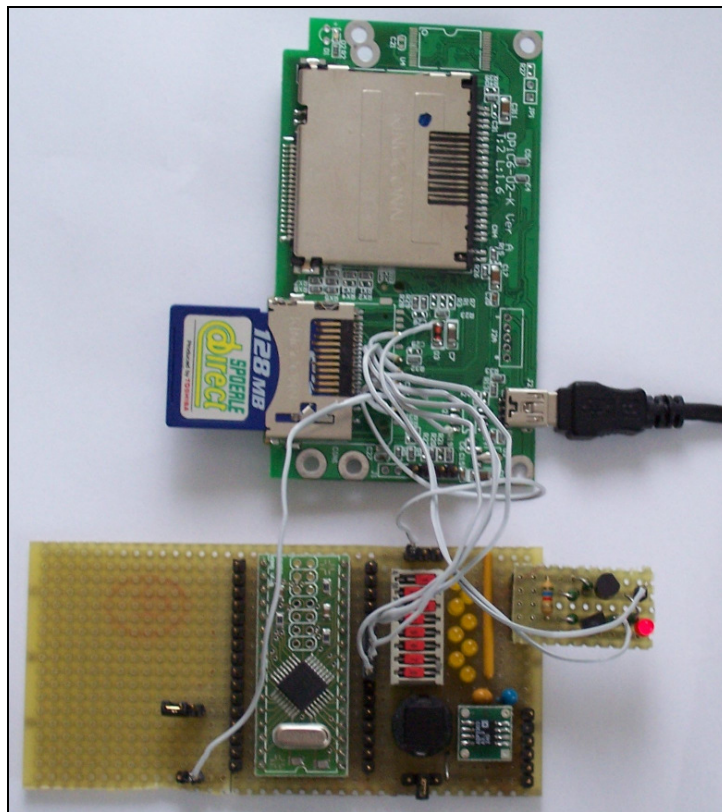


EDI

Embedded Datalogger Interface

Wie man einen preiswerten USB-Kartenleser in einen mikrocontrollergesteuerten universellen Datenlogger mit SD- / MMC-Karte umfunktioniert zeigt dieses Projekt. Die Möglichkeit der Anbindung von EDI an den PC per USB-Wechseldatenträger zur Auswertung der gespeicherten Daten ist dabei bereits automatisch vorhanden.



Beitrag für den Elektor R8C-Design-Wettbewerb

Autor: Michael Gaus

Version: V1.0

Datum: 2006-09-01

Hinweis:

Alle Informationen zu dem in diesem Projekt verwendeten USB-Kartenleser wurden experimentell durch Analyse der Schaltung ermittelt. Alle Informationen ohne Gewähr und ohne Anspruch auf Vollständigkeit und Richtigkeit. Um evtl. Datenverluste zu vermeiden, sollten wichtige Daten auf der SD/MMC-Karte immer gesichert werden, bevor die Karte für dieses Projekt eingesetzt wird.

Beschreibung:

Um große Datenmengen per Mikrocontroller aufzeichnen zu können, eignet sich eine SD- oder MMC-Karte als Speichermedium, da diese neben dem relativ komplizierten SD- bzw. MMC-Protokoll auch über einen einfach anzusteuern SPI-Modus verfügen. Zudem sind diese Speicherkarten sehr preisgünstig erhältlich und bieten mehrere MB Speicherplatz. Anstatt nun einen SD/MMC-Kartenslot an einen Mikrocontroller anzuschließen, wird hier ein etwas anderer Weg gegangen: es wird ein kompletter USB-Kartenleser verwendet, der u.a. auch über einen SD/MMC-Slot verfügt. Solche USB-Kartenleser sind sehr preiswert erhältlich, das hier verwendete Modell „Hama USB 2.0 Card Reader 35 in 1“ (Artikelnummer auf der Rückseite: 00055312, enthält den Chip IC1210-F128LQ von ICSI) kostet weniger als 10 Euro.

Es wird nun eine Umschaltmöglichkeit für den Zugriff auf die Speicherkarte benötigt:: entweder darf der Mikrocontroller auf die Karte zugreifen (μ C-Mode, LED2 leuchtet) oder aber der USB-Kartenleser (USB-Mode, LED3 leuchtet), der dann als USB-Wechselaufwerk am PC fungiert. Über einen kleinen Trick kann das ohne viel Hardware relativ einfach bewerkstelligt werden. Wenn der Kartenleserchip im Reset gehalten wird, dann sind die Pins des Kartenleserchips für die Ansteuerung der SD-/MMC-Karte im Tristate-Zustand, somit kann also von extern auf die Speicherkarte zugegriffen werden. Die Resetbeschaltung des Kartenlesers besteht aus 3 Bauteilen: ein R/C-Glied sowie eine Entladediode. Durch Überbrückung des Kondensators mit einem Transistor (T1) kann der Kartenleser im Reset gehalten werden. Der R8C13-Mikrocontroller kann mit einigen Portpins direkt mit den Pins des SD/MMC-Slots verbunden werden, da die Portpins auch tristate-fähig sind und dann somit hochohmig geschaltet werden können, wenn der USB-Mode aktiviert werden soll. Ein weiterer Transistor (T2) ist noch erforderlich, um die Spannungsversorgung der SD/MMC-Karte im μ C-Mode einschalten zu können (wird durch LED1 angezeigt). Beim USB-Mode ist hierzu im Kartenleserchip bereits ein FET integriert. Insgesamt müssen 8 Leitungen am Kartenleser angelötet werden. Diese sind gut zugänglich, sechs Leitungen können direkt an freien Löt pads am SD/MMC-Slot angelötet werden, eine Leitung wird an einer SMD-Diode angelötet und eine Leitung an einem Pin eines nicht bestückten SO8-SMD-ICs. Mit Jumper JP1 wird festgelegt, ob im USB-Mode (Jumper geschlossen) oder im μ C-Mode (Jumper offen) gestartet wird. Da keine Leitungen aufgetrennt werden müssen, kann der Kartenleser im USB-Mode (JP1 geschlossen) als Standard-Kartenleser weiterverwendet werden, um z.B. auf die anderen Kartenformate wie Compactflash usw. per PC zugreifen zu können.

Beim R8C13 wird die eingebaute SPI-Schnittstelle zur Ansteuerung der SD/MMC-Karte verwendet. Außerdem wird UART1 als RS232-Verbindung zur Außenwelt benutzt. Die erforderliche 3,3V Spannungsversorgung wird im Kartenleser durch einen Low-Drop-Spannungsregler aus der USB-Versorgungsspannung (ca. 5V) erzeugt. Im USB-Mode ist diese also automatisch vorhanden, im eigenständigen μ C-Mode ohne PC-Anbindung kann ein handelsübliches USB-Verlängerungskabel so umgebaut werden, dass eine externe stabilisierte Versorgungsspannung von 5V über die USB-Buchse zugeführt werden kann. Dabei ist dann die rote Ader des USB-Kabels die +5V-Leitung und die schwarze Ader GND, die weiße (D-) und die grüne (D+) Ader sind die USB-Datenleitungen und werden im μ C-Mode dann nicht benötigt.

Damit die im μ C-Mode aufgezeichneten Daten am PC beim Betrieb als USB-Wechselaufwerk auch ausgewertet werden können, müssen die Daten im FAT-Dateiformat auf der Karte gespeichert sein. Hier wird FAT16 verwendet, andere Formate wie z.B. FAT12 oder FAT32 werden von der R8C13-Mikrocontroller-Firmware nicht unterstützt. Da der Code für die Erstellung von Dateien in einem FAT16-Dateisystem sehr speicherintensiv ist (es müssen freie Cluster gesucht werden und zu einer Datei verkettet werden), wird hier ein kleiner Trick verwendet. Die Datei, die später die geloggtten Daten enthalten soll, wird als „leere Datei“ mit der gewünschten Maximalgröße am PC erstellt und im USB-Mode auf die Speicherkarte kopiert. Der Mikrocontroller muss dann anhand des Dateinamens den Startcluster der Datei im Hauptverzeichnis ermitteln und beim Schreiben der Daten am Ende eines Clusters die Position des nächsten Clusters aus der File Allocation Table auslesen und sich so durch die Datei durchhangeln, es ist jedoch nicht erforderlich, freie Cluster zu suchen und zu einer Datei zu verketten, da dies per PC ja bereits erledigt wurde.

Folgende Einschränkungen wurden der Einfachheit halber gemacht:

- Die Datei muss sich direkt im Hauptverzeichnis der Karte befinden, Unterverzeichnisse werden nicht ausgewertet.
- Der Dateiname muss im Format 8.3 gewählt werden (also max. 8 Zeichen plus 3 Zeichen Dateiergung), lange Dateinamen werden nicht unterstützt. Der Name der Logdatei wurde auf „LOG_DATA.TXT“ vordefiniert, auf Wunsch kann dieser Name im Quellcode in der Datei mmc_logger.c geändert werden (Variable „logfilename“)
- Innerhalb der Logdatei gibt es eine Endekennung, um erkennen zu können, wo das momentane Ende der Logdaten innerhalb der Logdatei ist. Diese Endekennung besteht momentan aus 2 Zeichen „@@“ in den ersten 2 Bytes eines Sektors, auf Wunsch kann dies im Quellcode in der Datei mcard_drv.c in den beiden Funktionen „setEndOfLogfile“ und „checkEndOfLogfile“ geändert werden.
- Aufgrund von zu geringer RAM-Größe des R8C13 erfolgt das Schreiben von Logdaten nur blockweise mit 512 Bytes. Wenn weniger als 512 Datenbytes pro Befehl gesendet werden, wird der Rest des Sektors nicht beschrieben und der nächste Befehl beschreibt dann den nächsten Sektor.

EDI kann über die RS232-Schnittstelle angesteuert werden. Die Baudrate ist auf 9600 Bd eingestellt, dies kann jedoch bei Bedarf im Quellcode in uart.c (Define BAUDRATE) geändert werden. Das Datenformat ist N,8,1, kein Handshake. Es werden verschiedene Befehle unterstützt, wobei nach Senden eines Befehls immer zuerst die Rückmeldung abgewartet muss, bevor weitere Befehle geschickt werden dürfen. Ein Befehl besteht immer aus mind. 2 Zeichen: einem <Esc>-Zeichen gefolgt vom Befehl.

Übersicht über die unterstützten Befehle:

- Firmwareversion abfragen:
Befehl: <Esc> v
Antwort: Versionskennung als 4stelliger ASCII-String gefolgt von <LF><CR>, z.B. „V1.0<LF><CR>“
- Begrüßungsmeldung und FAT-info ausgeben:
Befehl: <Esc> <Enter>
Antwort: ASCII-Text
- Letzten Fehlercode abfragen:
Befehl: <Esc> i
Antwort: 2-stelliger ASCII-Fehlercode
- Logdaten schreiben:
Befehl: <Esc> d <Binärdaten, max. 512 Bytes> <Esc> s
Hinweis: es können max. 512 Datenbytes pro Befehl geschrieben werden, soll ein <Esc>-Zeichen gespeichert werden, kann dies mit einem doppelten <Esc><Esc> erfolgen, das Ende der Datenbytes muss mit <Esc><s> markiert werden, es können auch weniger als 512 Bytes gesendet werden, allerdings wird dann der Rest des Sektors nicht beschrieben und der nächste Befehl beschreibt den nächsten Sektor.
Antwort: 2-stelliger ASCII-Fehlercode
- Umschalten in µC-Mode:
Befehl: <Esc> m
Hinweis: Dieser Befehl sollte nur dann verwendet werden, wenn zuvor im USB-Mode die Speicherkarte mit der Funktion „Hardware sicher entfernen“ beendet wurde, da dieser Befehl ohne Vorwarnung den Kartenleser in den Reset versetzt und somit eine evtl. laufende USB-Datenübertragung unterbrechen würde.
Antwort: 2-stelliger ASCII-Fehlercode
- Umschalten in USB-Mode:
Befehl: <Esc> u
Antwort: 2-stelliger ASCII-Fehlercode

Übersicht über die rückgemeldeten zweistelligen ASCII-Fehlercodes:

00	Kein Fehler
01	Logdatei voll
02	Logdatei nicht gefunden
03	Keine Endekennung in Logdatei gefunden
04	USB-Mode aktiv, kein Zugriff auf Speicherkarte möglich, es muss zuerst der µC-Mode aktiviert werden
10	E_MCARD_NOT_RESPONDING
11	E_MCARD_NOT_IDLE
12	E_MCARD_CARD_STAYS_IDLE
13	E_MCARD_NO_CID_RECEIVED
14	E_MCARD_NO_CSD_RECEIVED
15	E_MCARD_READ_RESPONSE_ERROR
16	E_MCARD_WRITE_RESPONSE_ERROR
17	E_MCARD_UNKNOWN_DATA_RESPONSE
18	E_MCARD_DATA_WRITE_ERROR
19	E_MCARD_DATA_CRC_ERROR
1A	E_MCARD_BLOCKCOUNT_RESP_ERROR
1B	E_MCARD_SETBLOCKLEN_RESP_ERROR
1C	E_MCARD_LOCK_UNLOCK_RESP_ERROR
1D	E_MCARD_MULTI_READ_RESP_ERROR
1E	E_MCARD_MULTI_WRITE_RESP_ERROR

Die Fehler im Bereich 1x sind von der Speicherkarte gemeldete Fehler.

Beim ersten Test empfiehlt es sich, mit einem Terminalprogramm am PC (z.B. Hyperterminal) zu arbeiten. Zuerst sollte im USB-mode gestartet werden (JP1 geschlossen), es sollten 4 Wechselaufwerke am PC dargestellt werden, eines davon ist für SD/MMC. Nun kann eine leere Datei (z.B. komplett mit Leerzeichen gefüllt) erstellt werden, in die ersten Bytes dieser Datei werden dann z.B. mit einem Texteditor 2 Zeichen „@@“ als Endekennung geschrieben und diese Datei wird dann auf die Speicherkarte kopiert. Nun wird das Wechselaufwerk beendet, Spannungsversorgung ausgeschaltet und JP1 geöffnet (für µC-Mode). Nach dem Einschalten der Spannungsversorgung mit aktiviertem µC-Mode (JP1 offen) dauert es eine kurze Zeit, bis die Speicherkarte initialisiert und analysiert wurde, danach wird ein zweistelliger ASCII-Fehlercode gemäß obiger Tabelle von EDI gesendet. Bei einer eingesteckten Karte mit vorhandener Logdatei „LOG_DATA.TXT“ und vorhandener Endekennung in der Datei sollte 00 geliefert werden. Nun kann mit dem Befehl <Esc><Enter> die Begrüßungsmeldung sowie die Informationen der FAT dargestellt werden.

Anwendungsmöglichkeiten:

Es gibt eine Fülle von Anwendungsmöglichkeiten für EDI. Beispielsweise können Langzeitaufzeichnungen von gemessenen Wetterdaten wie Temperatur, Luftdruck, Luftfeuchte usw. erfasst werden. Wenn die Daten im ASCII-Format als CSV-Datei abgelegt werden, ist sogar eine grafische Darstellung mit Excel möglich.

Wichtige Hinweise:

- Vor der Verwendung der Speicherkarte wichtige Daten zuerst sichern!
- Vor dem Einstecken bzw. Ausstecken der Speicherkarte die Versorgungsspannung abschalten.
- Bei Betrieb als Wechsellaufwerk am PC vor dem Abziehen des USB-Steckers immer zuerst den Datenträger mit der Funktion „Hardware sicher entfernen“ beenden.
- Der μ C-Mode funktioniert mit Standard SD- / MMC-Karten mit 3,3V Versorgungsspannung, spezielle Derivate wie MMCplus usw. wurden nicht getestet. Die übrigen Slots des Kartenlesers (Compactflash usw.) funktionieren im μ C-Mode nicht und es sollten dann auch keine Karten eingesteckt werden.
- Aufgrund der Vielzahl der Karten kann nicht gewährleistet werden, dass alle Modelle auch funktionieren. Folgende Typen wurden erfolgreich getestet: Toshiba 256MB SD, Extrememory 1GB SD, Sandisk 128MB SD, Extrememory 256MB MMC, Infineon 32MB MMC.
- Manchmal kann es sein, dass eine Speicherkarte im μ C-Mode nicht erkannt wird, dann hilft es meist, bei abgeschalteter Spannung die Karte einmal zu ziehen, nach kurzer Pause wieder einzustecken und dann die Spannungsversorgung wieder einzuschalten. Die genaue Ursache ist noch unbekannt.