Bau eines Planetenschleifenprojektors

Ein Projekt von Torben Bertram und Ole Gumbiowski

Projektkurs Astronomie 2016/2017
Carl-Fuhlrott-Gymnasium
Jung-Stilling-Weg 45
42349 Wuppertal
Kursleitung: Bernd Koch
Leitung Sternwarte: Michael Winkhaus
Abgabe am: 01.06.2017

Torben Bertram
Mastweg 1
42349 Wuppertal
tor.bertran@gmail.com

Ole Gumbiowski
Schwabhausen 47
42349 Wuppertal
ole.gumbiowski@gmx.de

Gliederung

1. Einleitung 2
2. Selbstgestellter Arbeitsauftrag 2
3. Motivation 2
4. Der Planet Mars 3
5. Planetenschleifen 4
   5.1 Was ist eine Planetenschleife 4
   5.2 Historie des Phänomens 5
6. Der Projektor 6
   6.1 Überblick Planetarium 6
   6.2 Aufbau des Projektors 7
      6.2.1 Standfuß 8
      6.2.2 Halterung 9
      6.2.3 Die vierte Achse 11
      6.2.4 Endschalter 12
      6.2.5 Anpassung an das Planetarium 13
7. Steuerung 14
   7.1 Steuerplatine und Programm 14
   7.2 Auslösung durch H-Term 17
8. Programmablauf 22
9. Ansteuerung 27
10. Ausblicke 28
11. Danksagung 28
12. Quellen 29
13. Abschlusserklärung 30

1. Einleitung

Diese Arbeit ist die Dokumentation eines Schulprojektes von Ole Gumbiowski und Torben Bertram. Sie wurde im Rahmen eines Projektkurses erstellt, welcher sich mit dem Thema Astronomie befasst. Während dieses Projektkurses arbeiteten wir zum größten Teil selbstständig an vorher selbst festgelegten Projekten.

2. Motivation

Im hauseigenen Planetarium des Schulzentrum Süd wird seit längerer Zeit mittels eines umgebauten Monokular und einem Stativ per Handsteuerung eine Marsschleife an den Abendhimmel projiziert. Unser Ziel war es, diesen Prozess zu automatisieren, indem wir einen „Marsschleifenprojektor“ konstruieren und bauen.

3. selbstgestellter Arbeitsauftrag

Unser Ziel war es, einen funktionstüchtigen Marsschleifenprojektor zu konstruieren und zu bauen. Dazu wollten wir den Verlauf des Mars am Abendhimmel mittels des bereits vorhandenen Monokular und eines Schrittmotors visualisieren. Die Ansteuerung des Schrittmotors sollte mittels Mikrocontroller erfolgen. Der Befehl, die Animation zu starten, sollte durch das Programm „PC-DIMMER“ kommen, welchen der Projektionsleiter selbstständig auslösen kann.

4. Der Planet Mars

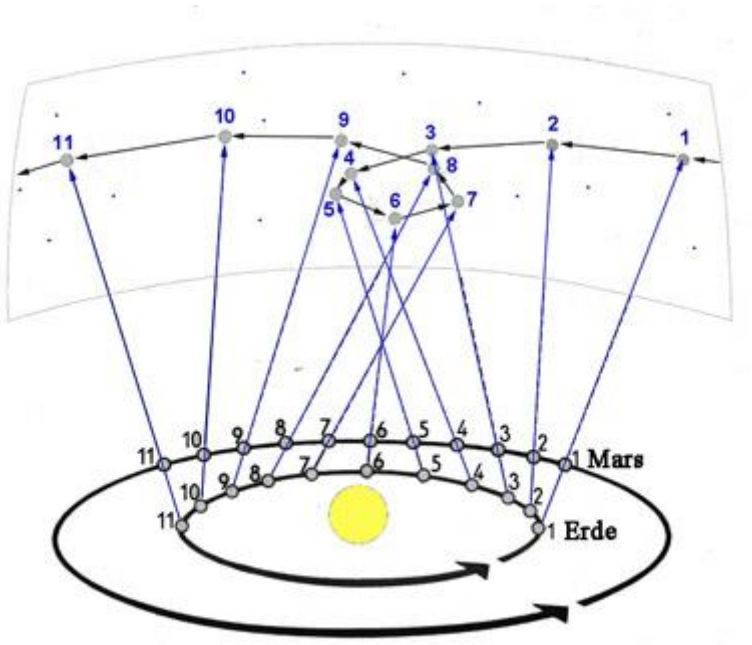
Der Mars ist der vierte Planet in unserem Sonnensystem. Er ist der Erde am ähnlichsten und schon seit der Antike bekannt. Anders als die Erde besitzt der Planet zwei Monde, Deimos und Phobos. Bei klaren Sichtverhältnissen ist der Mars von der Erde aus am Abendhimmel zu sehen und fällt durch seine deutliche Rotfärbung sofort auf.

Der Mars zwischen den Sternenbildern Walfisch und Wassermann (18.01.2017, 22:13 Uhr, ~51.23° NB, ~7.14° WL, 330m ü.NN.)

Unser Nachbar im Sonnensystem unterscheidet sich noch in weiteren Aspekten von unserem Heimatplaneten. So braucht er ca. 1,88 Erdenjahre für eine Rotation um die Sonne und die Schwerkraft dort beträgt nur das 0,385fache von den 1G welche wir erleben. Da der Mars große Ähnlichkeiten mit der Erde hat und mittels relativ kurzen Raumflügen von ungefähr 9 Monaten wird er seit längerem als Ausweichmöglichkeit betrachtet, sollte die Erde nicht mehr bewohnbar sein. Laut NASA ist dies jedoch noch ein fernes Ziel da zunächst die nötige Infrastruktur konzipiert, entwickelt und auf den Mars transportier werden müssen. Zusätzlich wird ein Übersiedeln erschwert, da dies nur in einem Zeitraum möglich ist in welchem Erde und Mars näher bei einander stehen als den Rest der Zeit. Dies findet jedoch nur alle zwei Jahre und zwei Monate statt.

5.1 Was ist eine Planetenschleife?

Planetenschleifen sind ein optisches Phänomen bei, welchem der beobachtete Planet auf seiner Bahn durch die Sternenbilder scheinbar an einem bestimmten Punkt die Richtung umkehrt, um einige Zeit später wieder in die originale Richtung fortzufahren. Der Effekt entsteht wenn die Erde einen Planeten bei der Sonnenumrundung überholt. Dieser Vorgang lässt sich in vier Phasen gliedern. In Phase Eins liegt der zu überholende Planet auf seiner Laufbahn vor der Erde, am Himmel bewegt er sich vorwärts (Punkt 1-4). Der Planet steht immer tiefer am Himmel. In Phase Zwei kommen sich die Planeten am nächsten(Punkt 5-6). Der Mars wandert immer langsamer vorwärts bis es schließlich so scheint als würde er sich zurück bewegen. Während Phase Drei, der direkten Opposition der Himmelskörper(Punkt 6), steht der Mars am tiefsten am Himmel. Die rückläufige Bewegung hält nach diesem Punkt noch ein paar weitere Tage an bis der Planet in Phase Vier seine ursprüngliche Richtung weiter verfolgt (Punkt 7-11).



Da dieser rein optische Effekt darauf beruht, dass der innere Planet den äußeren auf seiner Bahn überholt, kann das Phänomen von der Erde aus nur bei den Planeten Mars bis Neptun beobachtet werden. Befände man sich zum Beispiel auf dem Merkur ließe sich auch eine Erdenschleife am Himmel ausmachen. Diesem Prinzip folgend kann auf dem Saturn nur eine Schleifenbewegung der Planeten Uranus und Neptun beobachten. Anzumerken ist auch, dass auch in anderen Sonnensystemen Planetenschleifen am Himmel zu sehen sind wenn es mindestens einen zweiten weiter außen liegenden Himmelskörper gibt.

5.2 Historie des Phänomens

Die Planetenschleifen lassen sich seit der Entstehung des Sonnensystems wie wir es heute kennen beobachten. Wissenschaftlich untersucht wurden sie als erstes im antiken Babylon wo sie als Extrapolation ihrer Bahnen verstanden wurden. Im dritten Jahrhundert vor Christus setzte sich die Epizykeltheorie durch, nach welcher die Planeten auf kleinen Kreisbahnen laufen welche selbst auf einer Kreisbahn um die Erde wandern. Dieses Prinzip lässt sich heutzutage noch anwenden wenn man sich die Laufbahn des Erdmondes um die Sonne anschaut. Erst mit Aufkommen des heliozentrischen Weltbildes um 1500 durch Kopernikus entstanden andere Theorien wie die eigenartige Bewegung zu Stande kommt. Doch auch Kopernikus konnte sich nicht von der göttlichen Schöpfung lösen und ging davon aus die Planeten seien an Kristallspheren befestigt und würden sich durch diese bewegen. Durch die Beobachtung sich kreuzender Kometenbahnen konnte Tycho Brahe die Spherentheorie widerlegen. Der nächste große Durchbruch gelang Johannes Kepler, der als erstes von nicht perfekt runden Bahnen ausging. Dank Galileis Beobachtungen mit dem neu erfundenen Teleskop, wusste Kepler, dass die Sterne nicht auf einer einzelnen Ebene liegen, sondern über die Tiefen des Raumes verteilt sind. Jedoch blieb die Frage im Raum stehen wie die Planeten am Himmel bleiben und nicht herab stürzen. Eine Antwort darauf lieferte Newton mit seinen Erkenntnissen zur Schwerkraft und ihre Wirkung auf andere Objekte. Mit diesen Theorien ließen sich die Schleifenbewegungen als optisches Phänomen einstufen. Heute weiß man, dass die außergewöhnliche Himmelserscheinung durch den Wechsel des Beobachtungswinkels während der Opposition entsteht.

6. Der Projektor

6.1 Überblick Planetarium

Das Kellerplanetarium des CFG Wuppertal besteht aus einer Stoffkuppel die aus fünfeckigen Teilstücken zusammengesetzt wurde. Im Inneren befindet sich das Herzstück, der Sternenhimmel Projektor welcher 2008 von Thorben Beckert und Lukas Varnhorst gebaut wurde. Dieser ermöglicht es den Sternenhimmel von jedem Punkt auf der Erde zu visualisieren. Klappen, welche nachträglich in der Kuppel verbaut wurden, ermöglichten es weiter feststehende Projektoren zu installieren. Mittels dieser Projektoren lässt sich unter anderem der Orionnebel und diverse Visualisierungen von Sternenbildern projizieren. Ein weiterer Projektor ermöglicht die Darstellung von Polarlichtern. Gesteuert werden fast alle Funktionen durch einen Desktop PC und das Programm PC-Dimmer. Außerdem sind im Inneren Hartschaummatten verbaut, auf welchen die Zuschauer Platz nehmen können. Der Eingang des Planetariums ist mit Vorhängen versehen, um maximale Abschottung des Außenlichtes zu erreichen. Die Projektion unserer Marsschleife soll durch ein kleines Loch in der Kuppel geschehen, der Projektor befindet sich außerhalb der Kuppel hinter dem Planetarium. Um den sicheren Halt des Projektors zu gewährleisten, wurde ein Standfuß unter dem Projektor am Schrittmotor installiert.

Das Kellerplanetarium im Schulzentrum Süd (Quelle: Jens Stachowitz)

6.2 Aufbau des Projektors

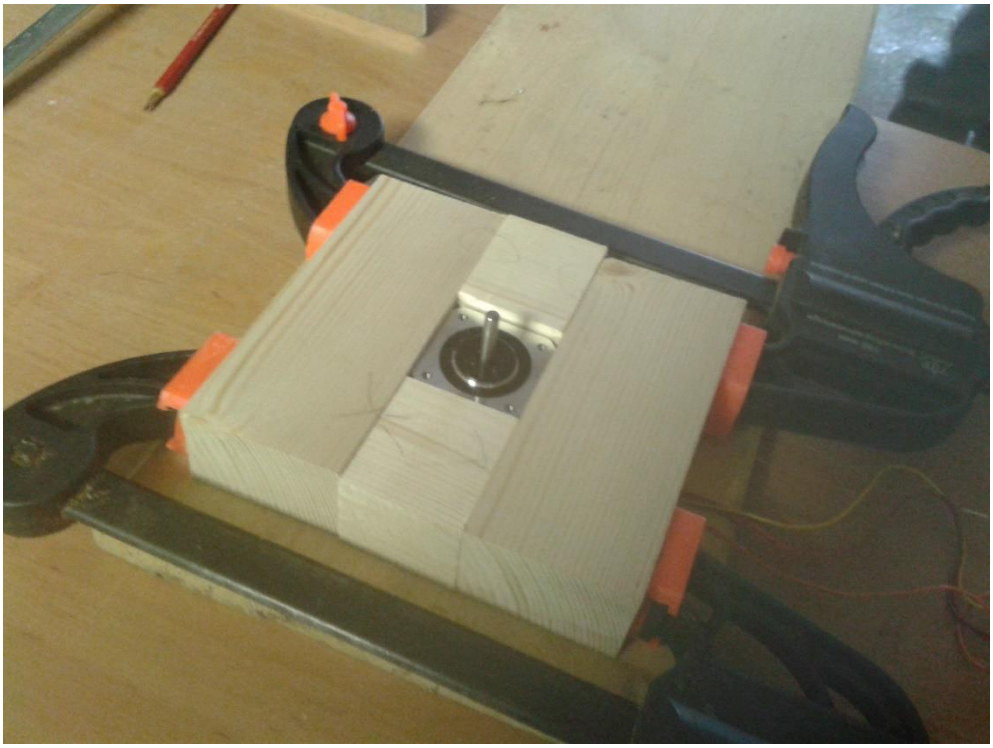
Der Rahmen des Projektors besteht aus einem Alusteckprofil der Firma ITEM. Das Unternehmen mit dem Hauptsitz in Solingen stellt diverse Metallprofile überwiegend für die Industrie her, aus welchen man dann Produktionsanlagen, Rollwagen, Montierungen etc. selber bauen kann. Die Firma ITEM wurde uns von einem Bekannten empfohlen, welcher bereits mehrfach mit den Profilen der Firma gearbeitet hat. Wir entschieden uns nach Recherchen auf der Website des Unternehmens für das Profil 6 (firmeneigener Name) mit den Kantenmaßen 30x30 mm. Das Profil hat auf jeder Fläche einen schienenförmigen Hohlraum, welcher dazu dient mehrere Stangen/Schrauben/Winkel etc. miteinander zu verbinden.

Profil 6 (Quelle: Item24.de)

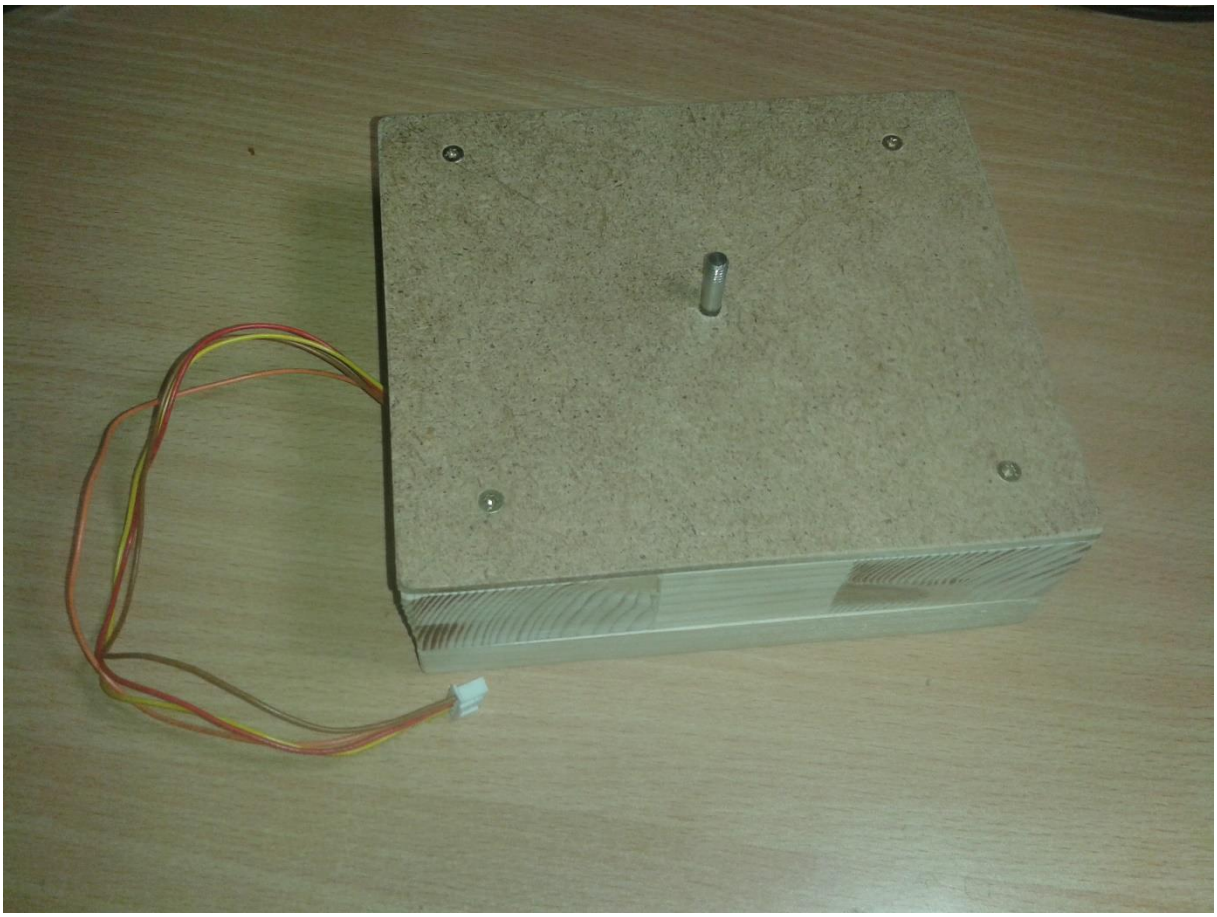
Da dieses Profil alle von uns gewünschten Eigenschaften aufwies, entschieden wir uns für dieses Profil. Nutensteine können in die Hohlschiene eingeführt werden und besitzen eine Durchbohrung, mit welchem man dann die Profilstange mit etwas Anderem (in unserem Fall mit der Schrittmotorwelle und weiteren Profilstangen) verschrauben kann. Um die Rahmenelemente miteinander zu verbinden, nutzten wir die bereits vorhandenen Löcher im Aluminiumprofil, welche sich am Ende des Profils befinden und einen Durchmesser von 4 mm haben. Diese Löcher verwendeten wir an den Stangen 2 und 3, auf deren Position und Aufgaben wird im Folgenden weiter eingegangen. An Stange 1 wurde an beiden Enden der Stange die Führungsschiene mittels Fräse entfernt, um die Stangen besser verschrauben zu können. Anschließend wurden die Stangen miteinander verschraubt, dafür verwendeten wir 6 cm lange Schrauben mit einem Durchmesser von 5 mm. Um die horizontale Stange und die Schrauben nicht unverhältnismäßig zu beschädigen, haben wir eine Vorbohrung durchgeführt, damit sich die Schrauben flüssiger durch das Material treiben lassen.

6.2.1. Standfuß

Um einen sicheren Halt des Projektors zu erreichen, haben wir um den Schrittmotor herum ein eigenes Gestell gebaut, welches aus Holzplatten und Kanthölzern besteht. Die Kanthölzer mussten auf die Höhe des Motors abgehobelt werden um möglichst wenig Spielraum in der Vertikalen zu ermöglichen. Die fertig bearbeiteten Holzlatten wurden so mit Holzleim verbunden, dass der Motor stramm in der Mitte liegt. In eines der Seitenteile musste eine Fuge gesägt werden um die Kabel des Motors herausführen zu können.



Unter die Kanthölzer schraubten wir bündig mit den Kanten eine Holzplatte mit einer Dicke von 15mm um eine gerade Grundfläche zu schaffen. Für die obere Platte berechneten wir eine maximale Dicke von 2 mm, da wir von der Motorwelle noch mindestens 12 mm benötigten. Um die Welle angemessen mit der Profilstange und dem Nutenstein zu verbinden, benötigten wir 6 mm für den Nutenstein und 3 mm, um die Dicke der Führungsschiene zu überbrücken. Um die Achse mit der Halterung zu verbinden musste auf die Motorwelle ein M6 Gewinde geschnitten werden.



Der fertige Standfuß

6.2.2. Die Halterung

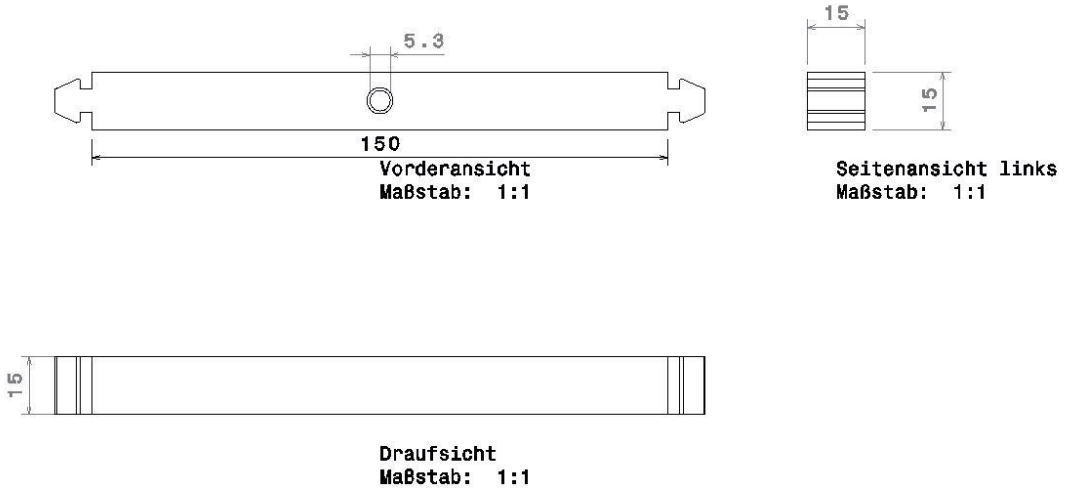
Um das Monokular nicht direkt auf dem Motor fixieren zu müssen entschieden wir uns mittels einer Aluminiumstange (Profil 6 Natur der Firma ITEM) ein Gerüst zur Fixierung zu bauen. Hierzu schnitten wir aus der Rohstange zwei Stangen a 12cm für die Vertikale und eine Stange mit 21cm für die Horizontale. Um die Teile zu verbinden musste in das Profil der kurzen Stücke ein 6mm Innengewinde geschnitten werden. In das längere Stück mussten Löcher mit einem Durchmesser von 6,5mm gebohrt werden, um mittels Schrauben die Teile zu verbinden. In der Mitte der Horizontalen Achse musste ein Nutenstein fixiert werden um einen Verbindungspunkt zwischen Gestell und Motorwelle zu schaffen. Eine Stabile und langlebige Fixierung konnte allein durch den Nutenstein nicht gewährleistet werden. Um den Halt zu verstärken nutzten wir zusätzlich einen Zwei-Komponenten-Epoxitkleber wie er in der Industrie verwendet wird.

Nutenstein mit Klebstoff

Der Rahmen des Projektors, ohne gedruckte Trägerachse

6.2.3. Vierte Achse

Die Achse, auf welcher auch das Monokular montiert ist, verläuft parallel zur horizontalen Stange. Sie wurde speziell für dieses Projekt in einem 3D-Drucker erstellt. In der Achse wurde eine 0,25“ Gewindeschraube eingelassen, auf welchem ein Kugelkopfgelenk geschraubt wurde. Mit diesem Kugelkopfgelenk ist das Monokular direkt verbunden, ebenfalls über eine 0,25“ Schraube, welche durch die Kugel vollkommen frei ausrichtbar ist. Da die Achse in der ersten Version noch nicht schwenkbar ist, wird sie vorerst mit den vertikalen Stangen verbunden.



Technische Skizze der gedruckten Achse

6.2.4 Endschalter

Der Vollständige Ablauf der Bewegung ist weniger als eine volle Drehung. Würde man die Projektion also beenden und eine weitere starten wollen würde sie an einer anderen Stelle beginnen. Um dies zu verhindern entschieden wir uns einen Kontaktschalter einzubauen. Nach der beendeten Projektion beginnt der Motor zurück in die Ausgangsstellung zu drehen, sobald der entsprechende Befehl am PC ausgelöst wird. Da aber ein genaues Einstellen mittels identischer Schrittzahl wie in Projektionsrichtung ein großer Aufwand wäre und die Projektion anderer Planeten als des Mars als Standardprogramm erschweren würde, da andere Intervalle gefahren werden, nutzen wir diese elektronische Lösung. Dreht sich der Motor und damit das Aluminiumgestell in Richtung Ausgangsposition, so aktiviert dieses an einem bestimmten Punkt den Kontaktschalter, dessen Signal an die Platine geleitet wird. Im Programmtext ist definiert, dass der Motor die Bewegung bei behalten soll bis das Signal des Schalters empfangen wird.

Endschalter am Standfuß mit Haltegestell

6.2.5 Anpassung an das Planetarium

Da die Steuerplatine über eingebaute LEDs verfügt welche bei Betrieb automatisch angesteuert werden musste eine Licht undurchlässige Ummantelung für die Mikroelektronik geschaffen werden. Wir nutzten ein für solche Zwecke konzipiertes Kunststoffgehäuse in welches zunächst Durchführungen für die Verbindungskabel eingefügt werden mussten. Um einen Ansatz zu haben sägten wir als erstes mit einer Feinsäge Kerben an die Stellen an welche die Löcher gesetzt werden sollten. Im zweiten Schritt feilten wir die Kerben unter Zuhilfenahme einer Rundfeile in Form

Verkleidung mit Kerbe

Verkleidung mit fertigen Öffnungen

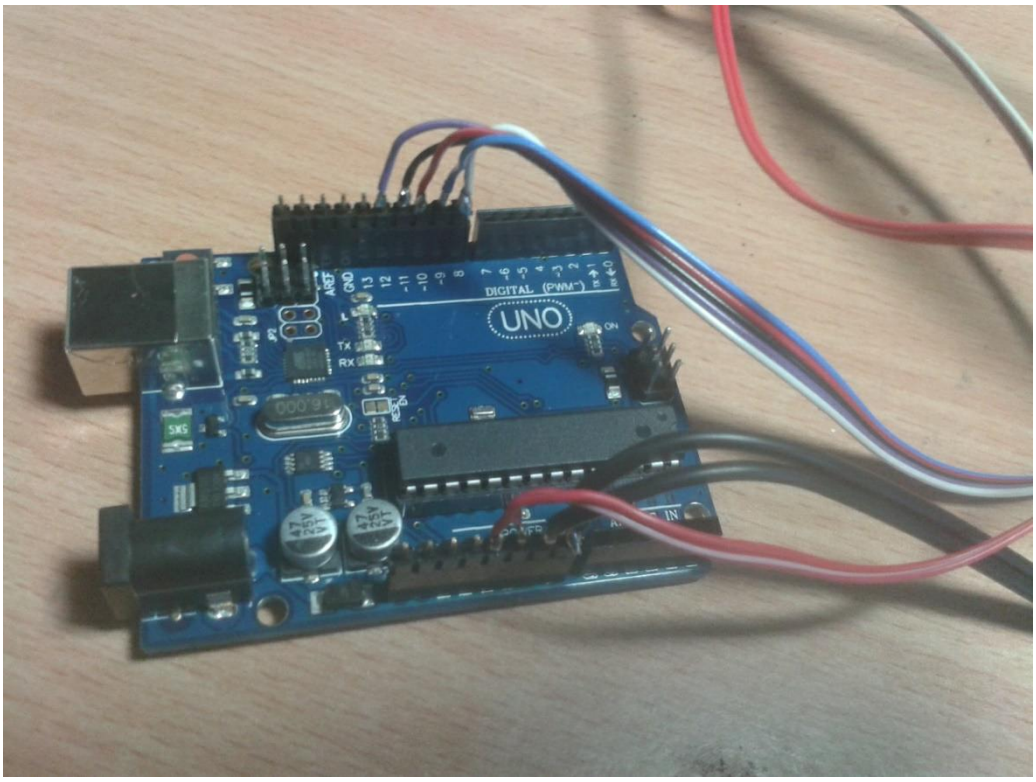
Um hier Lichtverschmutzungen zu vermeiden setzten wir zusätzlich noch Kabelklemmen in die Löcher ein welche zusätzlich auch eine Fixierung darstellen.

7. Die Steuerung

7.1 Steuerplatine

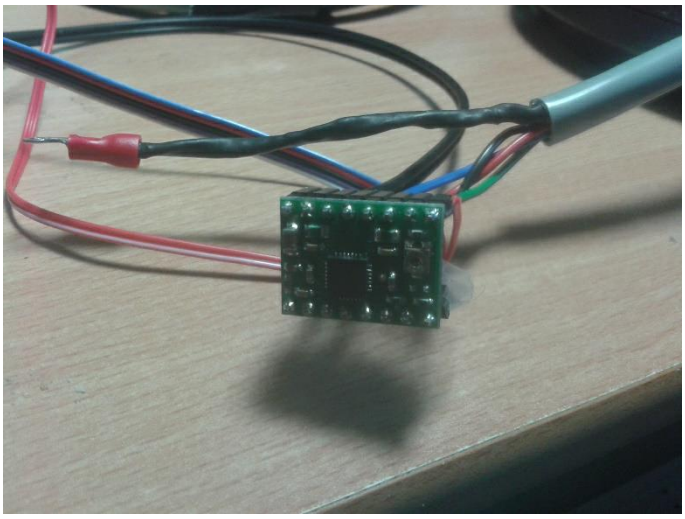
Um die Schrittmotoren anzusteuern, das heißt dem Motor einen konkreten Befehl geben, welchen dieser daraufhin ausführen soll, benötigt man zunächst einen sogenannten Mikrocontroller. Ein Mikrocontroller ist ein Bauteil aus der Elektrotechnik, welcher einen am Computer erstellten Befehl abspeichert und auf Kommando (ausgelöst von H-Term, siehe Punkt 9) hin ausführt. Wir nutzten zu diesem Zweck den vorverbauten ATMega328P. Dieses Bauteil ist auf einer Platine, einer Arduino UNO, aufgebaut. Die Platine muss für unseren Aufbau zusätzlich mit einem Schrittmotortreiber verbunden werden, welcher die Schnittstelle zwischen Ausgang an der Platine und Schrittmotor darstellt. Außerdem passt der Schrittmotortreiber die anliegende Spannung (5V) sowie die anliegende Stromstärke (1,5A) auf die vom Schrittmotor benötigten 2,79V bzw. 1,33A an.

Die Platine verfügt ebenfalls über mehrere Anschlüsse: Ein Hauptanschluss, in welchen das Netzgerät eingesteckt wird, um die Platine mit Strom zu versorgen. In unserem Fall handelt es sich um das Netzgerät PPI-12150-GS der Firma Logitec®. Des Weiteren gibt es einen USB und einen seriellen Anschluss, welche die Platine mit einem Computer (USB) oder dem Programmierer verbinden. Der Begriff „seriell“ wird im Folgenden weiter erläutert. Die weiteren Anschlüssen sind Ausgänge oder Eingänge, diese werden in unserem Fall durch die Motoren belegt. Sie erfüllen verschiedenste Zwecke, es gibt Ein- oder Ausgänge (Input/Output, im folgendem auch I/O) zur Versorgung (Arduino UNO) oder Aufnahme (Treiber) elektrischer Spannung. Desweiteren gibt es I/O, welche Daten von der Hauptplatine auf den Treiber übertragen, z.B. Schrittweiten, die Anzahl der Schritte, Pausen und deren Dauer zwischen den einzelnen Schritten und Schrittrichtung.

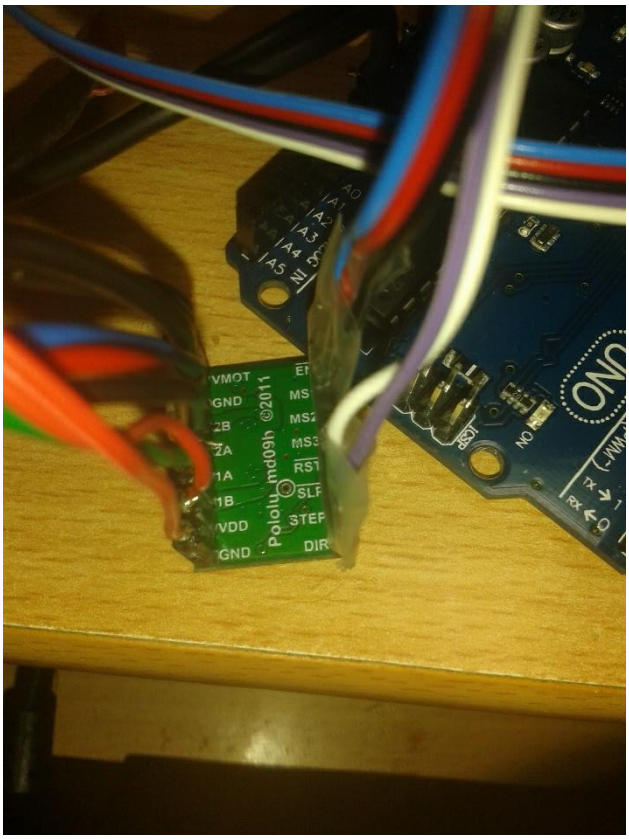


Steuerungsplatine Aduino Uno 328P

Die Ausgänge Acht bis zwölf der Platine wurden mit den Anschlüssen MS1, MS2, MS3, STP und DIR verbunden. RST und SLP mussten kurz geschlossen werden. Die Ausgänge GND mit GND am Carrier, Vin mit VMOT und der zweite Ausgang GND mit VDD. Die letztgenannten Verbindungen sind für die Stromversorgung des Motors zuständig. Um die oben genannten Werte für Spannung und Stromstärke zu erreichen musste der in der Carrier Platine integrierte Strombegrenzer so eingestellt werden, dass die benötigte Spannung sowie Stromstärke erreicht werden. Die sich am Carrier befindenden Ausgänge 1A, 1B, 2A und 2B mussten nun mit den entsprechenden Kontakten des Motors verbunden werden. Die Festigkeit der Verbindungen konnte durch das verlöten der Kabel gewährleistet werden.



Pololu A4988 StepStick Schrittmotortreiber (Treiber=schwarzes Bauteil in der Mitte)



Pololu A4988 StepStick Stepper Motortreiber Carrier, Black Edition (Ansicht von unten)

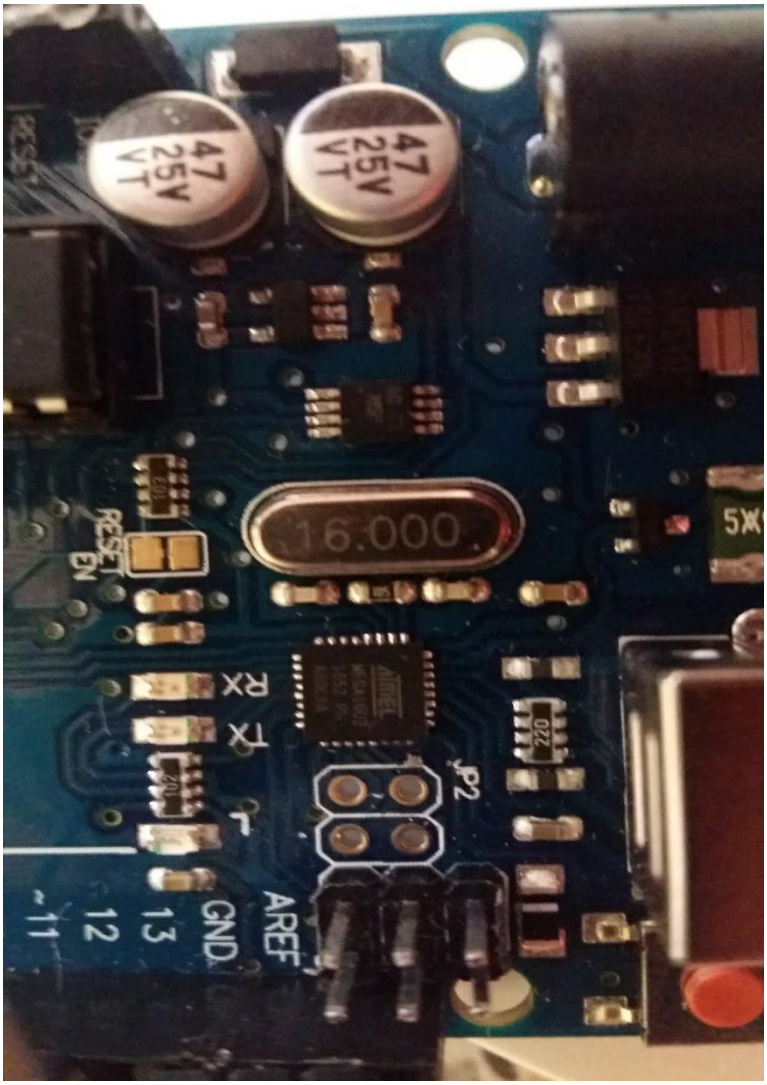
7.2 Steuerprogramm

Das Programm des Marsschleifenprojektors wurde mit der Software „ATMEL® Studio 7.0“ in der Programmiersprache „C“ erstellt. Das Unternehmen ATMEL® ist bekannt für Entwicklungssoftwaren sowie integrierte Schaltungen und hat seinen Hauptsitz in Kalifornien, USA. Die Programmiersprache „C“ wurde für das Unternehmen Microsoft® entwickelt. Um uns mit der Programmiersprache etwas vertrauter zu machen, besuchten wir die Universitätsbibliothek der Universität Wuppertal am Campus Freudenberg. Dort liehen wir folgende Werke aus, um uns mit dem Programmieren ein wenig vertrauter zu machen:

- Das Eagle PCB-Design, Autor: Herbert Bernstein (2004)
- AVR-Mikrocontroller Praxis, Autor: Safinaz Volpe (2002)
- Softwareentwicklung für Mikroprozessoren, Autor: Jörg Wiegelmann (2009)
- Programmieren des AVR RISP II Mikrocontroller mit BASCOM AVR, Autor: Claus Kühnel (2004)
- Programmieren in C, Autor: Robert Klima (2007)
- C als erste Programmiersprache, Autor: Manfred Dausmann (2008)

Mit diesen Büchern sowie mit fachkundiger Unterstützung bereiteten wir uns auf das Programmieren vor.

Der erste und wichtigste Schritt zur Erstellung eines Programmes ist das Definieren von unbekannten Zahlenwerten (Variablen, ähnlich wie in der Mathematik) sowie das Einfügen von I/O-Bezeichnungen. Letzteres bedeutet konkret, dass den vielen verschiedenen Ein- und Ausgängen Namen zugeteilt werden, welche dann im weiteren Programmtext dazu genutzt werden, die einzelnen I/O spezifisch anzusteuern. Ebenso muss ein Wert für den Schwingquartz festgelegt werden. Ein Schwingquartz dient dazu, die Frequenz der elektronischen Schwingungen festzulegen, welche der Mikrocontroller maximal leisten kann. Wir entschieden uns den allgemeinen Richtwert von 8000000Hz einzuhalten.



Der Schwingquartz (ovales Bauteil) des Arduino UNO

Um zu gewährleisten, dass alle vier Litzen des Schrittmotors eine eigene Stromversorgung haben, wurde jedes Kabel einzeln am Treiber verlötet. Der Treiber kann dadurch ein eingegangenes elektronisches Signal (Byte) direkt der entsprechenden Litze zuordnen, da das Signal vom Mikrocontroller aus kommt. Auch an den I/O des Mikrocontrollers ist je ein I/O für eine Litze vorgesehen, damit das Datensignal nicht versehentlich falsch verarbeitet wird. Je nach Größe des elektrischen Signals (z.B. 1 Byte, 2 Byte, 4 Byte, 8 Byte) muss im Programmtext eine entsprechende Variable gewählt werden. Dies wird durch die so genannten Variablentypen übernommen. Diese sind plattformabhängig festgelegt und lauten in diesem Fall wie folgt:

-char = 1 Byte

-short int = 2 Bytes

-int = 4 Bytes

-double = 8 Bytes

-long double = 16 Bytes

Ebenso kann man im Programmtext Bedingungen oder Abfragen stellen. Als Beispiel dient hier „if“. Das englische Wort für „Wenn“ erfüllt genau das, was sein Name vermuten lässt. Hier ein Beispiel:

```
(

If (wert==1);

Then(turn.left);

Else(turn.right);

)
```

(Wenn der Wert 1 entspricht, dann erfolgt eine Linksdrehung. Ist der Wert ungleich 1 erfolgt eine Rechtsdrehung)

Das Simikolon am Ende einer Zeile kennzeichnet das Ende eines Befehl, d.h. es wird ein Befehl erstellt, welche das Programm selbstständig überprüfen soll. Die Klammern (Im Programmtext sind diese geschwungen) kennzeichnen den Beginn sowie das Ende einer Funktion. So kann das Programm zuordnen, welche Einzelbefehle zu welcher Funktion gehören und diese auch gezielter abarbeiten.

Nun muss man festlegen, bei welchem Zahlenwert ein Port-Pin zu einem Eingang und bei welchem zu einem Ausgang wird. Dazu kann man als Beispiel den Port B auf 1 oder auf 0 setzen. Liegt der Zahlenwert bei 1, so handelt es sich um einen Ausgang, bei 0 genau umgekehrt. Diese Festlegung übernimmt der Mikrocontroller beim programming, dieser Begriff wird im Folgenden weiter ausgeführt. Hat man nun all diese Einstellungen getroffen, so kann das genaue festlegen der einzelnen Schritte beginnen. Dazu zeichneten wir den Verlauf der Marsbahn während einer Marsschleife auf Papier vor und überlegten uns zunächst, welche einzelnen Schritte in welche Richtung mit welchem zeitlichen Abstand erfolgen müssen. Nachdem wir diese grobe Planung beendet hatten, mussten die „groben Gedanken“ in präzise angepasste Bewegungen umgesetzt werden. Dabei gestaltete sich die optische „Entschleunigung des Mars“ kurz vor der eigentlichen Schleife am schwierigsten, da hier die Pausen zwischen den einzelnen Schritten sowie die Schrittweite komplett neu angepasst werden mussten.

Zu Beginn der Marsprojektion geht der Mars am Horizont auf. Daher musste zu Beginn der Animation keine beschleunigte Bewegung umgesetzt werden. Der Mars läuft also zunächst ganz normal auf den Horizont. Dazu ließen wir den Motor zu Beginn der Animation 400 Schritte a 0,225 Winkelgrad, also insgesamt 90 Winkelgrad drehen, wovon etwa ein Drittel tatsächlich am Abendhimmel zu sehen sind. Anschließend wurde der Motor „entschleunigt“, indem wir die Viertelschritte auf Achtelschritte, also 0,1125 Winkelgrad, kürzten. Ebenso wurde die Anzahl der Schritte von 400 auf 20 gekürzt, um das „Auslaufen“ des Mars nicht unnötig herauszuzögern. Weiterhin wurde die Schrittweite von Achtelschritten auf Sechszehntelschritte (0,05625 Winkelgrad) gekürzt, damit die Bewegung nicht zu abrupt stoppt. Zwischen den Änderungen der Schrittweiten wurde jeweils eine Pause von 1 ms eingelegt, damit die Animation fließend verläuft. Alle bisherigen Schritte verliefen im rechtslauf.

Nach den Sechszehntelschritten stoppt die Animation für 1000 ms. Diese Zeit würde in der Weiterentwicklung des Projektors genutzt werden, um die Bewegung in vertikale Richtung hervorzuheben. Anschließend dreht der Projektor im Linkslauf weiter, um die Schleife zu visualisieren. Hier lief die Reihenfolge der Schrittweite genau andersherum: Die Drehung begann mit kleinen 16tel Schritten, ging über in die Achtelschritte und verlief mit Viertelschritten weiter. Es wurden ebenfalls wieder jeweils 20 16tel bzw. Achtelschritte gemacht, wieder mit einer Trennpause von 1 ms. Nach der Beschleunigungsphase folgen erneut Viertelschritte, diesesmal jedoch nur 170, da das Hineinlaufen in den Horizont hier wegfällt. Nach diesen 170 Schritten läuft der Motor genau wie nach der ersten Drehung mit 20 Achtel- und 16telschritten mit 1 ms Pause aus.

Im letzten Teil der Animation läuft der Mars wieder aus dem Horizont heraus. Dafür wird genau wie bei der Linksdrehung beschleunigt und der Mars läuft wieder mit insgesamt 90 Winkelgrad dieses Mal aus dem Bild.

Um den Algorithmus einfacher programmieren zu können, erstellten wir folgende Funktion. Mit dem Befehl „mehrereSchritte“ ließ sich der Grundbefehl aufrufen. Hinter diesen Befehl wurde eine Klammer gesetzt, in welche wir vier Werte eintragen mussten. Dies waren in angegebener Reihenfolge:

-Anzahl der Schritte

-Drehrichtung (0=Rechtsdrehung, 1=Linksdrehung)

-Pausenlänge zwischen den einzelnen Schritten (hier: 10ms)

-Schrittart (1=Vollschritt, 2=Halbschritt, 4=Viertelschritt usw.)

Anschließend wurde der Befehl „_delay_ms(xxx);“ aufgerufen. Dieser erstellt den Befehl, dass das Programm hier eine gewisse Zeitspanne x warten soll. Die Einheit ließ sich frei wählen, wir entschieden uns für Mikro- und Millisekunden, um eine ruckartige Bewegung des Mars am Abendhimmel zu vermeiden.

```
_delay_ms(1000);
mehrereSchritte(400,0,10,4);
_delay_ms(1);
mehrereSchritte(20,0,10,8);
_delay_ms(1);
mehrereSchritte(20,0,10,16);
_delay_ms(1000);
mehrereSchritte(20,1,10,16);
_delay_ms(1);
mehrereSchritte(20,1,10,8);
_delay_ms(1);
mehrereSchritte(170,1,10,4);
_delay_ms(1);
mehrereSchritte(20,1,10,8);
_delay_ms(1);
mehrereSchritte(20,1,10,16);
_delay_ms(1000);
mehrereSchritte(20,0,10,16);
_delay_ms(1);
mehrereSchritte(20,0,10,8);
_delay_ms(1);
mehrereSchritte(400,0,10,4);
```

Der Algorithmus der Marsschleife, erstellt von Ole Gumbiowski

In der Programmiersprache „C“ lassen sich ebenfalls Kommentare erstellen. Diese Funktion stellte sich für uns als ausgesprochen hilfreich heraus, um an verständnisschwierigen Stellen im Algorithmus einfach kurz die „ausgesprochene Bedeutung“ anzuknüpfen. Um einen Kommentar zu erstellen, reicht es aus, die Zeichenfolge „//“ (Im Text ohne Anführungsstriche) im Programmtext einzufügen. Danach wird alles in der Zeile geschriebene nicht mehr als Programmtext,, sondern als Kommentar gewertet und Grün eingefärbt.

7.3 Auslösung durch HTerm

Um die Marsschleife zu starten und die Nullstelle anzufahren, müssen am PC bestimmte Befehle eingegeben und verarbeitet werden. Dies geschhieht über das Programm „HTerm“. Das Programm wurde zur Befehlgebung für Mikrocontroller entwickelt und arbeitet ähnlich wie das im Planetarium bereits verwendete Programm „PC-DIMMER“. Sobald man die Werte des Mikrocontroller im Programm angepasst hast, werden PC und Mikrocontroller über die Funktion „Connect“ mittels USB-Kabel digital miteinander verbunden. Anschließend muss im Textfeld ein Befehl eingegeben werden, welcher vorher speziell im Programmtext des Mikrocontroller festgelegt wurde. In unserem Fall sind dies der Befehl „h“ für den Beginn der Marsschleife und „u“ zum Anfahren der Nullstellung.

8. Programmablauf

Präprozessoranweisungen

Der Präprozessor bearbeitet Anweisungen zum Einfügen von Quelltext (#include) und zum Ersetzen von Makros (#define). Er wird hier verwendet, um vorhandene Quelltext-Bibliotheken der Programmiersprache C für vordefinierte Standard-Funktionen und Pin-Belegungen des Mikrocontrollers nutzen zu können, sowie um per Makro-Definitionen das Programm anschaulicher und leichter lesbar zu gestalten.

Variablen und Funktionsprototypen

Die beiden Variablen „Data_in“ und „status“ werden global definiert, um sie im gesamten Programm zur Verfügung zu haben.
Mit Hilfe von Funktionsprototypen wird dem Compiler die Schnittstelle einer Funktion mitgeteilt, bevor diese verwendet wird. So kann der Compiler bei jedem Aufruf der Funktion überprüfen, ob die Funktionsparameter und der Rückgabewert sinnvoll verwendet werden. Die vollständige Umsetzung einer Funktion kann so zu einem späteren Zeitpunkt erfolgen. Hier werden Funktionsprototypen für die Funktionen in der Interrupt-Service-Routine verwendet: „void TX_Byte(unsigned char data)“ und „void Sende_CR_LF(void)“. „void“ kennzeichnet einen leeren oder nicht vorhandenen Übergabe- oder Rückgabewert.

Interrupt

Bei bestimmten Ereignissen im Mikrocontroller wird ein so genannter Interrupt ausgelöst. Interrupts machen es möglich, beim Eintreten eines Ereignisses sofort informiert zu werden, ohne permanent irgendeinen Status abzufragen, was Rechenzeit kosten würde. Dabei wird das Programm unterbrochen und ein Unterprogramm aufgerufen. Wenn dieses beendet ist, läuft das Hauptprogramm ganz normal weiter. In diesem Projekt wird der Empfangsinterrupt der seriellen Schnittstelle (USART) verwendet.

ISR(USART_RX_vect)

Dies ist die Service-Route, also Funktion, für den Empfangsinterrupt der Schnittstelle. Sie wird aufgerufen, sobald über die Schnittstelle ein neues Byte empfangen wurde. Dieses wird sodann als ASCII-Zeichen ausgewertet und die Variable „status“ entsprechend gesetzt sowie eine Bestätigung über die Schnittstelle gesendet:

Empfangenes Zeichen	Gesetzter Wert Variable „status“	Gesendete Bestätigung	Bedeutung
H	1	H	Sequenzstart
S	0	S	Status zurücksetzen (im Falle einer Fehlfunktion)
U	2	U	Rücklauf in Ausgangslage

Initialisierung

Zuerst muss der eingesetzte Mikrocontroller initialisiert, das heißt eingerichtet werden. Für die benutzten Port-Pins wird festgelegt, ob sie Ein- oder Ausgänge sind. In unserem Fall werden die Pins 0 bis 4 des Ports B als Ausgang deklariert, alle anderen bleiben Eingänge.

Im nächsten Schritt wird die serielle Schnittstelle (USART) eingerichtet. Die Baudrate wird auf 19200 gesetzt, Parity auf „even“, der Modus auf „asynchron“ sowie acht Datenbits und ein Stopbit eingestellt.
Der Empfangsinterrupt der seriellen Schnittstelle wird aktiviert und danach werden die Interrupts global aktiviert.

Funktionen

void init(void)

Diese Funktion dient der Zusammenfassung der oben erklärten Initialisierung.

void TX_Byte(unsigned char data)

Es wird erst gewartet, bis der Sendepuffer der USART leer ist, danach wird der Übergabewert „data“ in das Senderegister geschrieben. Die eigentliche Übertragung über die serielle Schnittstelle erledigt die USART-Einheit des Mikrocontrollers selbstständig.

void Sende_CR_LF(void)

Sendet ein “Carriage Return” und “Linefeed” über die serielle Schnittstelle, also einen Zeilenumbruch.

void mehrereSchritte(double schritte, double richtung, double pause, double schrittart)

Diese Funktion steuert den Schrittmotor an. Sie hat vier Übergabewerte:

Schritte	Anzahl der Schritte, die mit diesen Einstellungen erfolgen sollen
Richtung	Drehrichtung: 0 = rechts, 1 = links
Pause	Pause zwischen den Schritten in Millisekunden
schrittart	2 = Halbschritte, 4 = Viertelschritte, 8 = Achtelschritte, 16 = Sechzehntelschritte, alles andere bewirkt Vollschritte

Als erstes wird die Drehrichtung abgefragt und der Ausgang für die Steuerleitung DIR entsprechend gesetzt.
Danach erfolgt die Auswertung der Schrittart mit Ansteuerung der Signalleitungen MS1-3.

Jetzt wird in einer while-Schleife so oft das Signal STEP auf „high“ und danach wieder auf „low“ gesetzt, bis die gewählte Anzahl Motorschritte erreicht ist. Nach jedem Flankenwechsel wird eine Pause von zwei Mikrosekunden eingehalten, um ein stabiles und sicheres Signal zu erhalten. Nach jedem Einzelschritt erfolgt eine Pause gemäß Übergabewert.

Als letzte Maßnahme in der Funktion werden die Steuerleitungen MS1-3 wieder auf die Grundkonfiguration Vollschritt gesetzt, um einen definierten Zustand beim nächsten Aufruf der Funktion zu haben.

int main (void)

Dieses ist die Hauptfunktion des Programmes, in die der Mikrocontroller nach seinem Start gelangt.

Zuerst wird die Initialisierungsfunktion init() aufgerufen um die notwendigen Einstellungen vorzunehmen.
Das restliche Programm wird in einer endlosen while-Schleife ohne Abbruchbedingung aufgerufen, damit jederzeit eine neue Sequenz gestartet werden kann.

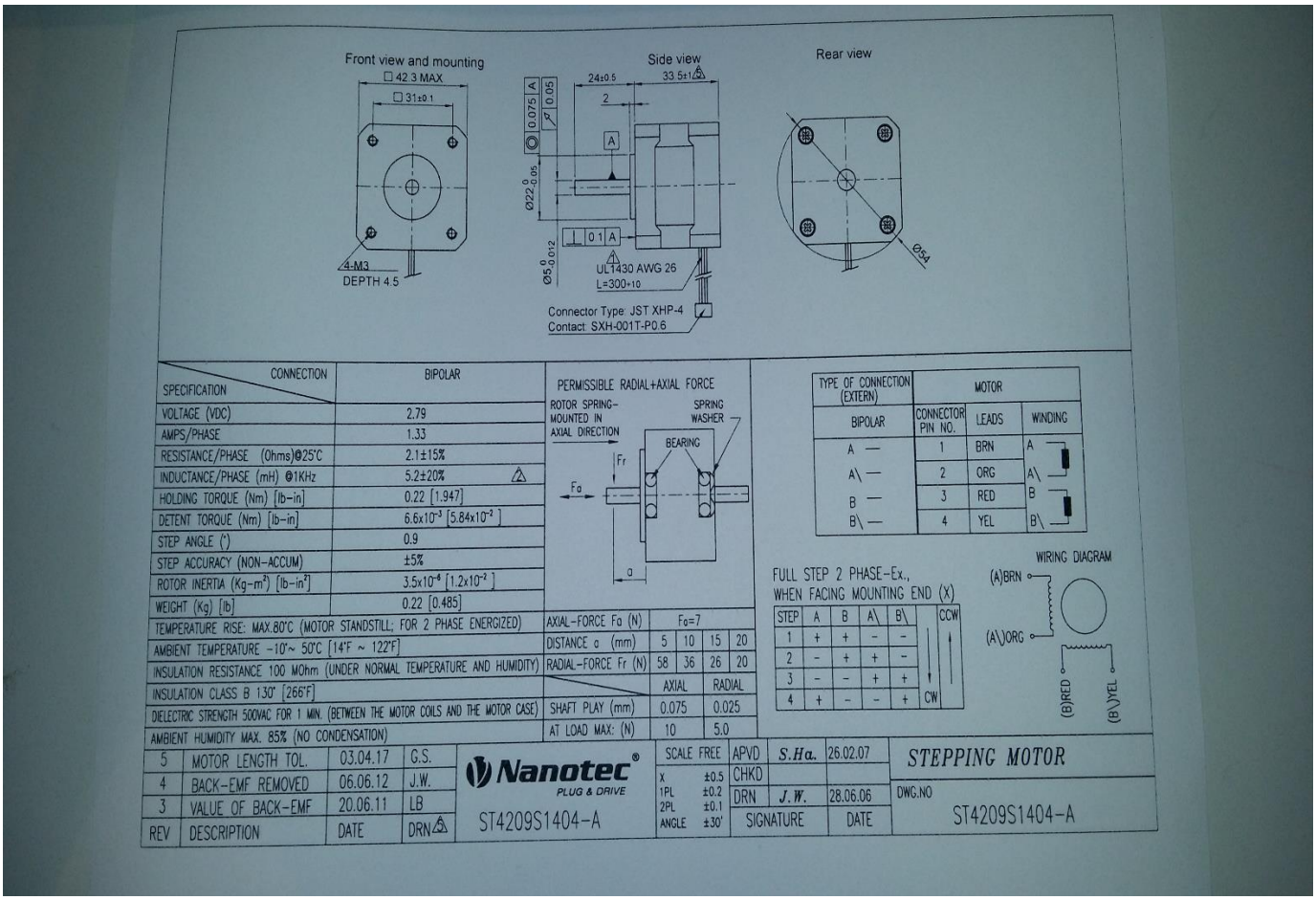
Jetzt wird in einer while-Schleife so lange gewartet, wie die Variable „status“ den Wert „0“ hat.

Ändert sich der Status, wird geprüft, welchen Wert dieser hat und ob der Endlagenschalter gedrückt ist:

Ist der Wert „1“ und der Endlagenschalter gedrückt, wird der Sequenz-Algorithmus des Schrittmotors gestartet und danach status wieder auf „0“ gesetzt; bei nicht gedrücktem Schalter wird der Wert der Variablen sofort wieder auf „0“ gesetzt.

Wenn der Wert der Variable „status“ „2“ ist, wird der Schrittmotor so lange in einzelnen Viertelschritten mit nachfolgender Pause von 20 Millisekunden in Richtung Ausgangslage gedreht, bis der Endlagenschalter wieder gedrückt ist.

Abschließend wird die Variable „status“ wieder auf den Wert „0“ gesetzt und das Programm springt an den Anfang der while-Schleife.



SPECIFICATION \ CONNECTION	BIPOLAR
VOLTAGE (VDC)	2.79
AMPS/PHASE	1.33
RESISTANCE/PHASE (Ohms)@25°C	2.1±15%
INDUCTANCE/PHASE (mH) @1KHz	5.2±20%
HOLDING TORQUE (Nm) [lb-in]	0.22 [1.947]
DETENT TORQUE (Nm) [lb-in]	6.6×10^{-3} [5.84×10^{-2}]
STEP ANGLE (°)	0.9
STEP ACCURACY (NON-ACCUM)	±5%
ROTOR INERTIA (Kg-m²) [lb-in²]	3.5×10^{-4} [1.2×10^{-2}]
WEIGHT (Kg) [lb]	0.22 [0.485]

TEMPERATURE RISE: MAX.80°C (MOTOR STANDSTILL; FOR 2 PHASE ENERGIZED)
AMBIENT TEMPERATURE -10°~ 50°C [14°F ~ 122°F]
INSULATION RESISTANCE 100 MOhm (UNDER NORMAL TEMPERATURE AND HUMIDITY)
INSULATION CLASS B 130° [266°F]
DIELECTRIC STRENGTH 500VAC FOR 1 MIN. (BETWEEN THE MOTOR COILS AND THE MOTOR CASE)
AMBIENT HUMIDITY MAX. 85% (NO CONDENSATION)

AXIAL-FORCE Fa (N)	Fa=7			
DISTANCE a (mm)	5	10	15	20
RADIAL-FORCE Fr (N)	58	36	26	20

	AXIAL	RADIAL
SHAFT PLAY (mm)	0.075	0.025
AT LOAD MAX: (N)	10	5.0

TYPE OF CONNECTION (EXTERN)	MOTOR		
BIPOLAR	CONNECTOR PIN NO.	LEADS	WINDING
A —	1	BRN	A
A\ —	2	ORG	A\
B —	3	RED	B
B\ —	4	YEL	B\

STEP	A	B	A\	B\
1	+	+	-	-
2	-	+	+	-
3	-	-	+	+
4	+	-	-	+

REV	DESCRIPTION	DATE	DRN
5	MOTOR LENGTH TOL.	03.04.17	G.S.
4	BACK-EMF REMOVED	06.06.12	J.W.
3	VALUE OF BACK-EMF	20.06.11	LB

SCALE FREE		APVD	S.Ha.	26.02.07
X	±0.5	CHKD		
1PL	±0.2	DRN	J.W.	28.06.06
2PL	±0.1	SIGNATURE		DATE
ANGLE	±30'			

9. Ansteuerung

Schnittstellenaufbau

Das Arduino-Board besitzt einen USB-Anschluss für den Anschluss an einen PC. Als Schnittstellenwandler wird auf dem Board ein Atmega16U2 von Atmel eingesetzt, der die USB-Daten so aufbereitet, dass der Hauptcontroller auf dem Board, der ATMega328p, mit seiner seriellen Schnittstelle diese verarbeiten kann. Dem PC gegenüber stellt dieser sich als virtuelle klassische serielle Schnittstelle (COM-Port) dar.
Die serielle Schnittstelle des Mikrocontrollers ist eine USART (Universal Serial Asynchronous Receiver and Transmitter), die flexibel konfiguriert werden kann.
Um eine hohe Kompatibilität zu erreichen, wurde sie in diesem Projekt gemäß des Standards RS-232 konfiguriert.

Schnittstellenparameter

Die Schnittstelle ist wie folgt konfiguriert:

Baudrate	19200
Parity	Even
Datenbits	8
Stoppbits	1

Diese Parameter, sowie der von Windows an die virtuelle Schnittstelle vergebene COM-Port, müssen in dem auf PC-Seite verwendeten Steuerprogramm eingestellt werden.

Befehle

Folgende Befehle kennt der Mikrocontroller:

H	Startet die Sequenz des Projektors
U	Fährt den Projektor zurück in die Ausgangslage
S	Kann, im Falle einer Fehlfunktion, den Status des Programmes zurück auf den Ausgangswert setzen

Für die Übertragung der Befehle kann beispielsweise ein Terminalprogramm genutzt werden. Wir benutzen dazu das Programm „HTerm“.

10. Ausblicke

Momentan lässt sich nur die Planetenbewegung in X-Richtung darstellen. Beobachtet man jedoch den Oppositionsvorgang am Nachthimmel so sieht man ebenfalls eine Auslenkung in Y-Richtung. Um dies zu integrieren wird die noch unbewegliche gedruckte Achse durch eine drehbare Achse ersetzt welche mittels eines zweiten Schrittmotors angetrieben wird. Da die Projektion möglichst originalgetreu ausfallen soll muss nun für jeden Schritt in X-Richtung eine Schrittweite in Y-Richtung festgelegt werden. Das Steuerprogramm für Motor 2 läuft simultan zum Steuerprogramm des ersten Motors. Ist diese Hürde überwunden lässt sich das Design auf einen zweiten Projektor umsetzten. Die Kombination der beiden Projektionen kann genutzt werden um die sich überlappenden Planetenschleifen darzustellen welche zum „Stern von Bethlehem“ führten.

11. Danksagung

Wir möchten uns im Rahmen dieser Projektarbeit zunächst bei Herrn Dennis Jechow bedanken. Ohne ihn wäre die Umsetzung des Projektors für uns sehr schwierig gewesen, da er uns gerade im Bereich Hardware und Programmieren geholfen hat. Desweiteren möchten wir Britta Thielen-Gumbiowski und Sandra Bertram für ihre logistische Unterstützung und weiteren Mühen danken. Ebenso danken wir Herrn Michael Winkhaus für die Zugangsmöglichkeiten zum Schulplanetarium und für die Kostenabwicklung. Der Firma August Welz danken wir für die Bereitstellung diverser Werkzeuge und technische Ratschläge. Herrn Bernd Koch danken wir für die Ermöglichung und Betreuung des Projektkurses sowie die Unterstützung bei der Bildbearbeitung.

12. Quellenverzeichnis

https://www.astronomie.de/das-sonnensystem/planeten-und-monde/der-mars/grunddaten/

https://astrokramkiste.de/geozentrisches-weltbild/30-astronomiegeschichte/weltbilder/geozentrische-weltbilder

https://eckstein-shop.de/Uno-R3-ATmega328P-Board-ATmega16U2-mit-USB-Kabel-Arduino-Uno-R3-kompatibel

https://www.heise.de/download/product/hterm-53283

http://downloadsf.com/5521549999de/ddl/103790/iivnMqaxnowrYFuCGkQEcpCAxUkIwLwGpwsOSkbiscivVjHQbm?b_adgroupid=1156686305806290&b_adid=72292904787702&b_bidmatchtype=be&b_campaignid=265607600&b_device=c&b_match=e&b_orderitemid=72292963733459&utm_campaign=265607600_1156686305806290&utm_medium=cpc&utm_source=bingads&utm_term=notepad+++

https://de.wikipedia.org/wiki/Schrittmotor

http://product.item24.de/produkte/produktkatalog/produktdetails/products/konstruktionsprofile-6-1001042790/profil-6-30x30-leicht-natur-41906/

https://de.nanotec.com/produkte/451-st4209-schrittmotor-09-nema-17/

https://www.mikrocontroller.net/

Das Eagle PCB-Design, Autor: Herbert Bernstein (2004)

AVR-Mikrocontroller Praxis, Autor: Safinaz Volpe (2002)

Softwareentwicklung für Mikroprozessoren, Autor: Jörg Wiegelmann (2009)

Programmieren des AVR RISP II Mikrocontroller mit BASCOM AVR, Autor: Claus Kühnel (2004)

Programmieren in C, Autor: Robert Klima (2007)

C als erste Programmiersprache, Autor: Manfred Dausmann (2008)

13. Abschlusserklärung

Hiermit versicheren wir, dass wir diese Arbeit selbstständig angefertigt, keine anderen Quellen als die genannten Quellen und Hilfsmittel benutzt und die Stellender Facharbeit, die dem Wortlaut oder dem Inhalt nach aus anderen Werken entnommen wurden, in jedem einzelnen Fall mitgenauer Quellenangabe gekennzeichnet haben.

Wir sind damit einverstanden, dass die von uns verfasste Projektarbeit der schulinternen Öffentlichkeit in der Bibliothek der Schule zugänglich gemacht wird.

Ort, Datum

Unterschrift (Torben Bertram)

Ort, Datum

Unterschrift (Ole Gumbiowski)