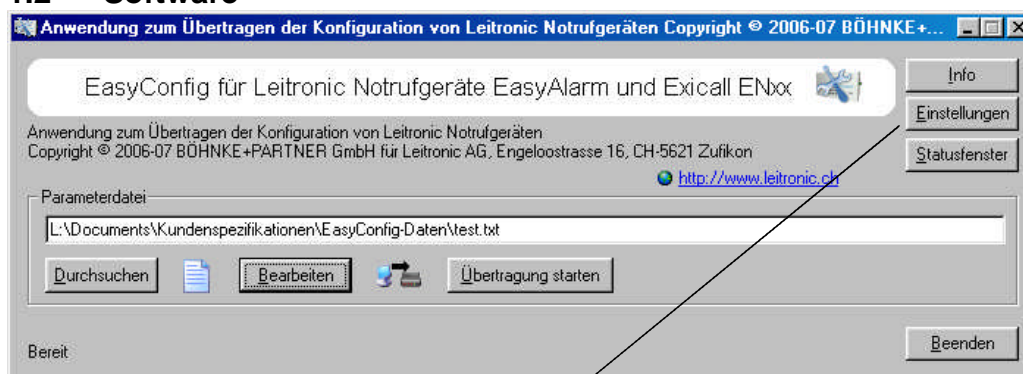
1. Voraussetzung

1.1 Hardware

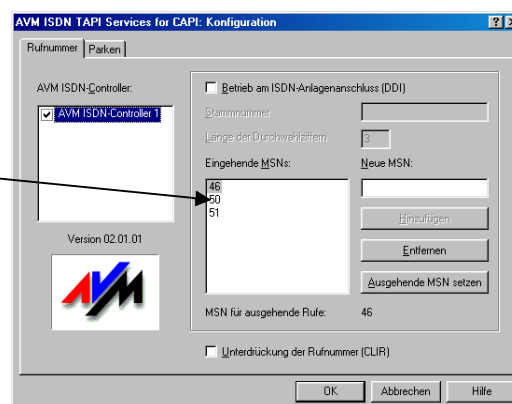
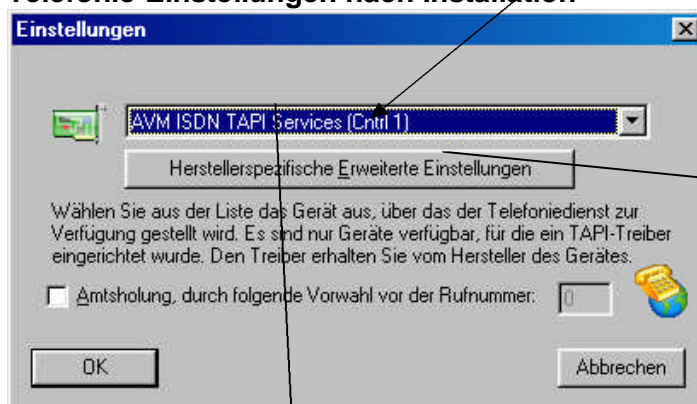
Windows PC mit TAPI fähiger ISDN-Karte (Vorzugsweise AVM FRITZ!Card => www.avm.de)



1.2 Software



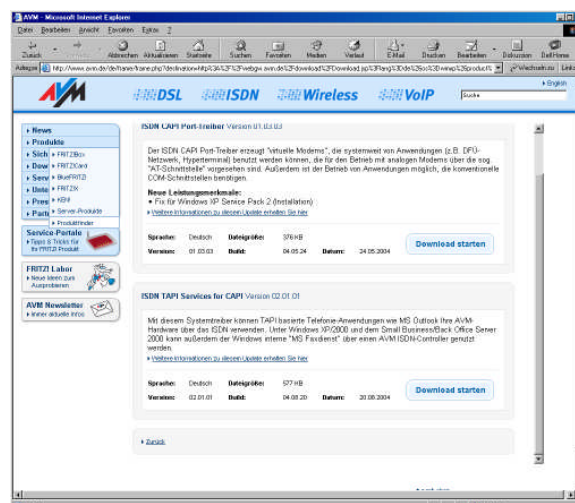
Telefonie-Einstellungen nach Installation



Abgangsseitige MSN-Nummer(n) eintragen

Wichtiger Hinweis: Telefoniedienst (TAPI)

AVM ISDN TAPI Services (Cntrl1) TAPI auswählen
Falls nicht vorhanden diesen Treiber direkt unter www.avm.de herunterladen und installieren!



2. Funktion

2.1 Text-File bearbeiten

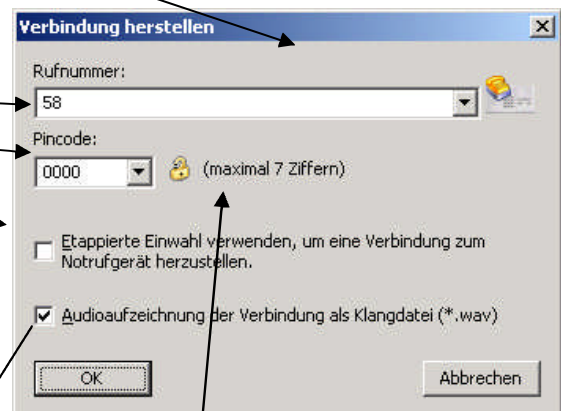
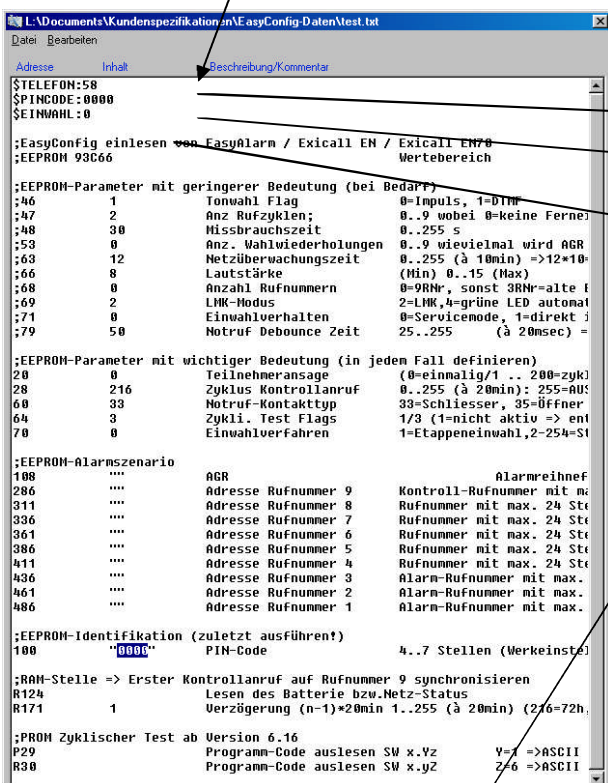
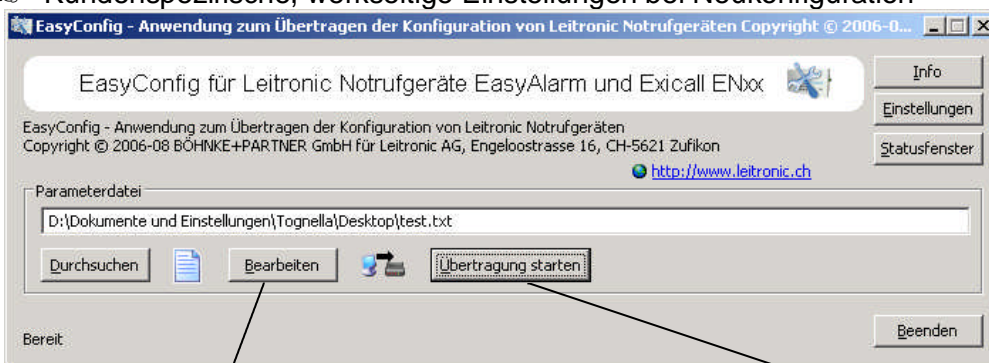
Mit Hilfe der Einträge eines Text-Files können

- A) EEPROM-Stellen (lesen / schreiben)
- B) RAM-Stellen (lesen / schreiben)
- C) PROM-Stellen (nur lesen)

eines EasyAlarm / Exicall definiert werden.

Zweck:

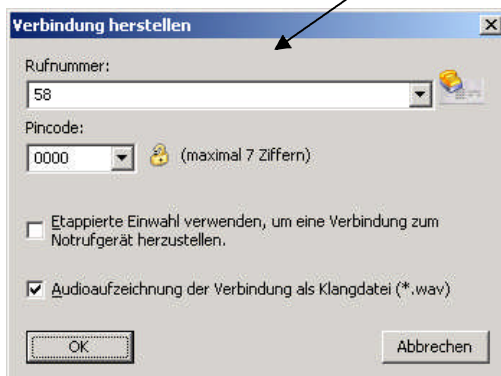
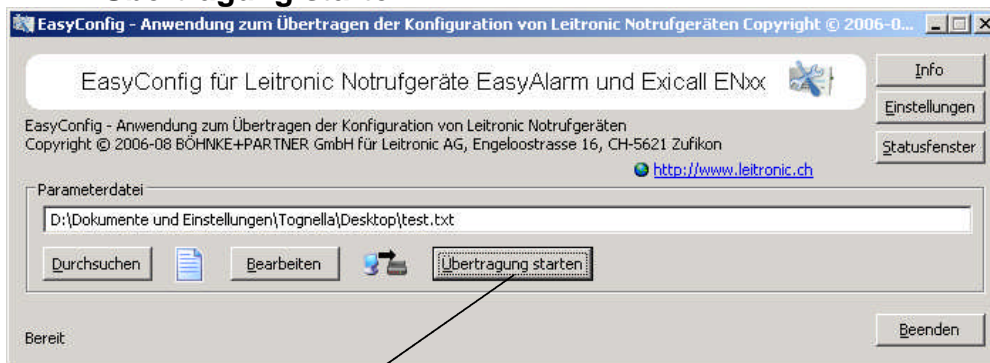
- ☞ Fernwartung bestehender Anlagen
- ☞ Kundenspezifische, werkseitige Einstellungen bei Neukonfiguration



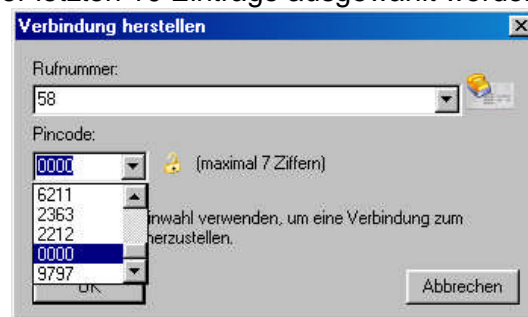
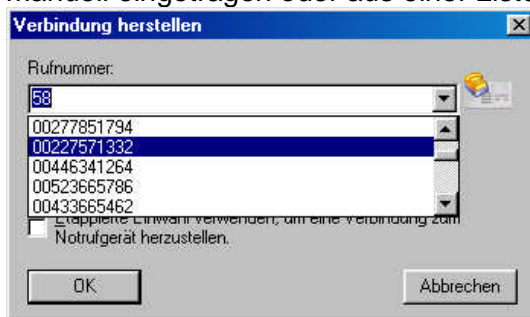
Die Einwahlnummer, der PIN-Code und das Einwahlverhalten sind Bestandteil des Text-Files!

Es besteht die Möglichkeit die Audioaufzeichnung als WAVE-File zu speichern
Speicherort: Eigene Dateien\EasyConfig\Waves
Syntax: Out_<Rufnummer><PIN>.wav

2.2 Übertragung starten

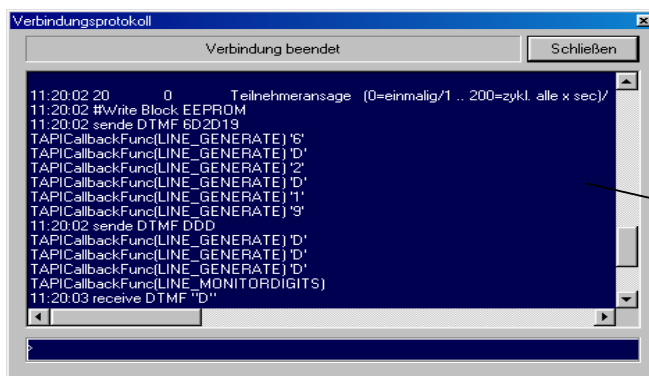


Falls die Einwahlnummer, der PIN-Code und das Einwahlverhalten im Textfile definiert sind, werden diese zu den Default-Einstellungen. Die Einwahlnummer bzw. der PIN-Code kann auch manuell eingetragen oder aus einer Liste der letzten 10 Einträge ausgewählt werden.



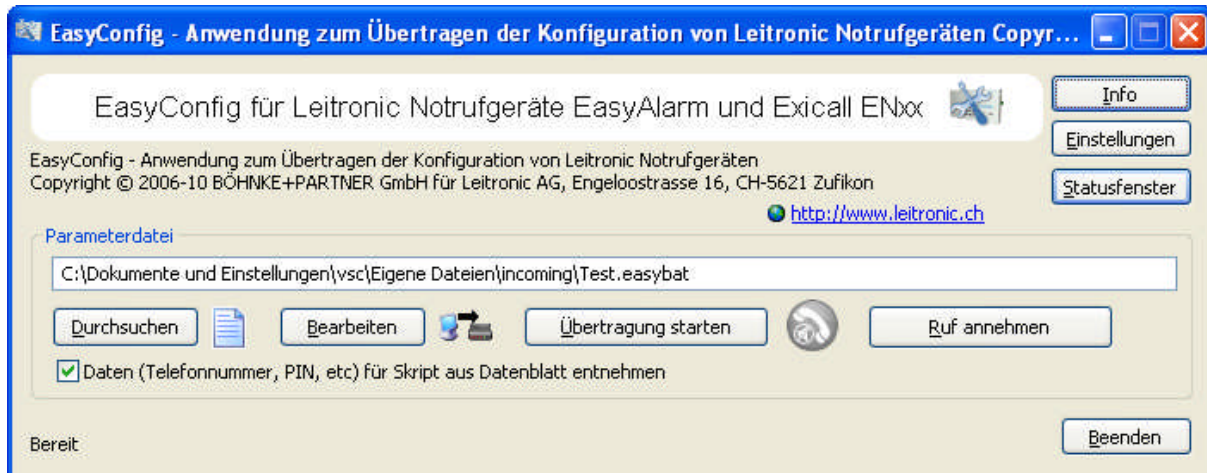
Während der Übertragung ist im Statusfenster der Protokollaustausch ersichtlich.

☞ Debug-Möglichkeit



Neuerungen in der Anwendung EasyConfig

Vsc/Stand Juli 2010



3. Einführung in Neuerungen

3.1 Beschreibung

Zu den Funktionen, die Sie bereits aus der vorherigen Version von EasyConfig kannten, sind weitere hinzugekommen. Eine kurze Beschreibung der neuen Funktionen entnehmen Sie bitte dieser Dokumentation. Folgende Funktionen sind neu hinzugekommen:

- Anrufe entgegennehmen
- Übertragen einer Batchdatei an mehrere Empfänger
- Auswahl der Empfänger aus einer Liste
- Definieren von Bedingungen in einer Konfigurationsdatei.

3.2 Anrufe entgegennehmen

EasyConfig nimmt Anrufe entgegen und überträgt, wenn gefordert, die Konfigurationsdatei an ein EasyAlarm Gerät. Die Aufforderung Anrufe entgegen zu nehmen und Daten zu übermitteln, wird durch den Knopf "Ruf annehmen" gegeben. Fehlt eine solche Aufforderung und EasyConfig wird angerufen, so werden diese Anrufe an Notruf oder EasyParrot weitergeleitet. Die Rufnummer, unter der EasyConfig zu erreichen ist, wird im Dialog "Einstellungen" im Feld "Anschlussnummer für Anrufe der Geräte" eingetragen. Im Modus "Anrufe entgegennehmen", können nur einzelne Dateien (.txt) übertragen werden, eine Übertragung von Batchdateien ist nicht möglich (=> siehe EasyBat).

3.3 Übertragen einer Batchdatei an mehrere Empfänger (EasyBat)

Neben der Übertragung einer Konfiguration an einen Empfänger ist es nun möglich, eine Konfigurationsdatei an mehrere Teilnehmer zu senden. Dazu wurde neben dem bereits vorhandenen

einfachen Textformat ein neues Dateiformat eingerichtet, in das mehrere Teilnehmer mit ihrer Telefonnummer, mit dem PIN des Gerätes und der Art des Einwahlverfahrens eingetragen werden können. Zusätzlich zu diesen Daten muss ein Dateiname angegeben werden, unter der später die veränderte Konfigurationsdatei für diesen Teilnehmer abgespeichert werden soll. Das neue Dateiformat nennt sich EasyBat und hat die Dateiendung ".easybat". Der Aufbau des neuen Dateiformats entspricht dem bereits bekannten Textformat:

Die einzelnen Teilnehmer werden durch ein "===" (mindestens drei "="-Zeichen!) voneinander getrennt (siehe Beispiel). Danach kommt das eigentliche Skript. Die EasyBat Datei bleibt während der gesamten Übertragung unverändert. Die neue Konfigurationsdatei wird in das gleiche Verzeichnis gespeichert in dem sich die EasyBat Datei befindet. EasyBat Dateien können wie alle Textdateien in jedem Editor, so auch in dem integrierten Editor von EasyConfig, geöffnet und bearbeitet werden.

3.4 Auswahl der Empfänger aus einer Liste.

Als zweite Möglichkeit, eine Konfiguration an mehrere Teilnehmer zu senden, bietet EasyConfig die Option "Daten wie Telefonnummer, PIN. etc für Skript aus Datenblatt entnehmen", die mit dem gleichnamigen Kästchen aktiviert werden kann, an. Dabei wird entweder ein Aufzug oder mehrere Aufzüge aus einer Liste ausgewählt, die dann nacheinander angerufen werden. Die Daten für die Kommunikation mit den EasyAlarm Geräten (wie Telefonnummer, PIN etc) werden aus der Datenbank gelesen.

3.5 Definieren von Bedingungen in einer Konfigurationsdatei.

Eine Neuerung in EasyConfig ist, dass mit einer Basic ähnlichen Programmiersprache Bedingungen in einer Konfigurationsdatei definiert werden können. Damit ist eine Konfigurationsdatei nicht mehr länger nur auf eine einzige Geräteversion beschränkt, sondern die Version kann am Gerät abgefragt werden um entsprechende Konfigurationsschritte auszuführen. Näheres zu diesem neuen Feature, entnehmen Sie bitte dem nächsten Kapitel.

IF, THEN, ELSE - Bedingungen in einer Konfigurationsdatei.

Zusätzlich zu den bisherigen Parameter für die Einstellung und Abfrage eines EasyAlarm Geräts, bietet EasyConfig nun auch die Möglichkeit, Teilbereiche des Skripts unter bestimmten Voraussetzungen auszuführen. So kann beispielsweise in einem Skript die Übertragung eines bestimmten Blocks so gesteuert werden, dass dieser nur ausgeführt wird, wenn die Versionsnummer mit einem Wert übereinstimmt. Solche Vergleiche werden durch den in EasyConfig integrierten Interpreter durchgeführt, der die einzelnen Befehle analysiert und ausführt. Da Befehle erst zur Zeit der Übertragung analysiert werden, erkennt EasyConfig Fehler auch erst zu diesem Zeitpunkt was zum sofortigen Abbruch der Übertragung führt! Daher sollte vor Übertragung das gesamte Skript auf Fehler untersucht werden!

Dem Benutzer stehen die folgenden 6 Gruppen an Befehlen zur Verfügung:

- Befehl für Lesezugriff
- Befehl für Schreibzugriff
- Variablen, Operanden und Strings
- Sprungbefehle (Sprungmarke, GOTO, GOSUB und RETURN)
- Bedingungen (IF, THEN, ELSE und ENDIF)
- Kommentare

3.6 Befehl für Lesezugriff:

Aufbau:¹

<Adresse>[TAB][TAB]<Kommentar>

<Adresse>	Speicherstelle, die am EasyAlarm Gerät ausgelesen werden soll.
[TAB]	Entspricht der Taste "TAB" auf der Tastatur.
<Kommentar>	Beschreibung welche Speicherstelle angesprochen wird.

3.7 Befehl für Schreibzugriffe:

Aufbau:

<Adresse>[TAB]<Wert>[TAB]<Kommentar>

<Adresse>	Speicherstelle auf die <Wert> gesetzt werden soll. Wert kann dabei auch eine Variable sein (siehe Variable).
<Kommentar>:	Beschreibung.

3.8 Variablen, Operanden und Strings:

Aufbau Variablen:

\$<Name der Variable>=<Wert>

¹ < > bezeichnet ein Teil des gesamten Befehls, der unbedingt angegeben werden muss
[] sind Tastatureingaben.

\$	Variablen sind immer mit einem \$ als Präfix gekennzeichnet.
<Name der Variablen>	Bestimmt den Namen der Variable. Das erste Zeichen muss ein Buchstabe sein, alle folgenden Zeichen können aus Buchstaben, Zahlen und dem "_" – Zeichen bestehen. (Gültig ist z.B.: \$var1, \$var_1, \$var1_ und \$var). Variablen speichern Werte, mit denen dann ein Vergleich (mit if) durchgeführt werden kann.
<Wert>	Bestimmt die Adresse, dessen Wert aus dem EasyAlarm Gerät ausgelesen werden soll und in diese Variable gespeichert wird.

Operanden:

=<>

Operanden sind bei Vergleichen wichtig (siehe Befehl „if“). Dabei haben diese folgende Bedeutung:

< kleiner, > größer, <= kleiner gleich, >= größer gleich, <> ungleich und = gleich.

Strings:

Strings sind durch zwei " - Zeichen gekennzeichnet und können alle Buchstaben, Zahlen und Sonderzeichen (,.-?) enthalten.

3.9 Sprungbefehle (GOTO, GOSUB und RETURN)

Führen einen direkten Sprung in der Konfigurationsdatei aus. Sprung bedeutet, an dieser Stelle in der Konfigurationsdatei wird die Ausführung fortgesetzt.

Aufbau Sprungmarke:

<Name der Sprungmarke>:	Das ":"-Zeichen ist Bestandteil einer Sprungmarke. Eine Sprung-marke bestimmt eine Stelle im Skript, an der die Ausführung fortgesetzt werden soll. Sprungmarken werden außerdem dazu benutzt, immer wiederkehrende Aufgaben zu definieren, die dann alle unter einem Namen ausgeführt werden (siehe Befehl GOSUB).
-------------------------	---

GOTO <Name der Sprungmarke>	Der Name der Sprungmarke ist hier ohne ":"-Zeichen! Unbedingter und sofortiger Sprung an die Stelle (<Name der Sprungmarke>) im Skript. Die Ausführung des Skripts wird an dieser Stelle fortgeführt.
-----------------------------	--

GOSUB <Name der Sprungmarke>	Der Name der Sprungmarke ist hier ohne ":"-Zeichen. Auch hier wird an die Stelle im Skript gesprungen die als Sprungmarke definiert ist. Die Ausführung des Skripts wird an dieser stelle fortgeführt. GOSUB leitet sich von den beiden Wörtern GO und SUB ab. SUB ist das Kürzel für Subroutine. In einer Subroutine können immer wiederkehrende Aufgaben unter einem Namen definiert werden, die nach dem Sprung ausgeführt werden. Dabei merkt sich der Interpreter die Stelle, an der GOSUB aufgerufen wurde, um nach Beendigung der Subroutine (siehe RETURN) an die Stelle nach dem GOSUB zu springen. Nach dem Rücksprung wird das Skript an dieser Stelle fortgesetzt. Eine Subroutine kann dabei
------------------------------	--

RETURN

wieder alle Befehle der 6 Gruppen beinhalten (auch ein erneutes GOSUB). Das Ende einer Subroutine wird durch RETURN gekennzeichnet.

Steht allein und markiert das Ende der Subroutine. Die Ausführung des Skripts wird an der Stelle fortgeführt, an der zuletzt GOSUB aufgerufen wird. Dadurch sind verschachtelte GOSUB möglich (in einer Subroutine kann wieder ein GOSUB aufgerufen werden).

3.10 Bedingungen (IF, THEN, ELSE und ENDIF)

Schon mehrmals wurde davon gesprochen, Bereiche eines Skripts nur dann auszuführen, wenn eine Bedingung erfüllt ist. Diese Möglichkeit ist durch ein oder mehrere ineinander geschachtelten if möglich. Der Bereich, der ausgeführt werden soll befindet sich dann zwischen einem IF und einem ENDIF (oder ELSE). Diesen Bereich nennen wir hier IF-Block. Zwischen dem IF und dem ENDIF kann außerdem ein alternativer Block definiert werden, der dann ausgeführt wird, wenn der Vergleich negativ ausgewertet wird. Diesen Bereich nennen wir ELSE - Block. Somit bestehen Bedingungen aus max. zwei Bereichen: IF - ELSE, ELSE - ENDIF.

Wird der Vergleich positiv ausgewertet, wird der Block "IF" bis "ELSE" ausgewertet, ansonsten der Block "ELSE" bis "ENDIF".

Aufbau:²

```
IF <Vergleich> THEN
    <hier folgen die Anweisungen, die bei einem positiven Vergleich ausgeführt werden
sollen.>
{ELSE
    <bei einem negativen Vergleich werden die Anweisungen aus diesem Bereich
ausgeführt.>}
ENDIF
```

<Vergleich> Zahlen und Variablen können miteinander verglichen werden. Dazu können die Vergleichsoperanden kleiner (<), größer (>), kleiner gleich (<=), größer gleich (>=), ungleich (<>) und gleich (=) benutzt werden. Auch wenn es wenig Sinn macht, können auch Zahlen miteinander verglichen werden (1=1). Ein Vergleich könnte so aussehen: \$var1=9. Dabei wird der Inhalt der Variable \$var1 mit dem Zahlenwert (nicht mit der Adresse!) 9 verglichen. Dabei muss die Variable vorher einen Wert zugewiesen bekommen haben, da ansonsten die Übertragung mit einem Fehler unterbrochen wird!

Es existieren drei verschiedene Definitionen eines IFs:

IF <Vergleich> THEN <Anweisung>

IF <Vergleich> THEN <Anweisung>

² < > bezeichnet ein Teil des gesamten Befehls, der unbedingt angegeben werden muss
[] sind Tastatureingaben.
{ } kennzeichnen Optionen, die nicht unbedingt im Skript vorhanden sein müssen.

ELSE <Anweisung>

```
IF <Vergleich> THEN
  <Anweisung>
ENDIF
```

```
IF <Vergleich> THEN
  <Anweisung>
ELSE
  <Anweisung>
ENDIF
```

Die erste Definition zeigt das nicht unbedingt nach einem IF, ein ELSE - Block oder ein ENDIF folgen muss. Ist der <Vergleich> positiv, so wird die <Anweisung> direkt hinter THEN ausgewertet, andernfalls geht die Programmausführung direkt unterhalb des ifs weiter. Folgt nach dem THEN eine Anweisung, ohne eine neue Zeile zu beginnen, muss kein ENDIF folgen.

Das ENDIF kann auch dann entfallen, wenn die Anweisung in einer Zeile mit dem ELSE steht, wie in dem zweiten Beispiel zu sehen ist.

Sollen nach einem positiven Vergleich mehrere Anweisungen ausgeführt werden, so muss dies in einem Block (IF-ENDIF) geschehen (siehe Beispiel 3).

Mehrere Anweisungen können, wie beim THEN - Block auch in einem ELSE - Block definiert werden. Wichtig ist, dass Blöcke immer mit einem ENDIF beendet werden (siehe viertes if Beispiel)

Kommentare

Kommentare erleichtern die Lesbarkeit des Skripts. Sie beschreiben, was das Programm leistet. Es existieren zwei Arten von Kommentaren:

;Kommentar

und

<Anweisung>[TAB]Kommentar.

Ohne eine Anweisung, kann ein Kommentar mit ";" - Zeichen definiert werden. Hinter einer Schreib- und Leseanweisung kann ein Kommentar, getrennt durch TAB, angegeben werden.

4. Beispiele

4.1 Rufnummern bearbeiten (3 oder 9 Rufnummern)

```
;Skript Rufnummern bearbeiten
;Autor: Tog
;Datum: 28.7.10
$Rufnummern=68 0=9RNR, sonst 3RNR.
if $Rufnummern=0 then
  ;EEPROM 93C66 EA-40, EA-8, Exicall ENxx
  ;286 " " Adresse Rufnummer 9
  ;311 " " Adresse Rufnummer 8
  ;336 " " Adresse Rufnummer 7
  ;361 " " Adresse Rufnummer 6
  ;386 " " Adresse Rufnummer 5
  ;411 " " Adresse Rufnummer 4
  ;436 " " Adresse Rufnummer 3
  ;461 " " Adresse Rufnummer 2
  ;486 " " Adresse Rufnummer 1
else
  ;EEPROM 93C56 EA-4
  ;180 " " Adresse Rufnummer 3
  ;205 " " Adresse Rufnummer 2
  ;230 " " Adresse Rufnummer 1
endif
```

4.2 Ansage-Sprache auswählen (Sprachchip: ein-, zwei- oder vier-sprachig)

```
;Skript Sprache auswählen
;Autor: Tog
;Datum: 28.7.10
$Version=0
if $Version>26 then
  ;ISD400x
  $Sprache=33
  if $Sprache>=16 then
    ;3316 ;1. Sprache (DE) mit ISD4003
    ;3317 ;2. Sprache (FR) mit ISD4003
    ;3318 ;3. Sprache (GB) mit ISD4003
    ;3319 ;4. Sprache (IT) mit ISD4003
  else
    ;330 ;1. Sprache (DE) mit ISD4002
    ;3310 ;2. Sprache (FR) mit ISD4002
  endif
else
  ;ISD2560 => EA-4(0) => Sprache nicht umschaltbar!
endif
```

4.3 Kontrollanruf auf WinMOS®300 aktivieren

```
;Skript Kontrollanruf auf WinMOS aktivieren
$Version=0
$Rufnummern=68
if $Version<16 then
    ;Kontrollrufe WinMOS erst ab Version 6.16
else
    64 3
    28 218
    R171 1
    if $Rufnummern=0 then
        ;EA-40, EA-8, Exicall ENxx
        286 " " Adresse Rufnummer 9
    else
        ;EA-4
        180 " " Adresse Rufnummer 3
        108 "12" Rufnummer-Reihenfolge 12 (ohne 3)
    endif
endif
endif
```

4.4 Batch-File Protokoll P100 deaktivieren

Der Header des Batch-Files kann anhand der Excel-Tabelle generiert werden

\$TELEFON:	\$PINCODE:	\$EINWAHL:	\$DATEINAME:	
00719502755	1234	0	00719502755_1234.txt	\$TELEFON:00719502755 \$PINCODE:1234 \$EINWAHL:0 \$DATEINAME:00719502755_1234.txt =====
00434952160	0000	0	00434952160_0000.txt	\$TELEFON:00434952160 \$PINCODE:0000 \$EINWAHL:0 \$DATEINAME:00434952160_0000.txt =====
		0	_txt	\$TELEFON: \$PINCODE: \$EINWAHL:0 \$DATEINAME:_txt =====
		0	_txt	\$TELEFON: \$PINCODE: \$EINWAHL:0 \$DATEINAME:_txt =====
		0	_txt	\$TELEFON: \$PINCODE: \$EINWAHL:0 \$DATEINAME:_txt =====
		0	_txt	\$TELEFON: \$PINCODE: \$EINWAHL:0 \$DATEINAME:_txt =====
		0	_txt	\$TELEFON: \$PINCODE: \$EINWAHL:0 \$DATEINAME:_txt =====

```
$TELEFON:00719502755
$PINCODE:1234
$EINWAHL:0
$DATEINAME:00719502755_1234.txt
=====
$TELEFON:00434952160
$PINCODE:0000
$EINWAHL:0
$DATEINAME:00434952160_0000.txt
=====
$Protokoll=30
if $Protokoll=22 then
    30 6 P100 ausschalten => Protokoll-Register 30 auf 6 setzen
endif
R171 1 Kontrollanruf starten
```