Erik Bartmann

Spaß mit dem Amiga

Amiga 500

Die Arduino Uno DrawBridge

Autor	Erik Bartmann
Internet	https://erik-bartmann.de/
Thema	Spaß mit dem Amiga - Arduino-Amiga-Bridge
Version	1.02
Datum	17. März 2022

Inhaltsverzeichnis

Ein 3.5“-PC-Diskettenlaufwerk für den Amiga

	**Worum geht es überhaupt?**
In diesem Papier werden folgende Themen besprochen. • Das Lesen und Schreiben von Amiga-Disketten über ein Arduino-Mikrocontroller-Board mit einem 3,5“-PC-Diskettenlaufwerk	

Zu Beginn sei gesagt, dass die hier vorgestellte Entwicklung von *Robert Smith* ist und ich lediglich das System in einer Form präsentiere, dass es dem Einsteiger oder Interessierten hoffentlich leichtfällt, das Ganze recht einfach umzusetzen. Alle Detail-Informationen sind unter der folgenden Internetseite zu finden

	**Hyperlink - Robert Smith**
https://amiga.robsmithdev.co.uk/	

Ich muss sagen, dass ich von der Idee bis hin zur Umsetzung begeistert bin und es ganz neue Horizonte eröffnet, Software auf einen richtigen und realen Amiga mithilfe von 3,5“-Disketten zu bekommen.

Das Problem

Wer einen Amiga sein Eigen nennt und das ist eine ganz tolle Sache in meinen Augen, der möchte sicherlich manchmal seine Disketten in einer bestimmten Art und Weise sichern. Natürlich ist das am Amiga selbst über eine Kopierfunktion möglich, doch manchmal möchte man einfach ein im Internet frei verfügbares Disketten-Image in Form eines *ADF* (Amiga-Disk-File) auf eine 3,5“-Diskette speichern, um diese dann auf dem Amiga zu nutzen. Heutige PCs oder Laptops verfügen in der Regel über kein 3,5“-Diskettenlaufwerk mehr, was die Sache etwas erschwert, doch nicht ganz unmöglich macht. Derartige Laufwerke gibt es immer noch neu oder gebraucht zu kaufen. Nun gibt es aber ein Dilemma, dass es so ohne weiteres nicht möglich ist, eine auf einem PC-Diskettenlaufwerk erstellte Diskette in einem Diskettenlaufwerk eines Amigas zu nutzen, denn Sie sind einfach nicht lesbar. Dieser Weg scheidet also aus. Nun kann man natürlich eine Amiga-Emulationssoftware wie zum

Beispiel *WinUAE* unter Windows nutzen, so dass die Verwendung der ADF-Files machbar ist, doch das wahre Feeling an einem richtigen Amiga zu sitzen, kommt dabei nicht auf.

Die Lösung

Nun gibt es aber eine Möglichkeit, über ein PC-Diskettenlaufwerk, das über einen Mikrocontroller in Form eines Arduino mit der USB-Schnittstelle verbunden ist, die genannten ADF-Dateien unter dem Betriebssystem Windows zu lesen und zu speichern. Im Anschluss können sie dann in einem Amiga-Diskettenlaufwerk zur Anwendung kommen. Wenn man etwas geschickt im Umgang mit elektronischen Bauteilen ist und etwas Löten kann, ist alles sehr schnell zusammengebaut. Es gibt natürlich auch einen fertigen Adapter zu kaufen, doch ich möchte hier den „*harten Weg*“ beschreiten und alles selbst zusammen frickeln. Was ist dafür an Hardware erforderlich? Nun, das ist nicht viel.

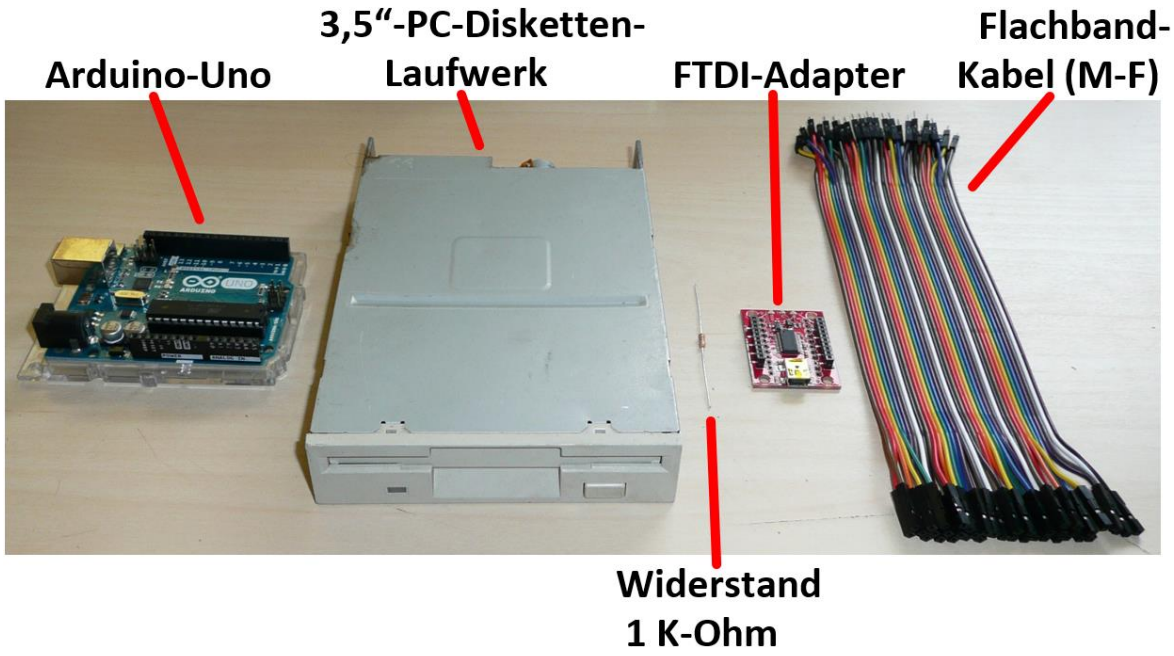


Abbildung 1 Die erforderlichen Komponenten

Was ich nicht aufgezählt habe, sind das Breadboard und die USB-Kabel zum Anschluss des Arduino an den PC beziehungsweise des FTDI-Adapters an den PC. Das hier verwendete Arduino-Board ist ein *Arduino-Uno*, wobei es auch mit dem *Arduino-Pro-Mini* oder dem *Arduino-Nano* zu realisieren ist. Näheres ist auf der genannten Internetseite zu finden. Für mich ist das Arduino-Uno-Board immer noch das *Non-Plus-Ultra*, doch es stellt lediglich meine persönliche Meinung dar und muss nicht zwangsläufig die der anderen Leser sein. Ich möchte zu Beginn etwas auf die einzelnen Komponenten eingehen, denn es kann für einen Einsteiger sicherlich sinnvoll sein, etwas darüber Bescheid zu wissen.

Das Mikrocontroller-Board

Das Herz des ganzen Projektes bildet ein Mikrocontroller-Board auf der Basis eines Arduino. Das Arduino-Uno-Board ist auf der folgenden Abbildung zu sehen.

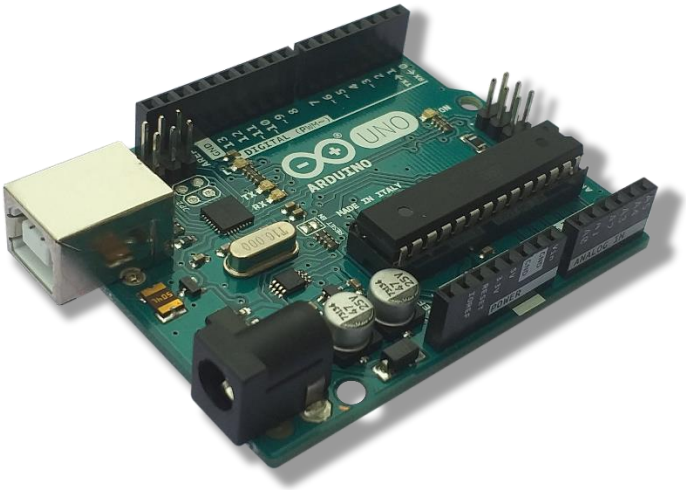


Abbildung 2 Das Arduino-Uno-Board

Das Board verfügt über zahlreiche digitale und analoge Ein- beziehungsweise Ausgänge, die mittels Header sehr leicht zugänglich sind. Auf der folgenden Abbildung sind die einzelnen Zugangs-Buchsen zu sehen.

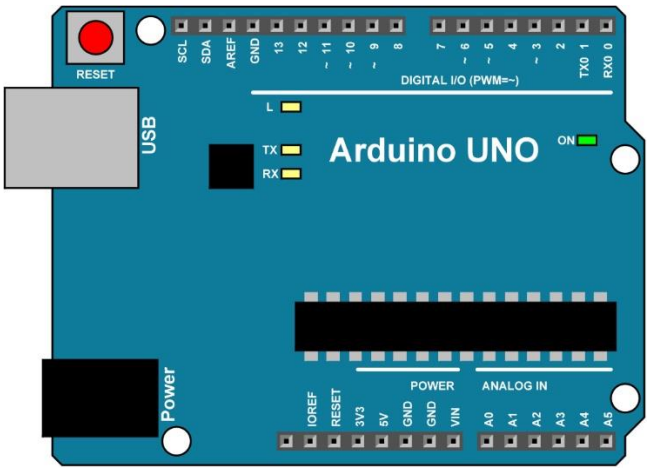


Abbildung 3 Die Zugangsbuchsen des Arduino-Uno

Detailinformationen sind unter der folgenden Internetadresse zu finden.

	Hyperlink - Arduino Uno
https://docs.arduino.cc/hardware/uno-rev3	

Wer sich näher mit der Programmierung des Arduino-Uno befassen möchte, der kann sich zum Beispiel mein aktuelles Arduino-Buch einmal näher anschauen, das unter der folgenden Internetadresse zu bekommen ist.

Hyperlink - Arduino-Buch

https://www.bombini-verlag.de/shop/arduino/

Die Arduino-Entwicklungsumgebung

Um den Arduino-Mikrocontroller programmieren zu können, ist die Arduino-Entwicklungsumgebung natürlich sehr gut geeignet, die unter der folgenden Internetadresse zu finden ist.

Hyperlink - Arduino-Entwicklungsumgebung

https://www.arduino.cc/en/software

Ein Installationsanleitung ist auf meiner Internetseite zu finden und kann heruntergeladen werden.

Hyperlink - Arduino-IDE - Anleitung

https://erik-bartmann.de/userfiles/downloads/Arduino/ArduinoIDEInstallationVersion20.pdf

Die Entwicklungsumgebung präsentiert sich nach dem Start wie folgt.

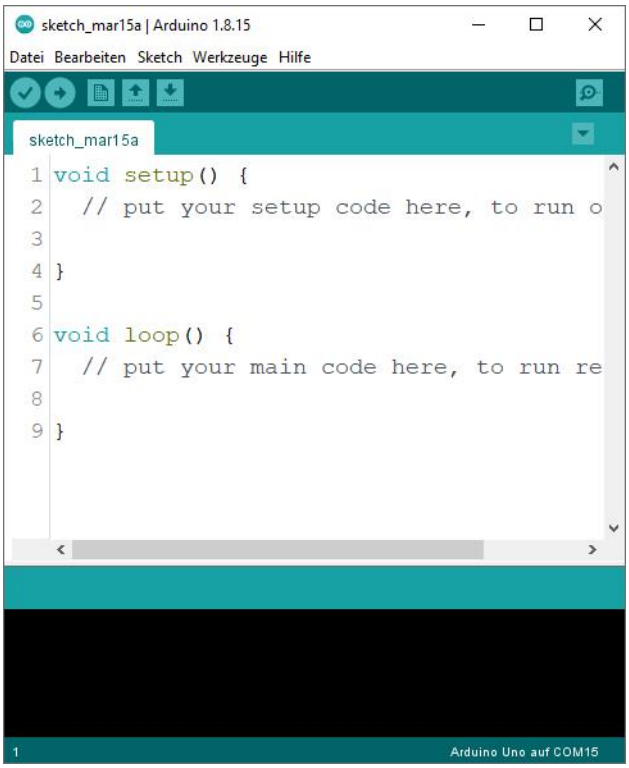


Abbildung 4 Die Arduino-Entwicklungsumgebung

Der FTDI-Adapter

Die Firma *Future Technology Devices International* hat den sogenannten *FTDI-Adapter* entwickelt, der ein USB-UART-Interface-Chip bereitstellt, mit dem es möglich ist, eine serielle Schnittstelle vom Typ RS-232 über einen Pegelwandler mit USB zu verbinden. Dieser wird dazu genutzt, um über ein spezielles Programm mit dem Namen *DrawBridge* Kontakt mit dem Arduino-Board aufzunehmen. Es ist u.a. dafür zuständig, die ADF-Dateien zu handhaben und danach über das Arduino-Board eine Verbindung mit dem Diskettenlaufwerk aufzunehmen.

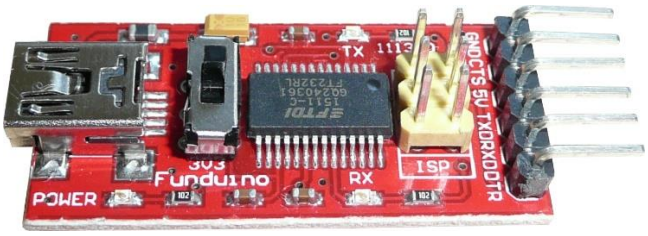


Abbildung 5 Der FTDI-Adapter

Ein derartiger Adapter besitzt in der Regel die folgenden Pins.

- DTR
- RX
- TX
- Vcc
- CTS
- GND

Diese Beschriftungen sind - je nach Adapter - auch auf der Platine aufgedruckt. Es ist wichtig, dass es ein FTDI-Adapter mit 5V ist. Die meisten können über einen Jumper zwischen 3,3V und 5V konfiguriert werden.

Das PC-Diskettenlaufwerk

Das PC-Diskettenlaufwerk ist natürlich dafür da, um eine 3,5“- DD-Diskette (DD = Double Density) zu lesen und zu beschreiben. Bei den Diskettenlaufwerken für den Amiga (AmigaDOS) handelt es sich eben um 3,5“ DD-Laufwerke mit einer Datenrate von 500KBit pro Sekunde. Diese Daten sind identisch mit 720KByte-Laufwerken, die für den PC verwendet werden. Tatsächlich formatiert der Amiga die Diskette jedoch mit 880KByte pro Diskette.

Abbildung 6 Das PC-Diskettenlaufwerk

Beim Anschluss des Diskettenlaufwerkes an einen Arduino ist natürlich die Pinbelegung (Daten und Power) wichtig, die auf der folgenden Abbildung zu sehen ist.

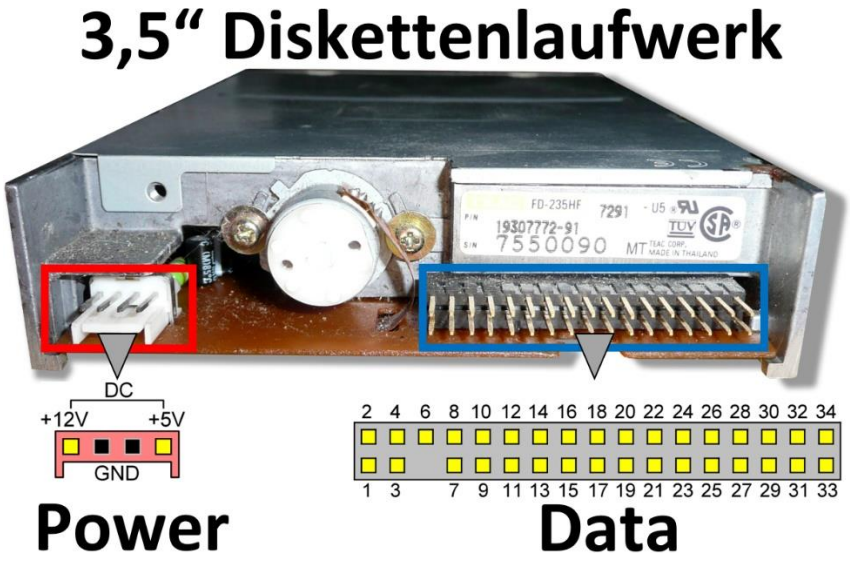


Abbildung 7 Die Pinbelegung des PC-Diskettenlaufwerkes

Natürlich ist es wichtig zu wissen, welche Pins welche Bedeutung haben, was dann auf der folgenden Abbildung zu sehen ist.

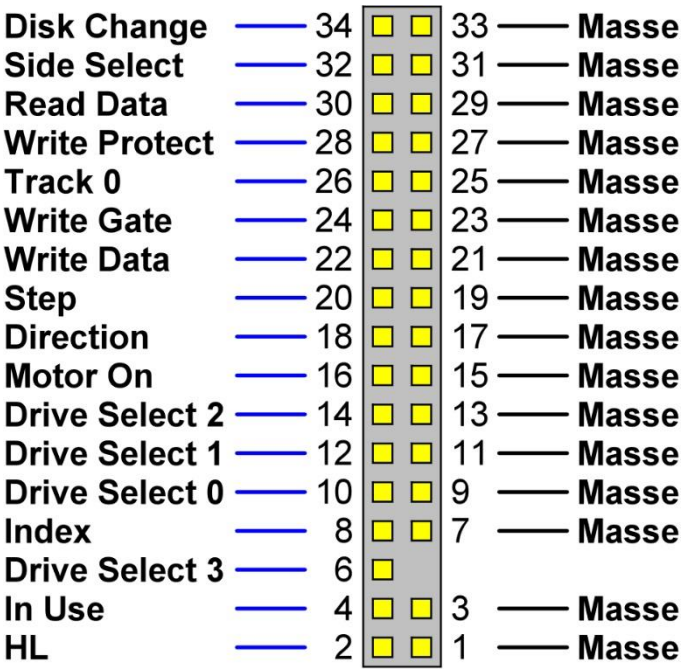


Abbildung 8 Die Pinbelegung der Disketten-Schnittstelle

Ein Pullup-Widerstand

Letztendlich ist noch ein Widerstand von 1-KOhm (1000 Ohm) erforderlich, der als Pullup-Widerstand in Erscheinung tritt und später natürlich im Schaltplan zu sehen ist. Er besitzt den folgenden Farbcode.

Abbildung 9 Widerstand 1 K-Ohm

Das Breadboard

Um alles miteinander zu verbinden, habe ich ein Breadboard verwendet, das auf einer Plastikplatte aufgebracht ist und u.a. auch des Arduino-Uno aufnehmen kann, wie das auf der folgenden Abbildung zu sehen ist.

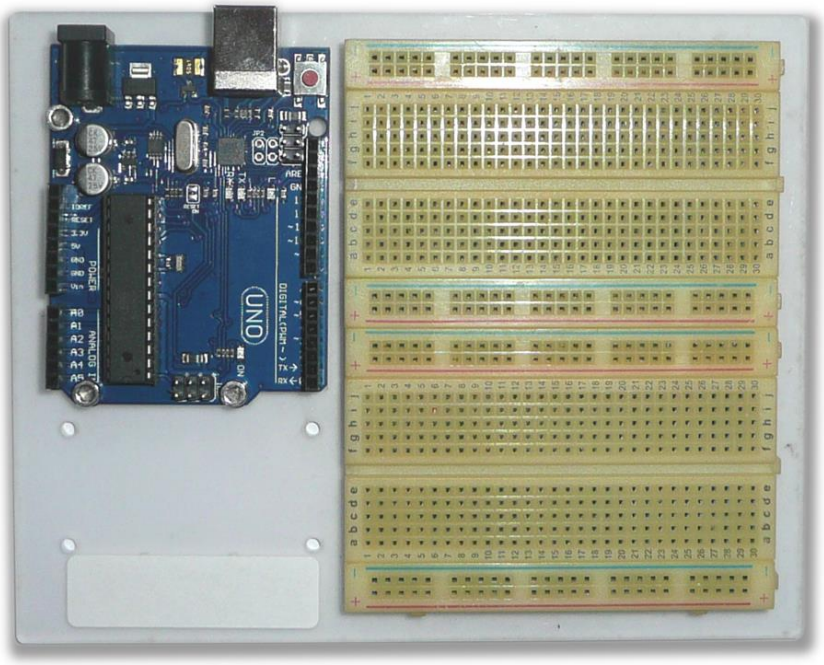


Abbildung 10 Arduino + Breadboard

Natürlich kann sich jeder selbst eine geeignete Möglichkeit suchen, um alle Verbindungen herzustellen und es gibt unzählige gute Varianten.

Patch-Kabel und Flachbandkabel

Um die elektrischen Verbindungen herzustellen, habe ich sogenannte *Patchkabel* verwendet, wie sie auf der folgenden Abbildung zu sehen sind, die es auch in unterschiedlichen Längen und Farben gibt. Zusätzlich habe ich auch noch zwei unterschiedliche Breadboards eingefügt, die hier und da sicherlich nützlich sind.

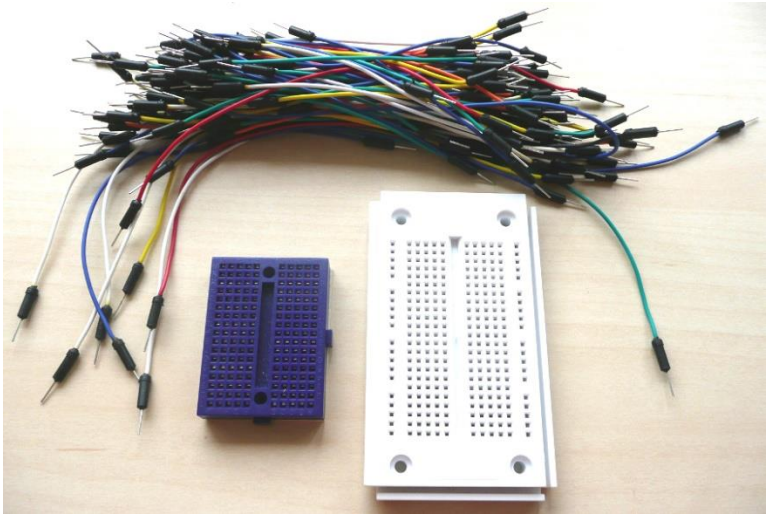


Abbildung 11 Patch-Kabel und Breadboards

Zusätzlich habe ich ein breites Flachbandkabel (Multicolored Dupont Wire 40pin Male to Female) verwendet, das auf einer Seite mit Steckern, auf der anderen Buchsen versehen ist.

Abbildung 12 Flachbandkabel

Diese sind gerade bei der Herstellung der Verbindungen zwischen Diskettenlaufwerk und Arduino-Board sehr hilfreich.

Der Schaltplan

Kommen wir nun zum Schaltplan, der alle erforderlichen Komponenten und deren elektrischen Verbindungen untereinander aufzeigt.

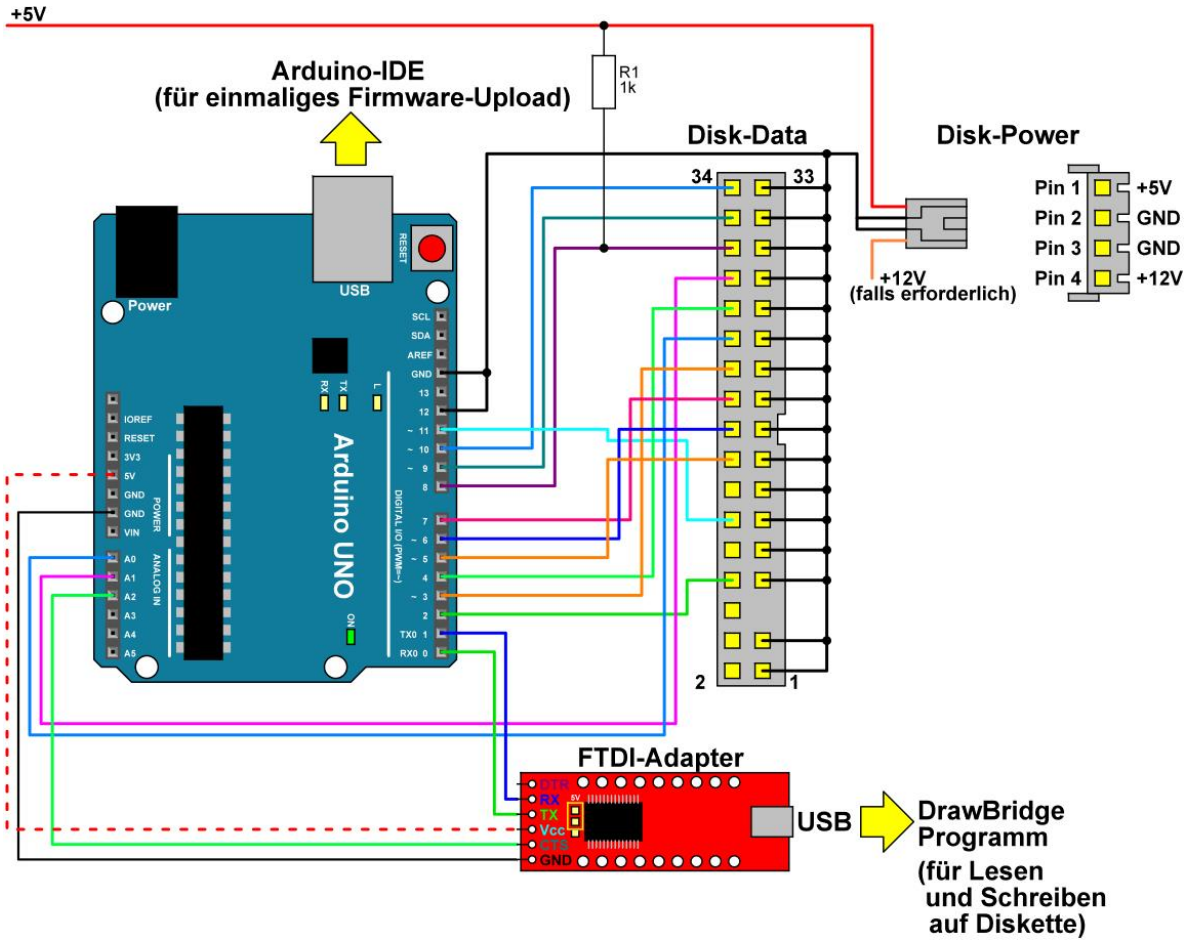


Abbildung 13 Der Schaltplan

Ich habe alles zu einem ersten Test natürlich erst einmal auf einem kleinen Breadboard zusammengesteckt, und das hat wirklich nicht lange gedauert. Es muss natürlich genauestens darauf geachtet werden, welche Pins der einzelnen Komponenten miteinander verbunden werden sollen. Da kann es schon mal Verwechselungen kommen und diese führen dann dazu, dass nichts funktioniert. Der Frust ist groß und die Fehlersuche kann dauern. Also in dieser Phase sehr genau beim Herstellen der elektrischen Verbindungen aufpassen, denn auch ein Kurzschluss kann sehr unangenehme Folgen haben. Ich nehme da keine Verantwortung für etwaige Schäden. Also am besten alles 3-Mal überprüfen und ggf. die zuerst genannte Internetseite des Entwicklers konsultieren!

Der Schaltungsaufbau

Der Aufbau der Schaltung schaut bei mir wie folgt aus und ist natürlich sehr individuell.

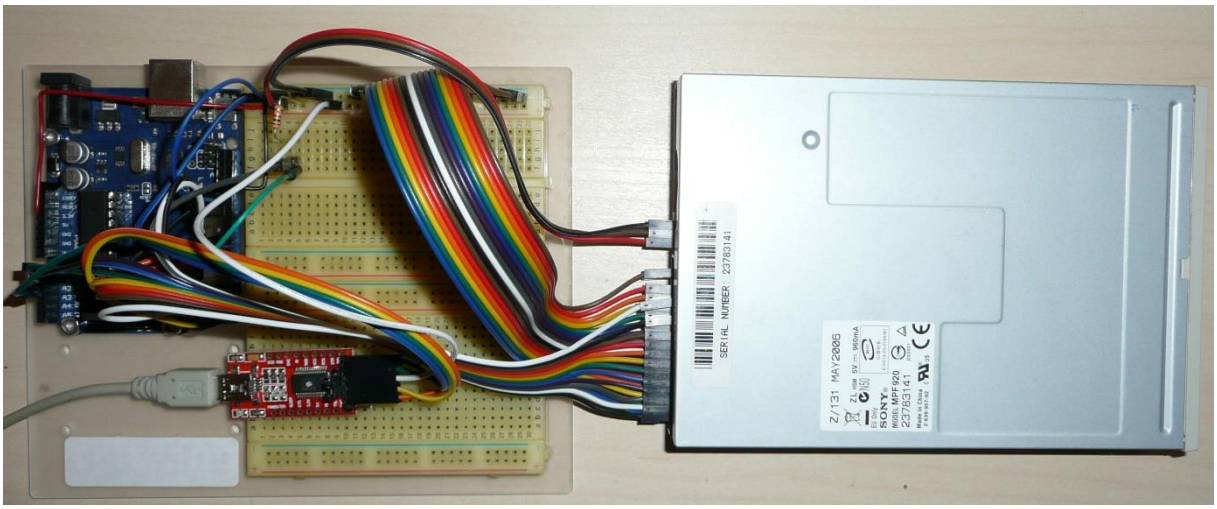


Abbildung 14 Der Schaltungsaufbau

Im Endeffekt zählt das Ergebnis und das es eben alles funktioniert. Für einen späteren Gebrauch ist es natürlich ratsam, das Ganze etwas robuster zu gestalten. Ich komme noch darauf zurück.

Die Vorgehensweise

Die folgenden Schritte habe ich zur Realisierung des Projektes durchgeführt.

Das Bereitstellen aller Hardware-Komponenten

Ich würde zuerst sehen, dass alle erforderlichen Hardware-Komponenten bereitliegen, denn erst dann kann es wirklich losgehen.

Das Herunterladen der DrawBridge-Software

Die spätere Kommunikation des Computers für das Laden oder Speichern der ADF-Dateien erfolgt über das *DrawBridge*-Programm, das unter der folgenden Internetadresse zu bekommen ist.

	Hyperlink - Das DrawBridge-Programm
https://amiga.robsmithdev.co.uk/download	

Nach dem Entpacken der ZIP-Datei muss später das Programm *DrawBridgeWin.exe* ausgeführt werden.

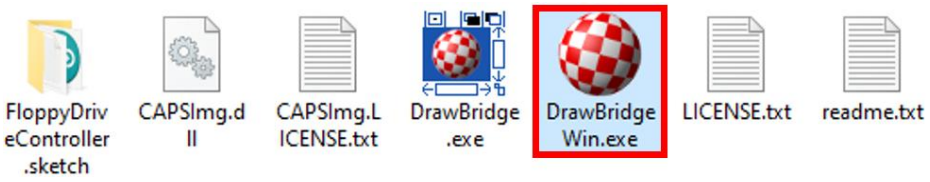


Abbildung 15 Das DrawBridge-Programm

Die Installation der Arduino-Entwicklungsumgebung

Die Installation der Arduino-Entwicklungsumgebung für das Hochladen der Firmware (Sketch) sollte nun erfolgen oder ist schon erfolgt, da ich diesen Schritt in diesem Papier ja schon etwas weiter oben zur Sprache gebracht hatte.

Der Upload der Firmware

Nun kann die Firmware auf den Arduino-Uno hochgeladen werden, wobei im Arduino-Umfeld die Firmware auch *Sketch* genannt wird. Diese Firmware ist unter der folgenden Internetadresse zu bekommen.

Hyperlink - Arduino-Firmware

https://github.com/RobSmithDev/ArduinoFloppyDiskReader/tree/master/

Am besten kopiert man sich den kompletten Github-Inhalt in einen Ordner auf dem Computer. Die Struktur schaut im Moment wie folgt aus.

.github	readme update	5 months ago
ArduinoFloppyReader	Small change to Linux makefile and added missing PLL files	6 days ago
FloppyDriveController.sketch	2.8.7 - Better Compatibility for UAE	15 days ago
LICENSE.txt	Initial Push from local VS	5 years ago
readme.md	V2.8.6 and firmware 1.9.23	last month

Nun wechselt man in den Ordner mit dem Namen *FloppyDriveController.sketch*, der den folgenden Inhalt aufweist.

FloppyDriveController.sketch.ino	2.8.7 - Better Compatibility for UAE
LICENSE.txt	Initial Push from local VS

Die erste Datei mit der Endung *.ino* beinhaltet die Firmware, die in die Arduino-Entwicklungsumgebung kopiert werden muss.

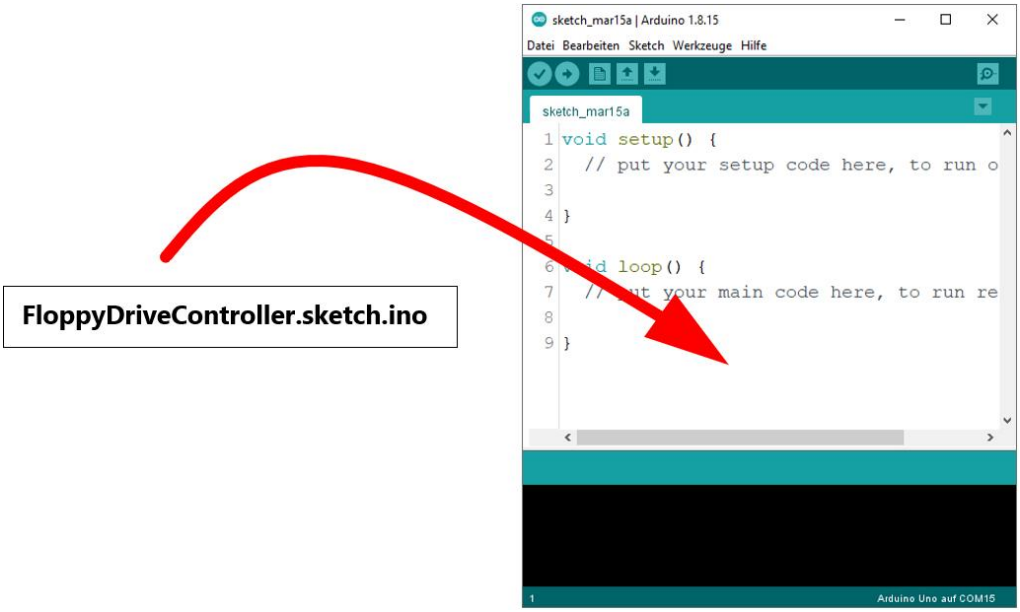


Abbildung 16 Die FloppyDriveController-Firmware

Bevor es nun zum Upload kommen kann, muss das Arduino-Board über die USB-Schnittstelle mit dem Computer verbunden werden.

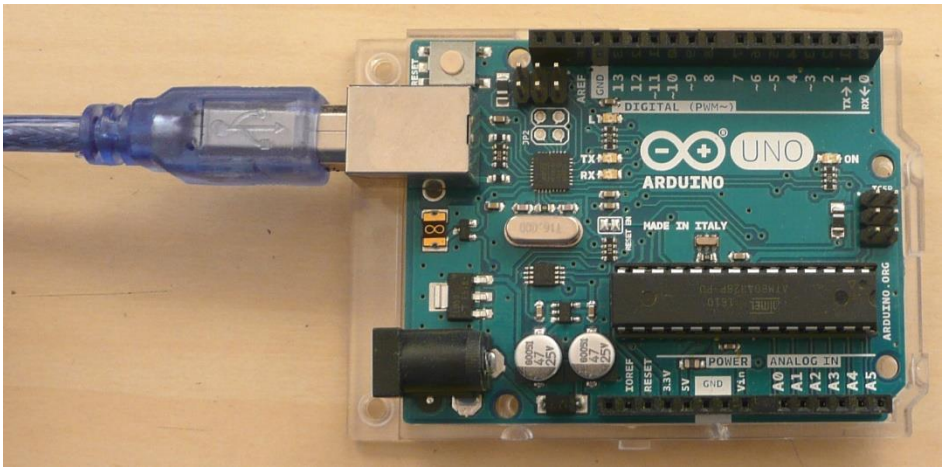


Abbildung 17 Der Arduino-Uno wird über USB mit dem Computer verbunden

Es sei an dieser Stelle erwähnt, dass dieser USB-Anschluss lediglich für den Firmware-Upload verwendet wird. Die spätere Kommunikation über das *DrawBridge*-Programm erfolgt dann über den schon gezeigten FTDI-Adapter. Die Spannungsversorgung des Boards erfolgt über den USB-Anschluss, der später jedoch ausschließlich über den FTDI-Adapter ermöglicht wird.

Für das Firmware-Upload muss natürlich sichergestellt sein, dass in der Arduino-Entwicklungsumgebung sowohl das richtige Board, als auch der richtige COM-Port gewählt wurden. Ich habe den Vorgang in meinen erwähnten Tutorial *ArduinoIDEInstallationVersion20.pdf* beschrieben. Hier dennoch in Kürze die erforderlichen Einstellungen, wobei der COM-Port natürlich individuell verschieden sein kann.

1. Die Wahl des richtigen Arduino-Boards

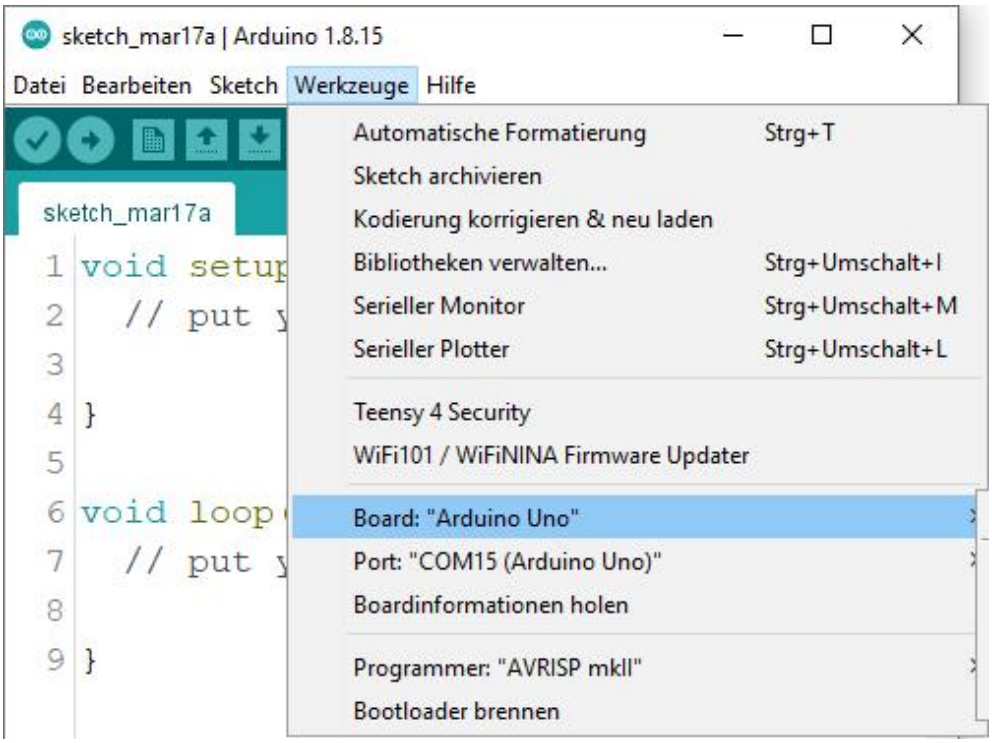


Abbildung 18 Die Wahl des richtigen Arduino-Boards - Arduino Uno

1. Die Wahl des richtigen COM-Ports

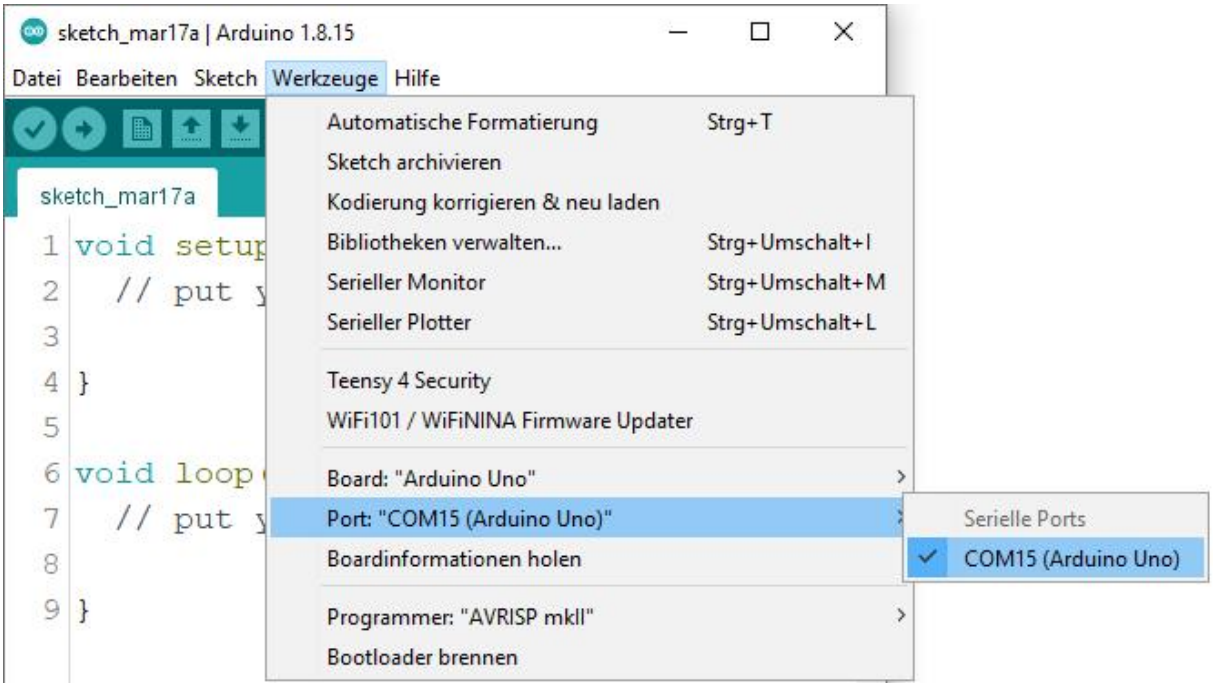


Abbildung 19 Die Wahl des richtigen COM-Ports

Die Verkabelung herstellen

Jetzt muss die USB-Verbindung des Arduino-Uno-Boards vom Computer getrennt werden, denn die Verkabelung sollte in einem stromlosen Zustand erfolgen. Ich rate dazu, den schon gezeigten Schaltplan auszudrucken und alle Verkabelungsschritte zu dokumentieren, also eine Markierung in Form von „ist erledigt“ auf dem Papier durchzuführen. Somit wird sichergestellt, dass auch nichts vergessen wird.

Der Anschluss der Schaltung mit dem Computer

Nach dem erfolgreichen Aufbau der Schaltung kann diese nun über den FTDI-Adapter mit dem Computer verbunden werden. Die Spannungsversorgung des Arduino-Uno erfolgt ab jetzt über den FTDI-Adapter via 5V-Pin, was bedeutet, dass die vorherige Spannungsversorgung über den USB-Anschluss nicht mehr erfolgen darf! Wenn man am 5V-Pin den Arduino-Uno mit Spannung versorgt und gleichzeitig eine USB-Verbindung etabliert, dann hat man quasi zwei 5V parallelgeschaltet. Es kann zu Ausgleichströmen kommen, da die beiden Spannungen nie gleich sein werden. Das kann es zu einer Beschädigung entweder des Arduino-Uno-Boards oder im schlimmsten Fall des Computers kommen!

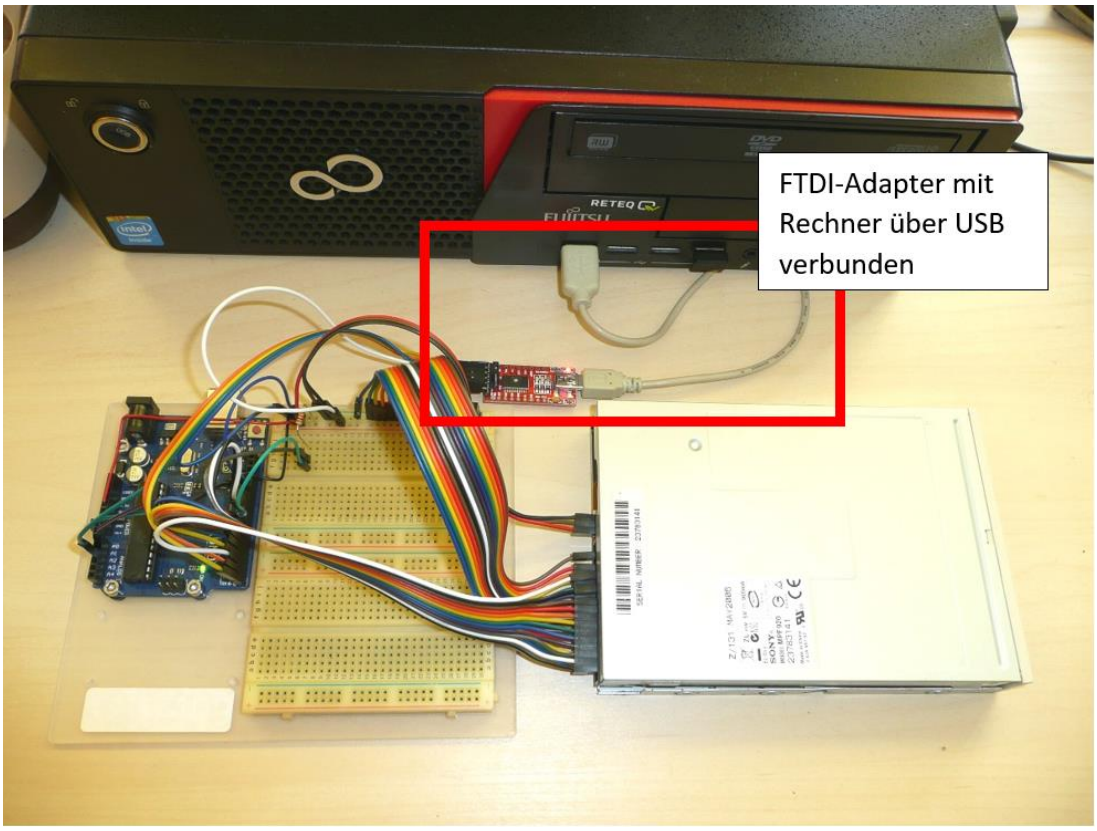


Abbildung 20 Die Spannungsversorgung über den FTDI-Adapter

Natürlich hat der FTDI-Adapter nicht nur die Aufgabe, die Schaltung mit Spannung zu versorgen. Die ganze Kommunikation erfolgt hierüber, denn das DrawBridge-Programm nutzt diesen Weg, um auf das Arduino-Board via serielle Schnittstelle und das Diskettenlaufwerk zuzugreifen.

Das DrawBridge-Programm

Starten wir also das *DrawBridge*-Programm und wählen den korrekten COM-Port dort aus. Er muss mit dem COM-Port des FTDI-Adapters aus dem Gerätemanager übereinstimmen, der bei mir *COM16* ist.

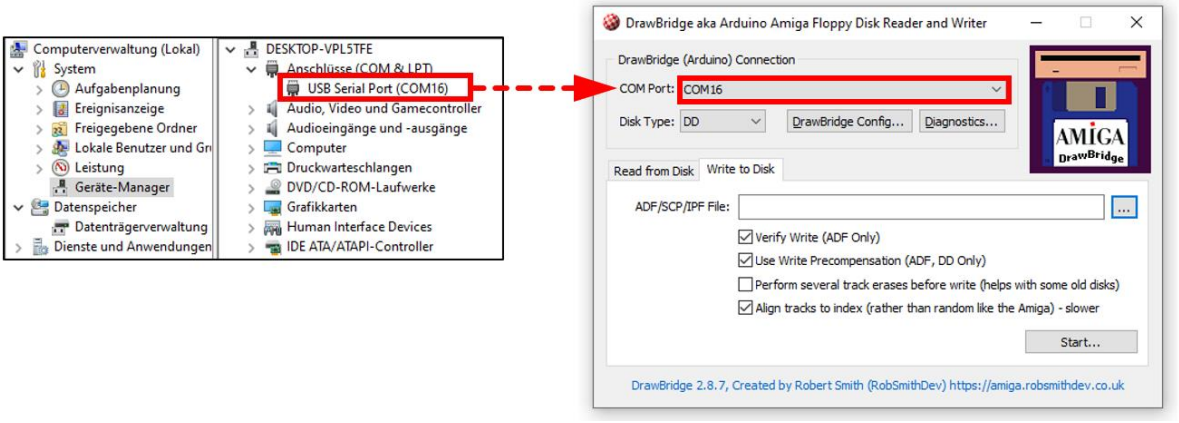


Abbildung 21 Der richtige COM-Port im DrawBridge-Programm

Da ich DD-Disketten verwende, setze ich den Disk-Type auf *DD*.

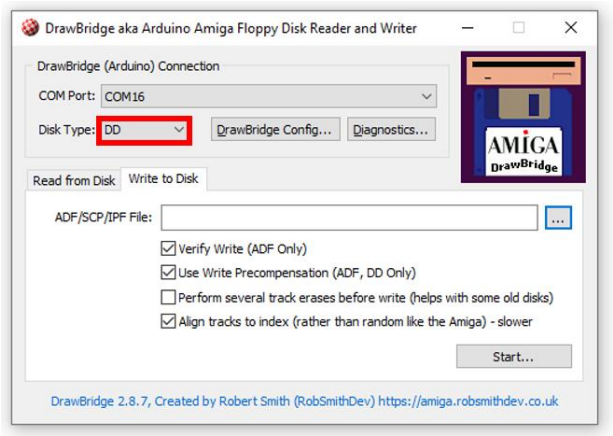


Abbildung 22 Der Disk-Type DD

Alle weiteren Einstellungen, die sich auf dem Tabulator *Write to Disk* befinden, belasse ich auf den gezeigten Standards. Um zu sehen, dass die Kommunikation zwischen dem DrawBridge-Programm und des Diskettenlaufwerks auch funktioniert, starten wir eine Diagnose über die *Diagnostics*-Schaltfläche. Dazu wird einerseits eine schreibgeschützte und voll funktionsfähige, im DD-Format vorliegende Diskette und andererseits eine im DD-Format und nicht schreibgeschützte Diskette benötigt, die beschrieben werden kann.

1. Schritt: Anklicken der Diagnostics-Schaltfläche

Es erscheinen die folgenden Meldungen:

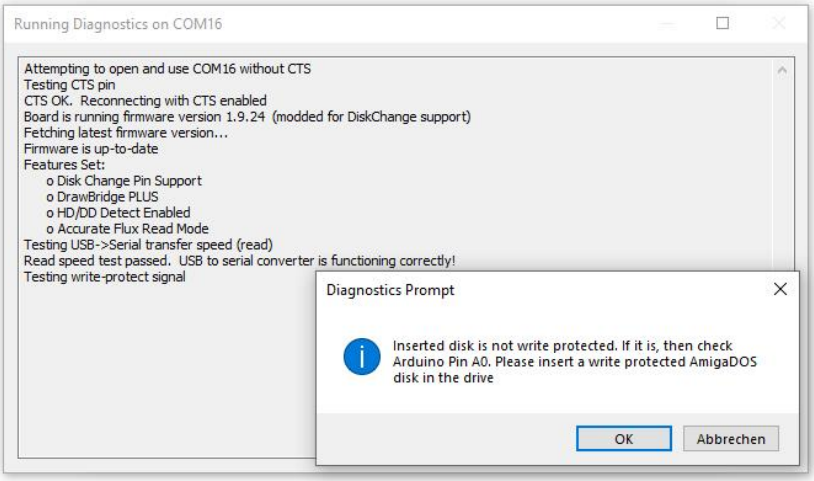


Abbildung 23 Diagnostics - Schritt 1

Es muss also eine schreibgeschützte DD-Diskette im AmigaDOS-Format eingelegt werden.

Schritt 2: Dreht sich die Diskette?

Es wird gefragt, ob sich die Diskette dreht und die LED leuchtet.

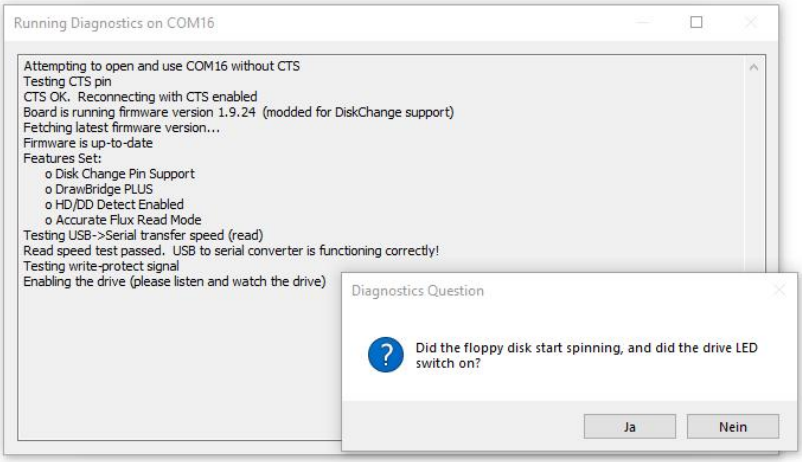


Abbildung 24 Diagnostics - Schritt 2

Schritt 3: Bewegt sich der Diskettenkopf?

Es wird gefragt, ob das kurzzeitige Bewegen des Diskettenkopfes zu hören ist.

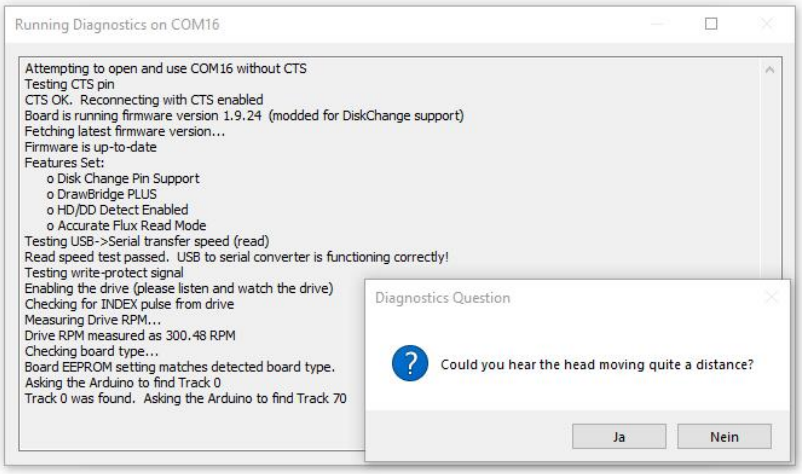


Abbildung 25 Diagnostics - Schritt 3

Schritt 4: Diskette entfernen und neue einlegen

Nun muss die schreibgeschützte Diskette entfernt und eine andere nicht schreibgeschützte Diskette eingelegt werden. Man hat jeweils 30 Sekunden dafür Zeit, bevor es zu einem Timeout kommt.

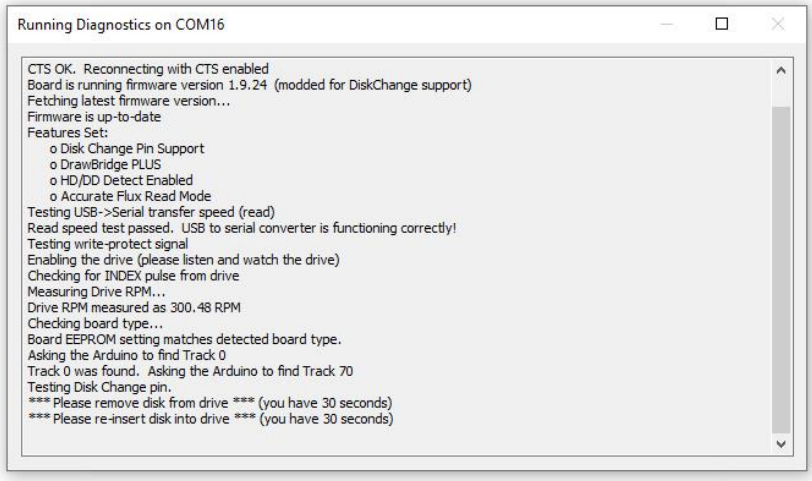


Abbildung 26 Diagnostics - Schritt 4

Schritt 5: Beschreiben einer Diskette

Die jetzt eigelegte Diskette wird auf einem bestimmten Track mit Daten beschrieben, die dann im Anschluss gelesen werden. Es sollte also eine Diskette sein, die nicht mehr in Benutzung ist, da im Anschluss die vorherigen Daten quasi korrupt sind.

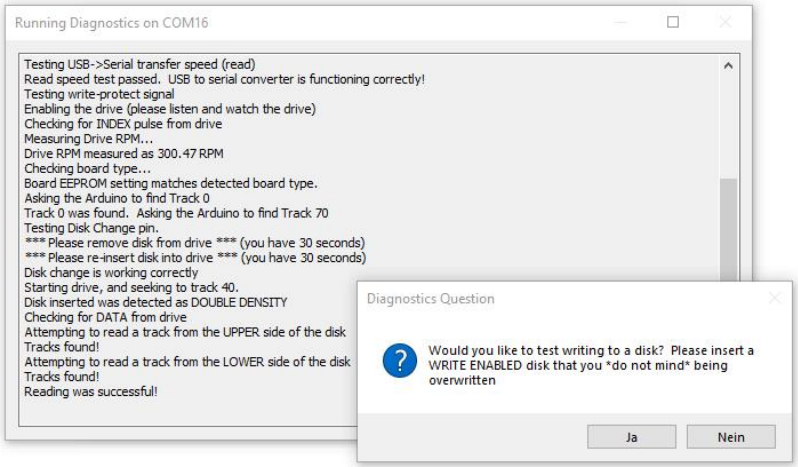


Abbildung 27 Diagnostics - Schritt 5

Schritt 5: Schreib- und Lesevorgang

Erfolgt die nachfolgende Meldung, so hat das Schreiben zumindest des genannten Tracks 41 auf der Ober- und Unterseite der Diskette funktioniert. Alles ist also bis dahin schon einmal sehr positiv verlaufen!

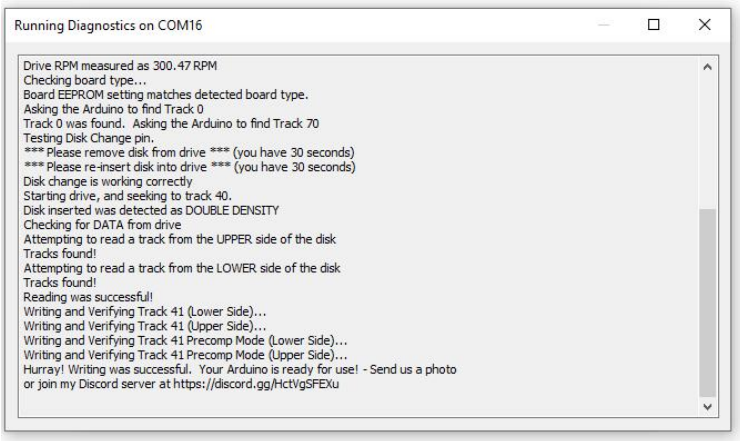


Abbildung 28 Diagnostics - Schritt 6

Das Dialog-Fenster kann geschlossen werden.

Das Schreiben eines ADF-Files

Nun ist es an der Zeit, eine ganze Diskette zu beschreiben und nutzen dafür ein ADF-File, das entweder schon vorhanden ist oder aus dem Internet heruntergeladen werden kann. Die Quellen dazu sind sehr leicht zu finden.

Beim Klick auf die drei kleinen Punkte, die ich rot markiert habe, öffnet sich ein Browser, um eine geeignete ADF-Datei auszuwählen.

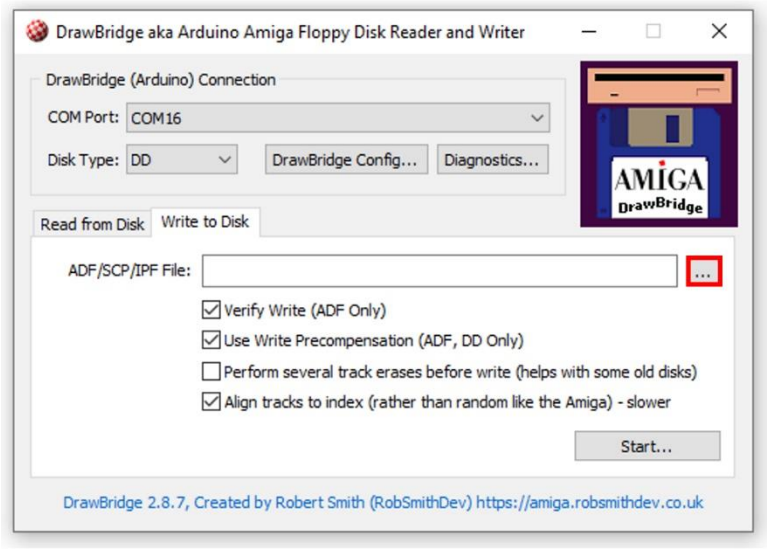
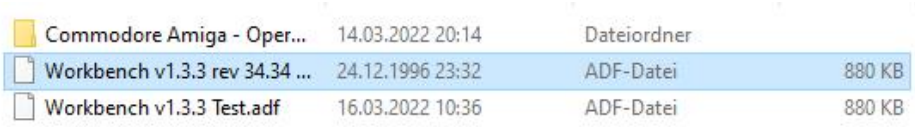


Abbildung 29 Die Wahl einer ADF-Datei

Ich wähle die gezeigte Datei aus und bestätige über die *Öffnen*-Schaltfläche.



Nun kann über die *Start*-Schaltfläche der Schreibvorgang begonnen werden, wobei natürlich zuvor eine nicht schreibgeschützte Diskette eingelegt sein muss.

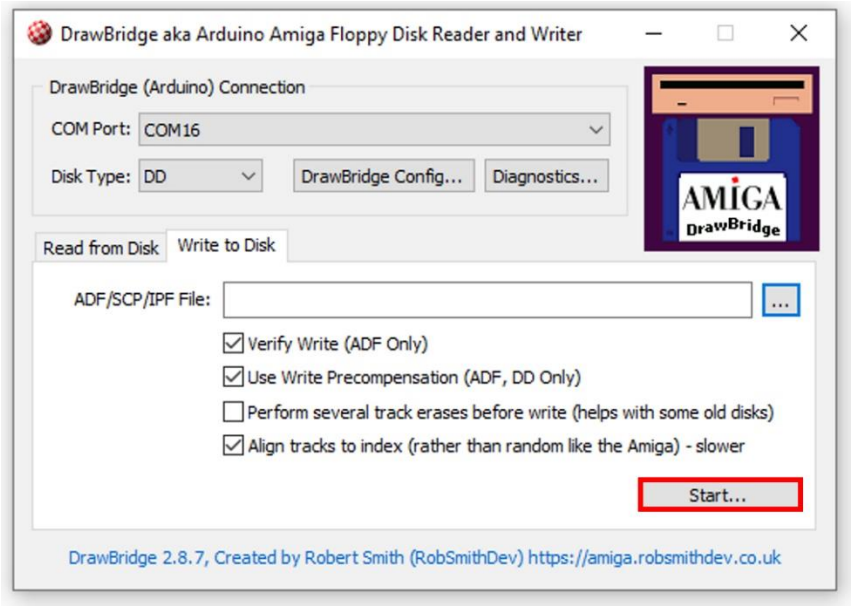


Abbildung 30 Der Schreibvorgang kann gestartet werden

Über eine Fortschrittsanzeige wird der Schreibvorgang protokolliert.

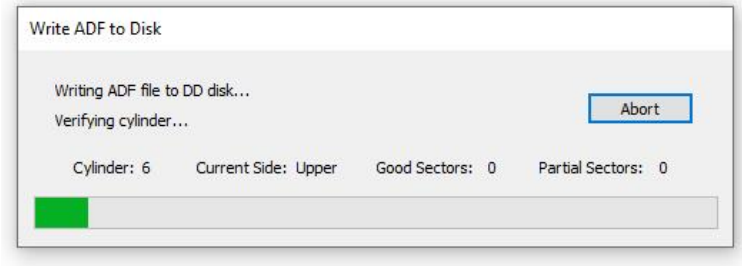


Abbildung 31 Der Schreibvorgang ist gestartet worden

Ist alles geschrieben und überprüft worden, wird die Fortschrittanzeige geschlossen und das DrawBridge-Programm bleibt weiterhin geöffnet. Falls es jedoch nach dem Schreiben und Lesen zu einem Fehler kommen sollte, wird dies auch angezeigt.

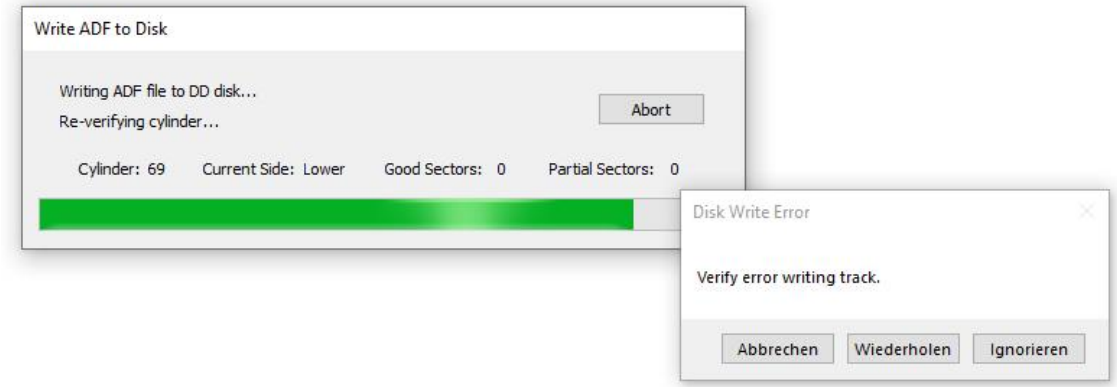


Abbildung 32 Ein Fehler ist aufgetreten

Nun kann der Vorgang wiederholt oder abgebrochen werden, um es vielleicht mit einer anderen Diskette erneut zu versuchen. Ein *Ignorieren* würde ich nicht empfehlen, da es mit Sicherheit zu einer korrupten Diskette führen wird, die nicht lesbar ist.

Abschließend

Ich werde das Ganze später noch in einem Video vorstellen. Für den Link bitte dann auf meiner Internetseite nachschauen.

Hyperlinks!

https://erik-bartmann.de/

https://erik-bartmann.de/?Projekte___Amiga-Disketten_am_PC

Frohes Frickeln!

Erik Bartmann