

## Installationsanleitung für die TAP-Entwicklungsumgebungen

Erstellt und getestet von TV-Junkie und FireBird. Danke an Jag für die Kontrolle der ersten Version.

### **Vorwort**

**TAP** steht für **Topfield Application Program**. Hierbei handelt es sich um kleine Programme, die direkt auf den Festplattenreovern von Topfield ausgeführt werden können. Da der TF4000PVR und der TF5x00PVR/TF6x00PVR(t) sowie der TMS (bisher nur der SRP-2100) unterschiedliche Prozessoren haben, sind die Programme nicht zwischen den Receivern kompatibel.

Diese Anleitung behandelt nur die Installation der Kompilierungsumgebung für den TF5x00PVR/6x00PVR+(t) und TMS unter Windows, und es werden diverse Programme zur Programmierung der TAPs vorgestellt. Der Einfachheit halber werden die TF5x00PVR/6x00PVR +(t), hier nur TF5000 genannt. Um mit dieser Anleitung TAPs erstellen zu können, müssen einige Voraussetzungen erfüllt sein. Diese sind im Wesentlichen folgende:

- Einen PC mit mindestens Windows 2000
- Administratorrechte auf dem PC (zur Softwareinstallation)
- Internetverbindung (zum Download der Software)
- Gute Kenntnisse über den Umgang mit Windows
- Mindestens Grundkenntnisse der Programmiersprache C

Da der Zugriff auf die Ordner „C:\Programme“ bzw. „C:\Program Files“ unter Microsoft Vista Probleme verursachen kann, wird auf die Installation der Kompilierungsumgebung dorthin verzichtet. Dies ist zwar nicht die sauberste Methode, da die Kompilierungsumgebung jedoch relativ viel Handarbeit benötigt, ist dies ein akzeptabler Kompromiss. Grundsätzlich ist die Wahl der Pfade natürlich frei, so dass die hier erwähnten Pfade nach Belieben geändert werden können. Nur muss darauf geachtet werden dies konsequent zu tun.

Ein Hinweis zum Entpacken der Pakete. Die meisten Entpacker wie [Winzip](#) (Shareware), [Winrar](#) (Shareware) oder [7-Zip](#) (kostenlos) bieten die Möglichkeit direkt per Kontextmenü die gepackten Archive über die Option „hier entpacken“ oder „entpacken nach hier“ zu entpacken. Hier in dieser Anleitung wird explizit darauf hingewiesen, wie die einzelnen Pakete ausgepackt werden!

Ein weiterer Hinweis: Alle benötigten Dateien werden aus dem Ursprungsordner „D:\Topfield“ verschoben. Kopieren ist natürlich auch möglich und kann sinnvoll sein, wenn es sich um Ordner mit mehreren tausend Dateien handelt. Prinzipiell funktioniert es aber auch mit dem Verschieben.

## Vorbereitung

Zuerst sollten einige Ordner angelegt werden. Wie bereits oben geschrieben, können die Pfade frei gewählt werden. In dieser Anleitung werden die Verzeichnisse, welche verwendet und auch getestet wurden, **dunkelgrün** markiert.

1. C:\Work\cygwin\opt für die „Linux“-Umgebung unter Windows
2. C:\Work\gccForTF5000\usr für den TF5000 Compiler
3. C:\Work\API\TF5000 für die TF5000 API
4. C:\Work\API\TMS für die TMS API
5. D:\Work\TAPs Hier liegen die eigenen Quelldateien
6. D:\Work\TAPs\SamplesTF5000 In den beiden Samples Ordner werden die Topfield-  
Beispieldateien für TF5000 und TMS gespeichert, da sie  
zueinander nicht kompatibel sind (gleicher Dateinamen, aber  
unterschiedliche API).
8. D:\Work\TAPs\SamplesTMS
  
9. D:\Topfield In diesen Ordner wird alles gespeichert und ausgepackt was  
aus verschiedenen Quellen kommt, z.B. Internet, um es später  
in den richtigen Ordner zu kopieren. Nach der Fertigstellung  
der Installation kann der Inhalt gelöscht oder an einer  
anderen Stelle abgelegt werden.

## Die benötigte Software

### 1. [Cygwin](#)

Cygwin ist die Umgebung unter der der gccForTAP später ausgeführt wird. Sie bietet dem Compiler unter Windows eine Linux-artige Umgebung an. Eine ausführliche deutsche Beschreibung von Cygwin findet sich z.B. in der [Wikipedia](#).

### 2. [gccForTAP](#)

gccForTAP ist der Compiler um den C-Quelltext in Maschinencode zu übersetzen. Aus lizenzrechtlichen Gründen darf Topfield diesen Compiler für die Erstellung der TF5000 TAPs nicht mehr selber anbieten. Er kann aber über Google gefunden werden. Die ZIP-Datei wird unter [D:\Topfield](#) gespeichert.

### 3. [TAP-API für TF5000](#)

Die TAP-API ist die Beschreibung der Schnittstelle zwischen den TAPs und dem Topfield. Die TAP-API bekommt man bei [Topfield](#) zum Download. Die aktuelle Version heißt „TAP API\_ver1.22\_Samples\_Uilities\_2005June03.zip“. Diese ZIP-Datei wird unter [„D:\Topfield“](#) gespeichert.

### 4. [FireBirdLib](#)

Bei der FireBirdLib handelt es sich um Routinen und Firmware-Hacks. Diese sind für den TF5000 und den TMS geeignet. Die FireBirdLib.zip wird in den Ordner [„D:\Topfield“](#) gespeichert und mit „Entpacken nach FireBirdLib“ entpackt. In dem Ordner „FireBirdLib“ befinden sich einige Ordner und Dateien, unter anderem auch die Dokumentation in Deutsch und Englisch. Diese kann in den Ordner [„D:\Work\TAPs“](#) kopiert werden. Bevor die FireBirdLib eingesetzt wird, sollte diese Dokumentation intensiv gelesen und auch verstanden worden sein. Die FireBirdLib greift sehr tief in die Firmware ein. Wenn man hier nicht genau weiß was man tut, kann dies sehr schwerwiegende Konsequenzen nach sich ziehen! Für die TAP-Erstellung auf dem TF5000 werden nur die Dateien „libFireBird.a“, „libFireBird.h“, „libFireBird\_generic.h“ und „libFireBird\_TF5000.h“ benötigt. Diese werden in den Ordner [„C:\Work\Api\TF5000“](#) kopiert. In den restlichen Ordnern und Dateien befindet sich der Quellcode der FireBirdLib. Dieser wird zur Erstellung eigener TAPs nicht benötigt.

Für die TAP-Entwicklung auf dem TMS werden die Dateien „libFireBird\_TMS.a“, „libFireBird.h“, „libFireBird\_generic.h“ und „libFireBird\_TMS.h“ benötigt. Die .h-Dateien werden in das Verzeichnis [„C:\Work\Api\TMS\include“](#) kopiert, „libFireBird\_TMS.a“ wird nach [„C:\Work\cygwin\opt\crosstool\lib“](#) befördert und danach auf „libFireBird.a“ umbenannt.

### 5. [crosstool\\_cywin.tar.gz](#)

Die Datei wird unter [„D:\Topfield“](#) gespeichert. Sie enthält den Compiler, um den C-Quelltext in Maschinencode zu übersetzen.

### 6. [TAP-API für TMS](#)

Die TAP-API ist die Beschreibung der Schnittstelle zwischen den TAPs und dem Topfield TMS. In der zu heruntergeladenen Datei befinden sich außerdem die Beispieldateien und die Dokumentation der TAP Operatoren. Die Datei [TAP Examples for TMS.zip](#) wird unter [„D:\Topfield“](#) gespeichert.

### 7. [JailBreak für TAP TMS](#)

In der Firmware des TMS befindet sich eine Liste mit verbotenen Linux-Systembefehlen. Verwendet ein TAP einen dieser Befehle, unterbindet der TMS den Start des TAPs. Diese Prüfung wird mit dem JailBreak ausgehebelt.

Achtung: Im Normalfall reichen die Befehle der API aus. Es sollte also dringend der Readme.txt in dem ausgepackten JailBreak Ordner gelesen werden.

Die JailBreak.zip wird von [hier](#) heruntergeladen und im Ordner [D:/Topfield](#) gespeichert und mit „Entpacken nach Hier“ ausgepackt.

## 8. [Happys Tapcheck](#)

Auf Grund eines Bugs im Topfield-Dateisystem der TF5000-Serie, werden Dateien, die 512 Bytes oder ein ganzzahlig Vielfaches davon groß sind, unvollständig geladen. Dieses Tool sorgt dafür, dass das TAP im Bedarfsfall um 1 Byte vergrößert wird, so dass der Ladevorgang korrekt durchgeführt wird. Die „*tapcheck.zip*“ wird auch in „*D:\Topfield*“ gespeichert. Da nichts im eigentlichen Sinne installiert wird, wird *tapcheck.zip* mit „Entpacken nach *tapcheck\*“ entpackt.

## 9. Die Editoren

Grundsätzlich kann für das Erstellen von TAPs ein beliebiger Text-Editor verwendet werden. Im Zuge der Installation werden jedoch ein paar Dateien verändert, die im Unix-Format vorliegen (Zeilenende durch line feed, statt carriage return und line feed). Daher muss ein Editor verwendet werden, der mit dem Unix-Format umgehen kann und beim Abspeichern der veränderten Datei diese nicht in das DOS-Format wandelt. Der kleine [MetaPad](#) (keine Installation notwendig) kann dafür genauso verwendet werden, wie z.B. [UltraEdit](#) (kein Freeware).

Ein anderer, interessanter Editor ist [SciTE](#) (Freeware), der Syntax-Highlighting, AutoComplete und Intellisense beherrscht. Das heißt, SciTE zeigt die benötigten Parameter unterhalb der C- und TAP-Anweisungen an. Speziell diese Funktion ist sehr aufwendig einzubinden, so dass dieser Editor als Komplett-Download incl. der TAP-Operatoren von uns angeboten wird (bisher nur TF5000).

## Cygwin installieren

- Auf <http://www.cygwin.com> gehen und „Install or update now!“ anklicken
- Die „*setup.exe*“ in „*D:\Topfield*“ abspeichern und von dort aus starten →Weiter
- Zur Installation von Cygwin muss eine Internetverbindung bestehen. Der Installer lädt die für die Installation notwendigen Dateien direkt aus dem Internet herunter.
- „Install from Internet“ auswählen →Weiter
- Pfad angeben (Root Directory) „*C:\Work\cygwin*“ →Weiter
- Die Felder „All Users“ und „Unix/binary“ sollten schon ausgewählt sein, sonst auswählen
- Local Package Directory Pfad angeben „*C:\Work\cygwin\packages*“ → Weiter
- Verbindungsart wählen (zu Hause normalerweise „Direct connection“) → Weiter
- Eine URL aus der Liste auswählen → Weiter
- Select Packages. Dazu auf das + Zeichen von z.B. Devel klicken und die Pakete durch klicken auf den Kreis neben dem Wort Skip auswählen. Zusätzlich benötigte Pakete werden automatisch angewählt und sollten nicht verändert werden! Da Topfield keine Angaben über Paketversionen macht, wird die erste Version genommen, die erscheint! Bitte die folgenden Pakete auswählen:
  - Devel - autoconf
  - Devel - automake
  - Devel - bison
  - Devel - flex
  - Devel - gcc
  - Devel - gettext-devel
  - Devel - libncurses-devel
  - Devel - libtool
  - Devel - make
  - Devel - patchutils
  - Devel - subversion
  - Web - wget
- Weiter, bis die Installation beendet ist. Das kann je nach Internetverbindung und Rechnergeschwindigkeit einige Zeit dauern. Je nach Geschmack ein Desktop- oder Startmenü-Icon anlegen lassen und danach auf Fertigstellen klicken.
- „*C:\Work\cygwin\bin*“ zur Windows-Pfadvariablen hinzufügen. Dazu unter Start, Systemsteuerung, System, Erweiterte Systemeinstellungen auswählen und auf Umgebungsvariablen klicken. Dort unter Systemvariablen „Path“ herausuchen und Bearbeiten anklicken, den Pfad mit einem ; getrennt zu anderen Pfadangaben eingeben, also zum Beispiel ;*C:\Work\cygwin\bin* (wenn diese Pfadangabe ganz hinten angehängt wird) oder *C:\Work\cygwin\bin*; (wenn diese Pfadangabe ganz vorne angefügt wird). Alle Fenster mit OK bestätigen.

## TF5000 TAP Compiler installieren

- „D:\Topfield\gcc\_for\_tap.zip“ mittels „Extrahieren nach hier“ entpacken
- Das durch das Entpacken entstandene Verzeichnis „gcc\_for\_tap“ öffnen. In diesem Verzeichnis befindet sich wiederum ein Verzeichnis „gcc\_for\_tap“. Und in diesen Verzeichnis der Ordner „local“, der nach „C:\Work\gccForTF5000\usr“ verschoben wird.

- Eine DOS-Box mit Administratorrechten öffnen. Dazu auf "Start" -> "Ausführen" klicken und "cmd" eingeben.  
Unter Vista auf den Startbutton klicken und im Suchfeld "cmd" eingeben. In der Ergebnisliste erscheint daraufhin die "cmd.exe". Diese mit der rechten Maustaste anklicken und "Als Administrator ausführen" auswählen. In dieser DOS-Box dann folgenden Befehl eingeben:

```
mount C:\work\gccForTF5000\usr /usr
```

- Die Datei „D:\Topfield\TAP\_API\_ver1.22\_Samples\_Uilities\_2005June03.zip“ umbenennen in „TapTF5x.zip“, dann mit „Entpacken nach TapTF5x“ entpacken und die darin befindliche ZIP „tap\_and\_samples\_2005June03.zip“ umbenennen in „TF5000.zip“, und diese ZIP mit „Entpacken nach TF5000“ entpacken. Diese ZIP enthält die eigentliche API und die Beispieldateien.
- Im Ordner „D:\Topfield\TapTF5x\TF5000“ befinden sich Unterverzeichnisse und einzelne Dateien. Sämtliche Unterordner (z.B. Fire) können in den Ordner „D:\Work\TAPs\SamplesTF5000“ verschoben werden. Sie enthalten die von Topfield gelieferten Beispielprojekte.
- „D:\Topfield\TapTF5x\TF5000\gcc.bat“ öffnen und die folgenden Zeilen mittels Copy und Paste in die Datei kopieren und den originalen Inhalt ersetzen. Die letzten 3 Zeilen (mips-gcc...) müssen in der „gcc.bat“ in einer Zeile stehen. Die Pfade sind ggf. den eigenen Erfordernissen anzupassen.

```
set PATH=C:\Work\gccForTF5000\usr\local\bin;C:\Work\cygwin\bin
set INCLUDE1=C:\Work\API\TF5000
set INCLUDE2=C:\Work\cygwin\usr\include
set INCLUDE3=C:\Work\gccForTF5000\usr\local\include
mips-gcc.exe -W -Wall -D_TF5000_ -Wstrict-prototypes -Wmissing-prototypes -
Wpointer-arith -O2 -c -mtap -mlong-calls -msoft-float -I %INCLUDE1% -I
%INCLUDE2% -I %INCLUDE3% %1 %2 %3 %4 %5 %6 %7 %8 %9
```

- „gcc.bat“ nach „C:\Work\cygwin\bin“ verschieben
- „D:\Topfield\TapTF5x\TF5000\TAP.LD“ öffnen und die Pfade der eigenen Installation anpassen:

```
SEARCH_DIR(./)
SEARCH_DIR(..)
SEARCH_DIR(C:/Work/gccForTF5000/usr/local/lib/)
SEARCH_DIR(C:/Work/gccForTF5000/usr/local/lib/gcc-lib/mips/2.95.3/)
SEARCH_DIR(C:/Work/API/TF5000/)
```

- Die folgenden Dateien in den Ordner „C:\Work\API\TF5000“ verschieben :  
„FONT.H“, „GIF.C“, „GIF.H“, „HDD.H“, „KEY.H“, „libtap.a“, „LINE.C“, „TAP.H“, „TAP.LD“, „Type.h“ und „WIN.H“
- Die „D:\Topfield\TapTF5x\TAP(Topfield's customizing API) v1.22.pdf“ kann nach „D:\Work\TAPs“ verschoben werden. In ihr befindet sich die Dokumentation zu den TAP-API-Funktionen.
- Die Datei „ADDPATH.BAT“ löschen
- Die Datei „D:\Topfield\tapcheck.exe“ nach „C:\Work\cygwin\bin“ verschieben.

## Testen der Installation

Im Ordner „D:\Work\TAPs\SamplesTF5000“ befinden sich die Beispiel-TAPs für den TF5000. Folgend wird die Demo „fireclock“ aus dem Ordner „FIRE“ kompiliert. Davor muss jedoch die „build.bat“, die den Compiler und Linker steuert, an die gerade eingerichtete Entwicklungsumgebung angepasst werden.

Tipp 1: Um das TAP mit Tapcheck überprüfen zu lassen, sollte die „build.bat“ in jedem Projekt um den Eintrag „tapcheck.exe /f“ erweitert werden.

Tipp2: Da sich die DOS-Box jedes Mal schließt, sollte als letzter Befehl ein „Pause“ in die „build.bat“ eingefügt werden, um die Fehler- und Warnmeldungen des Compilers lesen zu können und die Ausführung von Tapcheck zu überprüfen. Wird ein Editor verwendet, der die Standardausgabe in einem eigenen Fenster anzeigt (z.B. mit UltraEdit möglich), ist der pause-Befehl nicht notwendig.

So sollte die neue „build.bat“ aussehen:

```
call gcc.bat clock.c
call gcc.bat fire.c
mips-ld -o fireclk.elf -T C:\Work\API\TF5000\TAP.ld clock.o fire.o -l tap -l c -Map
fireclk.map
mips-objcopy -O binary fireclk.elf fireclk.tap
tapcheck /f
pause
```

Beim Doppelklick auf die „build.bat“ öffnet sich ein DOS-Fenster, einige Meldungen gehen durch und im Ordner „Fire“ werden ein paar neue Dateien angelegt, darunter auch das neue „fireclk.tap“. Beim Neukompilieren brauchen diese Dateien nicht gelöscht zu werden. Diese werden automatisch überschrieben.

## TMS

### TAP Compiler installieren

- „TAP\_Examples\_for\_TMS.zip“ mittels „Extrahieren nach TAP\_Examples\_for\_TMS.zip\“ entpacken. In diesem Ordner befinden sich nun einige Ordner und Dateien. Der Ordner „include“ und die Dateien „build“, „libtap.so“, „Makefile“ und „tapinit.o“ werden nach „C:\Work\API\TMS“ verschoben.

- „C:\Work\API\TMS\include\tool.mk“ mit einem Editor öffnen und die folgende Zeile erweitern:

```
CC      = $(CROSS_COMPILE)gcc -D_TMS_
```

- „crosstool\_cygwin.tar.gz“ aus „D:\Topfield“ nach „C:\Work\cygwin\opt“ verschieben

- cygwin starten und den TMS-Compiler entpacken

```
cd /opt
tar xvf crosstool_cygwin.tar.gz      (Topfield-Compiler entpacken)
rm crosstool_cygwin.tar.gz          (Gepacktes Archiv löschen)
mkdir /tapapi                        (mount point für die API erzeugen)
exit                                  (cygwin beenden)
```

- Editor, hier bitte Metapad starten, „.bash\_profile“ aus „C:\Work\cygwin\home\“ öffnen und die folgende Zeile am Ende der Datei anfügen:

```
export PATH=$PATH:/opt/crosstool/bin
```

- Aus dem Ordner „D:\Topfield\TAP\_Examples\_for\_TMS.zip“ kann die PDF-Datei „Topfield's customizing API v0.1 for TMS.pdf“ nach „D:\Work\TAPS“ verschoben werden. In ihr befindet sich die Dokumentation zu den TAP-API-Funktionen.

- Auch die darin noch enthaltenen Unterverzeichnisse können in den Ordner „D:\Work\TAPS\SamplesTMS“ verschoben werden. Sie enthalten die von Topfield gelieferten Beispielprojekte und werden für die Funktion des TMS-Compilers nicht benötigt.

- Jetzt muss nur mehr der Pfad zur TAP-API gesetzt werden, damit die API vom Compiler gefunden wird. Der erste Schritt wurde bereits oben durch das Anlegen des leeren Verzeichnisses gesetzt (mkdir /tapapi). DOS-Box mit Admin-Rechten öffnen. Dazu als Administrator auf "Start" -> "Ausführen" klicken und "cmd" eingeben.

Unter Vista auf den Startbutton klicken und im Suchfeld "cmd" eingeben. In der Ergebnisliste erscheint daraufhin die "cmd.exe". Diese mit der rechten Maustaste anklicken und "Als Administrator ausführen" auswählen. In dieser DOS-Box dann folgenden Befehl eingeben:

```
mount C:\Work\API /tapapi
```

## Testen der Installation

Ähnlich der „*build.bat*“ bei den TF5000-Projekten, müssen alle Projekt-Makefiles angepasst und eine Zeile geändert werden. Z.B. beim Topfield-Demoprojekt „*FireClock*“ bei der Datei „*D:\Work\TAPS\SamplesTMS\FireClock\Makefile*“ die BASE-Zeile ändern:

```
BASE = $(shell cd /tapapi/TMS; pwd)
```

Der Compiler lässt von DOS aus oder per Doppelklick starten, da ansonsten immer die gleichen Befehle in der Cygwin Bash ausgeführt werden müssten. Das lässt sich einfach umgehen, in dem eine „*build\_TMS.bat*“, ähnlich „*build.bat*“, gebaut wird:

```
set PATH=%PATH%;C:\Work\cygwin\opt\crosstool\bin
del /Q bin obj
bash -i -c make
pause
```

## JailBreak installieren

JailBreak.exe modifiziert ein kompiliertes TAP so, dass es trotz eingebauter Linux-Systembefehle vom Betriebssystem des TMS gestartet wird. Die im Paket mitgelieferte ausführbare Datei läuft unter DOS und der cygwin bash. Da der Source-Code beiliegt, lässt sich JailBreak auf jedem System übersetzen. Um ein TAP zu patchen, genügt es, Jailbreak mit dem Namen des TAPs aufzurufen, z.B.:

```
JailBreak FireClock.tap
```

Da dies zu mühsam ist, lässt sich der JailBreak auch automatisieren. Eine theoretische Möglichkeit ist das Einbinden in die im vorhergehenden Punkt beschriebene „*build\_TMS.bat*“. Da diese Batch-Datei den TAP-Namen aber nicht kennt, muss man jede „*build\_TMS.bat*“ selbst anpassen. Daher ist es ratsam, den JailBreak direkt in den make-Prozess einzubinden.

Dazu sind 3 Schritte notwendig:

- JailBreak.exe nach *C:\Work\cygwin\opt\crosstool\bin* kopieren.
- Bei *C:\Work\API\TMS\include\tool.mk* wird mit einem Unix-fähigen Editor wie MetaPad hinter der AWK-Zeile die JB-Zeile angefügt, so dass der Bereich der Datei so aussieht:

```
TOUCH    = touch
AWK      = awk
JB       = JailBreak.exe

# Option for quiet builds
```

Danach Speichern und den Editor beenden.

- Bei jedem Projekt muss die Datei *Makefile* geändert werden. Hier wird diese Datei wieder mit einem Unix-fähigen Editor geöffnet, nach dem Text „Linking“ gesucht und die markierte Zeile aus diesen Block eingefügt. Die 3. Zeile ist hier gekürzt dargestellt.

```
$(TAP_APP): ${TAP_OBJS}
@echo "[Linking... $@]"
$(Q_)$ (LD) -shared --no-undefined --allow-shlib-undefined -o $@ ...
$(Q_)$ (JB) $(TAP_APP)
```

## **Eine Quelldatei für zwei Systeme**

Durch eine kleine Erweiterung der „*tool.mk*“ bzw. „*gcc.bat*“ wurden die beiden Variablen „\_TF5000\_“ und „\_TMS\_“ deklariert. Dadurch kann bereits in den Source-Dateien zwischen den beiden Compilern unterschieden werden:

```
#ifdef _TMS_
    TAP_Print("This is a TMS build\n");
#else
    TAP_Print("This is a TF5000 build\n");
#endif
```

## **Debuggen der TMS TAPs**

Da der TMS keine serielle Schnittstelle hat, ist das Debuggen einer Applikation mittels TAP\_Print() etwas anders gelöst. Dazu muss man sich via Telnet auf den ftp-Port 21 verbinden und die Debug-Session starten. Die blauen Zeilen werden in einer DOS-Box eingegeben, die schwarzen Zeilen sind die Antworten des TMS:

```
telnet 192.168.2.34 21 (die IP-Adresse muss natürlich zum eigenen TMS passen)
220 Welcome message
user guest
331 Please specify the password.
pass 0000
230 Login successful.
tap debug
200 TAP DEBUG command succeeded
```

und das Ende...

```
quit
221 Goodbye.
Verbindung zu Host verloren.
```

## **Editoren**

### **UltraEdit**

UltraEdit ist ein sehr mächtiger Text-Editor, der unter anderen Syntax-Highlighting beherrscht, außerdem kann man direkt ein TAP (TF5000 und TMS) aus dem Programm kompilieren. UltraEdit kann als 30 Tage Demo Programm bezogen werden. Die „ue\_german.zip“ in einen beliebigen Ordner herunterladen, auspacken, auf die „ue\_german.msi“ klicken und den Anweisungen folgen.

#### 1. Einbau der build.bat und der build\_TMS.bat

UltraEdit bietet die Möglichkeit, Programme per Maus-Klick (sog. Tools) zu starten. Diese Funktion kann man verwenden, um den Topfield Compiler direkt aus dem Editor heraus zu starten. Folgende Schritte sind notwendig:

- Menü – Extras – Werkzeug-Konfiguration
- Beim Reiter „Befehl“ folgendes eintragen:
  - Bezeichnung für den Menüeintrag = TF5000 Compiler (oder TMS Compiler)
  - Befehlszeile = build.bat oder build\_TMS.bat
  - Arbeitsverzeichnis = %P
- Beim Reiter „Optionen“ ist folgendes zu aktivieren:
  - „Dos Programm“ und
  - „Aktive Datei zuerst Speichern“
- Beim Reiter „Ausgabe“ alle Checkboxes deaktivieren außer
  - „Ausgabe im Listenfeld“ ,
  - „Ausgabe aufzeichnen“
  - „Nicht ersetzen“ (unter „Markierten Text ersetzen durch“)
- Übernehmen-Knopf drücken, das Tool erscheint in der Liste unten, dann OK

Abschließend lässt die Werkzeug-Konfiguration als Knopf in ein eigenes Untermenü ablegen

- Rechts-Klick auf eine leere Fläche im Menü
- Menü anpassen
- Im linken Fenster auf „Menü“ klicken, so dass es blau hinterlegt ist
- Im rechten Fenster unter „Neues (Unter)Menü“ einen Namen eingeben, z.B. „Compiler“, dann auf den Pfeil rechts daneben klicken. Im linken Fenster wird das Menü „Compiler“ angelegt.
- Das Menü „Compiler“ kann mit dem Pfeil\_nach\_unten Knopf verschoben werden.
- Jetzt im rechten Fenster ziemlich weit nach unten scrollen, bis die Werkzeug-Symbole (Hammer) auftauchen. Davon das erste Symbol markieren, welches jetzt wie das „Compiler“ Menü blau hinterlegt wird. Durch den Pfeil\_nach\_links Knopf wird das „Werkzeug1“ Symbol nach links in das Menü „Compiler“ verschoben. Mit „Ok“ übernehmen.
- Oben im Menü wird jetzt der Menüpunkt „Compiler“ angezeigt, darin unsere angelegten „TF5000 Compiler“ und, falls angelegt, der „TMS Compiler“.

Tipp: Wenn die „build.bat“ und die „build\_TMS.bat“ wie oben beschrieben in UltraEdit eingefügt werden, sollte das „pause“ aus den Batch-Dateien entfernt werden!

#### 2. Hinzufügen der TAP-Syntax.

Für UltraEdit gibt es das sogenannte Syntax-Highlighting. Damit werden die Befehle automatisch farblich gekennzeichnet um einen besseren Überblick beim Programmieren zu haben. Folgender Abschnitt muss dazu lediglich in die Datei "wordfile.uew" (die Originaldatei ggf. sichern) unter **C:\Programme\IDM Computer Solutions\UltraEdit** (bei Standardinstallation) kopiert werden. Und zwar in den Abschnitt `"/L1"C/C++"" hinter "/C4"`.

Für die Operatoren- und Typenliste bitte auf das Klammersymbol klicken.

## SciTE

[Scintilla based Text Editor](#) ist ein freier Texteditor, der Syntax-Highlighting (farbliche Hervorhebung von Schlüsselwörtern), AutoComplete (Vorschlag aller möglichen Schlüsselwörter, während das Wort getippt wird) und Intellisense (automatische Anzeige der Funktionsparameter) unterstützt.

Diese [SciTE\\_TF-1.78.zip](#) ist speziell an die Topfield-Entwicklungsumgebung angepasst. Sie kann z.B. in den Ordner `D:\Topfield` gespeichert und dort mit „Entpacken nach SciTE\_TF-1.78“ entpackt werden. Der komplette Ordner wird nach `D:\Work` kopiert oder verschoben. Es ist natürlich auch jeder andere Ort, z.B. `C:\Programme` möglich. Im Ordner `SciTE_TF-1.78` befindet sich `SciTe.exe`. Hier macht es Sinn, sich von dieser Datei eine Verknüpfung in die Startleiste und/oder Desktop zu erstellen.

Nachdem SciTE gestartet ist, können Dateien unter „File/Open“, z.B. aus dem Beispielprojekt `D:\Work\Taps\TF5000\Fire` geladen werden. Hier kann man sehen, dass die `TAP_` in blau hinterlegt sind, `Type_` (hier im Beispiel nicht vorhanden, aber z.B. in der `quickepg.c` aus `D:\Work\Taps\SamplesTF5000\EPG\` in Zeile 106) in rot. Ggf. muss noch unter „Language“ `C(TF5000)` angewählt werden.

Im Menü unter „Tools“ kann jetzt auf „Build TF5000 TAP“ das TAP fireclk erstellt werden.

Tipp: Auch hier bitte das „pause“ aus der `build.bat` herausnehmen.

In einer neu angelegten Datei lässt sich sehr schön sehen, wie AutoComplete und Intellisense funktioniert. Wird ein „t“ eingetippt, geht ein Fenster auf, wo alle Anweisungen, die mit „t“ oder „T“ beginnen, aufgeführt sind. Wird weitergetippt, z.B. „tap\_“ tauchen schon die ersten TAP-Operatoren auf. Mit einem Doppelklick, z.B. auf „TAP\_CaptureScreen“ wird dieser Operator in die Umgebung eingefügt. Mit einer Klammer auf „(“ erscheint dann unter der Anweisung die Parameterliste, wo der erste Parameter blau geschrieben ist, die anderen grau. Fügt man jetzt einen Parameter ein und setzt ein Komma, wird der erste Parameter grau und der 2. blau usw.

Mit den TMS TAPs funktioniert das prinzipiell genauso, nur dass hier die leicht anders aufgebauten und/oder neuen TAP Operatoren und Typen noch nicht eingefügt sind.