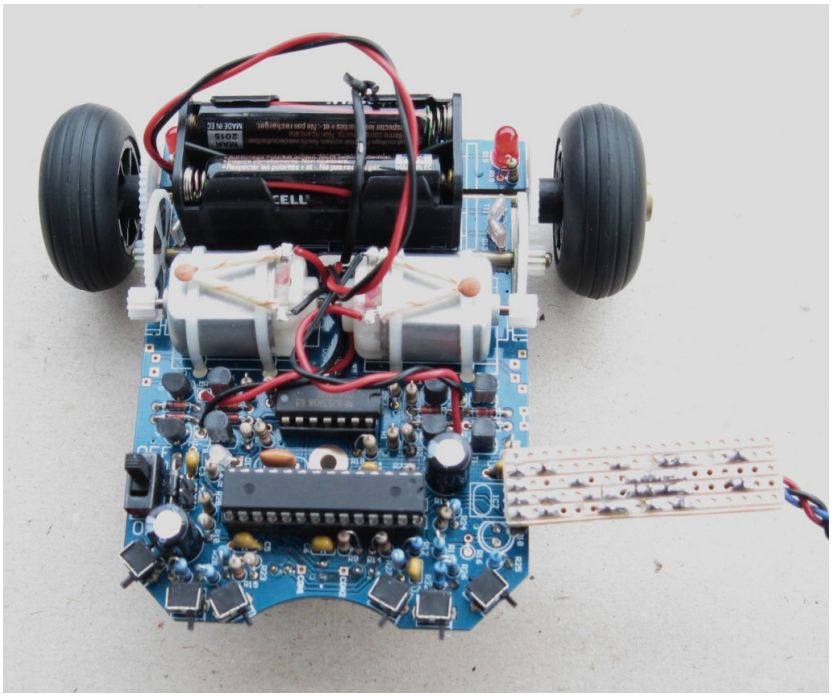
# Ein freier Asuro- Bootlader

Ralf Beesner

14. Mai 2012



## 1 Vorgeschichte

Der Asuro ist ein kleiner Roboter, der von Mitarbeitern des DLR für Ausbildungszwecke entwicklt wurde und von der niederländischen Firma Arexx vertrieben wird.

Ich hatte vor mehreren Jahren von den Kollegen einen Asuro- Bausatz geschenkt bekommen, ihn zusammengebaut und getestet, aber wurde mit der verwendeten Programmiersprache C nicht heimisch und hatte die bekannten, auch in vielen Foren diskutierten Probleme mit dem Flashen des Prozessors per Infrarot- Übertragung. Der Roboter lag dann einige Jahre in der Schublade.

Die Probleme sind aber geblieben. Vor einigen Tagen hervorgekramt, weigerte

er sich immer noch, sich flashen zu lassen. Genervt ersetzte ich die Infrarotstrecke durch eine Drahtverbindung, um Übertragungsfehler auf der Infrarotstrecke auszuschließen, aber es tat sich immer noch nichts.

Anschließend versuchte ich, mit einem ISP- Programmer den AtMega8 anzusprechen, aber auch das misslang. Sollte der AtMega tot sein? Nun wollte ich auf einen "frischen" AtMega den Asuro- Bootloader aufspielen. Die Suche auf der Website des Herstellers war ergebnislos; er möchte lieber programmierte AtMegas verkaufen.

Man findet zwar einen alternativen Bootloader, aber der erfordert ein spezielles Windows-only- Flashprogramm namens OCConsole.exe.

Auf der Suche nach einem besser geeigneten Ersatz stieß ich auf den kompakten Bootloader "Boofa" von Roland Riegel; da er unter der GPL steht und der Quelltext verfügbar ist, habe ich ihn ein wenig anpassen können.

## 2 Original- Bootvorgang

Der proprietäre Asuro- Bootloader benutzt (wie andere Bootloader auch) die serielle Schnittstelle des Ausuro. Eine Besonderheit ist die Infrarot- Übertragung. Während der IR- Empfängerchip die Demodulation der Signale intern vornimmt, so dass der demodulierte Bitstrom direkt auf den seriellen Eingang des AtMega gegeben werden kann, muss in Senderichtung ein 36 kHz- Signal erzeugt werden, auf das der Bitstrom des seriellen Ausgangs aufmoduliert wird.

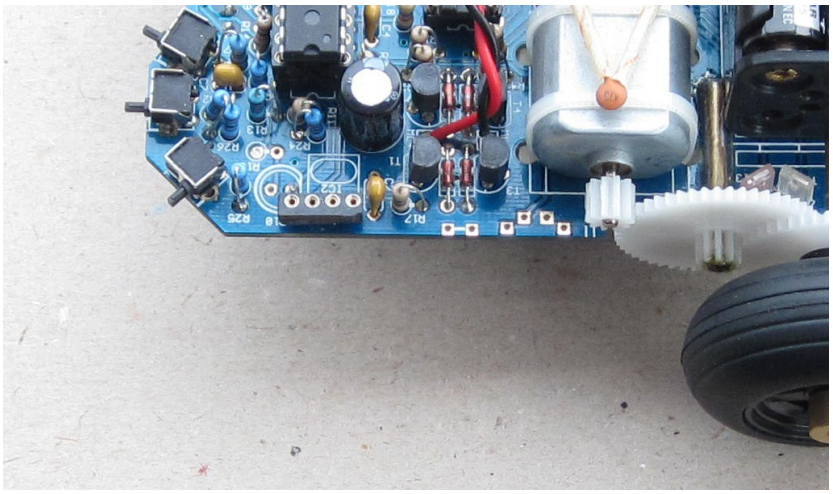
Beim Asuro ist das elegant und bauteilsparend gelöst: die IR- Sendediode hängt mit der Kathode an der TxD- Leitung des AtMega (Pin PD1), die Anode ist (über R1) mit Pin PB3/OC2 verbunden. Der Bootloader programmiert zunächst Timer 2 so, dass er ein 36kHz- Dauersignal an PB3 ausgibt. Ist TxD in Ruhelage (also low), liegt die IR- Diode mit der Kathode an Masse und strahlt ein 36 kHz-Signal ab; ist TxD high, wird eine Lücke in das IR- Signal getastet.

Man könnte also einen alternativen Bootloader recht einfach Asuro- kompatibel machen. Da die Infrarot- Hardware bereits abgewrackt und durch eine Steckverbindung ersetzt war, wollte ich das nicht weiter verfolgen. In früheren Experimenten mit der Infrarot- Übertragung serieller Signale hatte ich übrigens die Erfahrung gemacht, dass die direkte Modulation von ASCII- Signalen recht fehleranfällig ist und durch Leuchtstoffröhren und Röhrenmonitore leicht aus dem Tritt gebracht werden kann.

## 3 Hardware

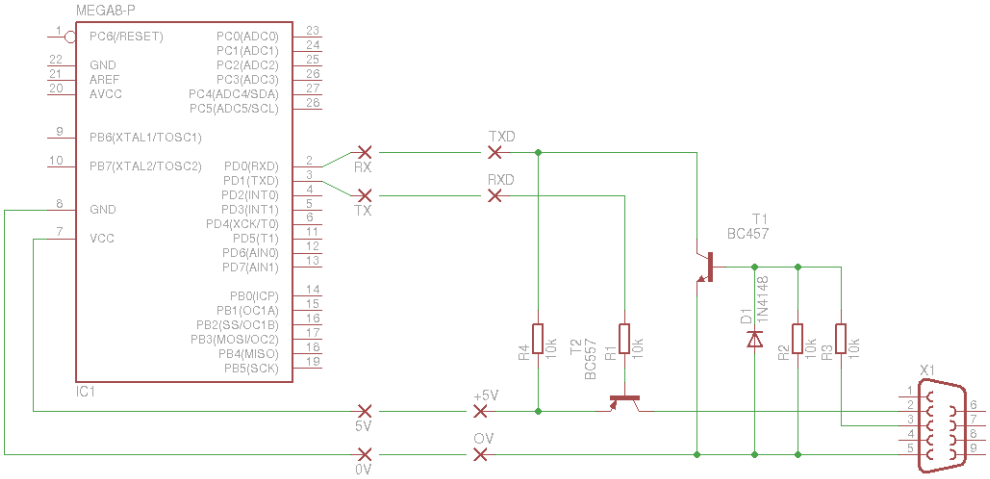
An Stelle der IR- Bauteile ist nun eine vierpolige SIL- Fassung in die Asuro-Platine eingelötet.

Nach Auslöten von R1, der IR- Diode und des IR- Empfängers musste ich zunächst ein kleines Loch so in die Asuro- Platine bohren, dass der vierte Pol der



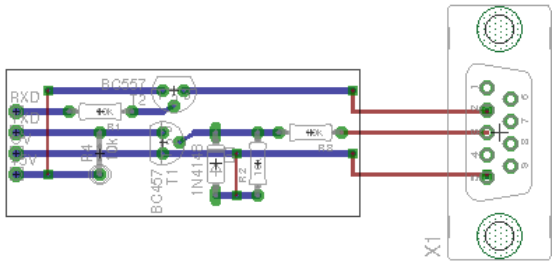
Fassung durch die Platine hindurchragt, ohne Kontakt mit der Massefläche zu bilden. Vor dort geht eine kleine Drahtbrücke zur an TxD liegenden Leiterbahn.

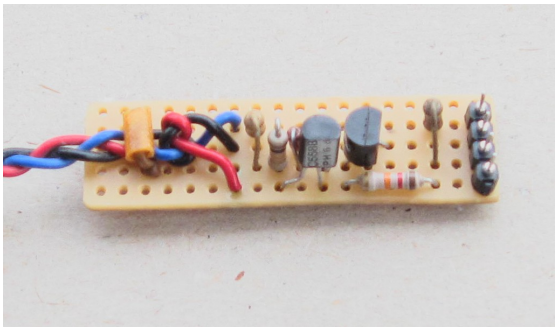
Die Pegelwandlung von TTL auf RS232- Pegel kann man mit einem (nur zur Hälfte genutzten) MAX232 vornehmen; zwei Transistoren, vier Widerstände und eine Diode tun es aber auch. Als Steckverbinder habe ich eine weitere vierpolige SIL- Fassung verwendet, in die ich zunächst kurze Drahtstückchen einlötete, die ihrerseits in die Streifenleitungsplatine eingelötet sind.




# 4 Software

Quelloffene und in Hochsprachen geschriebene Bootloader (z.B. der Arduino-Bootloader oder der BASCOM- Bootloader) belegen meist 2 kB Flash. Da der Asuro mit den 8 kB Flash des AtMega8 recht knapp ausgestattet ist, fiel die



Wahl auf den Boofa- Bootloader, der nur 1kB belegt.

Allerdings benötigt der Boofa- Bootloader einen freien Pin (im Originalcode PD6); liegt er bei Einschalten bzw. Reset des Mikrocontrollers an low, verzweigt der Boofa in die Flash- Routine, liegt er auf high, wird ein geladenes Programm gestartet.

Da der Asuro keinen völlig ungenutzten Pin aufweist, habe ich die Funktion auf PD2 (RxD) verlegt und den Code in der Datei boofa_config.asm um eine Schleife ergänzt: der Zustand des Pins wird nicht nur einmal abgefragt, sondern etwa eine Sekunde lang ein paar tausend mal.

Ist der Asuro über den Pegelwandler mit einem PC verbunden, liegt sein RxD-Eingang auf High. Startet man innerhalb der ersten Sekunde nach einem Mikrocontroller-Reset das Flashprogramm auf dem PC, zieht es mit dem ersten Startbit den RxD- Eingang auf low und schickt den Bootloader in die Flash- Routine.

Die serielle Bitrate wurde auf 19200 bit/s reduziert, um die Übertragungssicherheit zu verbessern. Damit man erkennt, dass der Asuro im Flash- Modus ist, werden über die Ports P0 und PD2 beide Segmente der Duo-LED eingeschaltet, so dass sie gelb leuchtet.

# 5 Handhabung

Der Boofa- Bootloader nutzt das avr109- Protokoll. Verwendet man das Kommandozeilentool avrdude, lautet der Aufruf:

```
avrdude -p m8 -c avr109 -b 19200 -U flash:w:<Programmname.hex>:i
```

Man kann den Befehl auch in eine Batchdatei schreiben; unter Linux habe ich mir einen Einzeiler namens "asu" angelegt:

```
avrdude -p m8 -c avr109 -b 19200 -U flash:w:$1.hex>:i
```

Unter Windows würde asu.bat lauten:

```
avrdude -p m8 -c avr109 -b 19200 -U flash:w:%1.hex>:i
```

Man kann dann mit dem Aufruf "asu <Programmname_ohne_hex>" das flashen starten.

Die erforderliche Reihenfolge:

- den Aufruf auf dem PC vorbereiten - den Ausuro einschalten - innerhalb einer Sekunde den Programmaufruf durch Drücken der Return- Taste absenden - die Duo- LED muss dann gelb leuchten - auch nach Übertragungsende bleibt der Asuro im Flash- Modus - nach Aus- und Wiedereinschalten und einer Sekunde Wartezeit startet dann das Programm

# 6 Vorbereiten des AtMega

Das Bootloader- Programm muss zunächst mit einem ISP- Programmer in den AtMega geflasht werden.

Anschließend wird der AtMega neu gefust. Das Fusebyte für die lfuse lautet CF (es aktiviert den Keramikresonator- Modus), das Fusebyte für die hfuse lautet C2 (es setzt die Bootsektorgröße auf 1 kByte bzw. 512 Worte und "verbiegt" den Startvektor so, dass er auf den Bootloader zeigt.

Mit avrdude und einem SP12- Programmer lautet der Aufruf:

```
avrdude -m8-c sp12 -U lfuse:w:0xcf:m hfuse:w:0xc2:m
```