Ein 1−Bit−Datenoszilloskop

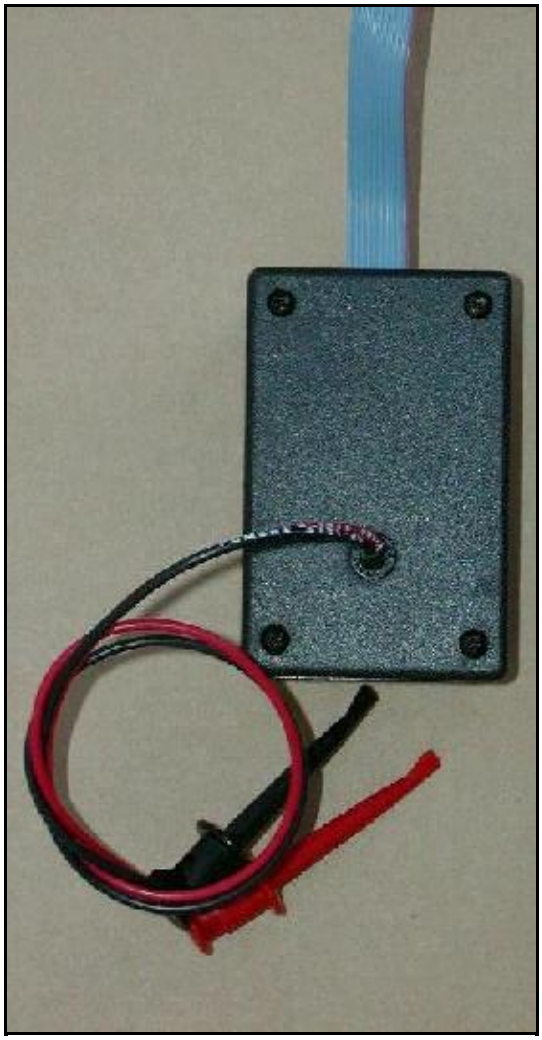
by Bob Smith
<bob(Q)linuxtoys.org>

About the author:

Bob ist ein Elektronik−Bastler und Linuxprogrammierer. Sie können sein neuestes Projekt auf www.linuxappliancedesign.com finden und seine Homepage auf www.linuxtoys.org.

Abstract:

Das 1−Bit−Datenoszilloskop (Logic−analyser) erfaßt ein Bit an Daten mit einer Rate von 46080 (oder 92160) Impulsen pro Sekunde und kodiert die Daten in einen Strom von RS−232−Zeichen, die mit 57.6 (oder 115.2) Kilobaud gelesen werden können. Das Datenoszilloskop ist ein ziemlich einfacher Projektaufbau, der weniger als sechs billige ICs braucht.



Einführung

In diesem Artikel wird der Entwurf, der Aufbau und die Benutzung des 1−Bit−Datenoszilloskops beschrieben. Ein Programm für die Erfassung und die Ausgabe des Datenstroms wird mitgeliefert.

Entwurf und Theorie der Ausführung

Entwurfsziele

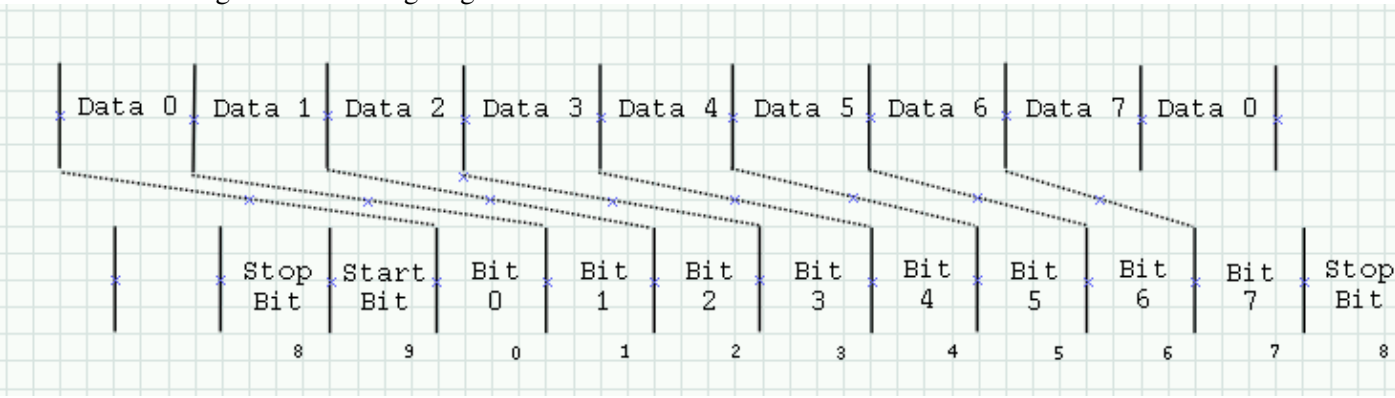
− Mindestbitmusterrate, um mehr als 8000 Muster pro Sekunde zu verarbeiten
− Ausgabe der Daten auf eine serielle PC−Schnittstelle
− Durchgehend Bitmusterdaten verarbeiten, ohne ein einziges Bit zu verlieren

– Strom von der seriellen Schnittstelle Strom abzapfen. Keine seperate Stromquelle.

Entwurfsüberblick

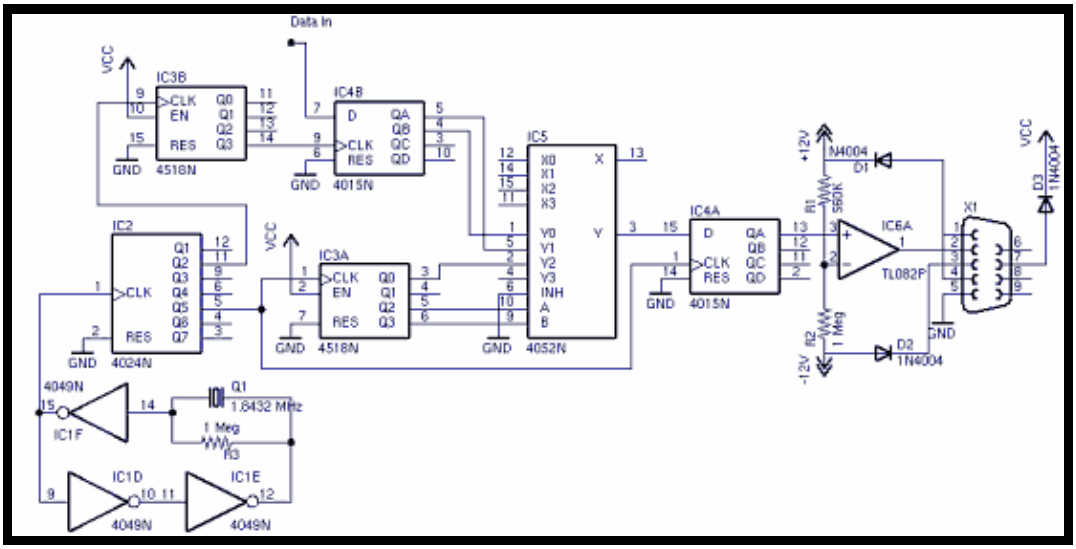
Das Ziel ist, mit einer geregelten Frequenz Eingaben zu erzeugen und die Daten in einem seriellen Zeichen auszugeben mit einem Startbit, acht Datenbits und einem Stopbit. Die Verwaltungskosten der Start– und Stopbits erfordern, daß die Bitdatenrate acht Zehntel der seriellen Baudrate entsprechen muß. Bei einer 57.6 Kbaud schnellen seriellen Schnittstelle ist die Datenrate (8 * 5.76 k) bzw. 46080 Bitmuster pro Sekunde.

Das Problem ist, daß wir die Eingabemuster zwischenspreichern müssen während wir die Start– und Stopbits versenden. Die folgende Zeichnung zeigt das Problem.



Während den Bits 0, 1, 2 und 3 wollen wir, daß die Daten um zwei Datentakte verzögert werden, damit wir Q(B) vom Schieberegister des 4015 holen können. Während den Bits 4, 5, 6 und 7 wollen wir die Daten verzögert um einen Datentakt, damit wir Q(A) vom Schieberegister holen können. Während den Stop– und Startbits (Bit 8 und 9) wollen wir eine Null und eine Eins, damit wir Q(0) vom Bitzähler holen können.

Schaltplan des 1–Bit–Datenoszilloskops



Der Schaltplan wurde mit einem sehr netten Softwarepaket namens "EAGLE" gemacht, erhältlich bei Cadsoft. Die Schemadatei für obigen Schaltkreis ist erhältlich als 1bitla.sch.gz (hier klicken).

Erzeugung des Taktes

Der Taktschaltkreis hat zwei Ausgänge: eine beim Baudtakt und eine bei acht Zehnteln des Baudtakts. Der Taktschaltkreis benutzt drei Inverter vom 4049, den 7−stufige Binärzahler im 4024 und ein Zehnerzähler einer Hälfte des 4518. Der Oszillator läuft mit 1.8432 MHz und Q(2) des 4024 hat eine Rechteckwelle bei 460.8 kHz. Der Q(3)−Ausgang des Zehntelzählers, 4518, hat 46.08 kHz. Der Q(5)−Ausgang des siebenstufigen asynchronen Zählers hat eine Rechteckwelle bei 57.6 kHz.

Um den Schaltkreis bei 115.2 kHz laufen zu lassen, benutzt man die Ausgänge des 4024 anstatt der Ausgänge von Q(2) und Q(5). Andere Baudraten sind möglich, wenn man andere Werte für den Kristall nimmt oder indem man den Kristalloszillator durch einen RC−Oszillator ersetzt.

Es wird sich wahrscheinlich herausstellen, daß der Kristalloszillator bei der gleichen Frequenz weniger Strom verbraucht als ein RC−Schaltkreis. Um Strom zu sparen, sollte man sichergehen, daß die unbenutzten Eingänge des 4049−Hex−Inverters auf Masse liegen.

Puffern der Eingabe

Es gibt zwei einfache Arten, wie man Eingabepufferung zu obiger Schaltung hinzufügt. Man kann den Dateneingang mit dem Eingang einer der unbenutzten Inverter im 4049 verbinden. Das bringt nicht wirklich viel, da der Eingang eine CMOS−Ladung darstellt und dazu noch CMOS−Schwellwerte hat (was in dieser Schaltung ungefähr drei Volt sind).

Ein besserer Eingabepuffer könnte der zweite Operationsverstärker im TL082 sein. Man kann einen Widerstandsteiler benutzen, den man an einen Operationsverstärkereingang klemmt und den anderen Eingang für die Eingabedaten benutzen. Dadurch kann man den Schwellwert sehr genau kontrollieren und es ergibt eine hohe Impedanz gegenüber dem Meßobjekt.

Konverter von CMOS nach RS−232

Eine Hälfte des TL082 Operationsvertärkers stellt eine Konvertierung von CMOS nach RS−232 zur Verfügung. Die Widerstandsteiler R1, R2 richten die negative Eingabe auf ungefähr 3 Volt. Sobald der Ausgang des 4015−Registers über und unter 3 Volt ist, schaltet der Ausgang des Operationsverstärkers auf positive und negative Stromversorgung. Wichig ist, daß die Bitmusterdaten auf der RS−232−Leitung invertiert werden und die datenlesende Software muß diese Umkehrung berücksichtigen.

Stromversorgung

Strom für die Schaltung wird von den RS−232−Leitungen des Computers geliefert. Tx vom Computer liefert die negative Spannung für den RS−232−Konverter (der TL082P) und DTR liefert die positive Spannung. Im

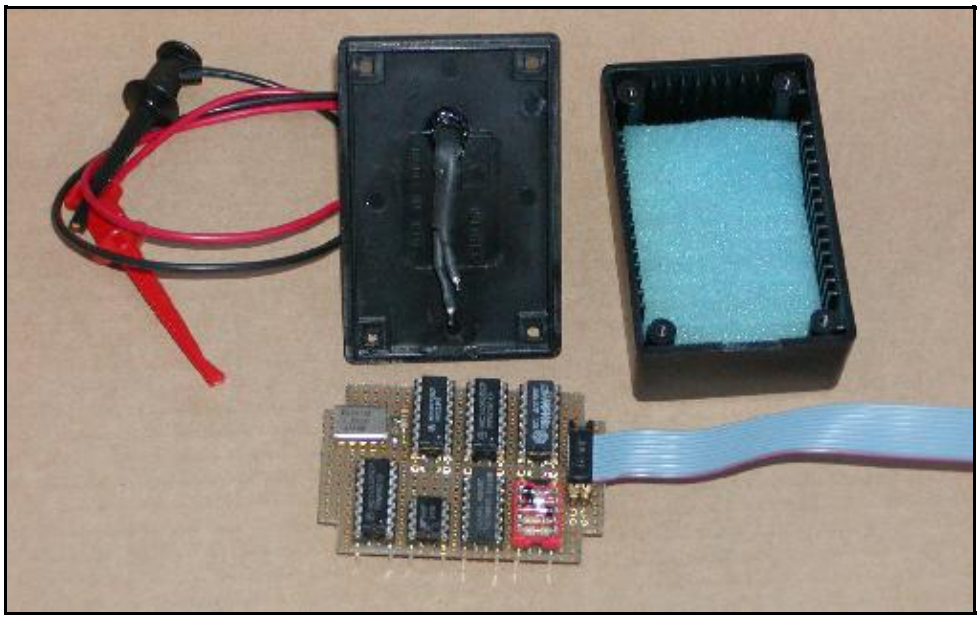
Prototypen waren die Dioden D1 unf D2 nicht wirklich nötig. Sie wollen die Schaltung vielleicht ohne sie ausprobieren.

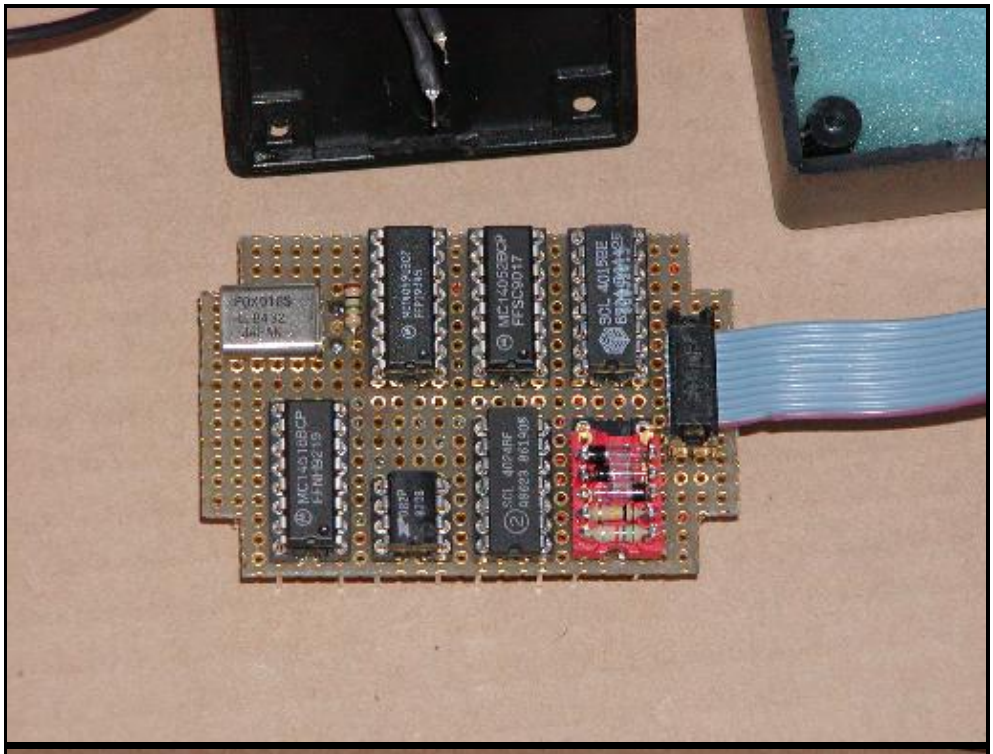
RTS stellt Strom (Vcc) für den Rest der Schaltung bereit. Im Prototypen war Vcc ungefähr 7 Volt stark. Ihre Spannung Vcc kann anders sein, das hängt von der RS−232−Schnittstelle ab. Die Software, die das Datenoszilloskop nutzt, muß sowohl DTR und RTS hochsetzen.

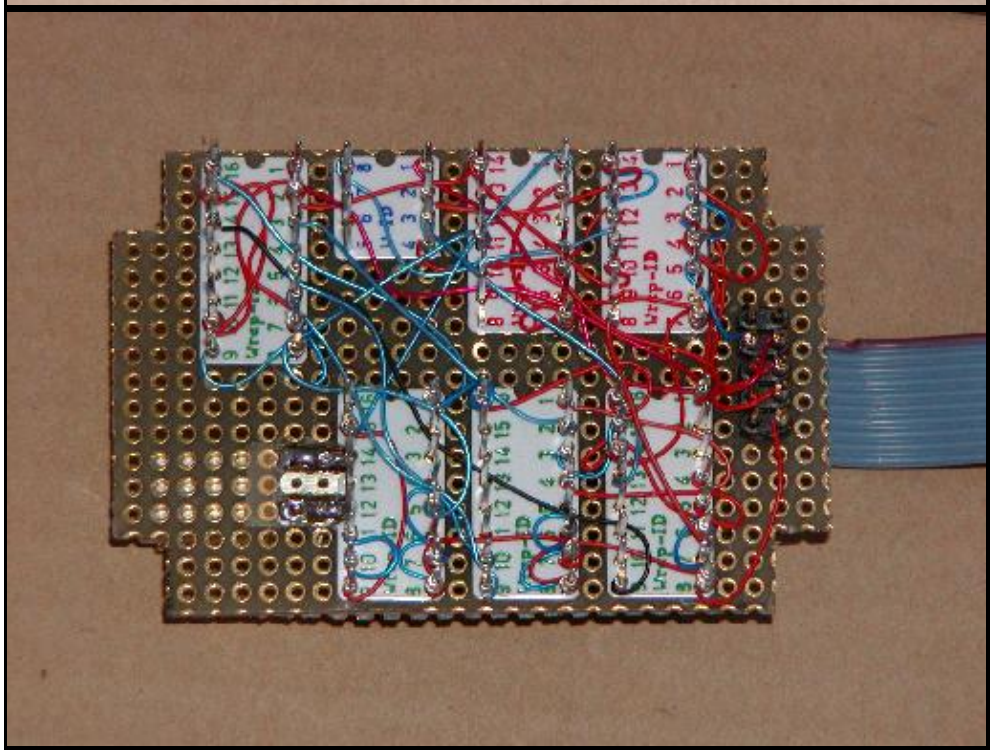
Aufbau

Die Frequenzen im Datenoszilloskop sind relativ niedrig und fast jede Bauform wird funktionieren. Wirewrap−Technik hat recht gute Ergebnisse gebracht. Vielleicht wollen Sie zuerst die Stromversorgung aufbauen und testen. Die ICs haben wie zu erwarten Vcc auf Pin 14/16, außer der 4049−Hex−Invertierer, der Vcc auf Pin 1 geschaltet hat.

Hier sind ein paar Fotos, um Ihnen eine mögliche Bauweise zu zeigen.







Einfaches Datenerfassungsprogramm

Bitmusterausgabe		
−r	−c	−t
03	0, 0	0, 0.000000
e3	1, 5	1, 0.000065
ff	0, 3	0, 0.000065
03	1, 10	1, 0.000217
e3	0, 6	0, 0.000130
ff	1, 5	1, 0.000109
03	0, 3	0, 0.000065
e3	1, 10	1, 0.000217
ff	0, 6	0, 0.000130
03	1, 5	1, 0.000109
e3	0, 3	0, 0.000065
ff	1, 10	1, 0.000217
03	0, 6	0, 0.000130
e3	1, 5	1, 0.000109
ff	0, 3	0, 0.000065
03	1, 10	1, 0.000217
e3	0, 6	0, 0.000130
ff	1, 5	1, 0.000109
03	0, 3	0, 0.000065
e3	1, 10	1, 0.000217
ff	0, 6	0, 0.000130
03	1, 5	1, 0.000109
e3	0, 3	0, 0.000065
ff	1, 10	1, 0.000217
03	0, 6	0, 0.000130
e3	1, 5	1, 0.000109
ff	0, 3	0, 0.000065
03	1, 10	1, 0.000217
e3		
ff		
03		
e3		
ff		
03		
e3		

Das einfache Programm 1bitla.c verarbeitet die seriellen Daten und kann die Daten in drei Formaten ausgeben. Das erste Format sind die Rohdaten, die ein Byte pro Zeile durch zwei Hex−Zeichen darstellen. Beachten Sie, daß das Programm die Daten invertiert und daß die Daten mit MSB zuerst verschickt werden. Rohdatenausgabe wird mit der Option −r aufgerufen. Das zweite Format ist ein Wert (0 oder 1) gefolgt von der Anzahl der 1−Bit−Muster, die der Wert hatte. Die Wert−/ Anzahlausgabe wird mit der Option −c aufgerufen. Die Auflösung bei 46080 kHz ist 21.7 Mikrosekunden. Die Wert−/Zeitausgabe wird mit der Option −t aufgerufen.

Das Programm wird mit

```
1bitla [option] serial_port
```

aufgerufen, wobei die Optionen –r, –c, oder –t sein können. Der bei Weglassen der Option wird Wert/Anzahl (–c) genommen.

Der Aufbau des Programms ist recht einfach. Wir verarbeiten die Kommandozeilenoptionen, öffnen die serielle Schnittstelle und machen eine unendliche Schleife und zeigen die Bytes an. Da wir keine Verarbeitung im Hintergrund laufen haben, benutzen wir ein blockierendes read.

Vielleicht sind Sie zufrieden damit, nur die Bitmuster auf Konsole auszugeben oder sie in eine Datei zur späteren Bearbeitung umzuleiten oder Sie können auch eine echte "Logik–Analyse" mit den Daten durchführen, indem Sie einen Automaten basteln der die Bitmuster verarbeitet. Beispielautomaten können Dekoderbefehle haben, die ihnen von einer Infrarotfernsteuerung geschickt werden oder können die Position eines funkgesteuerten Takts mit modulierter Ausgabe dekodieren.

Referenz

- Diesen Artikel herunterladen
- http://www.linuxtoys.org/, dieses und andere Hardwareprojekte für Linux

2005–01–11, generated by lfparser_pdf version 2.51