



USB-BITP-200

Hardware-Beschreibung

2010 Oktober

INDEX

<u>1. Einleitung</u>	4
<u>1.1. Vorwort</u>	4
<u>1.2. Kundenzufriedenheit</u>	4
<u>1.3. Kundenresonanz</u>	4
<u>2. Hardware Beschreibung</u>	6
<u>2.1. Einführung</u>	6
<u>2.2. Technische Daten</u>	7
<u>2.3. Ausgabekabel</u>	8
<u>3. Software</u>	10
<u>3.1. Benutzung unserer Produkte</u>	10
<u>3.1.1. Ansteuerung über grafische Anwendungen</u>	10
<u>3.1.2. Ansteuerung über unsere DELIB Treiberbibliothek</u>	10
<u>3.1.3. Ansteuerung auf Protokollebene</u>	10
<u>3.1.4. Ansteuerung über mitgelieferte Testprogramme</u>	11
<u>3.2. DELIB Treiberbibliothek</u>	12
<u>3.2.1. Übersicht</u>	12
<u>3.2.2. Unterstützte Betriebssysteme</u>	14
<u>3.2.3. Unterstützte Programmiersprachen</u>	14
<u>3.2.4. Installation DELIB-Treiberbibliothek</u>	15
<u>3.2.5. DELIB Configuration Utility</u>	17
<u>4. Programmier-Beispiel</u>	19
<u>5. Anhang</u>	24
<u>5.1. Revisionen</u>	24
<u>5.2. Urheberrechte und Marken</u>	25



Einleitung



1. Einleitung

1.1. Vorwort

Wir beglückwünschen Sie zum Kauf eines hochwertigen DEDITEC Produktes!

Unsere Produkte werden von unseren Ingenieuren nach den heutigen geforderten Qualitätsanforderungen entwickelt. Wir achten bereits bei der Entwicklung auf flexible Erweiterbarkeit und lange Verfügbarkeit.

Wir entwickeln modular!

Durch eine modulare Entwicklung verkürzt sich bei uns die Entwicklungszeit und - was natürlich dem Kunden zu Gute kommt - ein fairer Preis!

Wir sorgen für eine lange Lieferverfügbarkeit!

Sollten verwendete Halbleiter nicht mehr verfügbar sein, so können wir schneller reagieren. Bei uns müssen meistens nur Module redesigned werden und nicht das gesamte Produkt. Dies erhöht die Lieferverfügbarkeit.

1.2. Kundenzufriedenheit

Ein zufriedener Kunde steht bei uns an erster Stelle!

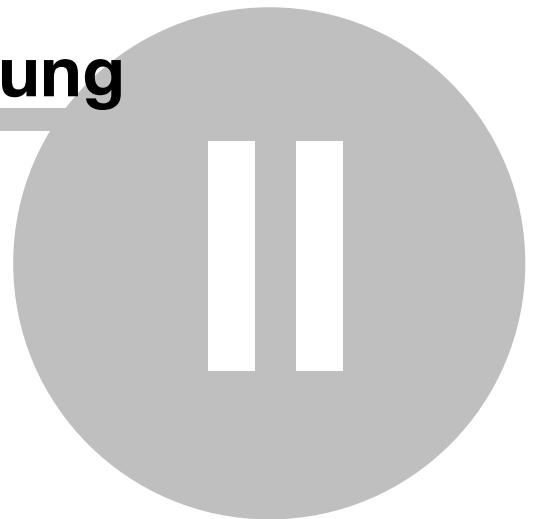
Sollte mal etwas nicht zu Ihrer Zufriedenheit sein, wenden Sie sich einfach per Telefon oder mail an uns.

Wir kümmern uns darum!

1.3. Kundenresonanz

Die besten Produkte wachsen mit unseren Kunden. Für Anregungen oder Vorschläge sind wir jederzeit dankbar.

Hardware Beschreibung

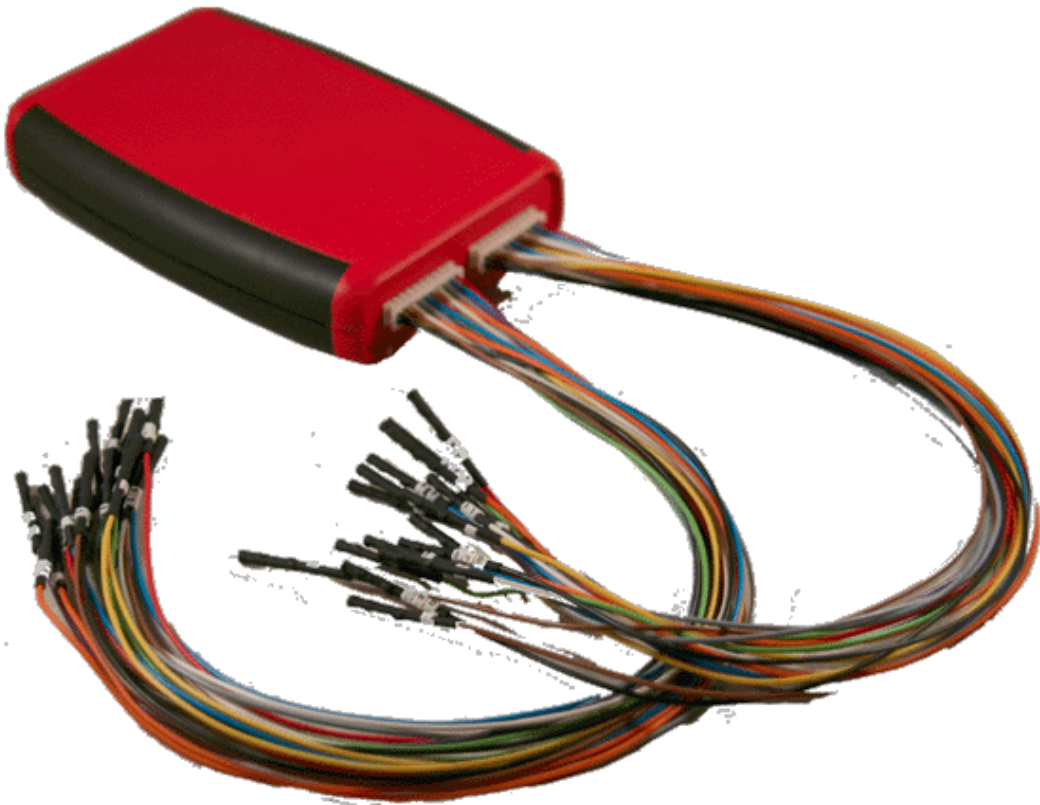


2. Hardware Beschreibung

2.1. Einführung

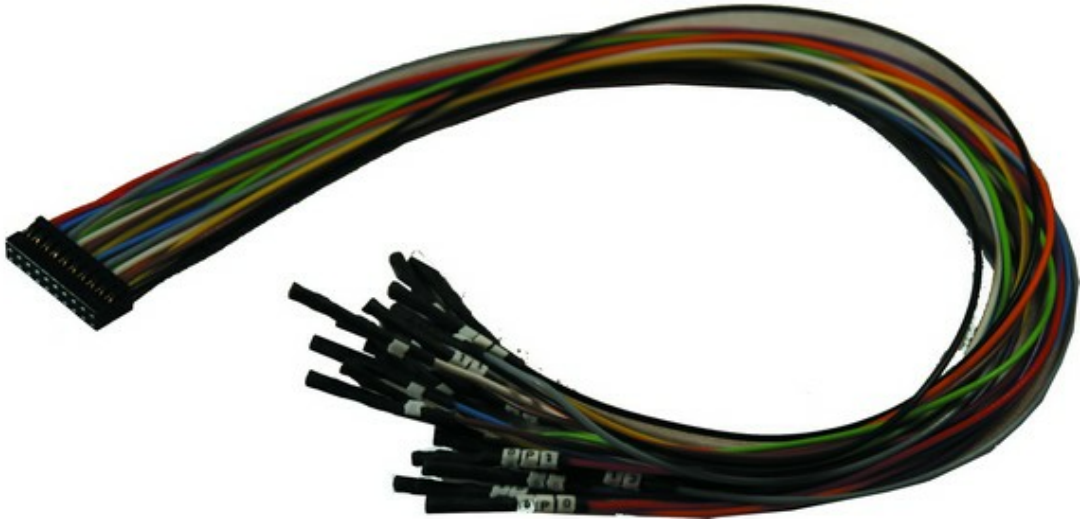
Der Bit Pattern Generator bietet die Möglichkeit, anhand einer einfachen Quellcode Software, ein per Software gegebenes Bitmuster, auszugeben. Die Bitmusterbreite beträgt 36 Bit und kann mit einer Geschwindigkeit von maximal 200MHz auf allen 36 Leitungen ausgegeben werden. Die Software bietet außerdem die Möglichkeit, eine frei wählbare Wiederholsequenz zu programmieren, um Bitmusterabfolgen zu wiederholen.

2.2. Technische Daten



- Versorgungsspannung über den USB-Port (max. 500mA)
- 36 Kanäle
- 36 Bit Bitmusterbreite
- Sample Puffer: 512 Kbit / Kanal
- Wiederholanzahl: Frei einstellbar 1..65535 / unendlich
- Steckverbinder: 2x 20pol Steckverbinder (Rastermaß 2.0 mm)
- Gehäuseabmessungen: 117x79x24 (LxBxH in mm)
- Temperaturbereich: 0°C bis +50°C

2.3. Ausgabekabel



Als optionales Zubehör für den **“USB-BITP-200”** ist ein spezielles Ausgabekabel mit farbigen und beschrifteten Einzeladern (Durchmesser ca. 0,8mm) möglich. Dieser kann z.B. auf Drähte, Stiftleisten oder sogenannte Micro-Kleps gesteckt werden. Micro-Kleps sind sehr feine Klemmprüfspitzen mit einer drehbaren Greifzange und sind ebenfalls bei uns erhältlich.

Software



3. Software

3.1. Benutzung unserer Produkte

3.1.1. Ansteuerung über grafische Anwendungen

Wir stellen Treiberinterfaces z.B. für LabVIEW und ProfiLab zur Verfügung. Als Basis dient die DELIB Treiberbibliothek, die von ProfiLab direkt angesteuert werden kann.

Für LabVIEW bieten wir eine einfache Treiberanbindung mit Beispielen an!

3.1.2. Ansteuerung über unsere DELIB Treiberbibliothek

Im Anhang befindet sich die komplette Funktionsreferenz für das Integrieren unserer API-Funktionen in Ihre Software. Des Weiteren bieten wir passende Beispiele für folgende Programmiersprachen:

- C
- C++
- C#
- Delphi
- VisualBasic
- VB.NET
- MS-Office

3.1.3. Ansteuerung auf Protokollebene

Das Protokoll für die Ansteuerung unserer Produkte legen wir komplett offen. So können Sie auch auf Systemen ohne Windows oder Linux unsere Produkte einsetzen!

3.1.4. Ansteuerung über mitgelieferte Testprogramme

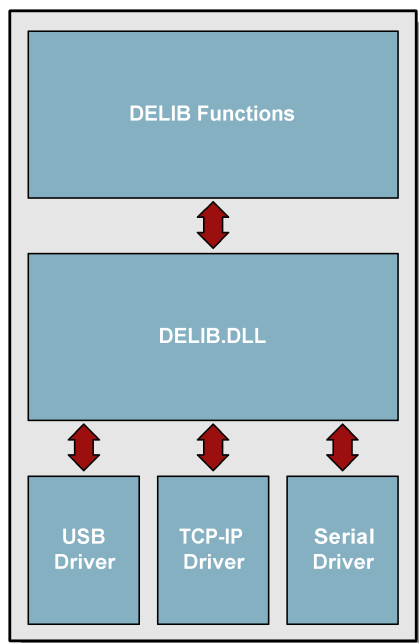
Für die wichtigsten Funktionen unserer Produkte stellen wir einfach zu bedienende Testprogramme zur Verfügung,. Diese werden bei der Installation der DELIB Treiberbibliothek direkt mit installiert.

So können z.B. Relais direkt getestet werden oder Spannungen am A/D Wandler direkt überprüft werden.

3.2. DELIB Treiberbibliothek

3.2.1. Übersicht

Die folgende Abbildung erläutert den Aufbau der DELIB Treiberbibliothek



Die DELIB Treiberbibliothek ermöglicht ein einheitliches Ansprechen von DEDITEC Hardware, mit der besonderen Berücksichtigung folgender Gesichtspunkte:

- Betriebssystem unabhängig
- Programmiersprachen unabhängig
- Produkt unabhängig

Programmieren unter diversen Betriebssystemen

Die DELIB Treiberbibliothek ermöglicht ein einheitliches Ansprechen unserer Produkte auf diversen Betriebssystemen.

Wir haben dafür gesorgt, dass mit wenigen Befehlen alle unsere Produkte angesprochen werden können.

Dabei spielt es keine Rolle, welches Betriebssystem Sie verwenden. - Dafür sorgt die DELIB !

Programmieren mit diversen Programmiersprachen

Für das Erstellen eigener Anwendungen stellen wir Ihnen einheitliche Befehle zur Verfügung. Dies wird über die DELIB Treiberbibliothek gelöst.

Sie wählen die Programmiersprache !

So können leicht Anwendung unter C++, C, Visual Basic, Delphi oder LabVIEW® entwickelt werden.

Schnittstellenunabhängiges programmieren

Schreiben Sie Ihre Anwendung schnittstellenunabhängig !

Programmieren Sie eine Anwendung für ein USB-Produkt von uns. - Es wird auch mit einem Ethernet oder RS-232 Produkt von uns laufen !

SDK-Kit für Programmierer

Integrieren Sie die DELIB in Ihre Anwendung. Auf Anfrage erhalten Sie von uns kostenlos Installationskripte, die es ermöglichen, die DELIB Installation in Ihre Anwendung mit einzubinden.

3.2.2. Unterstützte Betriebssysteme

Unsere Produkte unterstützen folgende Betriebssysteme:

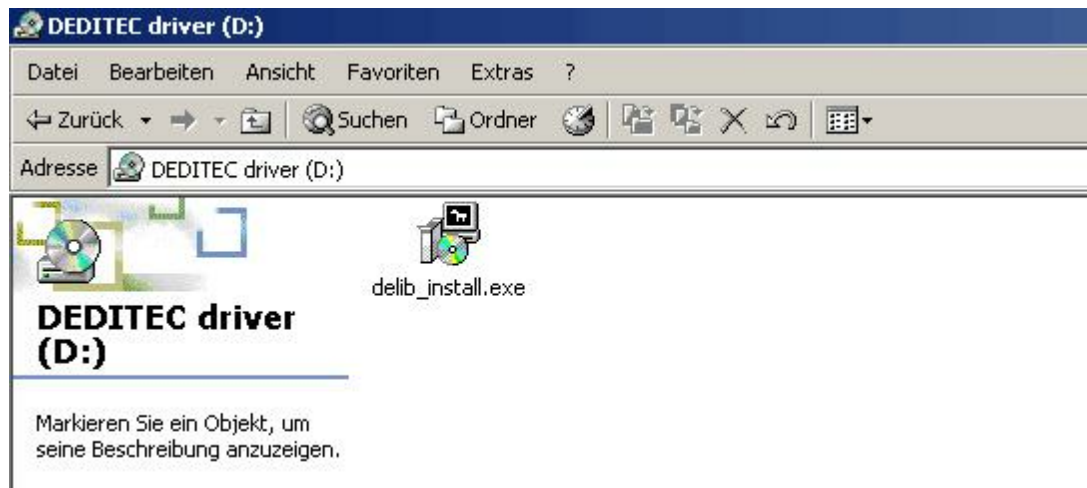
- Windows 2000
- Windows XP
- Windows Vista
- Windows 7
- Linux

3.2.3. Unterstützte Programmiersprachen

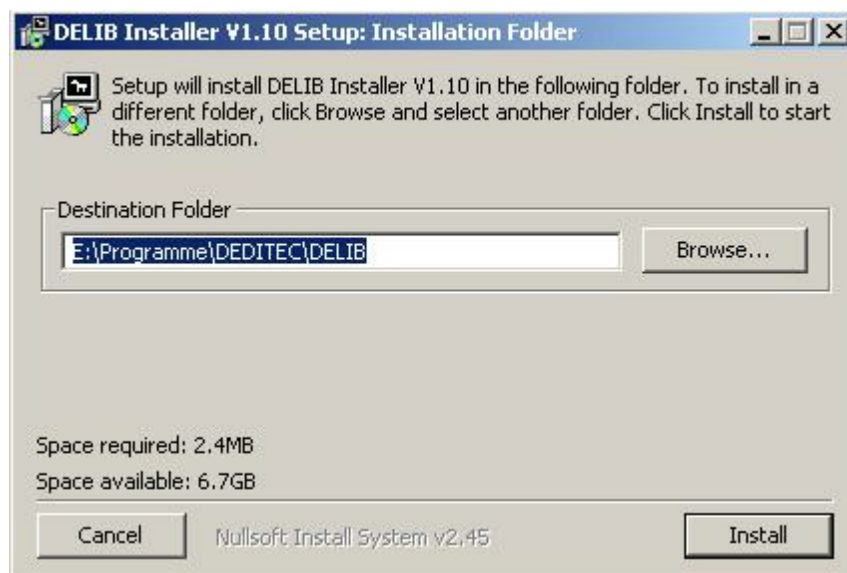
Unsere Produkte sind über folgende Programmiersprachen ansprechbar:

- C
- C++
- C#
- Delphi
- VisualBasic
- VB.NET
- MS-Office

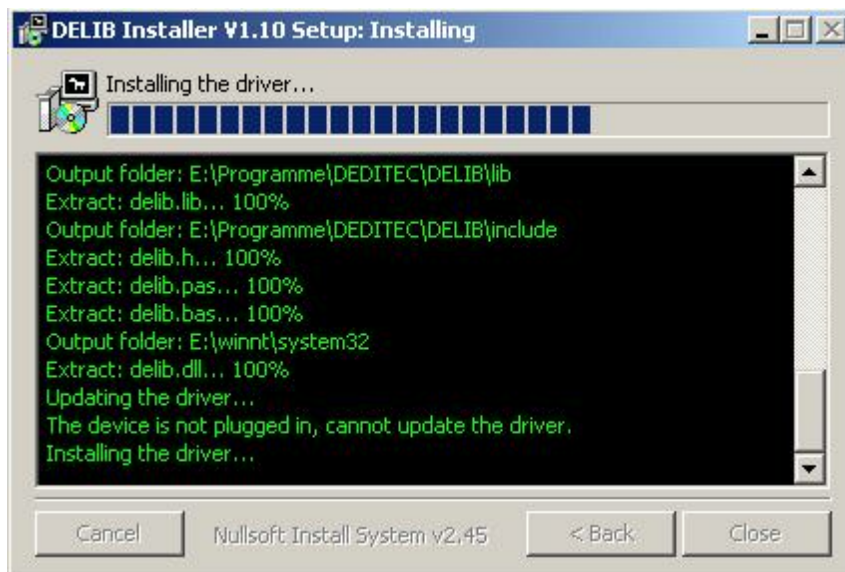
3.2.4. Installation DELIB-Treiberbibliothek



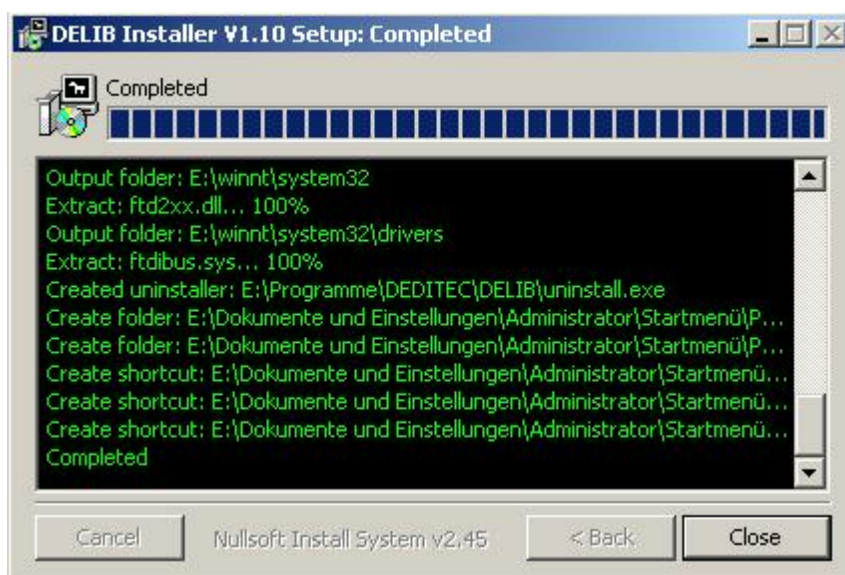
Legen Sie die DEDITEC driver CD in das Laufwerk und starten Sie "delib_install.exe". Die DELIB-Treiberbibliothek ist auch unter <http://www.deditec.de/delib> erhältlich.



Drücken Sie auf "Install".



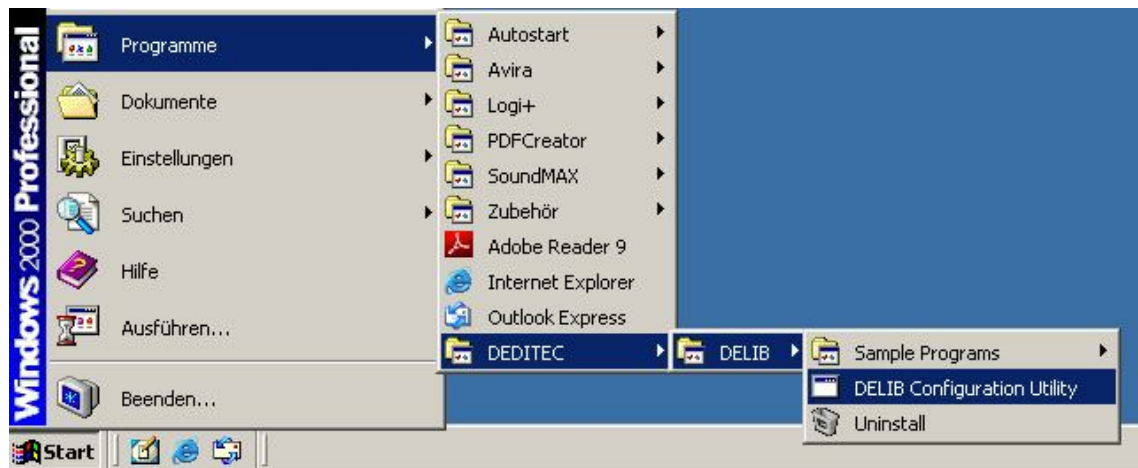
Die Treiber werden nun installiert.



Die DELIB Treiberbibliothek wurde nun installiert. Drücken sie auf **“Close”** um die Installation zu beenden.

Mit dem **“DELIB Configuration Utility”** (nächstes Kapitel) können Sie Ihr Modul konfigurieren (dies ist nur nötig, wenn Sie mehr als ein Modul ansprechen möchten).

3.2.5. DELIB Configuration Utility



“**DELIB Configuration Utility**” wird auf dem folgendem Weg gestartet:

Start → Programme → DEDITEC → DELIB → DELIB Configuration Utility.

Das “**DELIB Configuration Utility**” ist ein Programm zur Konfiguration und Unterteilung Identischer USB-Module im System. Dies ist aber nicht nötig falls nur ein Modul vorhanden ist.

Weiteres zum Inhalt der “**DELIB Installation**”, siehe “**Manual für DELIB Treiberbibliothek**”

Programmier-Beispiel



IV

4. Programmier-Beispiel

```
// *****
// *****
// *****
// *****
// *****
//
// (c) DEDITEC GmbH, 2009
//
// web: http://www.deditec.de
//
// mail: vertrieb@deditec.de
//
//
//
// bit-pattern-demo1.cpp
//
//
// *****
// *****
// *****
// *****
// *****

// Anweisung zum compilieren:
// -----
// Folgende DDL'S beim Linken mit einbinden
// DELIB.LIB (bindet dann DELIB.DLL mit ein)
//
//
#include "windows.h"
#include #include "conio.h"
#include "malloc.h"

#include "delib.h"

#include "bit-pattern_functions.h"

// -----
// -----
// -----
// -----
// -----

void main()
{
    ULONG ret;
    ULONG i;
    ULONG anz=10;
    ULONG device_nr=1;
    ULONG err=0;
    ULONG handle;
    ULONG ok;
```

```

ULONG a;
ULONG d0;

unsigned char * buff;

// Configuration
BP_Configuration bp_conf;

// -----
// Open Bit Pattern Generator
handle = DapiOpenModule(USB_BITP_200,0);

if(handle==0)
{
printf("No Module found\n");
printf("press any key\n");
getch();
return;
}

// -----
// Send ping to test the connection to the USB-device
printf("PING\n");
anz=100;
for(i=0;i!=anz;++i)
{
ret=DapiPing(handle, i); // DapiPing should return the value i
if(i==ret)
{
printf(".");
}
else
{
printf("E");
}
}
printf("\n");

// -----
// Test Memory
printf("Memory Test\n");
bp_memory_test(handle, 8*1024); // Teste den Speicher für max 512*1024

// -----
// Memory alloc for Pattern
buff = (unsigned char *) malloc(512 * 1024*5); // Memory of Bit Pattern
Generator
if(buff == 0)
{
printf("malloc buff Error\n");
getch();
return;
}

```

```

}

// -----
// Generate Test-Data

printf("Write Pattern Data to allocated-memory\n");

ULONG memory_lines = 100;
for(i=0;i!=memory_lines;++i)
{
    // Test Data (counter-values)
    d0 = i & 0xffff;
    d0|= ((~i) & 0xffff)<<16; // Inverted Data

    *(buff + i*5 + 0) = (unsigned char) (d0>>0) & 255;
    *(buff + i*5 + 1) = (unsigned char) (d0>>8) & 255;
    *(buff + i*5 + 2) = (unsigned char) (d0>>16) & 255;
    *(buff + i*5 + 3) = (unsigned char) (d0>>24) & 255;
    *(buff + i*5 + 4) = (unsigned char) (d0>>24) & 255;
}

// -----
// Transfere Memory to Hardware
ok=bp_write_ram(handle, buff, memory_lines*5); //buff= memory adress,
memory_lines*5 (36 Bit need 5 Bytes) -> size in Bytes
printf("Memory Data written to Bit Pattern Generator\n");

// -----
// Reset Hardware
bp_api_stop(handle);

// -----
// Set configuration

bp_conf.ram_adress_start = 2; // Set start address (only even addresses are
valid)
bp_conf.ram_adress_stop = 22; // Set stop address (only even addresses are
valid)
bp_conf.repeat_nr = 1; // counter for repeats (if = 0 -> don't stop output)

//bp_conf.sample_clock_divider = 0; // Ausgangstakt - Teiler für max. speed

bp_conf.sample_clock_divider = 76; // Ausgangstakt - Teiler für 1 MHz Takt

//bp_conf.sample_clock_divider = 75757572; // Ausgangstakt - Teiler für 1sec
Takt

// clock_divider T [ns]
// -----

```

```

// 0 6,6ns
// 1 53ns
// 2 66ns
// 3 80ns
// 5 105ns

// Formel:
// Ausgangstakt T[ns] = (3 + clock_divider) * 2 * 6,6 ns

bp_api_set_configuration(handle, bp_conf); // Configure Bit-Pattern Generator

// -----
// Start Bit Pattern generation
printf("Start Bit Pattern Generator now ...\n");
bp_api_start(handle);

// -----
// Check status
a=bp_api_get_status(handle);
printf("Sm=%x\n", a);

do
{
a=bp_get_adr_cnt(handle);
printf("RD Adress-Counter=%x bp_sm=%x ext_ram_sm=%x\n", bp_get_adr_cnt(handle),
bp_get_bp_sm(handle), bp_get_ext_ram_sm(handle));

//getch();
} while ((!kbhit()) && (bp_get_bp_sm(handle) != 2));

// -----
// Reset Hardware
bp_api_stop(handle);

printf("any key to leave demo\n");
getch();

// -----
// Close Hardware
DapiCloseModule(handle);

free(buff);

return ;
}

```

Anhang



5. Anhang

5.1. Revisionen

Rev 1.00	Erste DEDITEC Anleitung
Rev 2.00	Designänderung

5.2. Urheberrechte und Marken

Linux ist eine registrierte Marke von Linus Torvalds.

Windows CE ist eine registrierte Marke von Microsoft Corporation.

USB ist eine registrierte Marke von USB Implementers Forum Inc.

LabVIEW ist eine registrierte Marke von National Instruments.

Intel ist eine registrierte Marke von Intel Corporation

AMD ist eine registrierte Marke von Advanced Micro Devices, Inc.