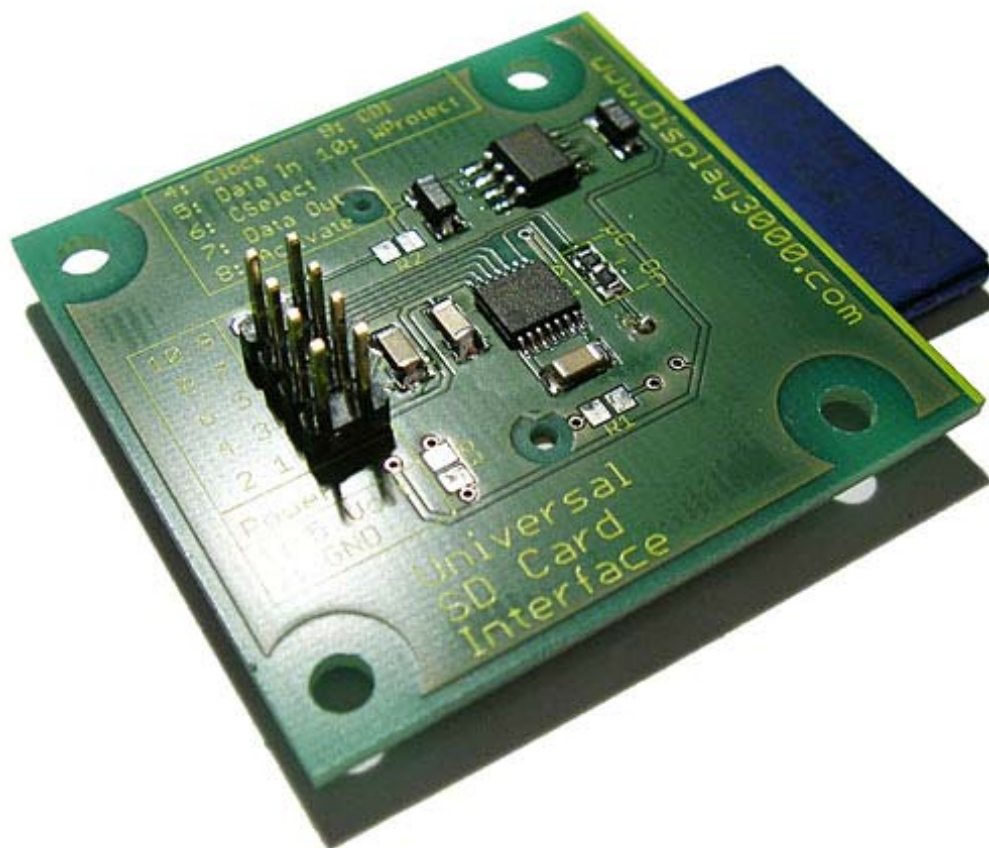


Handbuch für den Anschluss des SD-Karten Moduls P001

V 0.94
04. Mai 2008



© 2008 by Peter Küsters

Dieses Dokument ist urheberrechtlich geschützt. Es ist nicht gestattet, dieses Dokument zu verändern und komplett oder Teile daraus ohne schriftliche Genehmigung von uns weiterzugeben, es zu veröffentlichen; es als Download zur Verfügung zu stellen oder den Inhalt anderweitig anderen Personen zur Verfügung zu stellen. Zuwiderhandlungen werden verfolgt.

Herzlichen Glückwunsch zum Erwerb des SD-Kartenmoduls.

Mit diesem Modul können Sie zukünftig SD Karten als externe Speichereinheit für Ihren Mikrocontroller nutzen.

Folgendes WICHTIGES aber direkt vorab

- 1) SD-Karten sind kein RAM sondern Flash-Speicher. Jede Speicherzelle der Karte hat eine Lebensdauer von ca. 100.000 Schreibzyklen. Angeblich soll so eine Karte selber dafür sorgen, dass die Speicherzellen gleichmäßig beschrieben (verbraucht) werden (d.h. , wenn Sie eine Karte immer nur zu 10% beschreiben, demnach 1 Million Schreibzugriffe möglich wären). Trotzdem: Bei einer falschen Programmierung (Schleife mit permanentem Schreibzugriff) ist so eine Karte somit binnen 1 Sekunde zu zerstören. SD Karten sind also nur geeignet, um Daten längerfristig abzuspeichern bzw. als Datenlogger, nicht um ständig und permanent Daten dort upzudaten.
- 2) Die Software für die eigentliche Kommunikation mit der SD Karte stammt nicht von uns. Bitte sprechen Sie den Autor der Software-Treiber bei Fragen an.
- 3) Es gibt sicher tausende unterschiedlicher SD Karten am Markt. Nicht jede wird einwandfrei mit dem Mikrocontroller zusammenarbeiten wollen. Manche sind sehr langsam, obwohl sie als High-Speed ausgezeichnet wurden. Wenn also eine Karte nicht wie gewünscht funktioniert, probieren Sie andere Karten aus. Wir können hier nicht helfen.
- 4) Die übliche Software kann nur mit Dateinamen bis 8 Zeichen Länge arbeiten. Speichern Sie also nur Dateinamen ab, die nicht mehr als 8 Buchstaben haben (+3 für den Extender). Beispiel: „Logo1.bin“ ist OK, „Hintergrundbild.bin“ ist nicht OK.

Wichtiges zum Datenverlust:

Wenn Sie auf die Karte Schreiben möchten, so führt ein Reset, Abschalten oder Stromausfall etc. dazu, dass eine evtl. noch geöffnete Datei später nicht mehr lesbar ist, die Dateigröße nicht stimmt bzw. Cluster verloren gingen weil sie noch nicht in die FAT eingetragen wurden.

Daher: Vor dem Abschalten müssen zum Schreiben geöffnete Dateien immer geschlossen werden. Daher sind u.E. zwei Maßnahmen sinnvoll:

Wenn kleinere Datenmengen eher selten bzw. mit Abständen geschrieben werden sollen:

Datensätze zuerst im RAM anlegen und dann alle x Minuten die Daten auf einen Schlag abspeichern oder z.B. 512 Byte zum Schreiben „eingesammelt“ wurden. Dazu vorher die Datei Öffnen und danach wieder Schließen. Das Gleiche gilt beim „Anhängen“ von Daten. Auf diese Weise verliert man nur die Daten seit dem letzten Abspeichern - es sei denn, der Stromausfall passiert genau während des Schreibzugriffs – dann haben Sie Pech gehabt (evtl. Strategie: 2 Dateien gleichzeitig führen und nacheinander Beschreiben).

Wenn große Datenmengen geschrieben werden sollen

Dies tritt i.d.R. nicht permanent auf, denn dies würde (s.o.) schnell zu einem Erreichen der 100.000 Schreibzyklen führen. Daher gilt dann auch hier: Datei Öffnen – Daten ablegen/anhängen – Datei Schließen

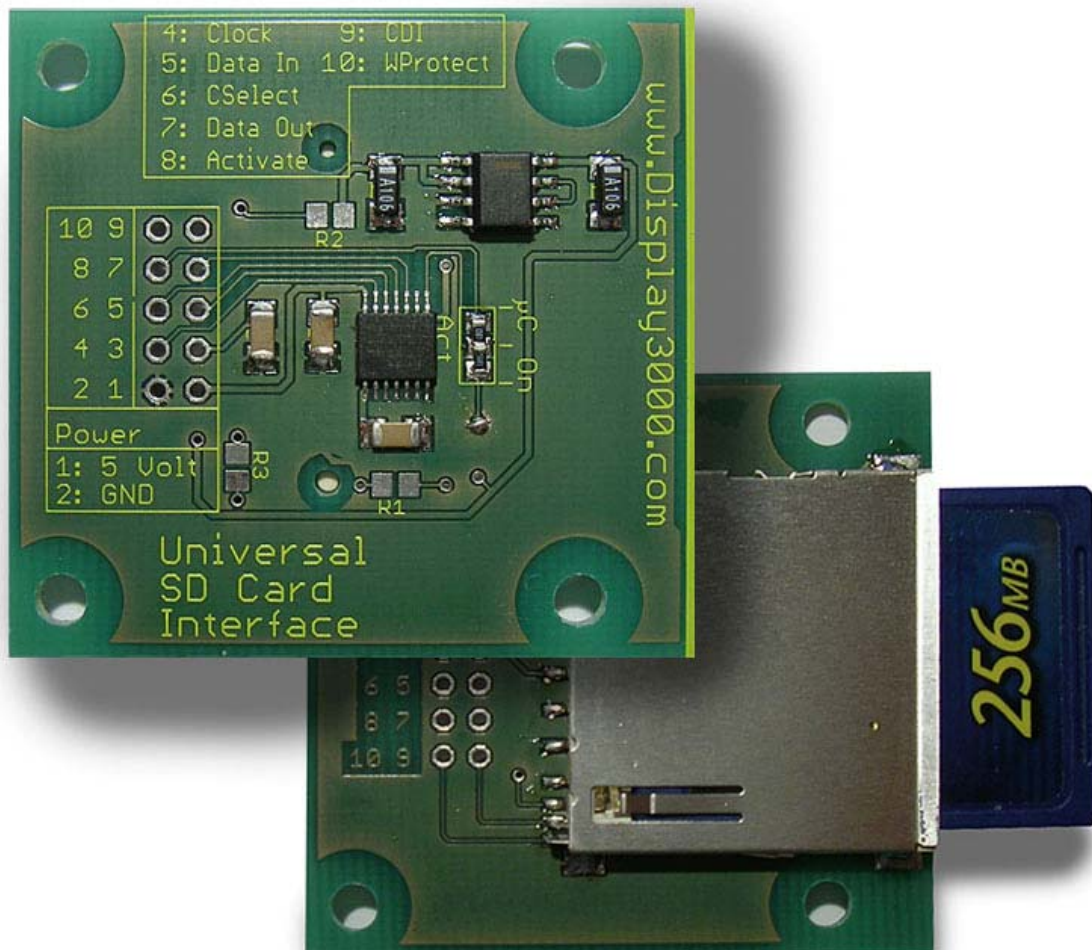
Das SD-Kartenmodul erfüllt mehrere Aufgaben:

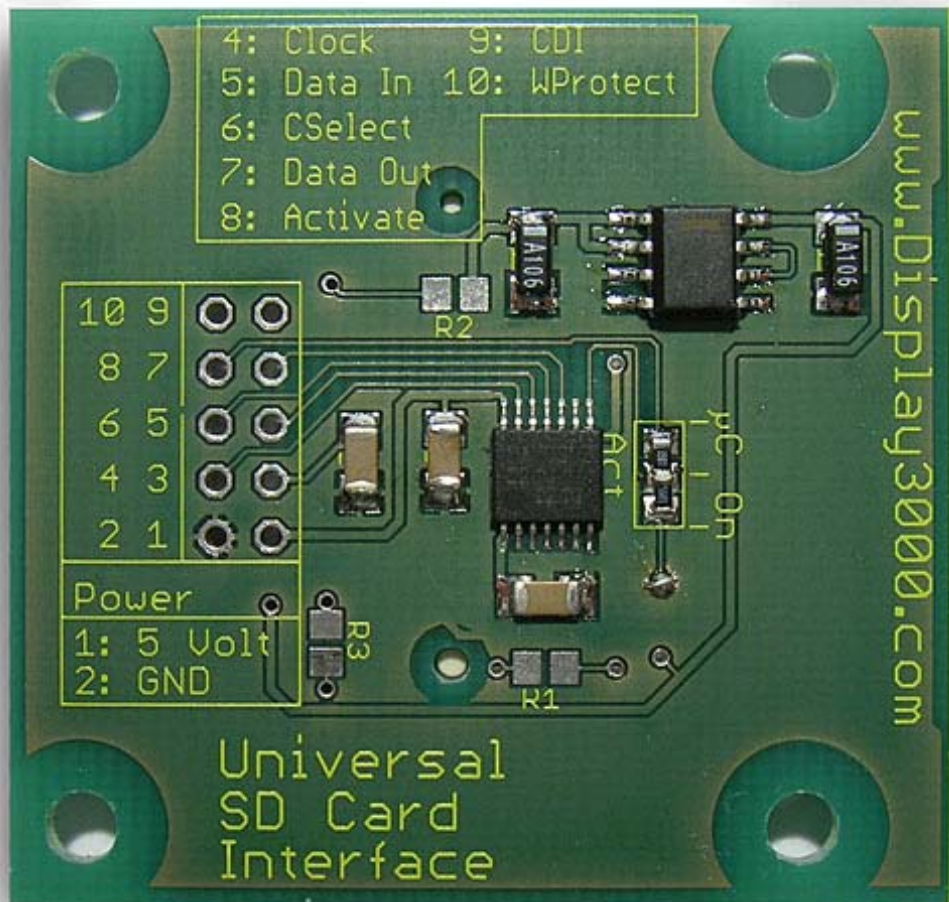
- Steckkontakt für SD-Karten
- Spannungsregler von 5 Volt auf 3 Volt
- Bidirektionaler Pegelwandler von 5 Volt (Mikrocontroller) zu 3 Volt (SD-Karte) und zurück
- Abkoppelung des kompletten Moduls vom SPI Bus durch Tristate-Ausgänge

Vielleicht stellen Sie sich die Frage, „Warum so kompliziert, im Internet kursieren Anschlusspläne, die mit ein paar Widerständen auskommen“.

Nun, im Anhang dieses Dokuments zeigen wir dies genauer. Sie erkennen dort deutlich, dass eine verlässliche Lösung ein wenig Aufwand benötigt.

Die Erweiterungsplatine





Die Platine ist mit Bestückungsdruck ausgestattet und erlaubt so die einfache Kontaktierung. Sie benötigen für den Betrieb der Karte die folgenden Leitungen:

- 1: 5 Volt** Eingang
- 2: Masse**
- 3: 3 Volt Ausgang** des Spannungsreglers an, sollten Sie also noch 3 Volt für ein anderes Gerät benötigen: hier können Sie sich bedienen (ca. 50mA)
- 4: Clock-Signal** des Mikrocontrollers
- 5: Datensignal (MOSI)** vom Mikrocontrollers (MOSI am Mikrocontroller)
- 6: CS** – Solange aktiv, geht die Karte auf Empfang für Daten / Befehle
- 7: Datensignal (MISO)** von der Karte (MISO am Mikrocontroller)
- 8: Activate** – 5 Volt an diesem Anschluss sind zwingend notwendig, um die Tristate Ausgänge zu aktivieren. Dieses Signal ist entweder vom Mikrocontroller mittels eines Ports zu schalten, oder wenn nicht notwendig, permanent mit 5 Volt zu verbinden um die SD-Karte ansprechen zu können
- 9: CDI: Kartenerkennung:** Liegt an Masse, wenn eine Karte eingesteckt ist
- 10: Write Protect:** Liegt an Masse, wenn der Schreibschutzschalters an der SD-Karte auf Position Schreibgeschützt liegt (Achtung: Das müssen Sie in der Software abfragen, die Karte selbst kann auch mit gesetztem Schreibschutz beschrieben werden)

Anschluss an den SPI-Bus / Mikrocontroller

Sinnvoll ist es, die SD-Karte über den SPI Bus des Mikrocontrollers zu betreiben, da dieser in der Regel hardwaremäßig vom Controller bedient wird und das Übertragen der einzelnen Bits keine Performance kostet.

Am Beispiel eines ATmega128 sind dann folgende Anschlüsse minimal notwendig:

- 1: 5 Volt Eingang
- 2: Masse
- 4: SCK (B1)
- 5: MOSI (B2)
- 6: beliebiger Port für das CS Signal z.B. B0 wie in unserem Software-Beispiel
- 7: MISO (B3)
- 8: beliebig bzw. sonst an 5V

Anmerkung: Der Stecker der SD-Kartenplatine kann nicht einfach per Flachbandkabel 1:1 mit einem unserer Controllermodule verbunden werden. Die Pinbelegung der Kontakte ist nicht identisch.

Selbstverständlich steht es Ihnen frei, die SD-Karte mittels Software-SPI an jeden beliebigen anderen Port anzuschließen. Dann fallen auch die im nächsten Abschnitt beschriebenen möglichen Probleme weg – allerdings geht dies zu Lasten der Geschwindigkeit. Wenn hohe Geschwindigkeit kein Muss ist, ist Software-SPI über beliebige andere Ports (eben nicht den festliegenden Hardware SPI Ports) die bessere Lösung.

Achtung: Über die Kabel laufen Signale mit mehreren Mhz Taktfrequenz ! Sie können hier kein Kabel von 30 oder 40 cm anschließen. Halten Sie das Kabel so kurz wie möglich. Mehr als 10 cm sollten es nach Möglichkeit nicht sein.

Tristate-Ausgänge

1) Neben der Spannungsversorgung an 5V und Masse MUSS der Eingänge *Activate* (Pin 8) am SD Kartenmodul an 5 Volt liegen, damit die SD Karte Daten erhält bzw. sendet. Sie können diesen entweder immer an 5 Volt legen, wenn keine Störungen durch andere Busteilnehmer zu erwarten sind, oder Sie legen *Activate* an einen Port des Mikrocontrollers und entfernen somit das komplette SD Kartenmodul vom Bus, indem Sie diesen Port auf Low ziehen.

Wenn Sie andere Busteilnehmer haben, sollten Sie so vorgehen, dass Sie den *Activate* Port auf 1 ziehen, bevor Sie auf die SD Karte zugreifen und direkt danach wieder auf 0.

2) SPI Geräte liegen zwar am gleichen Bus, können aber durchaus unterschiedliche Datenformate erwarten. Im Falle eines unserer Farb TFTs und der SD Karte liegt dieser Fall z.B.

vor:

TFT: Polarity = Low, Phase = 0

SD Karte : Polarity = High, Phase = 1

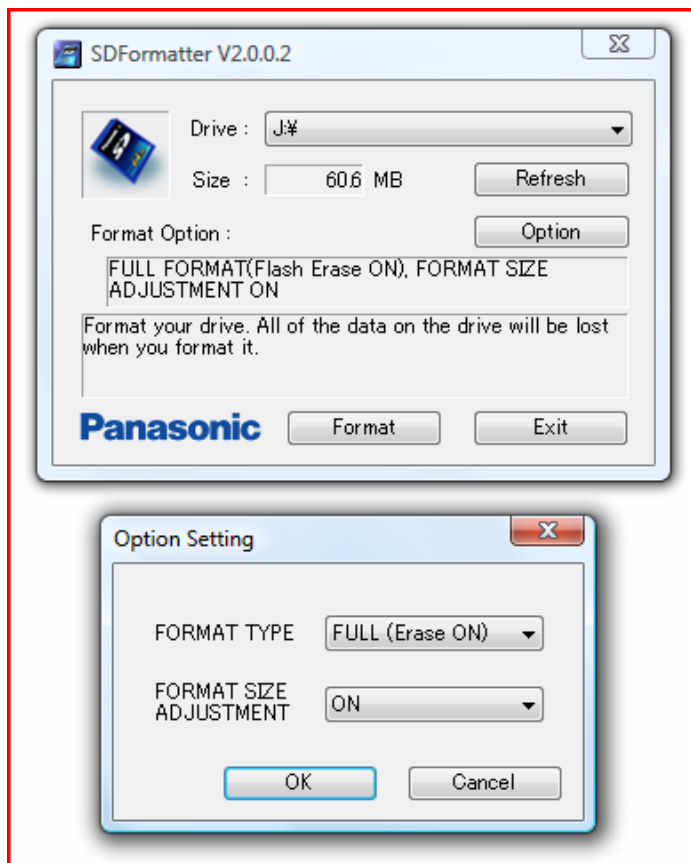
Dies bedeutet, Sie müssen vor einem Zugriff auf die Karte bzw. Display nicht nur *Activate* entsprechend schalten sondern auch den SPI Bus umkonfigurieren! Im Bascom Beispiel haben wir dies z.B. bereits für Sie durchgeführt. Dort gibt es zwei Routinen *Activate_Display* und *Activate_SD* die Sie einfach nur entsprechend aufrufen müssen. Tipp: Sie können z.B. *Activa-*

te_Display zusammen mit einer Abfrage, ob die Aktivierung überhaupt notwendig ist, auch in der Routine LCD_Window unterbringen – dann wird das Display wenn notwendig automatisch aktiviert.

Formatierung der SD Karte

Wenn die Karte beim ersten Test korrekt angesprochen wird, brauchen Sie sich um nichts zu kümmern. Wenn Sie auch nach einem Formatierungsversuch unter Windows nicht korrekt erkannt wird bzw. nicht angesprochen werden kann, versuchen Sie, die Karte mit einem speziellen Tool für SD Karten zu formatieren.

Unter folgendem Link: http://panasonic.co.jp/pavc/global/cs/sd/download/sd_formatter.html erhalten Sie von Panasonic das Programm SD Formatter, mit welchem Sie ihre SD Karte vor der Nutzung formatieren sollten. Aus Copyright-Gründen können wir hier nur den Link posten und Ihnen das Tool leider nicht direkt übermitteln.



Das Handbuch zu dieser Software erhalten Sie hier:

<http://panasonic.jp/support/global/cs/sd/download/ftp/manual2007e.pdf>

Als Backup falls der obige Link einmal nicht funktionieren sollte: Programm und Handbuch befinden sich auch hier: <http://www.sdcard.org/about/downloads/>

Die Software zum Ansprechen der SD Karte

Es gibt verschiedene Protokolle zum Auslesen und Beschreiben einer SD-Karte. Einige sind nicht für die Öffentlichkeit protokolliert und bedürfen einer Lizenzierung. Ein freier Zugriff ist allerdings mittels des SPI-Bus möglich. Dies ist zwar die langsamste Zugriffsart, aber für einen Mikrocontroller reicht die Geschwindigkeit in der Regel aus.

Die SD-Karte kann als Speichereinheit sektorweise beschrieben oder gelesen werden - dies ist per Software noch relativ einfach zu lösen aber oft doch unpraktisch, da man ein „richtiges“ Dateisystem gewohnt ist. So möchte man z.B. Daten von der Karte Lesen, die vorab mittels eines PCs auf sie geschrieben wurde, oder man möchte mit dem PC Daten auslesen, die der Mikrocontroller auf die Karte geschrieben hat. Dann geht kein Weg daran vorbei, das FAT Dateisystem der Karte zu nutzen. Dieses FAT Dateisystem muss der Mikrocontroller jedoch verwalten und das ist aufwändig und speicherintensiv. Ein kleiner Mikrocontroller mit wenig RAM wird hier u.U. bereits Probleme bekommen, Mind. 1 KByte RAM sollten schon verfügbar sein. Beim Lesen von der Karte und beim Schreiben auf die Karte gilt grundsätzlich: je größer der Puffer (Cache) im RAM hierfür, desto schneller ist dieser Vorgang.

Wie schon erwähnt, haben wir selbst haben keinerlei eigene Software für den Zugriff auf SD Karten programmiert. Nachfolgend geben wir Ihnen ein paar Hinweise und Hilfestellung zur Nutzung der SD Karten Treiber..

Wenn Sie Ihre Karte nicht ansprechen können, sind wir jedoch der falsche Ansprechpartner hierfür. Wir kennen diese Software nicht gut genug um Ihnen helfen zu können. Nutzen Sie dann bitte die diversen Internetforen oder bei Bascom auch das Supportforum des Herstellers unter www.mcselec.com

Zugriff auf die Karte mit Bascom Basic

Mit Bascom wird bereits AVR DOS mitgeliefert, welches den Zugriff auf die SD Karte erlaubt. Dieses stammt von Herrn Voegel. Seine Internetseite rufen Sie unter folgendem Link auf: <http://members.aon.at/voegel/>. 4 KByte Programmspeicher und ca. 1 K Ram werden hier mindestens benötigt.

Beachten Sie bitte, das AVR DOS nur für den privaten Gebrauch lizenziert ist. Wenn Sie kommerzielle Applikationen erstellen, dann müssen Sie AVR DOS unter obiger Internetseite lizenzieren. Eine Company-Lizenz kostet jedoch lediglich 99,95 Euro und ist somit erschwinglich. Unter dieser Internetseite erhalten Sie zudem alle wichtigen Informationen rund um die in Bascom integrierten Kommandos.

Für einen Test von AVR DOS mit einem unserer Grafikmodule erhalten Sie mit der Karte das Programm SDCardtest.bas sowie die dazugehörige Grafikdatei Grafik1.bin und Grafik2.bin, welche Sie bitte beide auf eine SD Karte kopieren (vorher bitte Ix wie weiter oben beschrieben, formatieren).

Zu allen weiteren Befehlen konsultieren Sie bitte die Webseiten zu AVR DOS bzw. die Onlinhilfe von Bascom. Und nochmals: Denken Sie immer daran, dass Sie jede Speicherzelle einer Flash Karte ca. 100.000 mal beschreiben können. Also programmieren Sie bitte keine Schleifen, die permanent mehrere Male pro Sekunde Daten auf die SD-Karte schreiben.

Schnellstart mit Bascom

Im Verzeichnis \SD_Card finden Sie ein Beispielprogramm Namens SDCardtest.bas sowie die notwendigen Bilder für Ihre SD Karte und die AVR_DOS Software.

Dieses Programm sowie Config_MMC.bas ist für unser Modul D071 oder D072 vorbereitet. Sie brauchen lediglich am Anfang von SDCardtest.bas Ihren Mikrocontroller sowie den genutzten Quarz festlegen. Alles Weitere ist bereits vordefiniert und Sie brauchen nur noch die SD Karte an das D071/D072 anschließen.

Die notwendigen Verbindungen:

Pin am SD Modul	Bezeichnung	Pin am D071/D072
1	5 Volt Eingang	Dort wo 5V anliegen
2	Masse	Dort wo Masse anliegt
4	Clock	B1
5	Data In (μ C->Karte)	B2 (Mosi)
6	CS	B0
7	Data Out (Karte-> μ C)	B3 (Miso)
8	Activate	E7
3, 9, 10		Hier nicht benötigt

Für unserer 2.1“ Display benötigt diese Software noch die üblichen 3 Librarydateien von Ihrer CD:

GLCD21_Display3000.bas
 GLCD21_fonts.bas
 Init21_Display3000.bas

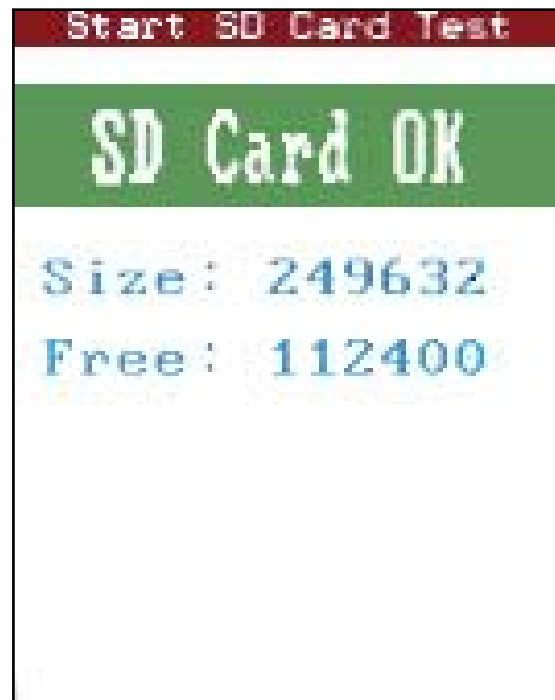
In Init21_Display3000.bas nehmen Sie bitte noch eine Änderung vor: damit wir die Daten auch schnell genug aus der SD Karte lesen können, müssen wir ein Array definieren, welches alle Pixel einer kompletten Zeile beinhaltet. Maximal können dies 176 Pixel sein – pro Pixel 2 Byte macht insg. 352 Byte. Das Array gibt es schon, aber es muss vergrößert werden. Bitte am Anfang dieser Datei die Arraygröße ändern in: **Dim Data_array(352) As Byte**

Kompilieren Sie die Software und übertragen sie dann auf das Modul:

Als erstes wird die Karte initialisiert und bei Erfolg die Kartengröße und der noch verfügbare Platz angezeigt (siehe Foto rechts).

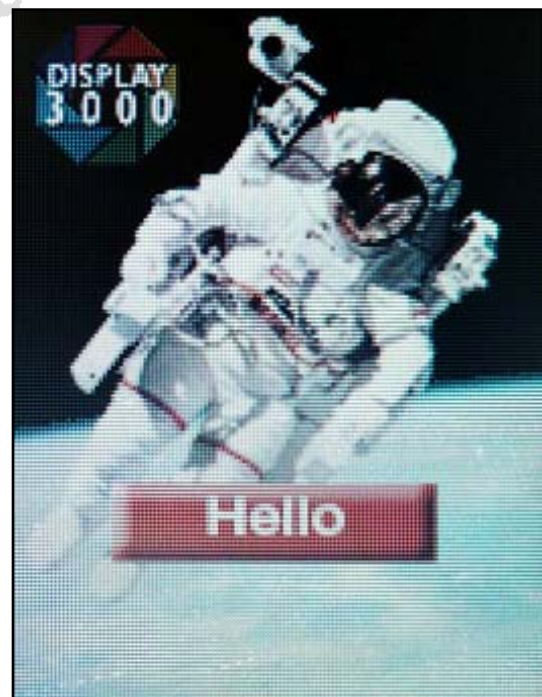
Sehen Sie auf dem Display allerdings mehrere Sekunden lang (ein kurzes Aufblitzen ist OK) eine rote Meldung „No SD Card?“, so wird die Karte überhaupt nicht angesprochen und vermutlich stimmt etwas nicht mit Anschluss oder der Software im Allgemeinen.

Wenn „No SD Card?“ nur kurz aufblitzt, aber dann auch nach mehreren Versuchen (Meldung „Retry“) immer noch kein Bild erscheint, so ist vermutlich Ihre Karte nicht OK oder nicht kompatibel (Formatieren ?).



Ist aber alles OK werden nach ca. einer Sekunde von der Karte 2 Bilddateien gelesen und dargestellt. Zuerst das Vollbild, danach wird der Button nachgeladen und dargestellt.

Hinweis: Gegenüber dem ursprünglichen Anwendungsbeispiel von Herr Voegel haben wir den Start der Initialisierungsroutine in Bascom etwas geändert. Es kann passieren, dass die Karte beim ersten Ansprechen nicht ordnungsgemäß reagiert und einen zweiten oder dritten Versuch benötigt. Daher brechen wir nicht direkt beim ersten Mal ab, sondern Initialisieren die Karte bis zu 10 Mal. Erst dann gilt sie als nicht verfügbar. **Dieses Vorgehen hat sich als sinnvoll herausgestellt und sollte von Ihnen übernommen werden.**



So, das Kartenmodul funktioniert und wir haben Ihnen den Einstieg mit Bascom Basic erleichtert. Für den Rest sind Sie nun zuständig!

Grundsätzlich geht der Zugriff mit Bascom wie folgt vonstatten:

- 1) Mit Config_MMC.bas wird die SD Karte eingelesen, Config_AVR-DOS.BAS lädt dann den Support für die meisten DOS Kommandos nach.
- 2) Config_MMC.bas MUSS von Ihnen noch an Ihre aktuelle Hardware angepasst werden.
- 2) Mit Open # wird eine bestimmte Datei eröffnet oder angelegt.
- 3) Mittels Get# und Put# etc. werden Daten gelesen bzw. geschrieben.
- 4) Close # schließt die Datei wieder. Dies ist wichtig uns sollte niemals unterbleiben – also nicht einfach den Controller vom Strom nehmen. Alles weitere erfahren Sie auf den Internetseiten von Herrn Vögel oder in der Bascom-Online-Hilfe.

Auflistung der Bascom-Befehle, die mit AVR DOS zusammenarbeiten:

Disk/Directory:

[InitFileSystem](#) (<Partition#>)
[DiskSize](#)
[DiskFree](#)
[Kill](#) <FileName>
[Dir](#) ([<FileName>])
[FileLen](#) ([<FileName>])
[FileDateTime](#) ([<FileName>])
[FileDate](#) ([<FileName>])
[FileTime](#) ([<FileName>])
[GetAttr](#) ([<FileName>])
[Name](#) <OldFileName> As <NewFileName>
[ChDir](#) <PathName>]
[MkDir](#) <PathName>]
[Rmdir](#) <PathName>])

File create, open, read, write, close:

[FreeFile](#)
[Open](#) <FileName> For Input/Output/Append/Binary as #<File#>
[Close](#) <File#>
[Flush](#) [<File#>]
[Print](#) #<File#>, Variable1; Variable2; ...
[Write](#) #<File#>, Variable1, Variable2, ...
[Input](#) #<File#>, Variable1, Variable2, ...
[Line Input](#) #<File#>, StringVariable
[Get](#) #<File#>, <Variable> [,<Position>]
[Put](#) #<File#>, <Variable> [,<Position>]
[Seek](#) #<File#> [,<New Position>]

Infos about opened files:

[EOF](#) (#<File#>)
[LOC](#) (#<File#>)
[LOF](#) (#<File#>)
[FileAttr](#) (#<File#>)

Others:

[Bload](#) <FileName>, <SRAM-Address>
[Bsave](#) <FileName>, <SRAM-Address>, <Length>

Mehr hierzu erfahren Sie auch in der Bascom-Hilfe. Suchen Sie einfach nach „AVR-DOS“.

Änderungen in der Datei Config_MMC.bas

Wenn Sie Hardware-SPI nutzen möchten, so passen Sie in der Datei Config_MMC.bas bitte am Anfang die Variable Const Cmmc_soft an.

Const Cmmc_soft = 0 bedeutet Hardware-SPI,
Const Cmmc_soft = 1 bedeutet Software-SPI.

Für das **Hardware-SPI** liegen bestimmte Portanschlüsse bereits fest.

Hier ATmega128 / ATmega2561 / AT90CAN128:

Clock: B1

MOSI: B2

MISO: B3

CS legen Sie selber fest und müssen der Software diese Leitung auch mitteilen. CS kann jeder beliebige Port sein. Und immer daran denken: CS bitte auch als Output definieren!

Alternativ: Für das **Software-SPI** legen Sie bitte dann weiter unten im Programm die entsprechenden Ports selber fest. Bitte denken Sie auch hier daran, sie als Eingang bzw. Ausgang zu definieren.

An der Datei Config_AVR.bas können Sie ebenfalls Änderungen vornehmen (z.B. Speicherbedarf, mehrere gleichzeitig geöffnete Dateien etc.). Bitte lesen Sie hierzu die Online-Hilfe von Bascom (Suchwort AVR-DOS). Für den normalen Betrieb brauchen Sie hier jedoch keine Änderungen vornehmen.

Zugriff mit C (WinAVR)

Hier gibt es verschiedene Projekte im Internet, die den Zugriff auf eine SD Karte erlauben. Eine Software finden Sie z.B. unter <http://www.ulrichradig.de/home/index.php/avr/mmc-sd>

Auch hier müssen wir Sie bitten, bei Fragen zum Ansprechen der SD-Karte mittels dieser Software an den Autor zu wenden.

Wir werden in den nächsten Tagen auch hier ein funktionierendes Beispiel mit dieser Source oder evtl. einer anderen zusammenstellen. Grundsätzlich funktioniert es jedoch genauso wie im Bascom Code gezeigt. **Auch für C gilt hier:**

Vor dem Ansprechen der SD Karte:

- SPI Schnittstelle definieren (falls zuvor z.B. für das Display geändert)
- die Activate Leitung auf High setzen

Nach dem Ansprechen der SD Karte

- Activate Leitung auf Low
- vor dem Ansprechen des Displays: SPI Schnittstelle definieren

Datenblöcke

Die SD-Karte kann als Speichereinheit blockweise beschrieben oder gelesen werden - dies ist per Software noch relativ einfach zu lösen. Oft möchte man jedoch Daten von der Karte Lesen, die vorab mittels eines PCs auf sie geschrieben wurde, oder man möchte mit dem PC Daten auslesen, die der Mikrocontroller auf die Karte geschrieben hat. Dann geht kein Weg daran vorbei, das FAT Dateisystem der Karte zu nutzen. Dies muss der Mikrocontroller jedoch verwalten und das ist aufwändig und speicherintensiv. Ein kleiner Mikrocontroller mit wenig RAM wird hier u.U. bereits Probleme bekommen, 2 KByte RAM sollten schon verfügbar sein. Beim Lesen von der Karte und beim Schreiben auf die Karte gilt grundsätzlich: je größer der Puffer (Cache) im RAM hierfür, desto schneller ist dieser Vorgang.

Wenn Sie genug Speicher zur Hand haben, lesen/schreiben Sie immer gleichzeitig 512 Byte (=1 Sektor). Dies geht wesentlich schneller, als einzelne Bytes zu übertragen.

Nutzung der SD Karte ab Ablage für Grafikdaten zusammen mit unseren Farb-TFTs

Die Ablage von größeren Grafikdaten auf einer SD-Karte bietet sich an, kostet doch dieser Speicherplatz verhältnismäßig wenig. Allerdings gibt es hier bei Nutzung der schnellen Hardware-SPI-Schnittstelle ein grundsätzliches Problem, welches umgangen werden muss:

Fakt ist:

- Es ist grundsätzlich möglich, mehrere Busteilnehmer am gleichen SPI Bus zu betreiben, in diesem Fall also Display und SD Karte
- Der Mikrocontroller hat weniger RAM als die einzuladende Bitmapdatei an Datenmengen hat (eine Vollgrafik bei z.B. 176x132 Pixeln benötigt 45 KByte)
- Somit muss die große Datenmenge in mehreren Schüben eingelesen und an das Display geschickt werden.
- Am SPI Bus (und somit an Mosi und Clock) hängen Display und SD Karte gleichzeitig.
- Display und Karte haben zwar eigene CS-Signale
- Zur Festlegung des Bildbereiches, wird dieser mittels LCD_Bitmap vorab definiert

Angenommen, wir möchten 45 x 1 KByte Bilddaten einlesen und diese dann jedes mal an das Display senden..... so führt dies zu einem Problem. Wir müssen, bevor wir zum 2. Mal die SD Karte ansprechen, die CS-Leitung des Displays auf 1 setzen (also deaktivieren), damit die Kommunikation mit der SD Karte nicht vom Display als Daten angenommen werden und als falsche Pixel erscheinen. Leider bricht jedoch das Display dann auch die aktuelle Funktion ab (wir hatten ja einen Bildbereich definiert, der nun nach und nach mit Pixeln gefüllt werden sollte). Nach dem Lesen des zweiten 1KByte-Blocks an Daten können wir nicht dort fortsetzen, wo das letzte Pixel gesetzt wurde. Als Ausweg böte sich an, 45 KByte RAM für das Einlesen zur Verfügung zu stellen, aber dies ist in der Regel viel zu aufwändig.

Hier hilft nur ein Workaround:

Eine eigene Bitmap-Routine für die Übernahme von Datenblöcken. Als Beispiel finden Sie auf der CD das Programm SDCardtest.bas, welches eine Bitmapgrafik inkl. dem Abruf von Daten der SD Karte automatisch zeilenweise durchführt – d.h. es wird immer nur eine Zeile an Daten angefordert und an das Display geschickt. LCD_Window wird also bei einer 100 Pixel hohen Grafik auch 100x aufgerufen. Die Y-Position ist dann jedes Mal um 1 erhöht, das Fenster jedes Mal auch nur 1 Pixel hoch.

Eines können Sie jedoch mit einer solchen Routine nicht mehr nutzen: Komprimierte Grafiken – durch den zeilenweisen Aufbau mit einem Abruf von genau so vielen Pixeln wie die Zeile breit ist. ist diese Option nicht mehr nutzbar.

Anmerkung: Theoretisch könnte die Routine natürlich erweitert werden, so dass auch die Nutzung von komprimierten Daten auf der SD Karte möglich wäre. Allerdings ist der Speicher auf der SD Karte in der Regel so reichlich bemessen, dass man die Komprimierung hier auch weglassen kann.

Umgekehrt ist dies übrigens dies kein Problem, da unser SD-Kartenmodul mit Tristate-Eingängen und -Ausgängen versehen ist, und das Display einfach nach Erledigung eines Tasks komplett vom Bus abgekoppelt werden kann.

Links:**Generell:**

<http://www.sdcard.org/home>

[http://www.sdcard.org/about/memory_card/pls/Simplified Physical Layer Spec.pdf](http://www.sdcard.org/about/memory_card/pls/Simplified_Physical_Layer_Spec.pdf)

<http://www.chipcatalog.com/Doc/BC6DC20FC60BAC7595E5D0419DB6126E.pdf>

AVR Bascom:

<http://members.aon.at/voegel/>

<http://www.mcselec.com/>

<http://staff.ltam.lu/feljc/electronics/bascom/>

Links zu C-Code:

http://elm-chan.org/docs/mmc/mmc_e.html

http://www.ulrichradig.de/site/atmel/avr_mmc/d/p/hitachi_hb28b128mm2.pdf

http://www.embedded-os.de/index.html?pcfat_port.htm

<http://www.avrfreaks.net/index.php?module=Freaks%20Files&func=viewFile&id=1436&showinfo=1>

<http://www.efsl.be/>

<http://www.avrfreaks.net/index.php?module=Freaks%20Files&func=viewFile&id=2251&showinfo=1>

<http://www.roland-riegel.de/sd-reader/doc/>

<http://www.cc5x.de/MMC/>

<http://dfn.dl.sourceforge.net/sourceforge/efsl/manual-0.2.8.pdf>

Kabellänge – Wichtig !

Über die Kabel laufen Signale mit mehreren Mhz Taktfrequenz ! Sie können hier kein Kabel von 30 oder 40 cm anschließen – das Übersprechen zwischen den Leitungen, die kapazitive Belastung und die eingefangenen Störungen würden jede Kommunikation mit der Karte zu nichte machen. Halten Sie das Kabel so kurz wie möglich. Mehr als 10-20 cm sollten es nach Möglichkeit nicht sein. Notfalls fangen Sie bei einem ersten Test mit einem kurzen Kabel an und versuchen dann ein längeres, wenn es denn unbedingt sein muss. Zusätzlich können Sie bei der SPI Kommunikation die SPI-Taktfrequenz heruntersetzen – dann sind auch längere Leitungen möglich (aber gleichzeitig sinkt natürlich der Datendurchsatz).

Übersicht mit funktionierenden Karten

BITTE: Senden Sie uns Größe, Hersteller und wenn möglich eine ID Nummer der Karte (meist auf der Rückseite) jeder Karte, die Sie als funktionsfähig identifiziert haben (perfekt wäre noch ein Foto der Karte auf einem weißem Blatt Papier als Hintergrund). Wichtig wäre auch die Information ob die Karte sofort funktionierte, oder ob sie zuerst formatiert werden musste. So können wir eine Datenbank aufbauen, die die funktionierenden SD Karten auflistet und auch die Karten zeigt, die evtl. Probleme machen. Dies kommt langfristig allen zu Gute.

Die jeweils aktuelle Liste ist dann unter folgendem Link abzurufen:

http://www.display3000.com/downloads/SD_CARD_Reports.pdf

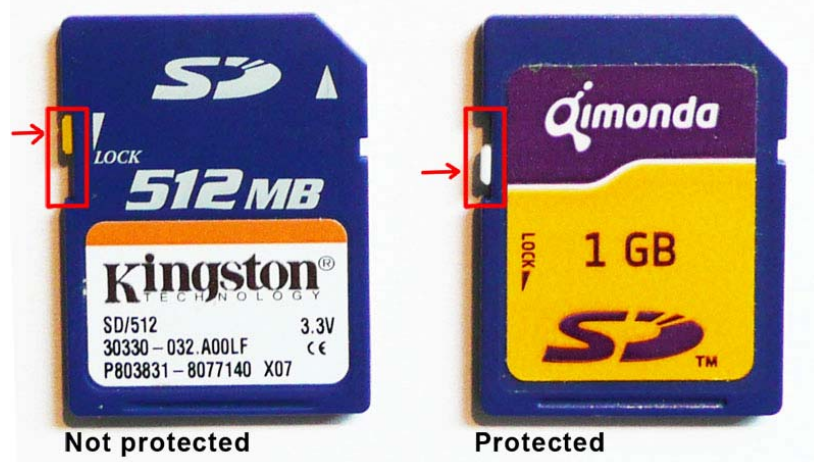
Dankeschön !!

Der Schreibschutz-Schalter:

Der Schreibschutz-Schieber an der SD-Karte wird nicht von der Karte selbst abgefragt. Es ist kein interner Schalter, sondern lediglich ein Kunststoffschieber ohne jegliche Funktion für die Karte selbst. Sie können somit die Karte immer beschreiben, egal in welcher Position sich der Schalter befindet.

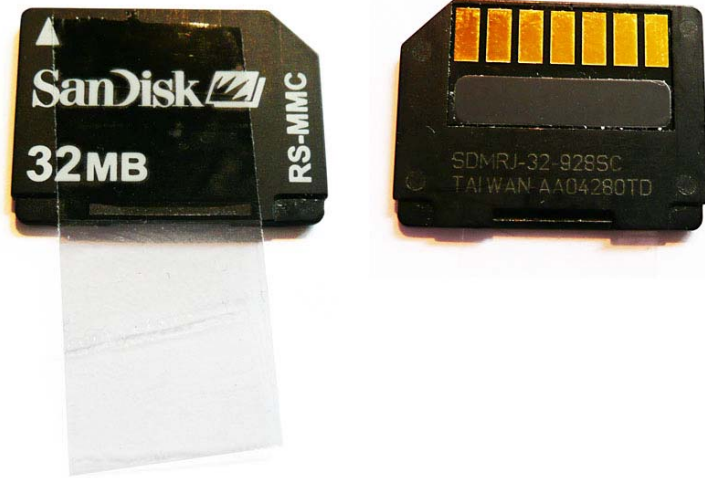
Die einzige Art der Auswertung geschieht durch den Schreibschutz-Kontakt an unserem Modul. Sie müssen also bei Bedarf diesen Schreibschutzschalter vom Mikrocontroller auswerten und in Ihrer Software das Schreiben unterbinden.

Wenn Sie keinen Schreibschutz benötigen, werten Sie den Kontakt am Modul einfach nicht aus.



MMC Karten

Mit unserem Modul können auch MMC Karten per SPI gelesen und beschrieben werden. Die Prozedur ist identisch zu einer SD Karte – auch die wichtigen Kontakte sind identisch zu einer SD Karte.



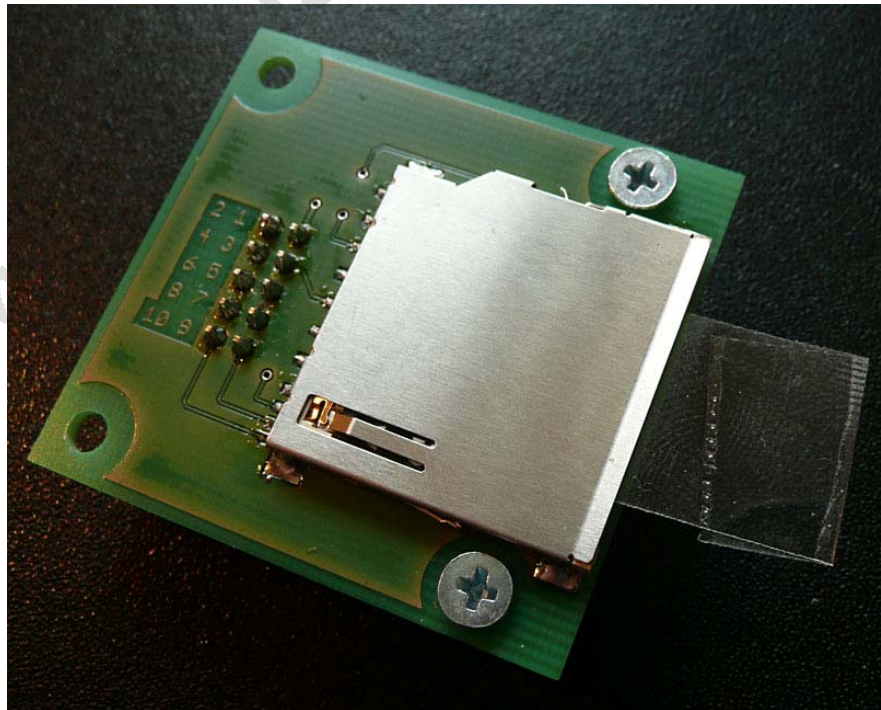
Eigentlich stände der Nutzung einer MMC Karte nichts im Wege eigentlich.... Es gibt nur ein winziges Problem dabei: Eine MMC Karte ist zu kurz und passt nicht einfach so in unser Modul.

Mit einem kleinen „Work around“ ist dies aber kein Problem:

- a) Die Karte muss beim Einsetzen von hinten noch „angeschoben“ werden, damit sie weit genug in das Modul gedrückt wird. Dies kann ein Stift sein oder eine andere Karte.
- b) Allerdings gibt es nach a) ein Problem, wenn Sie die Karte wieder entnehmen möchten – die Karte sitzt zu tief und Sie kommen nicht an sie heran um sie herauszuziehen.

Daher sollten Sie die Karte wie auf dem Bild gezeigt, mit einem Streifen Klebeband o.ä. versehen, damit Sie sie daran bei Bedarf auch wieder herausziehen können.

Wenn Sie nach a) nicht weiter gelesen haben oder sich das Klebeband gelöst hat, können Sie die Karte auch von den Lötkontakten des Kartenhalters aus herausdrücken: über den Lötkontakten ist im Kunststoffrahmen genug Platz um einen Klingeldraht, eine dünne Büroklammer o.ä. einzuführen, mit dem Sie die Karte nach vorne herausdrücken können.



Anhang: Warum gibt es diese SD Erweiterungskarte ?

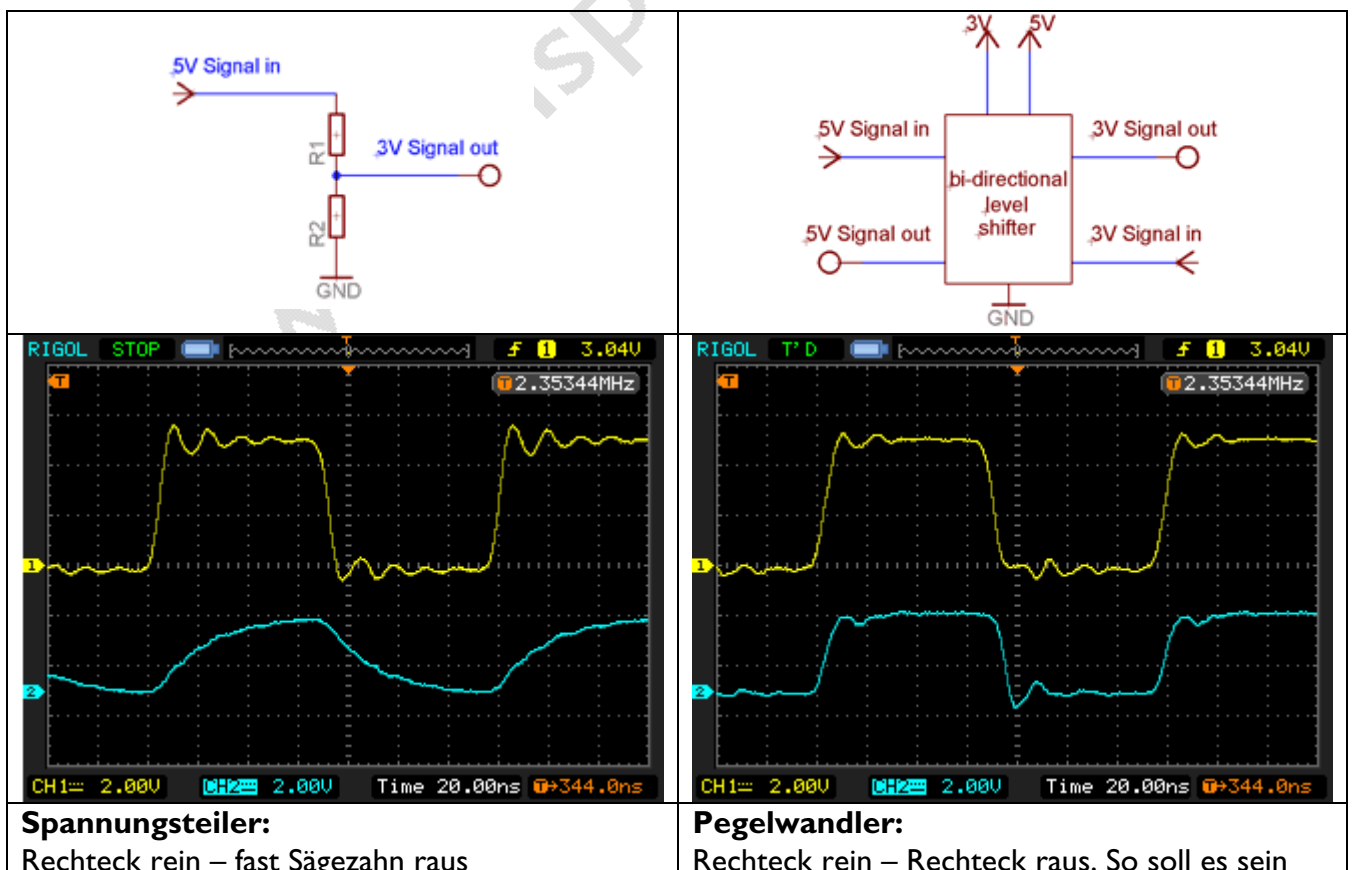
Eine SD-Karte benötigt eine Spannung von 2,7 bis 3,6 Volt – aber je nach Nutzung schwankt der Strombedarf der Karte von wenigen maA bis zu (kurzfristig) 100mA. Damit scheidet ein Spannungsteiler aus Widerständen schon aus, denn hier muss die Last schon gleich bleibend sein. Alternativ bieten sich hier evtl. mehrere Dioden an, die jeweils 0,7 Volt Spannungsabfall zeigen – aber eine professionelle Lösung ist dies nicht. Besser und langfristig sinnvoll ist also ein dezidiertes 3 Volt Spannungsregler wie auf unserem Modul.

Pegelwandlung

Die direkte Verbindung zwischen einer 5 Volt Mikrocontroller und einer SD Karte verbietet sich, da diese durch den 5 Volt Pegel der Signale zerstört würde. Die Signalpegel der Datenleitung müssen also von 5 Volt auf 3 Volt umgesetzt werden. Bei einem 3 Volt Mikrocontroller könnten Sie diesen natürlich auch direkt anschließen – Sie können allerdings trotzdem unsere Karte nutzen und von den Vorteilen der Tristate-Ausgänge profitieren.

Pegelwandlung (5V -> 3V) durch Spannungsteiler (2 Widerstände)

Hier könnte man nun meinen, dass zwischen jedem Ausgang des Mikrocontrollers und dem entsprechenden Signaleingang der Karte ein Spannungsteiler aus 2 Widerständen geschaltet werden könnte. Nun, bei langsamen Signalen ist dagegen auch nichts einzuwenden, aber sobald die Daten etwas schneller fließen sollen, wird es kritisch. Das nachfolgende Bild zeigt den Signalpegel hinter dem Spannungsteiler (so sieht dann also das 3 Volt Signal aus, das die Karte empfängt) – von einem Rechtecksignal ist hier keine Rede mehr.

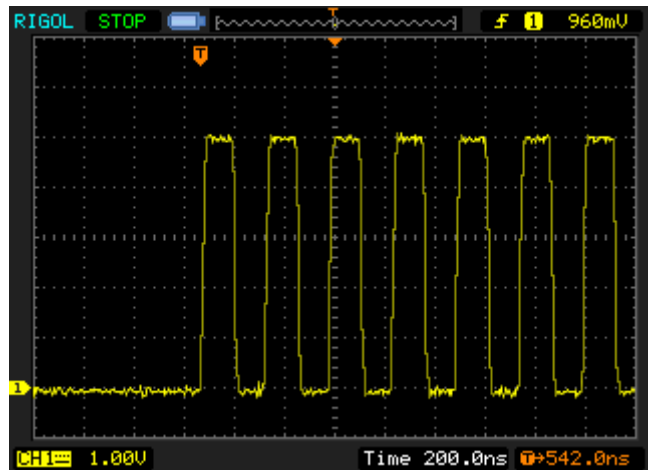


Pegelwandlung durch Transistoren

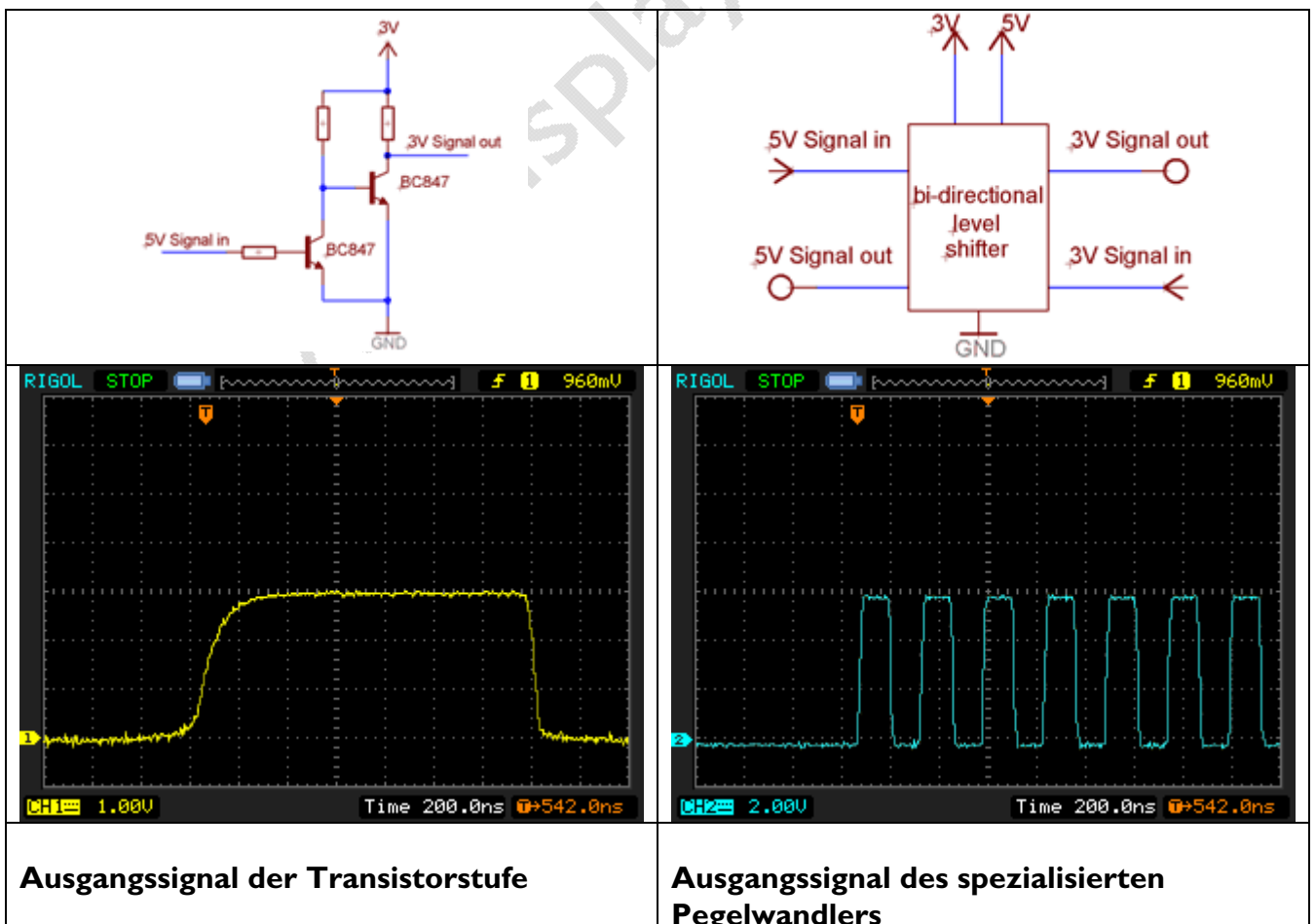
Jetzt wenden Sie vielleicht ein, dass man ja auch mittels zweier Transistoren den Pegel von 5 Volt auf 3 Volt senken kann. Das stimmt wohl, aber sie werden Schwierigkeiten haben, mit Transistoren einen Pegelwandler aufzubauen, der einen schnellen Datenverkehr zulässt. Die im Internet kursierende Schaltung mit BC547 / BC847 erlaubt dies sicher nicht.

Das nebenstehende Bild zeigt das Eingangssignal. Unten dann sehen Sie Ausgangssignal des Transistorpegelwandlers sowie des spezialisierten Pegelwandlers.

Deutlich erkennbar: Transistoren sind einfach nicht schnell genug für schnelle Pegeländerungen und damit für einen Pegelwandler unbrauchbar (statt der Clock-Signale kommt nur ein einzelnes langes Signal an, da die Transistoren nicht schnell genug schalten).



Eingangssignal 5V



Pegelwandlung in die umgekehrte Richtung (von 3 Volt auf 5 Volt)

Die SD Karte empfängt ja nicht nur Daten, sie soll sie auch an den Mikrocontroller senden. Da die Karte nur mit 3 Volt betrieben wird, beträgt natürlich auch das Ausgangssignal der Karte 3 Volt. Theoretisch könnte man nun diese Datenleitung direkt an den Controller anschließen, 3 Volt reichen in der Regel aus, um als High-Signal erkannt zu werden.

Aber Achtung – Böse Falle! Bei einer solchen Beschaltung müssen Sie sehr vorsichtig sein. Wenn auch nur einmal an dieser Leitung 5 Volt anliegen (sei es, dass noch ein anderer Busteilnehmer wie z.B. ein Programmieradapter auf der Datenleitung MISO etwas sendet, sei es, dass sie vom Controller versehentlich auf High geschaltet wird), dann liegen die 5 Volt direkt an der SD Karte an und zerstören sie vermutlich.

Daher kann eine langfristige Lösung nur ein Pegelwandler sein, der sowohl die 5 Volt-Signale des Controllers in ein 3 Volt-Signal für die SD Karte wandelt, aber auch das 3 Volt Datensignal der Karte in ein 5 Volt-Signale für den Controller überträgt. Dies erledigt ein bidirektionaler Pegelwandler.

Tristate-Ausgänge – Wichtig !

Die Welt besteht in der Regel nicht nur aus Mikrocontroller und SD-Karte. Meist gibt es noch andere Busteilnehmer wie z.B. ein Farbdisplay, ein Touchscreen-Controller oder andere Bausteine. Wenn ein solcher Baustein „gemein“ ist (und bei SD Karten ist dies u.U. der Fall), so behält eine Datenleitung des Teilnehmers nach der Übertragung einfach einen aktiven High-Pegel. Damit bricht der gesamte Busverkehr zusammen, denn diese Datenleitung kann nun nicht mehr von anderen Teilnehmern genutzt werden. Daher haben wir der Karte einen Pegelwandler mit Tristate-Kanälen spendiert: Durch die „Active-Leitung“ wird die Karte an den Bus komplett an- oder abgekoppelt und kann somit weder stören noch gestört werden.

Technische Daten

SD-Karten Adapterplatine P001

Artikel P001

Versorgungsspannung:	3,3 bis 5,0 Volt Gleichstrom
Strombedarf:	<0,5 mA der Platine ohne SD Karte Der Strombedarf der SD Karte selbst ist kartenabhängig und kann bis 100 mA betragen
Signale vom Mikroprozessor:	3,3 bis 5 Volt
Ca. Maße (ohne SD Karte) (HxB):	56 mm x 51 mm, Höhe: 9 mm
Ca. Maße (mit SD Karte) (HxB):	56 mm x 39 mm, Höhe: 9 mm

Hersteller:

Speed IT up
Inhaber Peter Küsters
Wekeln 39
47877 Willich
Telefon: (0 21 54) 88 27 5-0
Telefax: (0 21 54) 88 27 5-22

Weitere Informationen und Updates: www.display3000.com

Autor dieses Manuals: Peter Küsters.
© **aller Informationen: Peter Küsters**

Haftung, EMV-Konformität

Wenn Sie diese Baugruppe durch Anlöten von Kabeln, Erweiterung bzw. Gehäuseeinbau betriebsbereit gemacht haben, gelten Sie nach DIN VDE 0869 als Hersteller und sind verpflichtet, bei der Weitergabe des Gerätes alle Begleitpapiere mitzuliefern und auch Ihren Namen und Ihre Anschrift anzugeben.

Geräte, die aus Bausätzen selbst zusammengestellt werden, sind sicherheitstechnisch wie ein industrielles Produkt zu betrachten.

Derjenige, der den Bausatz zusammenbaut und in einem Gehäuse montiert, gilt als Hersteller und ist damit selbst für die Einhaltung der geltenden Sicherheits- und EMV-Vorschriften verantwortlich.

Für Schäden die durch fehlerhaften Aufbau entstanden sind, direkt oder indirekt, ist die Haftung generell ausgeschlossen.

Bei der Lieferung von Fremdprodukten als auch Software gelten über diese Bedingungen hinaus die besonderen Lizenz- oder sonstigen Bedingungen des Herstellers.