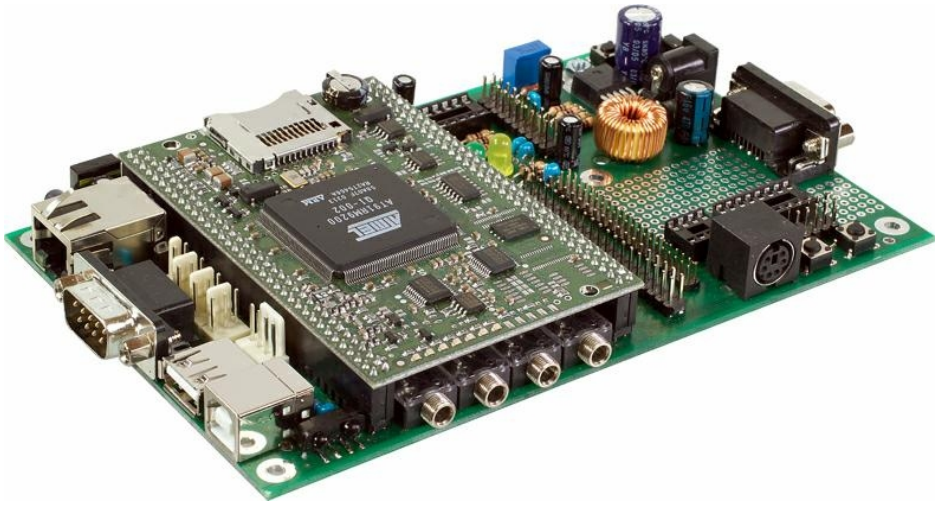
CentiPad Applikations Handbuch



Autor: Marcus Hasenstab
Version: 1.01
Datum: 061217
www.centipad.com

Notizen:

- 060818 SPI-ser3 Hinweis
- 060802 Hinweis auf Open Sound System
-
-

- Inhaltsverzeichnis

1. Einführung

1.1. Fokus

Herzlich willkommen beim „CentiPad Applikations Handbuch". Der Inhalt dieses Handbuchs ist so vielfältig wie sein Thema – die Entwicklung von Applikationen für das CentiPad.

Dieses Handbuch ist eine Ideensammlung und praktischer Ratgeber rund um das CentiPad.

Dieses Skript enthält Beispiele, welche in der Anwendung von CentiPad entstanden sind und ist ständig am wachsen.

Wir würden uns freuen auch Ihre Ideen in diese Sammlung aufnehmen zu dürfen.

1.2. Weitere Dokumentation

- CentiPad Hardware Dokumentation
- CentiPad Break Out Board Dokumentation mit Quickstart Guide
- CentiPad Break Out Board Schaltplan
- CentiPad Programmers Model
- CentiPad Applikations Handbuch
- Die neuesten Informationen sind jederzeit unter www.centipad.com verfügbar

1.3. Sicherheitshinweise

Bei Schäden, die durch Nichtbeachten dieser Bedienungsanleitung verursacht werden, erlischt der Garantieanspruch!
Für Folgeschäden wird keine Haftung übernommen!
Bei Sach- oder Personenschäden, die durch unsachgemäße Handhabung oder Nichtbeachten der Sicherheitshinweise verursacht werden, übernehmen wir keine Haftung!
In solchen Fällen erlischt jeder Garantieanspruch.

Verwenden Sie das Gerät nur in trockenen Räumen, in denen keine brennbaren Gase und Dämpfe vorhanden sein können.
Nehmen Sie das Gerät nicht sofort in Betrieb, wenn es von einem kalten in einen warmen Raum gebracht wurde. Das dabei entstandene Kondenswasser kann unter Umständen Ihr Gerät zerstören.
Wenn das Gerät *sichtbare Beschädigungen* aufweist, *nicht mehr arbeitet* oder *längere Zeit* unter *ungünstigen* Verhältnissen *gelagert* wurde, so ist anzunehmen, dass ein gefahrloser Betrieb nicht mehr möglich ist. An dieser Stelle ist das Gerät gegen unbeabsichtigte Inbetriebnahmen zu sichern, und falls erforderlich außer Betrieb zu nehmen.
Es dürfen keinerlei technische Veränderungen am Gerät vorgenommen werden.
Da das Gerät Wärme erzeugt, ist in der Endanwendung für eine ausreichende Luftzirkulation zum Abtransport der Wärme zu sorgen.

1.4. Garantiebedingungen und Produkthaftung

Die Garantie- und Produkthaftungsansprüche, auch während der gesetzlichen Gewährleistungspflicht entfallen, wenn das Gerät nicht entsprechend den Hinweisen, beschrieben in dieser Betriebsanleitung und am Gerät, betrieben wird.

Die Garantie- und Produkthaftungsansprüche, auch während der gesetzlichen Gewährleistungspflicht entfallen, wenn das Gerät geöffnet und/oder modifiziert bzw. unsachgemäß betrieben wird.
Auf Grund Ihrer begrenzten Lebensdauer sind Teile die einem besonderen Verschleiß unterliegen, von der gesetzlichen Gewährleistungspflicht ausgenommen. Dazu gehören z.B. Batterien, Steckverbinder etc..

1.5. Legende

Verdana Fett	Untertitel, markierter Text
Courier New fett	Befehl oder Befehlsrückmeldung sowie Programmcode
Courier New Fett Kursiv	Pfadangaben und lokale Verzeichnisse
Courier New	Lenkt die Aufmerksamkeit auf wichtige oder nützliche Hinweise
#	Kommentarzeichen in Shellskripten
$	Linux Benutzer Eingabeprompt
#	Linux Root Eingabeprompt
Under Construction	grün unterlegte Passagen sind derzeit im Aufbau

1.6. Rechtliches

Das vorliegende Werk ist in all seinen Teilen urheberrechtlich geschützt. Alle Rechte vorbehalten, insbesondere das Recht der Übersetzung, des Vortrags, der Reproduktion, der Vervielfältigung auf fotomechanischem oder anderen Wegen und der Speicherung in elektronischen Medien.

Ungeachtet der Sorgfalt, die auf die Erstellung von Text, Abbildungen und Programmen verwendet wurde, können weder der Autor noch Hersteller für mögliche Fehler und deren Folgen eine juristische Verantwortung oder irgendeine Haftung übernehmen.

Die in diesem Werk wiedergegebenen Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. können auch ohne besondere Kennzeichnung Marken sein und als solche den gesetzlichen Bestimmungen unterliegen.

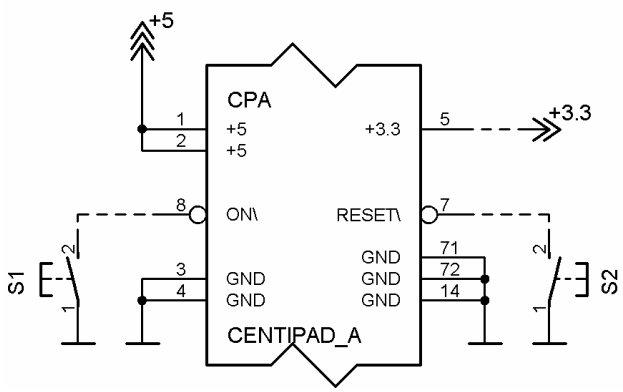
Technische Änderungen vorbehalten!

2. Application Notes

2.1. Layout Empfehlungen

- stabile Versorgungsspannung 5V
  - ein Linearregler auf dem CentiPad erzeugt die 3,3V Peripheriespannung und die 1,8V CPU-Corespannung
  - schlechte Versorgungsqualität kann die Funktion des Boards, insbesondere Sound, CAN und USB negativ beeinflussen
- Erweiterungs Bus
  - Maximal 60MHz
  - Zum Vermindern von Übersprechen Adress- und Datenleitungen mit 22Ω Serienwiderständen terminieren
- Ethernet
  - Signalleitungen bis zum Trafo parallel und so kurz wie möglich legen
- USB
  - Signalleitungen bis zum Stecker parallel und so kurz wie möglich legen
- CAN
  - Leitungsabstände aufgrund der hören Signalspannungen beachten
- Sound
  - Auf sauber Masseführung achten, AGND ist gefiltert gegenüber DGND

2.2. CentiPad Minimalsystem



Die einfachste Beschaltung des CentiPad ist im Bild dargestellt. Es wird nur die 5V Versorgungsspannung angelegt. Wird S1 geschlossen, so startet das Board sofort. Dieser Aufbau eignet sich z.B. ideal für Datenlogger Anwendungen.
An Pin 5 steht die 3,3V Peripheriespannung zur Verfügung, welche mit maximal 250mA belastet werden darf.

2.3. Demos

Unter **centidev/platform/demos** stehen verschiedene Beispielprogramme zur Verfügung. Diese Programme sind nicht unter **centidev/application** abgelegt, damit diese mittels **centidev/make menuconfig** aus einem Produktivsystem entfernt werden können.

2.4. Serial Port

Trotz Ihres begnadeten Alters ist die an RS232 angelehnte serielle Schnittstelle immer noch ein universelles Interface zwischen Baugruppen. CentiPad verfügt neben der Debug-Schnittstelle noch über vier weitere serielle Schnittstellen.

2.4.1. Übersicht Einstellungen für die serielle Schnittstelle

Im Kernel wird eine Zuordnung zwischen der vorhandenen Hardware und den Schnittstellen dev-Dateien gemacht.
In **centidev/platform/arch/arm/mach-at91rm9200/board-centipad.c** werden die vorhandenen seriellen Schnittstellen auf /dev/ttyS0..4 gemappt.
Standardeinstellung der Schnittstellen:

- 9600bps, 8, N, 1
- /dev/ttyS1 -> USART1
- /dev/ttyS2 -> USART2
- /dev/ttyS3 -> USART3
- /dev/ttyS4 -> USART0
- /dev/ttyS0 -> Debugschnittstelle

2.4.2. Schnittstellentest

- Auf dem CentiBOB ist die Schnittstelle 2 des CentiPad mit der männlichen SubD9-Buchse X9 verbunden.
- Über ein Nullmodem-Kabel kann man dies mit dem Hostrechner verbinden.
- Öffnen Sie ein Terminal Programm.
- Stellen Sie die Übertragungsrate auf 9600bps ein.
- Tippen Sie in der CentiPad-Konsole **echo „Schnittstellentest“ > /dev/ttyS2**.
- Auf dem Host-Terminal erscheint `Schnittstellentest`, sofern die Baudrate richtig eingestellt war
- Tippen Sie nun **cat /dev/ttyS2** auf dem CentiPad-Terminal
- Eine Zeichenfolge wird nach Drücken der Return-Taste auf dem CentiPad-Terminal angezeigt

2.4.3. Programmierbeispiel für Zugriff auf einen seriellen Port

Das Programm **demos/serialtest** zeigt ein einfaches Programmierbeispiel für die serielle Schnittstelle. Die seriellen Schnittstellen tty* verhalten sich wie Files. Zum Setzen der weiteren Parameter werden „termios“ verwendet.

Es werden folgende Verfahren demonstriert:

- Schnittstelle öffnen
  - **fd = open(sPortName, O_RDWR | O_NOCTTY | O_NDELAY);**
- Schnittstelle leeren
  - **tcflush(fd, TCIFLUSH);**
- Baudrate und weitere Parameter einstellen
  - **my_termios.c_cflag = B9600 | CS8 |CREAD | CLOCAL | HUPCL;**
  - **cfsetospeed(&my_termios, B9600);**
  - **tcsetattr(fd, TCSANOW, &my_termios);**

- Daten schreiben
  - `iOut = write(fd, psOutput, strlen(psOutput));`
- Daten lesen
  - `iIn = read(fd, psResponse, iMax-1);`

Beim Testen der Schnittstellen muss der eingestellte Puffermodus (character, line, full) beachtet werden. Für weitere Informationen siehe **`man setbuf`**.

2.5. Verwenden der General Purpose Input/Outputs

CentiPad verfügt über 6 GPIOs. Diese Leitungen GPIO0..5 sind mit den dem PORTC, Pin 0..5 verbunden. Auf dem CentiPad sind die GPIOs zusätzlich mit Leuchtdioden verbunden. Die LEDs leuchten, wenn der zugehörige Pin auf Low geschaltet ist.

Für die GPIOs wurde ein eigener Treiber entwickelt, das zugehörige Devicefile ist **/dev/gpio**. Der Benutzer greift auf den Treiber über die gpiolib zu. Deren Sourcen sind unter **/centidev/platform/gpiolib** abgelegt.

2.5.1. GPIOTEST

gpiotest ist ein Beispielprogramm, welches den Zugriff auf die GPIOs über die GPIOLIB demonstriert. Es eignet sich für Tests und als Grundlage für eigene Programme.

Funktionsweise:
Nach dem Start wartet gpiotest auf Benutzereingaben. Mit help stehen weitere Informationen zur Verfügung.

```
/ # gpiotest
help
Available commands:
        acquire <port> <pin>
        release <port> <pin>
        set_config <port> <pin> <mode> <dir> <pullup> <multidrive> <filter>
        get_config <port> <pin>
        set_output <port> <pin> <output>
        get_output <port> <pin>
        get_input <port> <pin>
        help
        exit
```

port:	1..3	-> PORTA..PORTC
pin:	0..31	-> Pinnummer
mode:	0	-> Pin als GPIO betreiben
dir:	0/1	-> Eingang/Ausgang
pullup:	0/1	-> Pullup aus/einschalten
multidrive:	0/1	-> Multidrive aus/einschalten
filter:	0/1	-> Filter aus/einschalten
output:	0/1	-> Ausgangspegel low/high

GPIOs müssen vor ihrer Verwendung mittels acquire für gpiotest reserviert werden. Mit set_config wird die Datenrichtung vorgegeben. multidrive und filter sollten auf 0 gesetzt werden (weitere Info im AT91RM9200 Datenblatt). Beispiel:

$ gpiotest Starte gpiotest	
acquire 3 5	GPIO5 (PORTC, Pin0) für gpiotest buchen
set_config 3 5 1 1 0 0 0	GPIO5 als Ausgang schalten
set_output 3 5 1	GPIO5 high schalten – LED erlischt
set_output 3 5 0	GPIO5 low schalten – LED leuchtet

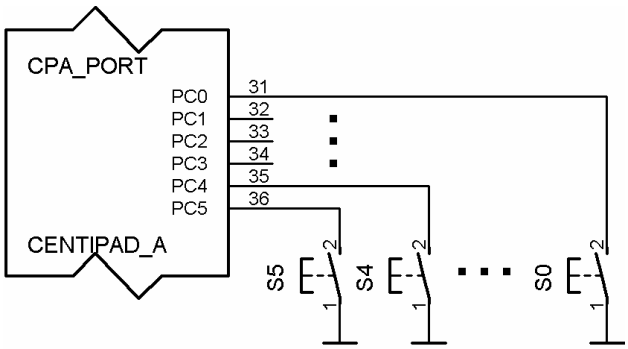
Hinweis: im GPIO-Treiber Hardwarezugriff ist die Port-Zählweise 0..2 für PORTA..C, in der GPIOLIB ist die Zählweise 1..3.

Bekannte Bugs: PullUp-Einschalten erfordert dass der GPIO kurz auf Ausgang geschaltet war. Dies ist Hardware-Bug im AT91RM9200. Änderung an der GPIOLIB derzeit nicht vorgesehen.

2.5.2. Lauflicht

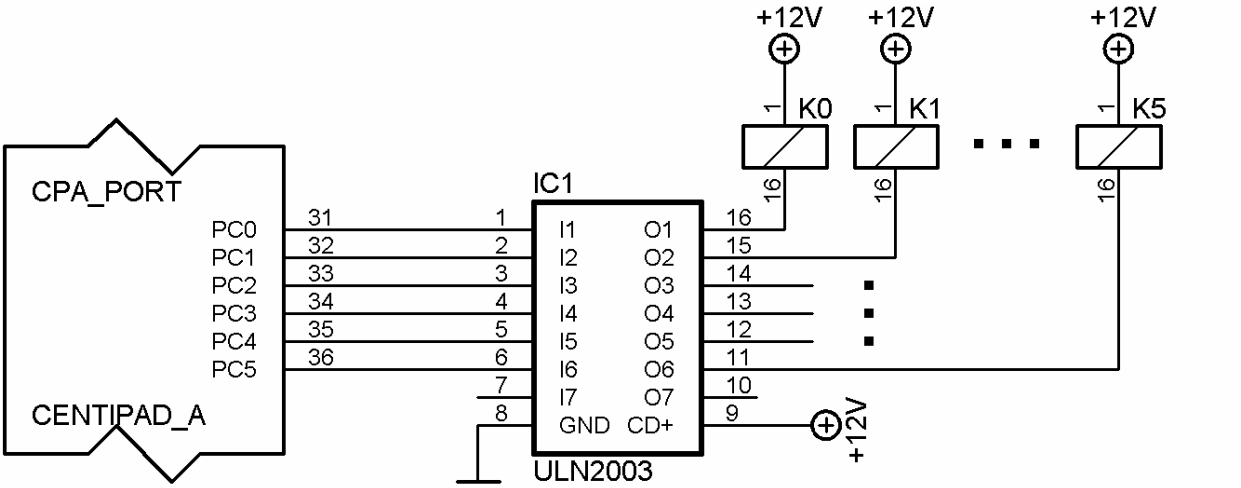
Das Programm **`lauflicht`** ist ein weiteres Beispiel für den Zugriff auf die GPIOs. **`lauflicht`** kann zum Beispiel während der Entwicklung beim Systemstart aufgerufen werden. Damit ist immer erkennbar, ob das CentiPad Linux noch läuft.
Verzeichnis: **`centidev/platform/demos/lauflicht`**

2.5.3. Taster

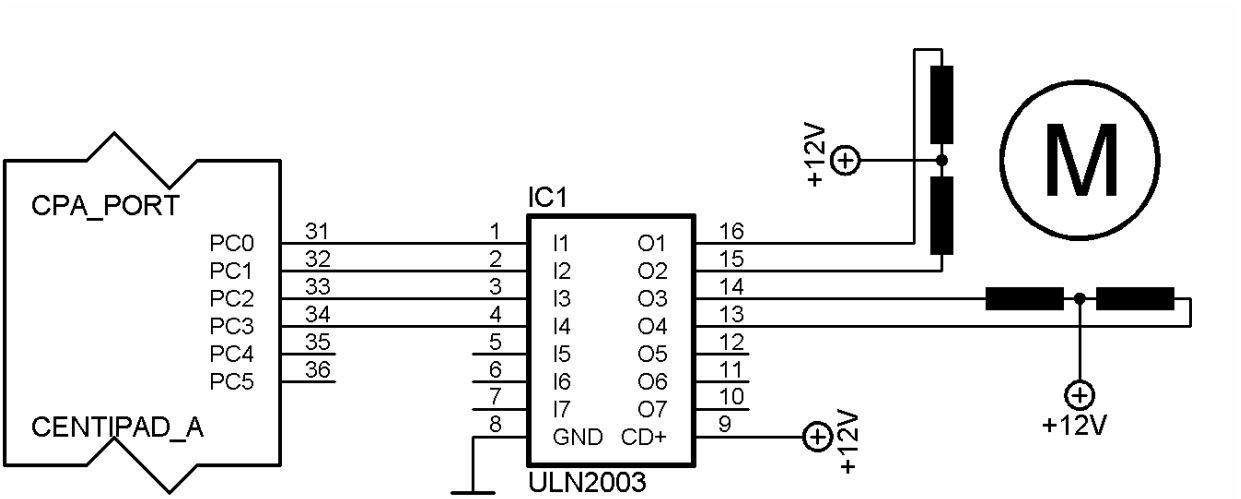


Taster können wie im Bild gezeigt einfach an die GPIO’s angeschlossen werden. Die GPIOs werden hierzu auf Eingang mit Pullup geschaltet. Wird ein Schalter geschlossen leuchtet die zugehörige LED auf und der Eingangspegel Low wird gemessen.

2.5.4. Leistungstreiber ULN2003

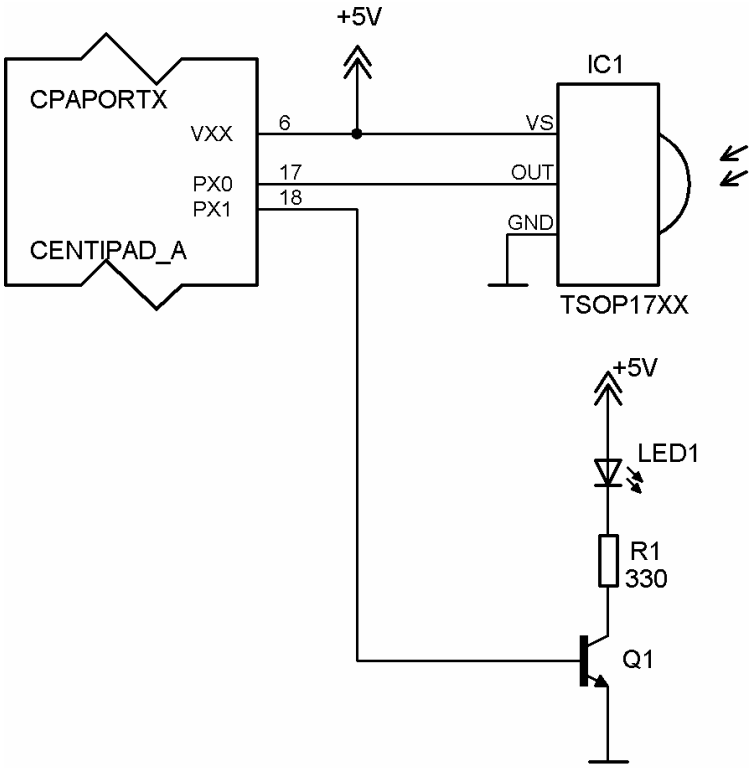


Bei höherem Spannungs-/Strombedarf kann ein ULN2003 wie im Bild dargestellt an die GPIOs angeschlossen werden.



Der Anschluss von unipolaren Schrittmotoren ist möglich, die Schrittfolge wird einfach von einem Anwenderprogramm erzeugt.

2.6. Infrarot Sender/Empfänger

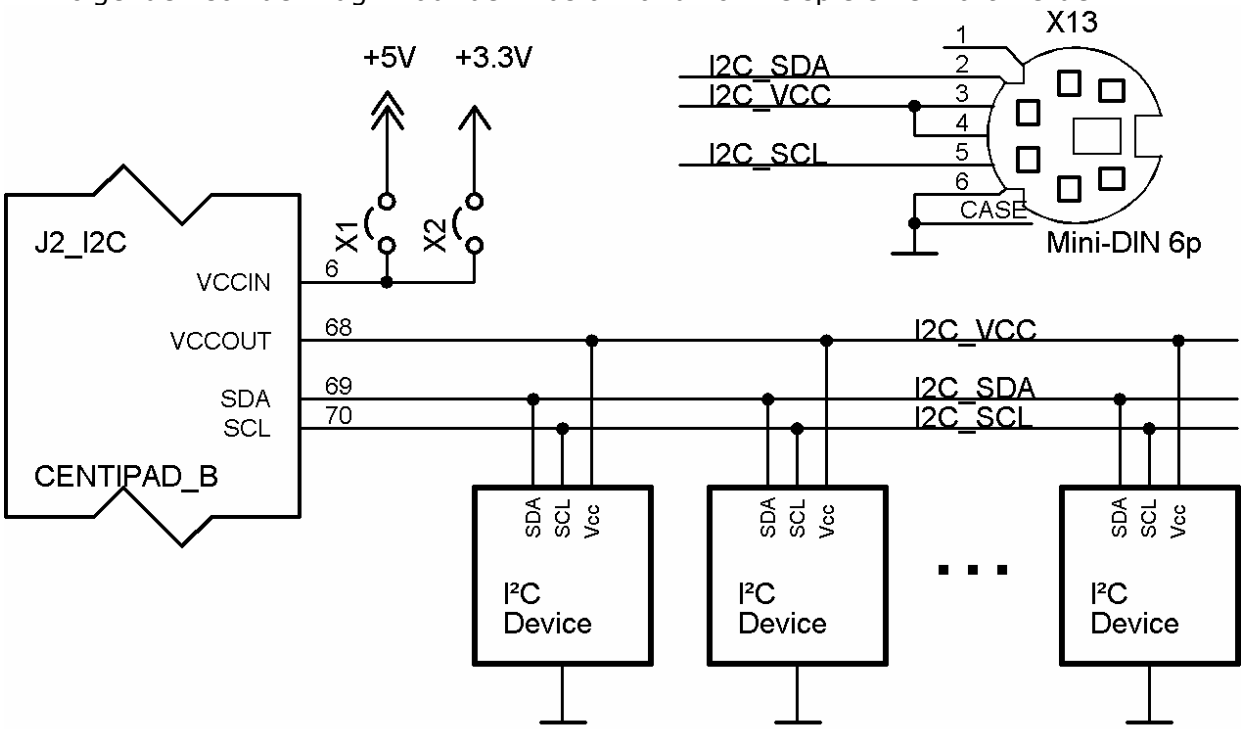


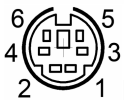
Die Schaltung zeigt, wie an die PX0/1 Schaltung ein IR-Empfänger/-Sender angeschlossen werden kann. Als Softwaretreiber kann z.B. LIRC eingesetzt werden.

2.7. Verwendung des TWI

2.7.1. Allgemein

Zunächst zur Begriffsklärung: Die Firma Philips entwickelte ein Bus-System für die „Inter Integrated Circuit Communication", den I^2C-Bus. Dieser Bus und auch sein Name sind geschützt und lizenzpflichtig. Um dies zu umgehen, bieten viele Hersteller auf ihren Bausteinen das „Two Wire Interface" an. Dieses TWI ist weitgehend, aber nicht hundertprozentig, zu I^2C kompatibel. Da auch Atmel nur ein TWI anbietet, wird in dieser Dokumentation auch vorwiegend dieser Begriff verwendet. Wenn doch einmal die Zeichenfolge „i2c" auftaucht, dann meistens, weil in bestehender Linux-Software der Begriff I^2C verwendet wurde.
Der TWI-Bus wird auf dem CentiPad ausgiebig genutzt. Am TWI sind der Soundchip, das Konfigurations-EEPROM, die RTC und der 1-Wire-Controller angeschlossen.
Im Folgenden soll der Zugriff auf den Bus anhand von Beispielen erklärt werden.





Draufsicht Mini-DIN 6p

2.7.2. Was hängt am Bus?

CentiDev enthält Treiber für die verschiedene TWI-Geräte. Ist ein Kernel-Treiber für ein Busgerät vorhanden, so übernimmt der Kernel hierfür die Zugriffskontrolle. Man sagt der Kernel ‚claimed' das Device.
Für diese Devices legt der Kernel im Verzeichnis **`/sys/bus/i2c/devices`** Treiberdateien an. Die Namenskonvention ist **`<Busadapater>-<Deviceadresse>.`**

An einem CentiPad pro auf CentiBOB sind folgende Devices vorhanden:

0-001a	EEPROM
0-0027	PCF
0-0051	RTC
0-0057	Soundchip

Das jeweilige Device-Verzeichnis enthält folgende Dateien/Verzeichnisse:

- Bus Bus Verzeichnis
- driver Treiber Verzeichnis
- name Name des Devices, z.B. pcf8574
- read char Device zum Lesen
- write char Device zum Schreiben
- uevent event Device

2.7.3. i2ctest

Das Programm **i2ctest** gehört zu den Beispielprogrammen im demos-Verzeichnis. Es ermöglicht den Zugriff auf TWI-Bus-Devices.
Vorsicht: Der direkte Zugriff funktioniert nur, wenn kein Kerneltreiber für das Device geladen ist! Hierzu ist der entsprechnde Treiber aus der Kernelkonfiguration zu entfernen.

Das Zugriffsformat ist:

```
8bit Zugriff:
i2ctest <i2c-device> <i2c-addr> r <reg>
i2ctest <i2c-device> <i2c-addr> w <reg> <value>

16bit Zugriff:
i2ctest <i2c-device> <i2c-addr> R <reg>
i2ctest <i2c-device> <i2c-addr> W <reg> <value>
```

2.7.4. Zugriff auf das interne EEPROM

Das Konfigurations-EEPROM enthält Bootloader Informationen. Ein Zugriff von Benutzerseite ist derzeit nicht vorgesehen. Bei Bedarf bitte Rückfrage.
Standardmäßig kontrolliert der CentiPad cpnvram-Treiber den Zugriff auf das EEPROM.

2.7.5. Zugriff auf die batteriegepufferte Echtzeituhr

Hinweis: Zugriff wird vom Kerneltreiber unterdrückt!
Die Echtzeituhr hat die Busadresse 0xA2, die Sekunden liegen im Register 0x02. Mit dem i2test-programm sieht der Zugriff wie folgt aus:
`i2test /dev/i2c/0 0x51 r 2`
Wird dieser Befehl wiederholt ausgeführt, so ändert sich jede Sekunde die Anzeige.
`watch –n 1 i2ctest /dev/i2c/0 0x51 r 2`
Die RTC enthält einige don’t-care-Bits. Diese verändern ihren Zustand aufgrund undokumentierter Gesetzmäßigkeiten. Daher müssen die RTC-Register über Bitmasken gefiltert werden.

Table 4: Binary formatted registers overview

Bit positions labelled as x are not implemented. Bit positions labelled with 0 should always be written with logic 0; if read they could be either logic 0 or logic 1.

Address	Register name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
00H	control/status 1	TEST1	0	STOP	0	TESTC	0	0	0
01H	control/status 2	0	0	0	TI/TP	AF	TF	AIE	TIE
0DH	CLKOUT control	FE	x	x	x	x	x	FD1	FD0
0EH	timer control	TE	x	x	x	x	x	TD1	TD0
0FH	timer	<timer countdown value>							

Table 5: BCD formatted registers overview

Bit positions labelled as x are not implemented.

Address	Register name	BCD format tens nibble				BCD format units nibble			
		Bit 7 2^3	Bit 6 2^2	Bit 5 2^1	Bit 4 2^0	Bit 3 2^3	Bit 2 2^2	Bit 1 2^1	Bit 0 2^0
02H	seconds	VL	<seconds 00 to 59 coded in BCD>						
03H	minutes	x	<minutes 00 to 59 coded in BCD>						
04H	hours	x	x	<hours 00 to 23 coded in BCD>					
05H	days	x	x	<days 01 to 31 coded in BCD>					
06H	weekdays	x	x	x	x	x	<weekdays 0 to 6>		
07H	months/century	C	x	x	<months 01 to 12 coded in BCD>				
08H	years	<years 00 to 99 coded in BCD>							
09H	minute alarm	AE	<minute alarm 00 to 59 coded in BCD>						
0AH	hour alarm	AE	x	<hour alarm 00 to 23 coded in BCD>					
0BH	day alarm	AE	x	<day alarm 01 to 31 coded in BCD>					
0CH	weekday alarm	AE	x	x	x	x	<weekday alarm 0 to 6>		

[aus RTC Datasheet]

Für die RTC stehen verschiedene Sourcen/Programme im CentiPad Entwicklungssystem zur Verfügung:

`rtcset 060223164012` – setzt die Echtzeituhr und die Systemzeit auf den 23.02.2006 16:40:15

`rtcset C` - setzt die Systemzeit entsprechend der Zeitinformationen aus der RTC. Dieses Programm sollte z.B. beim Systemstart aufgerufen werden. Gleichzeitig werden die Interrupt Flags der RTC gelöscht, um ein erneutes Ausschalten zu ermöglichen!

`rtcstatus` - zeigt die Register der RTC an. Hier werden alle Bits angezeigt – bitte mit dem RTC-Datenblatt vergleichen

2.7.6. Zeitgesteuertes Einschalten

Zugriff wird von Kerneltreiber unterdrückt!

Die RTC hat einen Open Collector Interrupt-Ausgang, welcher über J29 mit dem PWR_EN\ Signal des Spannungsreglers verbunden werden kann.
Durch Setzen der entsprechenden Register in der RTC kann eine Alarmzeit oder eine Verzögerungszeit eingestellt werden. Danach wird das CentiPad ausgeschaltet.
Nach Ablauf der Verzögerungszeit oder bei erreichen des programmierten Aufwachzeitpunkts startet das CentiPad neu.

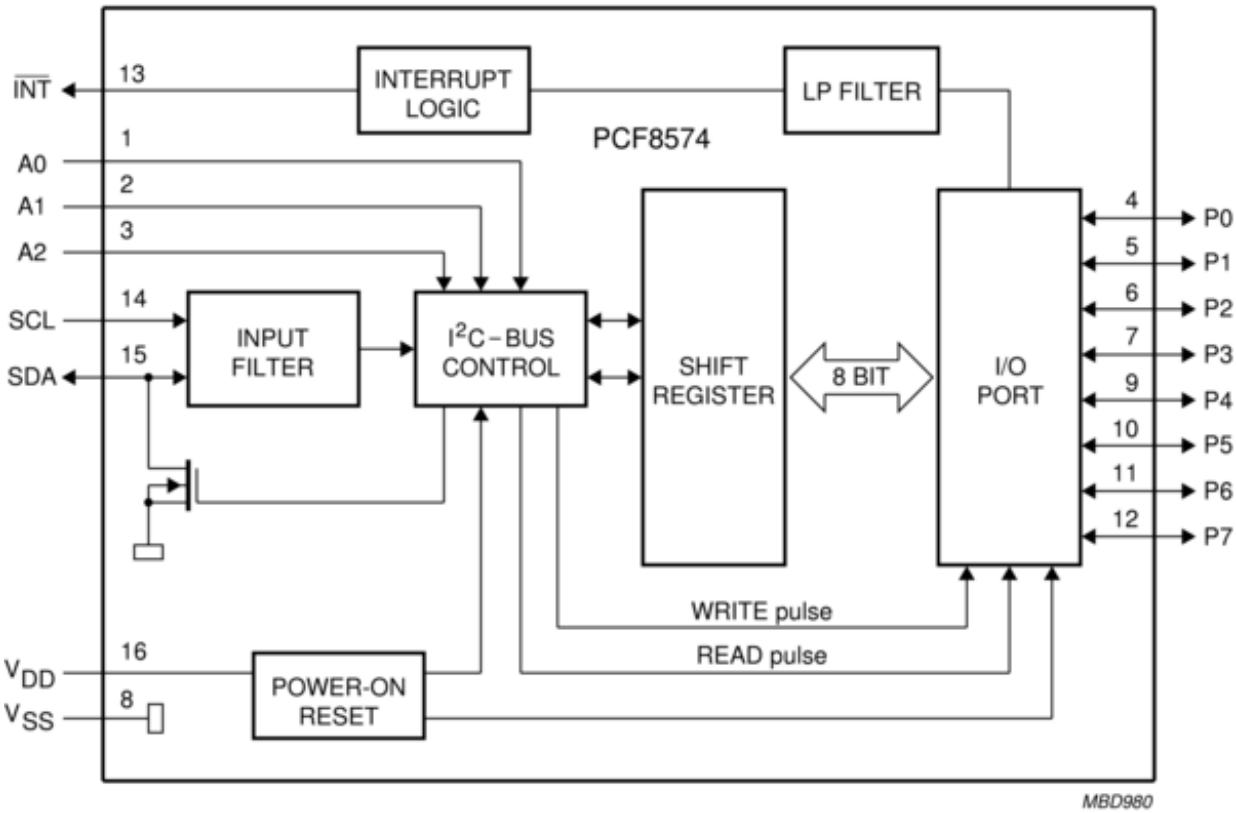
showdown - kontrollierter Shutdown für CentiPad. Beispielprogramm und Werkzeug. Im Folgenden die Online-Hilfe:

showdown c - configures centipad shutdown logic
showdown s - executes a sync and powers the system down
showdown r - reboots the system

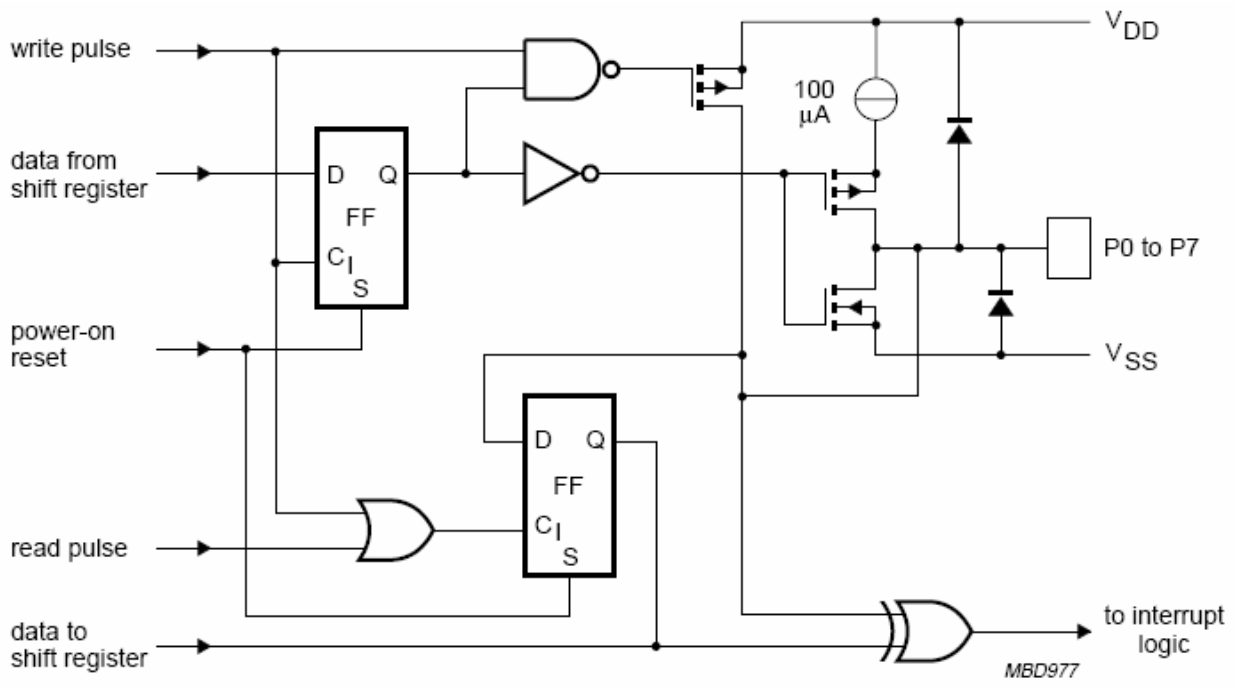
showdown a <ddhhmm> - restarts the system at the defined time
<.....> -> dotted time values will be ignored
dd: 01..23 hh: 00..23 mm: 00..59
showdown t <time> <unit> - shutdown, then restart after <time> <unit>
time: 1..255 unit: 'min' or 'sec'
Wichtig: CentiPad kann nur heruntergefahren werden, wenn die RTC-Interrupt-Flags gelöscht sind. **showdown**, genau wie **rtcset** löscht diese Interrupt Flags.

2.7.7. Zugriff Parallelport PCF8574

Zugriff wird vom Kerneltreiber unterdrückt!
Auf dem BOB ist ein Sockel für den PCF8574P vorhanden. Dieser Baustein ist ein Port-Expander für den TWI-Bus. Der PCF8574P verwendet Quasi-bidirectionale I/Os. Die Ausgänge verhalten sich ähnlich Open Collector Ausgängen mit integriertem Pullup-Widerstand.
Beim Einschalten sind die Ausgänge über eine schwache Stromquelle auf High geschalten. Wechselt ein Bit im Empfangsregister auf High, so schaltet sich ein stärkerer Pullup für einen SCL-Takt zu. Ein Low-Bit im Empfangsregister schaltet einen Transistor nach GND. Bevor ein IO als Eingang verwendet wird, sollte das entsprechende Bit auf High gesetzt werden.



[aus PCF8574 Datasheet]



[aus PCF8574 Datasheet]

Die TWI-Basisadresse des PCF8574P ist $20. Über J15..J17 kann die Adresse das Bausteins verändert werden, normalerweise sind die Brücken geschlossen, was eine Adresse $27 zur Folge hat. Der Baustein verfügt über keine Pulldown-Widerstände, bei Bedarf müssen die Adressleitungen über Drahtbrücken auf Low gezogen werden. Hinweis: Wahlweise kann ein PCF8574PA mit der Basisadresse $38 bestückt werden.

Beispiel Lesezugriff:
`i2ctest /dev/i2c-0 0x27 r 0`

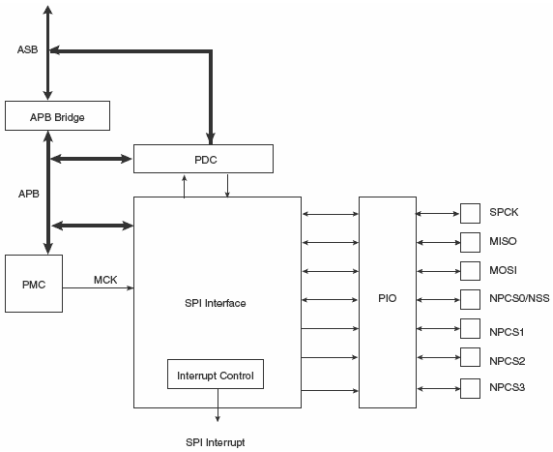
Schreibzugriff:
`i2ctest /dev/i2c-0 0x27 w 0 0x55`

Für weitere Informationen siehe PCF8574P Datenblatt.

Der Zugriff über Kerneltreiber:
`cat /sys/devices/platform/i2c-0/0-0027/read`
Ergebnis ist normalerweise 255, wird eine Taste auf dem CentiBOB gedrückt, reduziert sich der Wert um 128 bzw 64.

2.8. Verwendung des SPI-Bus

2.8.1. Allgemein



[aus AT91RM9200 Datasheet]

Der synchrone Serial Peripheral Bus des AT91RM9200 wird intern auf dem CentiPad verwendet und steht gleichzeitig für externe Erweiterungen zur Verfügung.
Der SPI-Controller besitzt vier Chip Select Leitungen. Bei Bedarf stehen dem Treiberentwickler noch die Signale Port0..5 zur Verfügung.

Systemanbindung:

Chipselect	**Device**	
NPCS0	Boot Flash	
NPCS1	CAN-Controller	
NPCS2	ExpansionPort CPA39	wird mit Ser3 geteilt!
NPCS3	ExpansionPort CPA40	wird mit Ser3 geteilt!
Port0..5	ExpansionPort CPA31..36	

Unter /dev stehen auf dem CentiPad die Devices spi0..spi3 für die jeweiligen Chipselects zur Verfügung.
Wird ser3 benötigt, so ist der SPI-Treiber zu modifizieren.

2.8.2. spitest

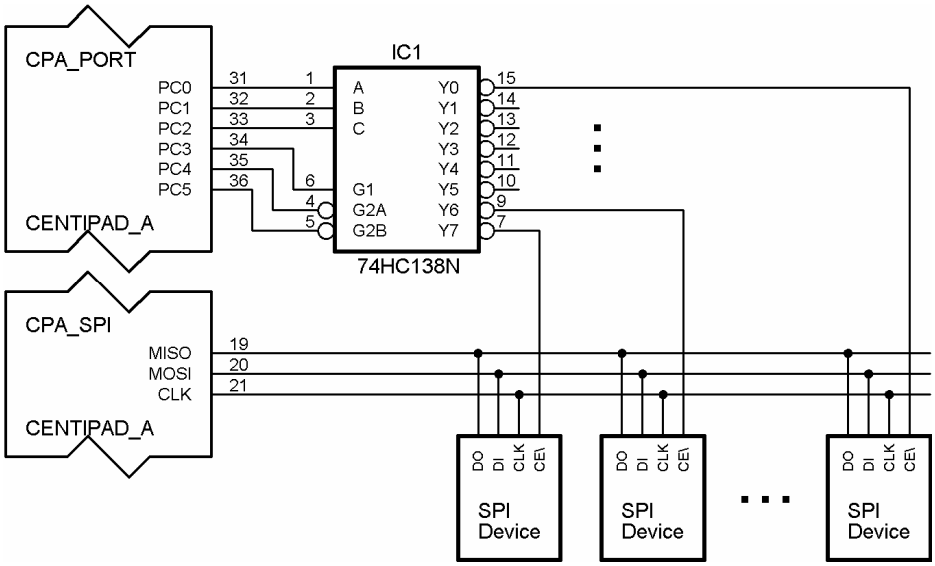
Der CAN-Controller ist über den SPI-Bus und NPCS1 mit der CPU verbunden. spitest demonstriert den Zugriff auf den SPI-Controller und sendet eine Folge von Kommandos an den CAN-Controller und damit an ein angeschlossenes Trinamic-Modul.

Die Demo wird mit **`spitest /dev/spi1`** gestartet. Der Motor des Trinamic-Moduls beschleunigt zunächst in eine Richtung bremst ab und dreht in die Ausgangsposition zurück.

`spitest` besteht aus folgenden Files:

- **`Makefile`**
- **`spitest.c`**
- **`types.h`**
- **`mcp2515.c`**
- **`mcp2515.h`**

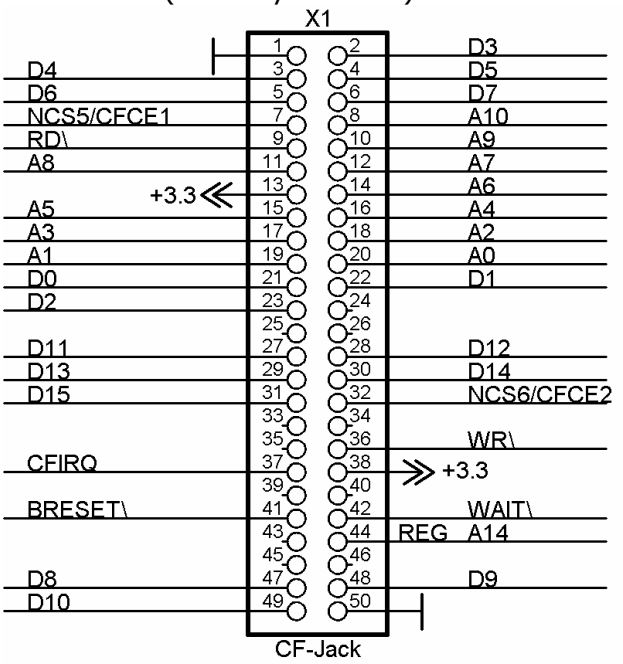
2.8.3. Anschluss eines Decoders für 8 SPI-Devices



2.9. Der Externe Datenbus

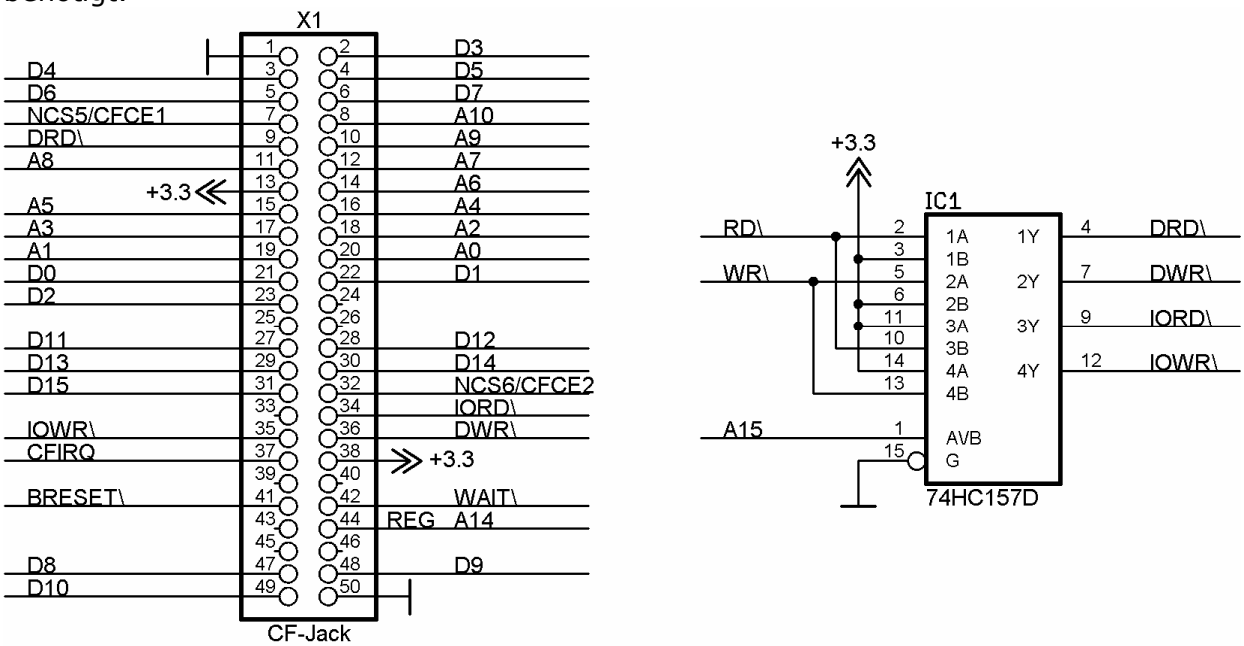
2.9.1. Anschluss von Compact Flash Devices

CF Devices (Memory Devices) können direkt ans CentiPad angeschlossen werden.

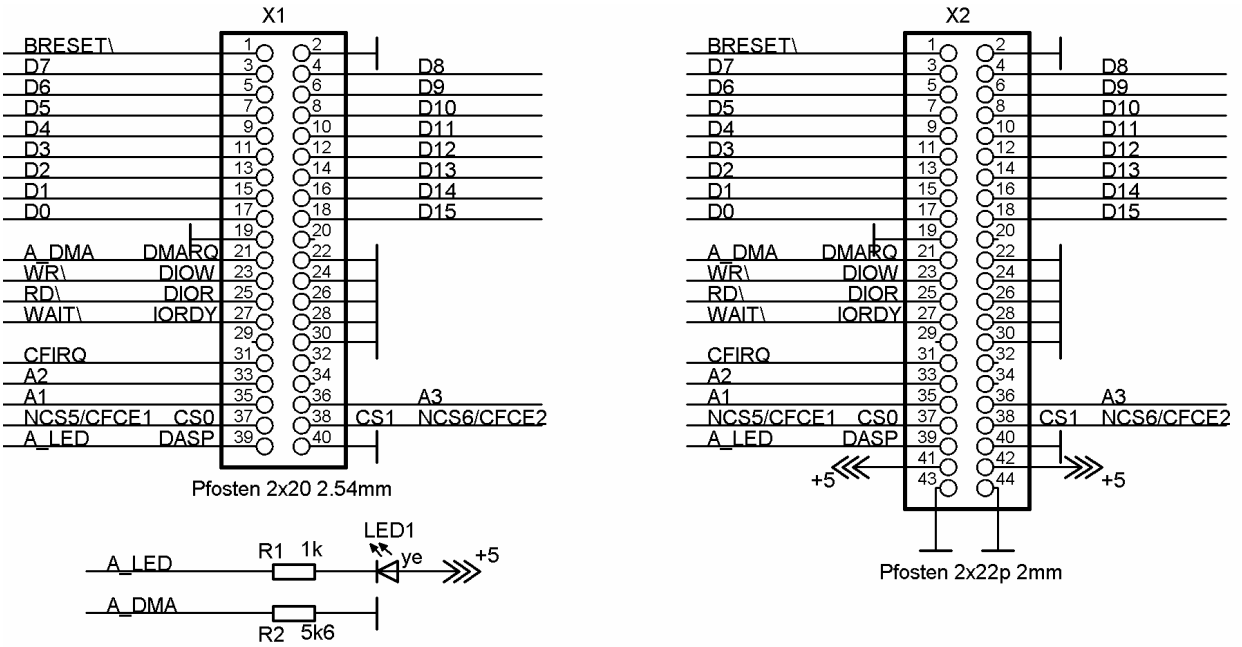


2.9.2. Decoder für CF+ Devices

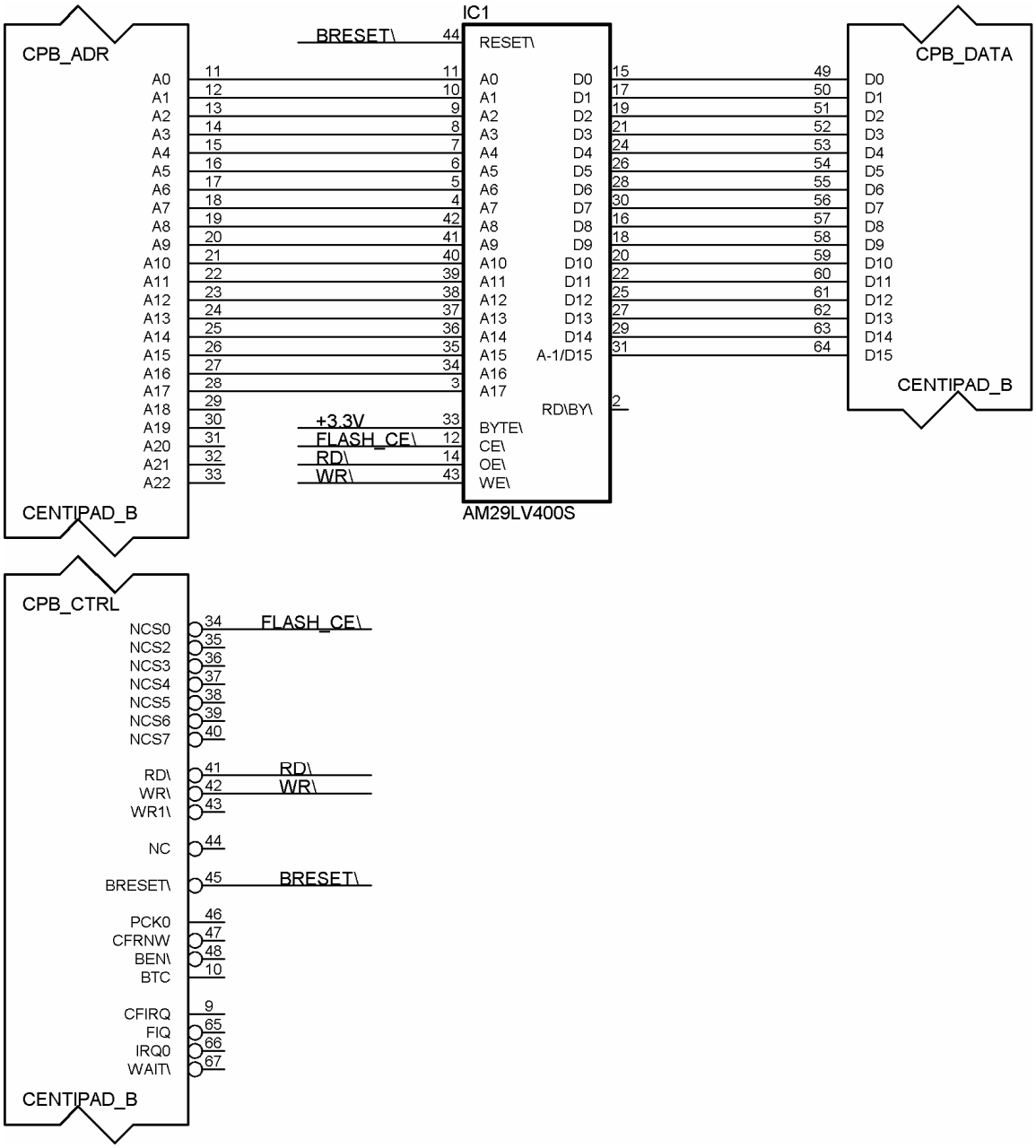
CF+ Devices benötigen haben ein erweitertes Interface, welches eine Dekoderlogik benötigt.



2.9.3. Anschluss von 3,5“ / 2“ IDE-Festplatten

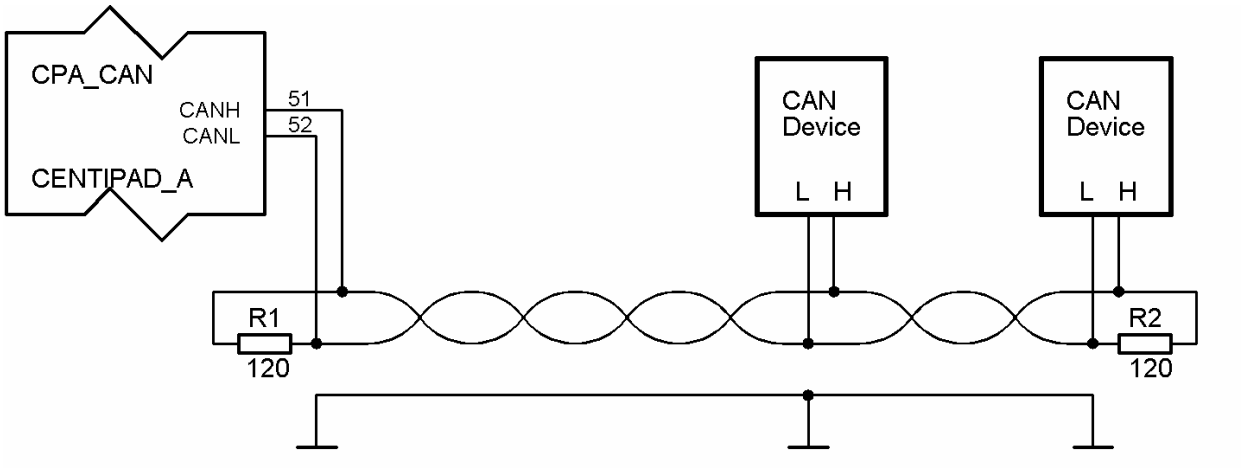


2.9.4. Anschluss eine Flash-Speichers an den externen Datenbus



2.10. Verwendung des CAN-Busses

2.10.1. Allgemein



2.10.2. cantest

Das Beispielprogramm cantest kommuniziert mit einem Trinamic PANdrive Modul. Hierfür stehen die Befehle ror, rol und stop zur Verfügung.
cantest dient als Beispiel für die Benutzung des CAN-Controllers und ist demonstriert gleichzeitig die Ansteuerung von Trinamic-Modulen.

cantest besteht aus folgenden Files:

- **Makefile**
- **cantest.c**
- **types.h**
- **at91_can.h**

Screenshot für die Bedienung von **cantest:**

```
/ # cantest
Usage:
 cantest ror <val>
 cantest rol <val>
 cantest stop
/ # cantest ror 100
Received message:
 - Identifier:  0x00000002 (standard)
 - Data Length: 7
 - Data:        0x01 0x64 0x01 0x00 0x00 0x00 0x64
/ # cantest stop
Received message:
 - Identifier:  0x00000002 (standard)
 - Data Length: 7
 - Data:        0x01 0x64 0x03 0x00 0x00 0x00 0x00
/ #
```

2.11.Verwendung des 1-Wire-Controllers

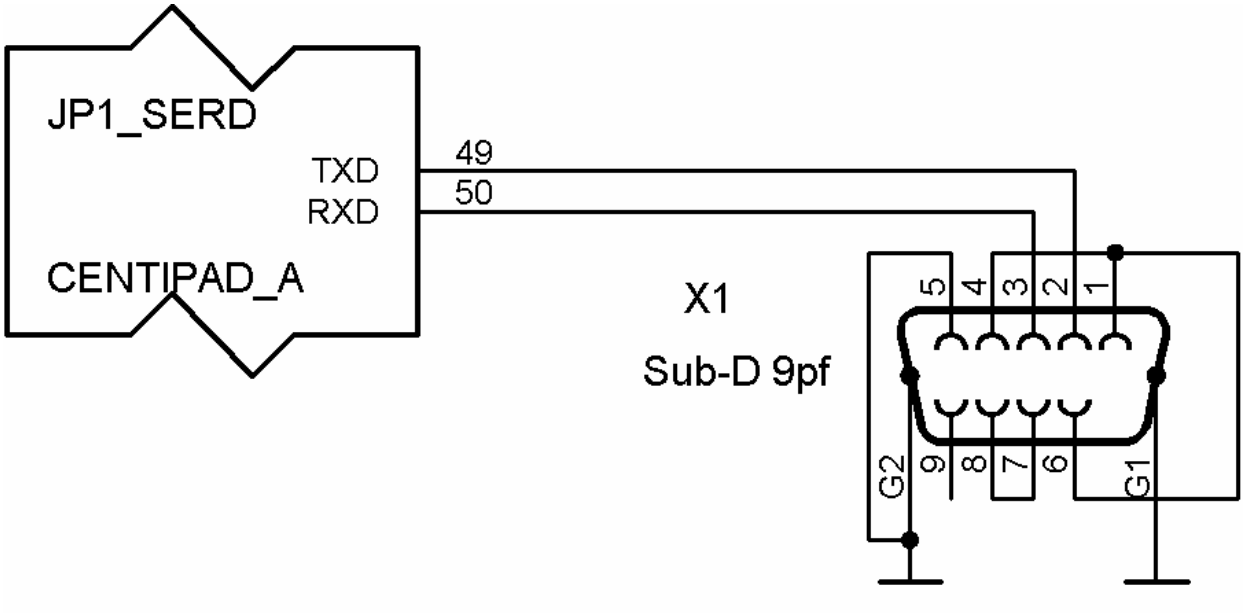
2.11.1. Applikations Zugriff auf den 1-Wire-Controller

Wie im TWI-Kapitel beschrieben, ist der 1-Wire-Controller physikalisch an das TWI angeschlossen. Die vorliegenden Bespiele zeigen den Zugriff auf das 1wire-Interface via TWI.
Ähnlich dem TWI sind auch für den 1wire-Bus Treiber vorhanden. Diese belegen die von Ihnen betreuten Devices. Alle im Folgenden beschriebenen Beispiele erfordern das Abschalten der Kerneltreiber.
Weitere Beispiele sind geplant.

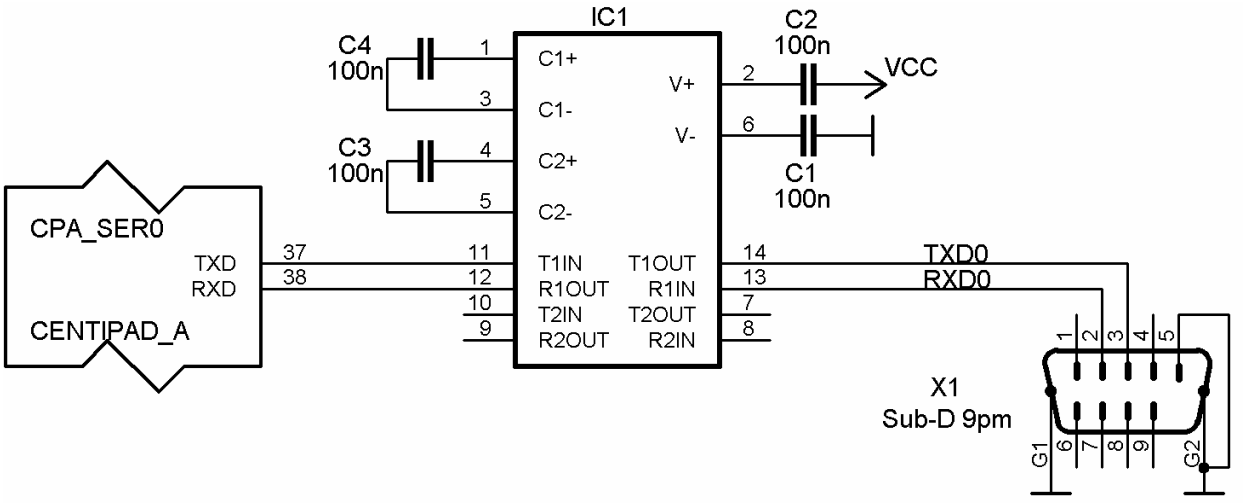
2.11.2. Temperatursensor DS18B20

BUG im original Linux-Treiber, zeigt falsche Temperatur. Korrektur in Planung.

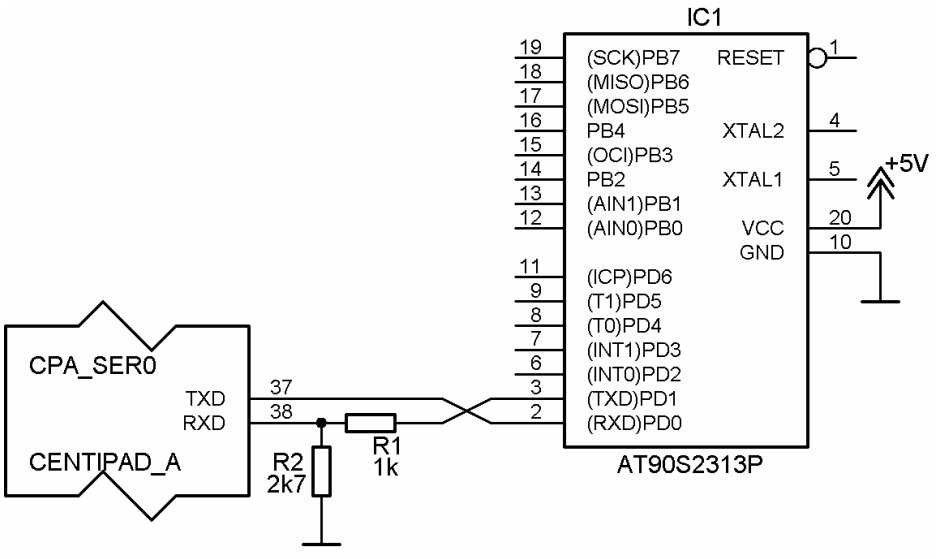
2.12.Serielle Schnittstelle – Debug



2.13.Pegelwandler für Ser0

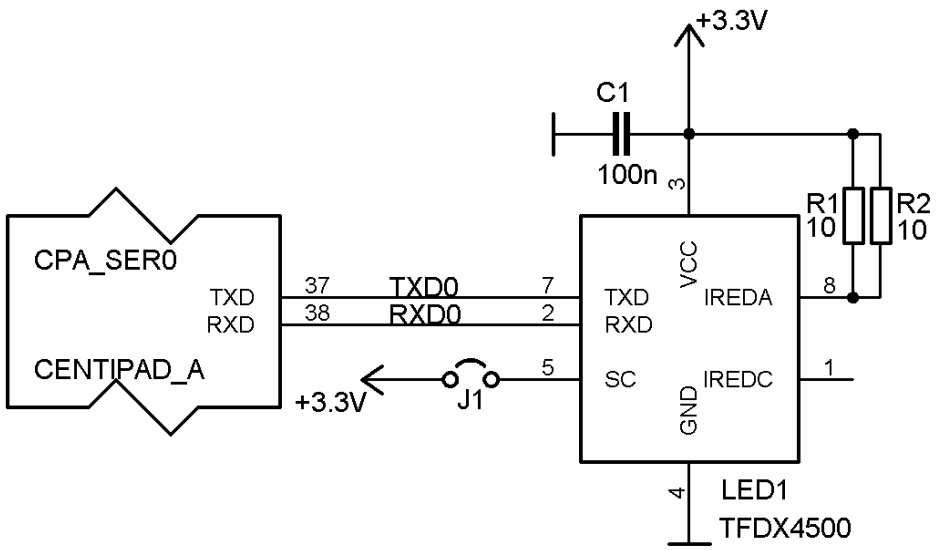


2.14.Anschluss eines 5V Peripheriegerätes an eine 3,3V serielle Schnittstelle



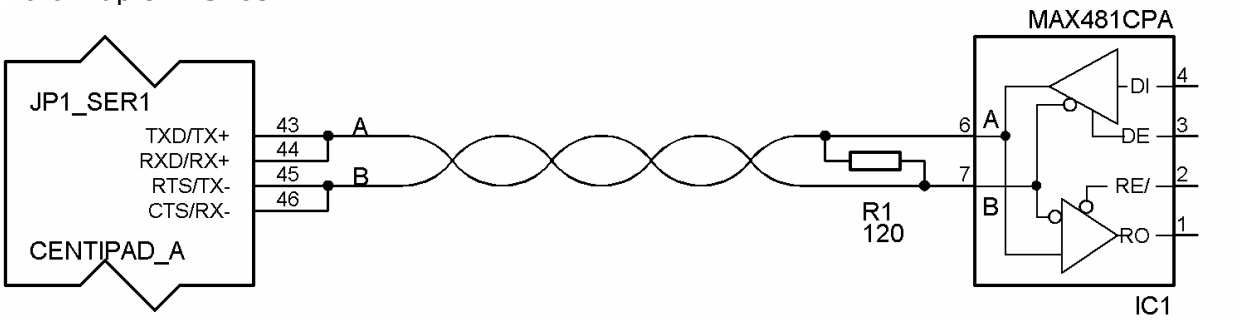
2.15.Ser0 mit exernem IRDA-Transceiver

Ser0 ist zum direkten Anschluss von IRDA-Transceivern geeignet. Die im Bild gezeigte Schaltung ist auf dem CentiBOB vorhanden.

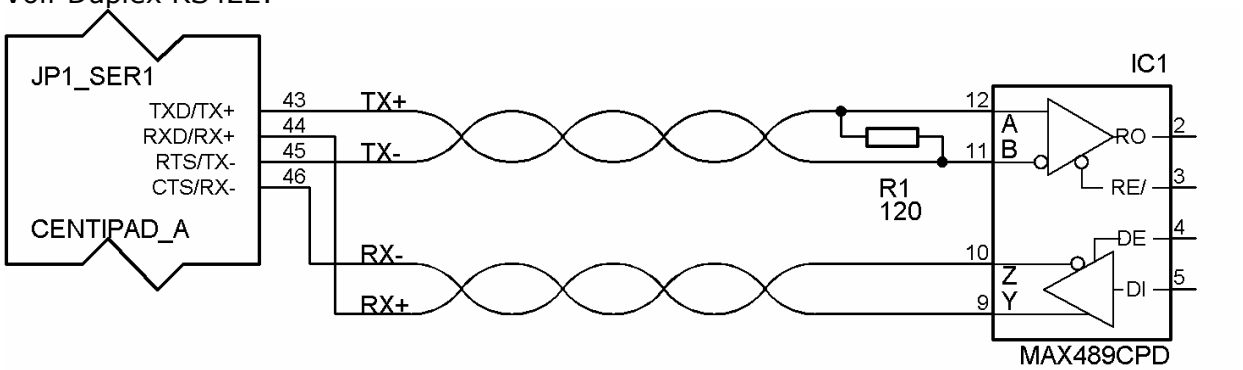


2.16.RS485

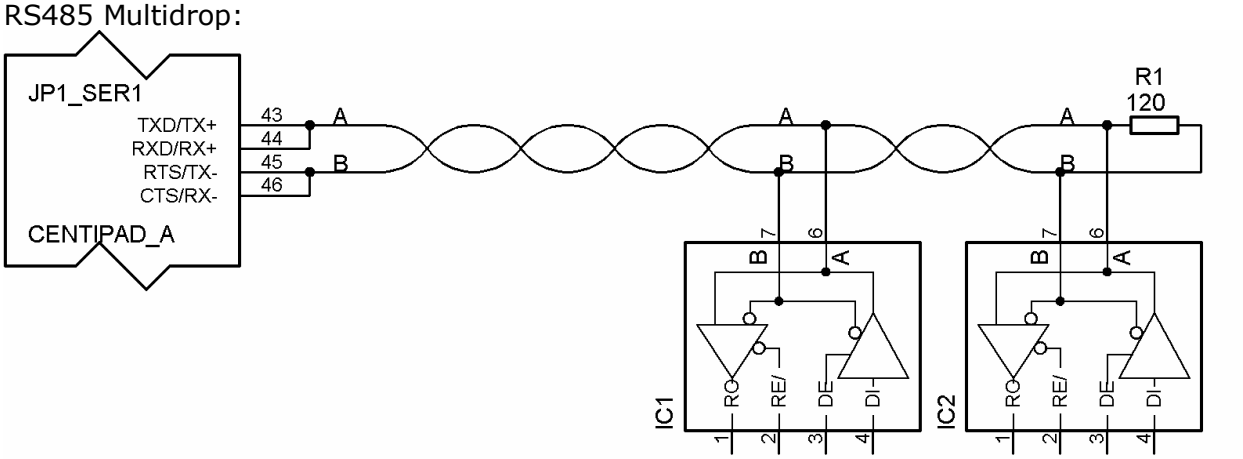
Halb-Duplex RS485:



Voll-Duplex RS422:

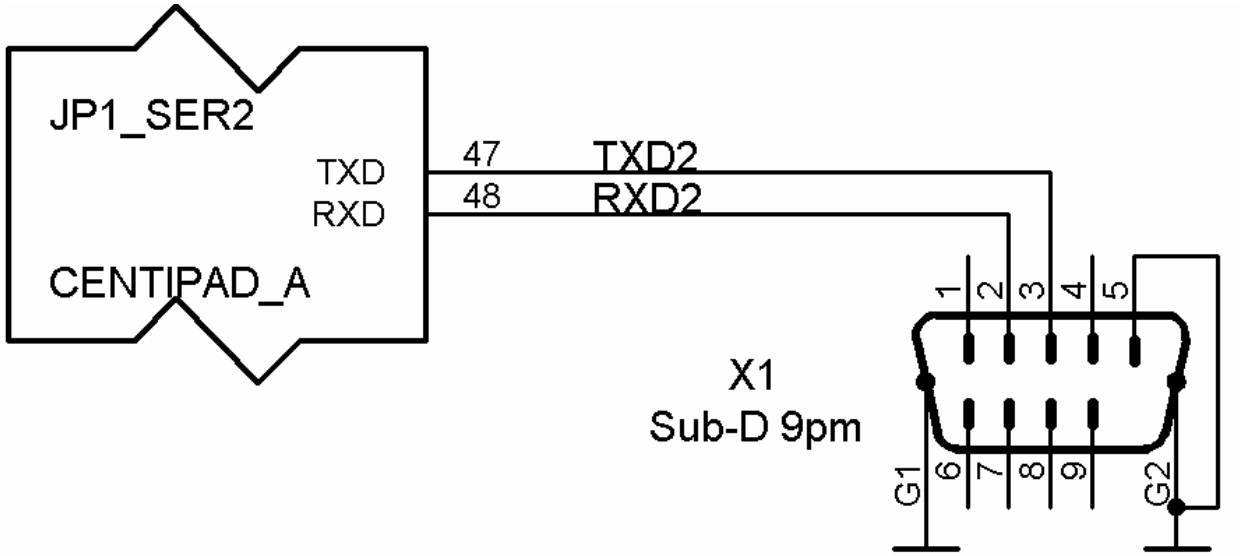


RS485 Multidrop:

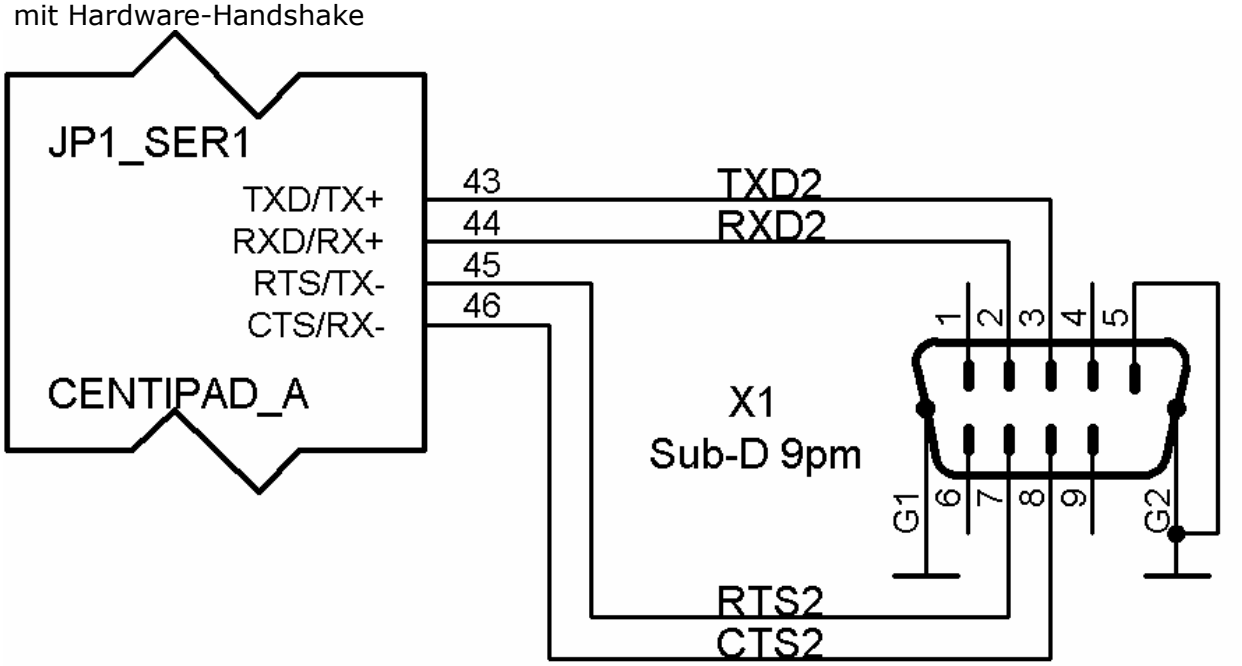


2.17.Serielle Schnittstelle Ser2

Software Handshake:



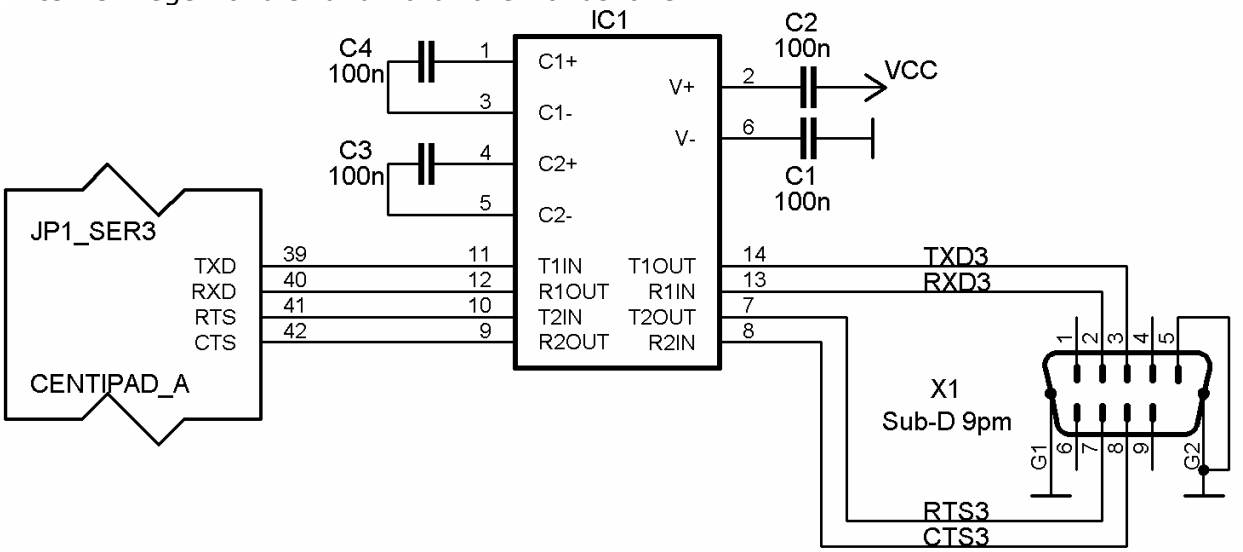
mit Hardware-Handshake



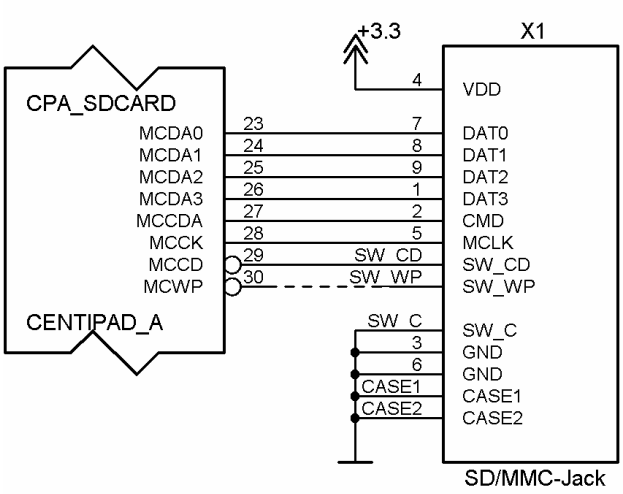
Wird eine serielle Schnittstelle mit Hardwarehandshake benötigt, so kann man Peripherie direkt wie im Bild gezeigt anschließen.

2.18.Serielle Schnittstelle Ser3

Externer Pegelwandler und Hardware Handshake:



2.19.Externe SDcard / MMC



Bei Bedarf kann der Anschluss für die SD-Card auch auf die Trägerplatine gelegt werden. Es darf jedoch immer nur eine Karte am Interface angeschlossen sein. Beim Layout ist auf kurze, HF-konforme Verlegung der Datenleitungen zu achten.

2.20.lighttpd - Mini Webserver

Die CentiPad Busybox enthält den Mini-Webserver **`lighttpd`**.

Die Web-Daten können beispielsweise unter ***`/application/www`*** abgelegt werden. In **`/application/www/Makefile`** wird dann im Abschnitt `install` der Befehl

```
mkdir -p ../demos/www
cp -v *.php ../demos/www
```

eingefügt.
Das Buildsystem kopiert dann die Daten automatisch ins Verzeichnis ***`/www`*** der `initrd`.
Wenn das lighttp Paket ausgewählt ist, dann wird automatisch ein Skript in die init.d kopiert, welches den Webserver startet.

2.20.1. PHP

CentiPad verfügt über einen PHP-Server. Programme hierfür werden wie Binärprogramme unter application editiert und mit entsprechenden Makefiles in das Image einkopiert. Beispiel: **`/demos/www`**.

2.20.2. CGI

Unter ***`/src/www`*** sind einfache Beispiele für CGI vorhanden. Hiermit kann über das Webinterface das **`lauflicht`** an- und ausgeschaltet werden.

2.21.USB Device Port – Bus Powered CentiPad

Ein USB Device darf vor der Enumeration laut Spezifikation USB Bus Powered Devices nur wenige Milliampere Strom aufnehmen. Aufgrund seines Stromverbrauchs hält CentiPad daher die Spezifikation für USB Bus Powered Devices nicht ein. Die Erfahrung (und Applikationen wie USB-Lampen und Batterieladegeräte) zeigt jedoch, dass viele Hostcontroller mehr Strom zur Verfügung stellen.
Daher kann CentiPad an vielen Hosts als Bus-Powered betreiben werden.
Wenn J3 gebrückt wird, dann ist die 5V-Versorgung des CentiPad über eine Diodenstrecke direkt mit der USB-Bus-Spannungsversorgung verbunden.
CPA/CPB Pin1 darf dann nicht mehr mit einer 5V Versorgung verbunden werden.

2.21.1. Serial Gadget Treiber

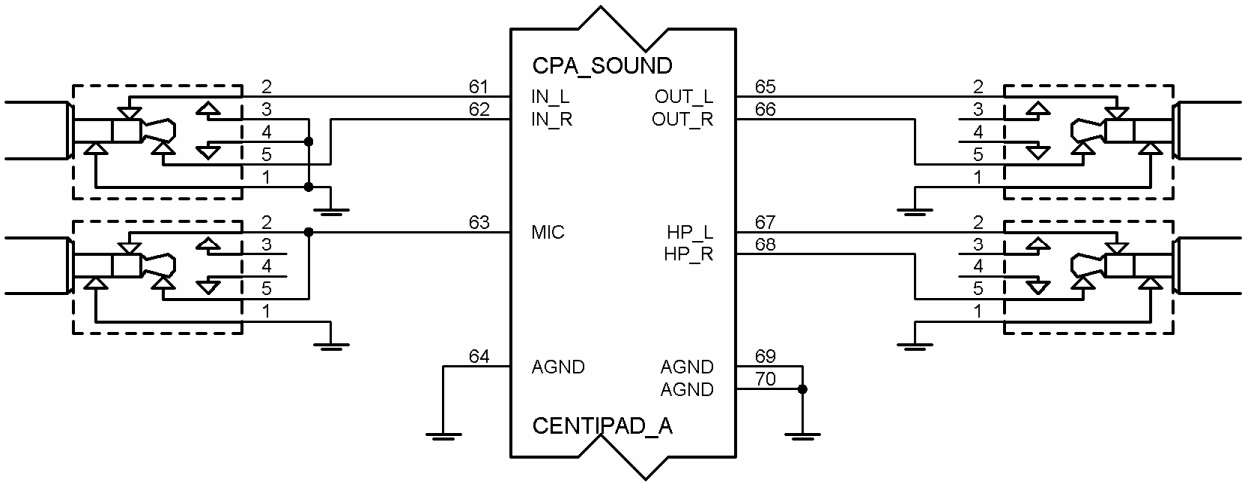
Für den USB Device Port ist ein Serial-Gadget-Treiber vorhanden.
Weitere Informationen sind unter **`Documentation/usb/gadget_serial.txt`** zu finden.

Auf dem Centipad:
`mknod /dev/ttygserial c 127 0`

Für Windows-Betrieb muss der Treiber mit der Option ACM geladen werden. Derzeit direkt im Quelltext eingestellt (**`"usb/gadget/serial.c"`** den **`define GS_DEFAULT_USE_ACM = 1`** gesetzt).

2.22.Sound auf dem CentiPad

2.22.1. Anschließen



2.22.2. Sound wiedergeben

Im folgenden Beispiel werden mp3-Files von der onboard SDcard wiedergegeben.

Falls noch nicht geschehen, muss die SDcard gemounted werden:
```
mount /dev/mmcblk0p1 /mnt/sdcard
```

Damit wir auch was hören, muss die Lautstärke im Kopfhörer passen:
```
echo "vol Vol 060" | smixer
```

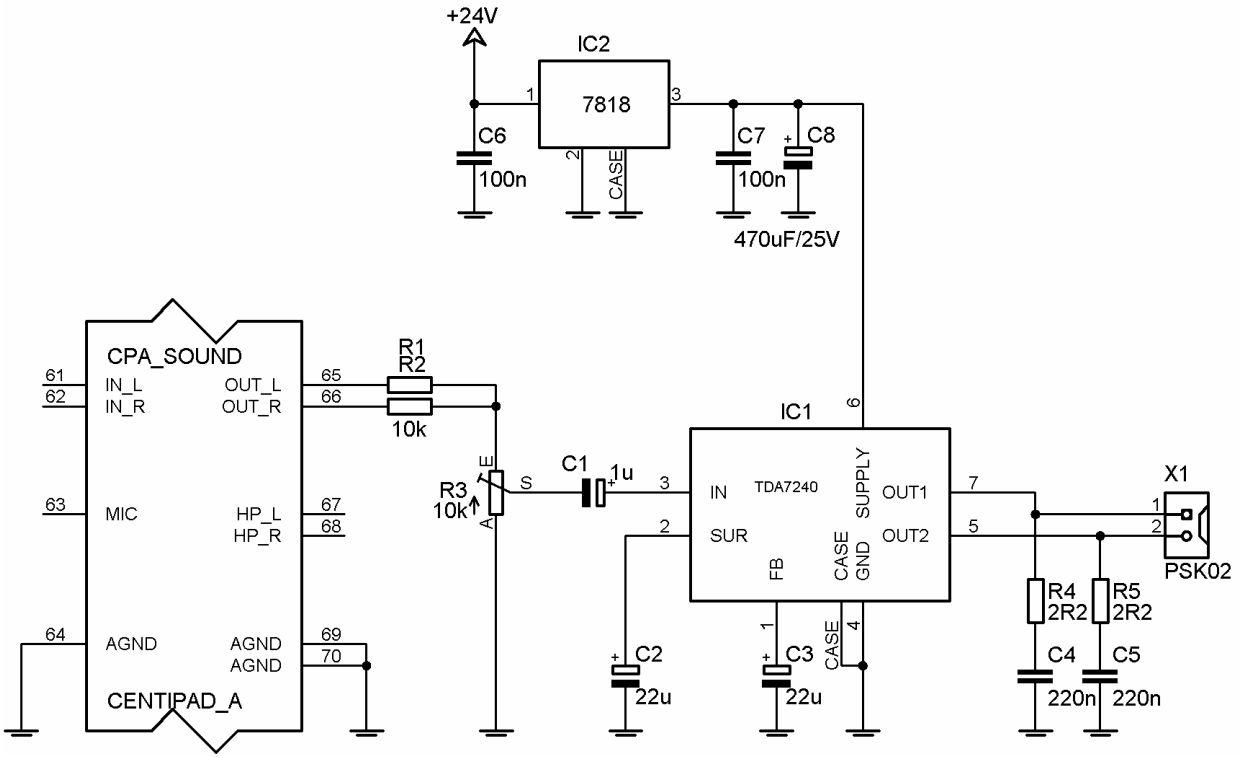
Zum Abspielen aller mp3-Stücke in der SDcard-root:
```
madplay -v /mnt/sdcard/*.mp3
```

Bitte beachten: madplay ist kommerzielle Software und wird nur zu Demozwecken beigelegt. Vor Verwendung in eigenen Projekten unbedingt die Lizenzrechte klären. Die kostenpflichtige Version von Madplay verwendet intern integer-Arithmetik und benötigt daher nur 8% der CPU-Zeit. Die Standardversion ist mit ca. 48% deutlich rechenzeitintensiver.

Alternativ könnte die Verwendung von lizenzfreien Kodieralgorithmen wie z.B. „Ogg Vorbis“ in Betrachtung gezogen werden.

2.22.3. 20W Leistungsverstärker

Der TDA7420A ist ein Audioverstärker IC für Car-Hifi-Anwendungen. Mit wenigen externen Bauteilen kann man so einen brauchbaren Verstärker aufbauen.
Die gezeigte Schaltung hält sich an das original Datenblatt. Die Materialkosten für die Elektronikkomponenten liegen deutlich unter 5€.
Wichtig ist, dass C1 einen geringen parasitären Widerstand hat. Als Versorgungsspannung dürfen 9V..18V angelegt werden. Die Verstärkung ist fest auf 40dB (Faktor 100) eingestellt. Zur Rauschreduzierung und Lautstärkevoreinstellung wird C1 am Mittelabgriff eines 10k Potentiometers angeschlossen. Wird C2 überbrückt, so schaltet der Verstärker stumm.
Als Lautsprecher können 4Ω oder 8Ω Typen verwendet werden.
Der Verstärker ist gegen Übertemperatur und Überstrom geschützt.
Wird Stereo gewünscht, kann man einfach zwei dieser Verstärker aufbauen.
Soll die Lautstärke per Software einstellbar sein, muss der Headphone-Ausgang verwendet werden, da der Soundchip nur hier über Lautstärkeregler verfügt.



2.22.4. Open Sound System

Das CentiPad Sound System basiert auf dem Open Sound System von Linux. Für eigene Anwendungen ist ein Blick in die Spezifikation sinnvoll.

2.22.5. Sound aufnehmen

Siehe **`centidev/platform/demos/soundcheck`**. Soundcheck gibt alle gesampelten Daten direkt zurück.

Zum Test:
`soundcheck 48000 1000 > /dev/null`

3. C++ auf dem CentiPad - cpptest

C++ bringt wesentliche Erweiterungen zur Sprache C. Der gewonnene Komfort wird jedoch durch eine umfangreiche Bibliothek libstdc++ erkauft. Diese stellt so große Anforderungen an die Systemresourcen, dass für CentiPad die Datei microstdcpp.c verwendet wird.

Ohne die C++ Standard-Bibliothek libstdc++ sind nur Funktionen der normalen libc verfügbar. Dies bedeutet insbesondere:

- keine Streams (cin, cout, cerr, etc...)
- keine globalen Objekte-Instanzen - alle Instanzen müssen entweder dynamisch erzeugt werden (mit new) oder lokal auf dem Stack liegen
- keine Objekt-Arrays (array = new Object()[200] - die Funktion zum iterativen Aufrufen der Konstruktoren fehlt in der microstdc++. Muss bei Bedarf implementiert werden. Normale Arrays sind kein Problem.
- keine Iteratoren, Container, Algorithmen die aus der STL kommen. Auch der Datentyp "string" gehört da dazu.

Jedoch funktionieren ohne Schwierigkeiten:

- virtuelle Methoden
- multiple Vererbung

Man gewinnt also ein C mit Objekten - als wirkliches C++ (nach Standard) kann man es so nicht mehr bezeichnen. Die Funktionalität ermöglicht jedoch die Entwicklung von objektorientierten Embedded Linux Applikationen mit optimiertem Speicherbedarf.

Das Verzeichnis ***centidev/platform/demos/cpptest*** enthält das Beispielprogramm **cpptest**. Im Verzeichnis ist die Bibliothek **microstdcpp.c** und ein **Makefile** zum Erzeugen von C++ Programmen.

Umfangreichere Beispiele sind geplant.