

Studienarbeit
Led-Globe

im Studiengang Technische Informatik
der Fakultät Informationstechnik
Sommersemester14

Clemens Weißenberg & Tobias Finke

Datum: 15. Dezember 2014
Prüfer: Prof. Dr.-Ing. Dipl.-Ing. Reiner Marchthaler

Inhaltsverzeichnis

1	Einleitung	1
1.1	Projektübersicht	1
1.2	Verwendete Tools	2
2	Elektronik	5
2.1	Schaltung	5
2.2	Spannungsversorgung	5
2.3	Atmel ATMega32P Mikrocontroller	7
2.4	Bluetoothmodul BTM222	9
2.5	SPI-PWM-Controller WS2803	9
2.6	SMD RGB-Leds	11
2.7	Reed-Kontakt für die Bildsynchronisation	12
3	Software-Architektur	14
3.1	Anforderungen	14
3.2	Simulation	15
3.3	Host-spezifisch	15
3.4	Target-spezifisch	16
4	Algorithmen	18
4.1	Schrift	18
4.2	Mode-Select-Menu	19
4.3	Pong	20
4.4	Snake	22
4.5	Flappy Bird	23
4.6	Pacman	25
4.7	Statische Bilder	29
4.8	Connway's Game of Life	30
4.9	Manuelle Farbwahl	31
4.10	Color-Fading	31
5	Mechanik	33
5.1	Aufbau	33
5.2	Motor	34
5.3	Schleifikontakte	34
5.4	Alte Revisions	35
6	Bedienungsanleitung	37
6.1	Herstellen der Bluetooth-Verbindung	37

6.2 Steuerung des Globe	37
6.3 Erweitern des Globe um eigene Modi	37
7 Fazit	40

Abbildungsverzeichnis

1.1	Led Globe	1
2.1	Trägerplatine für die WS2803	5
2.2	Scahlung des Globe	6
2.3	Blockdiagramm des WS2803	7
2.4	Blockdiagramm des WS2803	10
2.5	Blockdiagramm des WS2803	10
2.6	Pinbelegung des WS2803	11
2.7	Kommunikationsprotokoll des PWM-Controller WS2803	12
2.8	Verkabelung der Leds	13
3.1	Simulation des Display	16
4.1	ASCII-Tabelle	18
4.2	Darstellung der ASCII-Tabelle auf dem Globe	19
4.3	Darstellung des Menü auf dem Globe	20
4.4	Darstellung von Pong auf dem Globe	21
4.5	Darstellung von Snake auf dem Globe	22
4.6	Darstellung von Flappy Bird auf dem Globe	24
4.7	Konzept für die Darstellung des Vogels auf dem Globe	25
4.8	Konzept des Pacman Spielfelds	26
4.9	Darstellung der Geister und Pacman	27
4.10	Flussdiagramm der Navigationslogik der Geister	28
4.11	Darstellung einer Weltkarte auf dem Globe	29
4.12	Darstellung von Game of Life auf dem Globe	30
4.13	Manuelle Auswahl der Fahrben	31
4.14	Darstellung von Color-Fading auf dem Globe	32
5.1	Aufbau des Globe	33
5.2	Aufbau des Globe(Ansicht von oben)	34
5.3	Motor	34
5.4	Schleifkontakte	35
5.5	Probleme bei den WS2811 LEDs	35

Tabellenverzeichnis

Listings

1 Einleitung



Abb. 1.1: LED Globe

1.1 Projektübersicht

1.1.1 POV: Persistence-of-Vision

Das POV-Prinzip baut darauf auf, dass das menschliche Auge ein Objekt nur bis zu einer bestimmten Geschwindigkeit wahrnehmen kann. Beim Film wird mit einer Bildanzahl von 25 Bildern pro Sekunde gearbeitet um einen flüssigen Filmeindruck beim Zuschauer zu hinterlassen. Diesen Effekt kann man auch in anderen Anwendungen in leicht abgewandelter Form zu Nutze machen.

1.1.2 Prinzip des LED-Globe

Der Globe besteht aus einem rotierenden Streifen aus LEDs, der in einem Halbkreis vertikal um eine Drehachse gelagert ist. Durch Rotation des LED-Streifens entsteht für den Betrachter eine Kugel. Die LEDs werden durch einen Mikrocontroller angesteuert. Dadurch ist es möglich verschiedene Farben darzustellen, komplexe Muster anzuzeigen, oder den Globe als Spielfeld für rudimentäre Spiele zu benutzen. Der Mikrocontroller teilt die darzustellende Grafik in maximal 80 Spalten auf. Diese werden nun nacheinander auf dem LED-Streifen, mit einer kurzen Pause dazwischen, ausgegeben. So entsteht der Bildeindruck einer kompletten Kugel, die aus vielen Streifen besteht. Da jede LED im Steifen ebenfalls einzeln ansteuerbar ist, ergibt sich eine Darstellungsfläche, die mit einem normalen Display vergleichbar ist.

1.2 Verwendete Tools

Für die Entwicklung der Software und der Dokumentation wurden verschiedene Tools genutzt. Zur Anwendung kamen Entwicklungsumgebungen, Versionsverwaltungen und Programmdokumentationen. Eine Herausforderung war das Betriebssystem. Es wurde sowohl auf Mac, als auch auf Windows entwickelt und deshalb mussten Programme gefunden werden, die auf beiden Systemen kompatibel sind oder zumindest mit den gleichen Quelldaten arbeiten können.

1.2.1 IDEs

Als Entwicklungsumgebung wurde sowohl Eclipse auf Mac und Visual Studio auf Windows genutzt. Eclipse ist zwar ein universell einsetzbares Programm und mit Windows kompatibel, bereitete jedoch Probleme bei der Verwendung der Compiler. Daher wurde zum Entwickeln unter Windows das Microsoft eigene Programm Visual Studio verwendet.

Eclipse

Das mit der von Sun entwickelten IDE Eclipse und dem dazugehörigen Plugin CDD für die Unterstützung der Programmiersprache C wurde auf OSX entwickelt. Dies hatte den Vorteil, dass die AVR-GCC Toolchain leicht integriert und der Mikrocontroller direkt aus der IDE geflasht werden konnte.

MS Visual Studio

Visual Studio wird bei der Entwicklung auf Windows benutzt. Bei Windows bereitet die Compiler-Toolchain beim Einbinden Probleme, weshalb hier der leicht einzurichtende Windowscompiler benutzt wird.

1.2.2 Versionsverwaltung

Ein solch umfangreiches Projekt erfordert eine strukturierte Versionsverwaltung auf die schon am Anfang des Projektes gelegt wurde.

Git

Als Versionverwaltung wurde Git gewählt. Es ist eine OpenSource-Anwendung die in der Community weit verbreitet ist. Deren Nutzung kommt Studenten mit langem Anfahrtswegen mit den öffentlichen Nahverkehr zu gute, da offline entwickelt werden kann. Die Verwendung von Branches erlaubt, Funktionen gesondert zum Projekt zu entwickeln und diese später mit dem Hauptzweig zu “mergen“, also ins Projekt einzupflegen.

Das Git Repository wurde auf einem eigenen Rootserver angelegt, welches unter folgender Adresse mittels des PHP-Tools Gitlist eine Webansicht bereitstellt. <http://net-zwerg.org/gitlist/led-globe.git/tree/master/led-globe/>

SourceTree

Um das Versionsverwaltungstool Git nicht über die Kommandozeile bedienen zu müssen, kommt das plattform-unabhängige Tool SourceTree zum Einsatz. Damit ist es möglich über eine GUI, sehr bequem Branches zu erstellen, Diffs anzuzeigen und Programmcode in die Versionsverwaltung zu commiten.

1.2.3 Schaltungslayout

Eagle

Eagle ist eines der gängigen Tools bei der Entwicklung von Schaltungen. Es zeichnet sich durch seine große Bibliothek an Bauteilen aus, die einfach in das Schaltungslayout eingefügt werden können. Die Schaltung wurde im Projekt über Eagle entworfen. Die beiliegenden Eagle-Datei bietet die Möglichkeit, später eine Platine zu ätzen und die etwas klobigen Prototypen zu ersetzen. Dafür wurde Eagle in der Version 6.1.0 für Mac OSX verwendet.

1.2.4 Dokumentation

LateX

Um die Dokumentation zu erstellen wird LateX verwendet. LateX ist ein Programm, dass aus simplen binären Textdateien, die mit jedem üblichen Texteditor bearbeitet werden können, grafisch hochwertige PDF-Dateien generiert. Dies hat in Verbindung mit dem Versionierungstool den großen Vorteil, die Dokumentation in mehrere Bereiche aufzuteilen und parallel daran arbeiten zu können. Änderungen im GIT sind durch den Commit-Verlauf sehr gut ersichtlich.

Doxxygen

Um den Source-Code zu dokumentieren wird Doxygen genutzt. Dazu werden im Quellcode mit einer bestimmten Syntax Kommentare eingefügt, die unter Anderem Funktionen und Parameter beschreiben oder eine Übersicht über Module geben. Danach kann aus dem dokumentierten Quellcode eine übersichtliche Darstellung generiert werden, die mit den gängigen Webbrowsersn betrachtet werden kann. Um die übliche Darstellungsform aufzuwerten wurde das Design mit Hilfe des Web-Frameworks Bootstrap angepasst. Damit entspricht es dem Auftritt der Hochschule Esslingen.

1.2.5 Messinstrumente

Oszilloskop

Bei der Entwicklung der Hardware war vor allem das Oszilloskop unerlässlich. Damit konnten während des Projektes mehrere Probleme aufgespürt und behoben werden. Dies galt vor allem dem Timing, welches durch die extremen **Ressourcenknappheit (Welche Ressourcenknappheit?! Meinst du der geringen Performance?)** sehr eng gestaltet war.

Beim Oszilloskop handelt es sich um ein bla bla mit folgenden Daten

8-Kanal Logic-Analyzer

Der 8-Kanal Logic-Analyzer war wie das Oszilloskop notwendig. Hier können Daten die der Mikrocontroller an die LED-Treiberbausteine sendet, verifiziert werden. Auch bei der Entwicklung der Kommunikation über Bluetooth war der Logic-Analyzer sehr hilfreich.

Die Software Logic ist sehr umfangreich, plattformunabhängig und einfach zu bedienen. Die benötigte Hardware ist verhältnismäßig günstig und könnte in Laboren der Hochschule Anwendung finden.

Daten der Hardware: **Bilder und daten einfügen**

2 Elektronik

2.1 Schaltung

Der Entwurf der Schaltung (siehe Abb. 2.2) wurde in Eagle realisiert. Um schnell einen verwendbaren Prototypen herstellen zu können, wurde das Projekt auf Lochrasterplatten realisiert. Ein weiterer wichtiger Punkt bei der Erstellung des Layouts für die Platinen war der Massenschwerpunkt. Für ein möglichst ruhiges Drehen des Globes ohne Umwucht ist es nötig, alle Komponenten innerhalb des Globe möglichst symmetrisch anzurichten. Deshalb wurde für das Unterbringen der benötigten acht WS2803-Chips ein Design gewählt, bei dem auf beiden Seiten der Achse jeweils vier Chips angebracht sind. Das hat zur Folge, dass sich der Schwerpunkt der Chips auf der Achse befindet. Der grün-grau verdrillte Draht in Abb. 2.1 leitet das Signal von der oberen zur unteren Platine weiter, da die WS2803 in einer Reihe angeordnet sind und wie Schieberegister funktionieren.

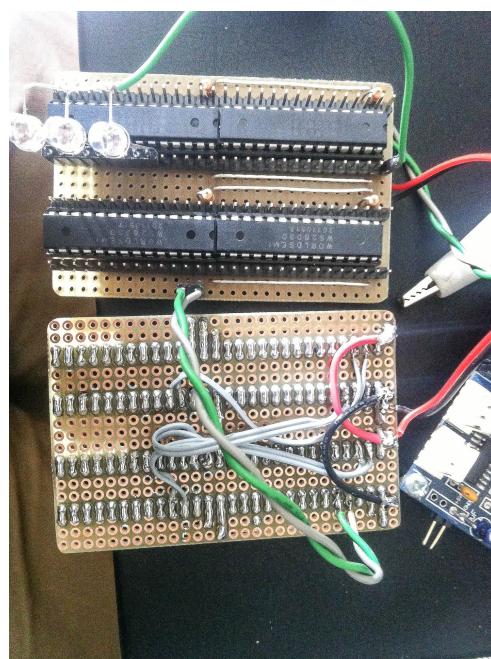
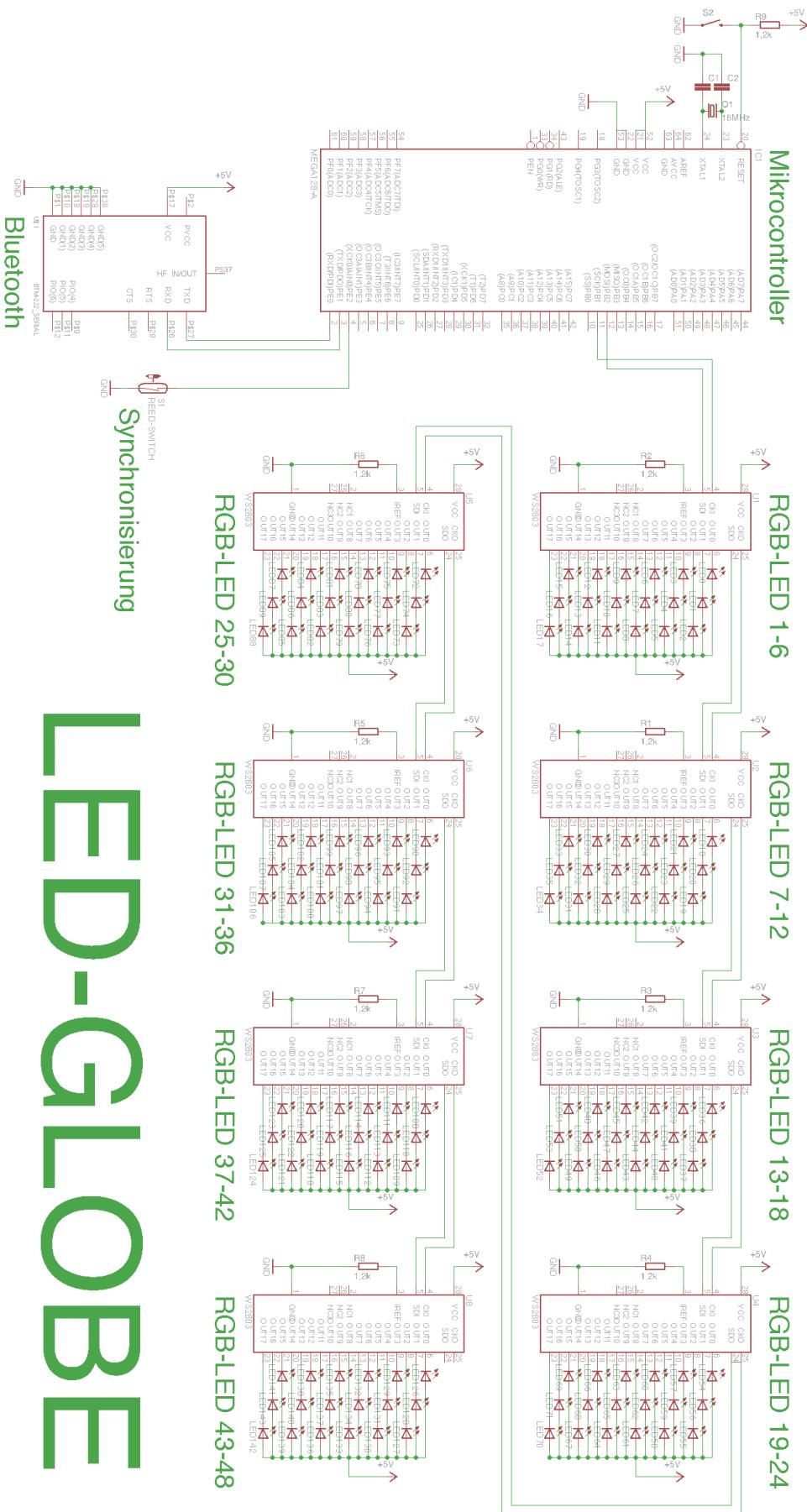


Abb. 2.1: Trägerplatine für die WS2803

2.2 Spannungsversorgung

Die Spannungsversorgung stellt aufgrund der Konstruktion eine besondere Herausforderung dar.



Zuerst wurde versucht die Spannungsversorgung innerhalb des rotierenden Teils durch 9V-Batterieblöcke sicherzustellen. Diese Möglichkeit wurde aber schnell verworfen, da die Lebenszeit der Batterien viel zu kurz war und es nach kurzer Zeit zu Verbindungsabbrüchen beim Bluetooth-Modul kam. Die durch die schweren Batterien erzeugte Umwucht, konnte ebenfalls nicht zufriedenstellend beseitigt werden.

Als Alternativen gibt es bei solchen rotierenden Konstruktionen entweder die Möglichkeit per Induktionsstrom zu übertragen oder Schleifkontakte anzubringen, die entlang der Achse die Spannungsversorgung sicherstellen. Aufgrund der technischen Möglichkeiten fiel die Wahl auf einen Schleifkontakte, der durch eine Kugelschreiberfeder realisiert wurde. Diese Feder schleift an der aus einer Gewindestange bestehenden Achse des Globe entlang.

Eine Spannung von 24V wird von einem 2A-Labornetzteil bereitgestellt und über den Schleifkontakt in das Innere des Globe geleitet. Ein Step-Down-Spannungswandler stabilisiert die Spannung auf konstante 5V. Zuerst wurde ein Linearregler verwendet. Da sich der Regler stark erhitzte, was auf eine zu hohe Leistung schließen lässt, wurde er vorsichtshalber ausgewechselt. Nachträglich wird die Spannung mit einem Goldcap-Kondensator noch weiter geglättet um Spannungsabbrüche abzufangen, die durch den etwas instabilen Schleifkontakt sporadisch entstehen. Durch die große Kapazität des Goldcaps ist es möglich, die Spannungsversorgung bei einem Spannungsabbruch des Schleifkontakte für etwa eine Sekunde aufrecht zu erhalten.

2.3 Atmel ATMega32P Mikrocontroller

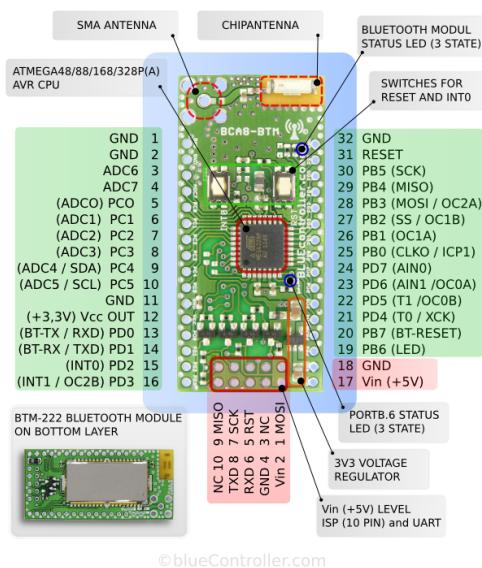


Abb. 2.3: Blockdiagramm des WS2803

Als Mikrocontroller wird der Atmel ATMega32P verwendet. Er bietet sich durch eine leichte Bedingung und einen geringen Preis für Einsteiger-Projekte, wie es beim Globe der Fall ist, sehr

gut an. Außerdem baut das bekannte Konzept mit dem Namen Arduino auf den Mikrocontrollern der ATMega-Reihe von Atmel auf. Zudem gibt es hier viel bestehendes Know-How das durch Projekte mit diesem Controller-Typ in der Vergangenheit vorhanden ist.

2.3.1 Technische Daten

- 8-Bit Mikrocontroller
- Advanced RISC-Architektur
- 20MIPS bei 20MHz Taktfrequenz
- 32 kB Flash
- 1 kB EEPROM
- 2 kB SRAM
- Timer: (2x 8-Bit, 1x 16-Bit, RTC)
- 6x PWM
- 8x ADC
- Serielle Schnittstelle (UART)
- SPI-Interface
- 1,8 - 5,5 V Versorgungsspannung
- 23 GPIOs

2.3.2 Verwendete Peripherie

Der Mikrocontroller zeichnet sich durch eine Fülle von verschiedenensten Peripherie-Elementen aus, die im Projekt Verwendung finden.

Externe Interrupts

Interrupts sind Funktionen die bei auftreten eines Ereignisses ausgeführt werden. Dabei unterbrechen sie den aktuellen Programmablauf und setzen diesen nach der Ausführung der jeweiligen Funktion fort. Ein externer Interrupt wird dann ausgelöst, wenn sich an einem Pin des Mikrocontrollers der Zustand ändert oder eine bestimmte Flanke auftritt. Bei dem Globe wird durch den Reed-Kontakt ein Interrupt generiert, wenn dieser in die Nähe des Neodym-Magneten kommt. Dieser Interrupt wird dafür genutzt, um das dargestellte Bild immer an der gleichen Position zu halten und den Mikrocontroller zu informieren, dass er nun wieder die erste Spalte anzeigen soll.

SPI-Interface

Das SPI-Interface wird für die Kommunikation mit den SPI-PWM-Controllern der LEDs verwendet. Dabei werden die benötigten Farbwerte bitweise über die MOSI-Leitung des Mikrocontrollers übertragen. Die SCK-Leitung ist hierbei für den Takt verantwortlich. Für die Kommunikation sind lediglich 2 Leitungen nötig, was die Verkabelung zusätzlich vereinfacht. Die Funktionsweise der Kommunikation wird im Kapitel 2.5 genauer beschrieben.

UART-Interface

Das UART-Interface ist ein zeichenbasiertes Protokoll, welches für die Kommunikation mit dem Bluetooth-Modul verwendet wird. Es bietet die Möglichkeit vom PC oder Handy Kommandos zu empfangen, um beispielsweise Farben zu ändern, im grafischen Menü zu Navigieren oder Spiele darauf spielen zu können.

ADC

Der ADC findet in diesem Projekt bei der Erstellung von Zufallszahlen Anwendung. Beim Erzeugen von Zufallszahlen verwendet man eine Funktion aus der Standard-C-Bibliothek. Dies setzt allerdings voraus, dass man die random-Funktion mit einem sich immer ändernden Wert initialisiert. Üblicherweise genügt die aktuelle Uhrzeit, was bei einem Mikrocontroller nicht ohne weiteres möglich ist. Deshalb wird hier der ADC genutzt, der einen nicht angeschlossenen ADC-Pin des Mikrocontrollers ausliest. Der dabei gemessene Wert entspricht einem zufälligen Wert, der dann als Seed, also als Initialisierungswert verwendet wird und somit für hinreichend zufällige Werte sorgt, die sich von Start zu Start unterscheiden.

2.4 Bluetoothmodul BTM222

Das Bluetooth-Modul stellt das Bindeglied zwischen dem Mikrocontroller und dem PC dar. Auf dem PC kann sehr einfach eine Verbindung über das SPP hergestellt werden. Nach Eingabe des VerbindungsCodes (Standard ist 1234) wird auf dem PC bei den Seriellen Schnittstellen ein neues Gerät angezeigt, über das dann Zeichen gesendet werden können. Die Zeichen werden über Bluetooth an das BTM222 gesendet und dort per UART an die serielle Schnittstelle des Mikrocontrollers weitergeleitet und vice versa.

2.4.1 Technische Daten

2.5 SPI-PWM-Controller WS2803

Für den Globe werden mindestens 48 RGB-LEDs benötigt. Da jede Farbe einer RGB-LED einen eigenen PWM-Channel benötigt ergibt das in Summe mindestens 144 PWM-Channel. Neben der Tatsache, dass der Mikrocontroller allein nicht genügend freie Pins zur Verfügung hat, sind auch

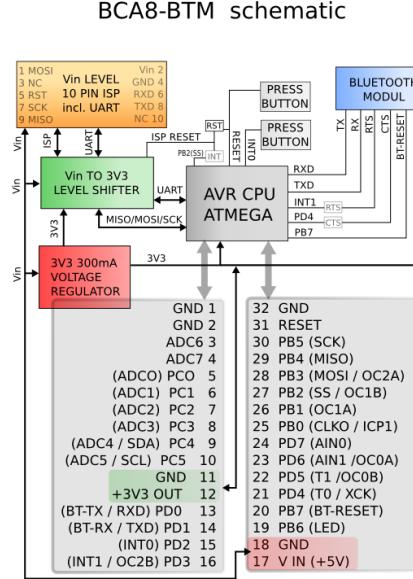


Abb. 2.4: Blockdiagramm des WS2803

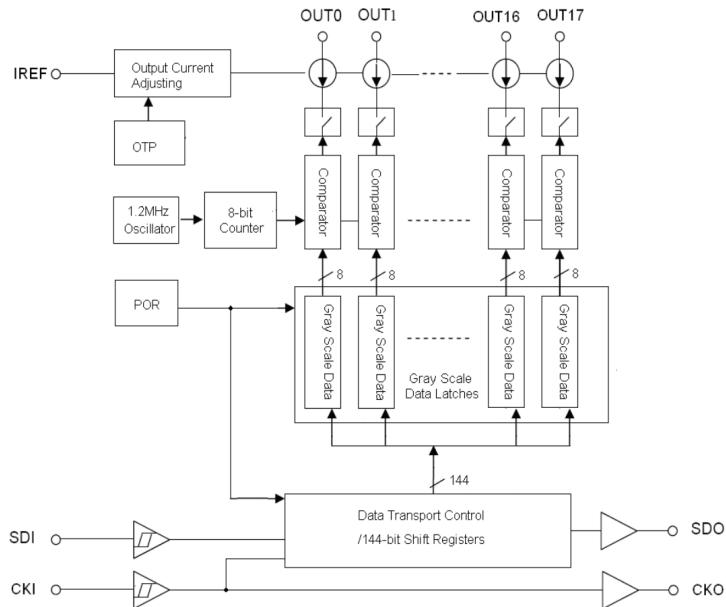


Abb. 2.5: Blockdiagramm des WS2803

nur eine begrenzte Anzahl an PWM-Channel als Hardware-Peripherie verfügbar. Die Realisierung einer Software-PWM mittels effizienter Algorithmen wie Bit-Angle-Manipulation ist nicht möglich, da die erforderliche PWM-Frequenz viel zu langsam ist. Deshalb wird der WS2803-Chip verwendet. Es handelt sich um einen PWM-Controller der über ein serielles Protokoll gesteuert wird. Der Controller zeichnet sich vor allem durch seine schnelle PWM-Frequenz aus, die für das Projekt benötigt wird. Höhere Geschwindigkeiten wären in diesem Fall sogar noch besser.

gewesen.

2.5.1 Technische Daten

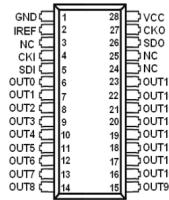


Abb. 2.6: Pinbelegung des WS2803

- 18 Konstantstrom-Quellen Treiber
- 8-Bit PWM Graulevel-Steuerung
- Ausgangsspannung max. 30V
- max. Taktfrequenz 25MHz
- Freilauf-Funktion
- Überhitzungsschutz
- PWM-Frequenz mit 4,7 kHz
- Treiber als Konstantstrom- oder Konstantspannungs-Quelle nutzbar

2.5.2 Ansteuerung

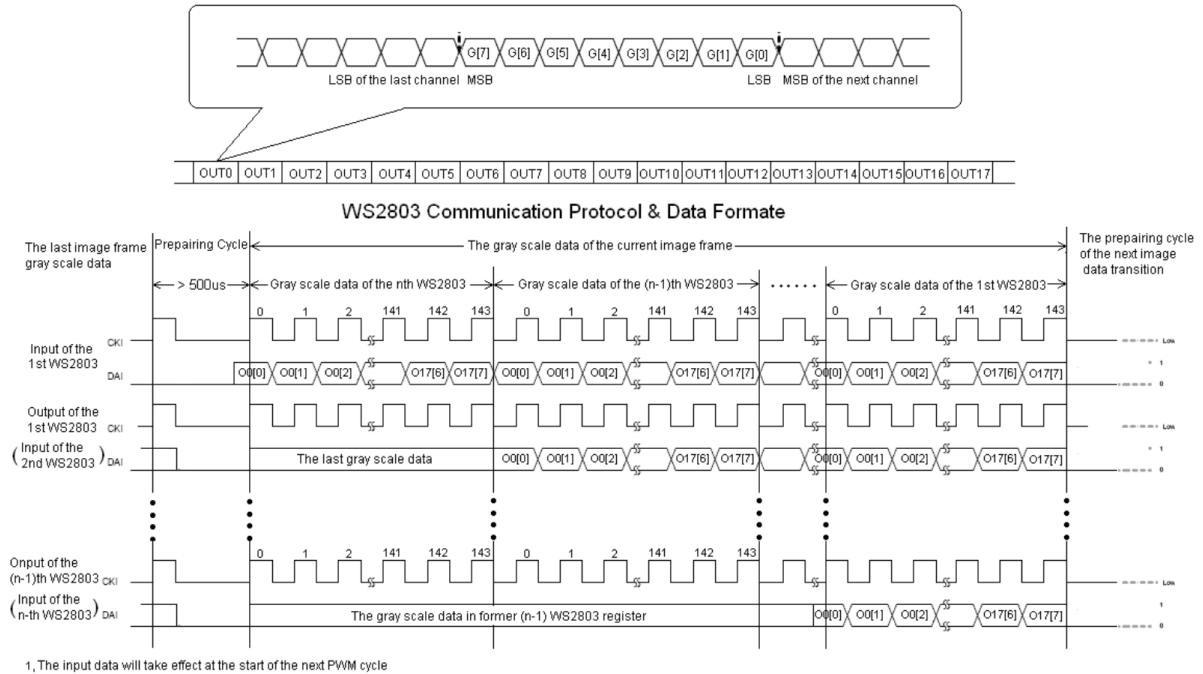
Die Ansteuerung des WS2803 erfolgt über das in Abb. ?? dargestellte Schema. Der Mikrocontroller steuert nur den ersten der 8 WS2803 an, da das Signal alle weiteren über eine Reihenschaltung erreicht.

2.5.3 Probleme

Die Geschwindigkeit der PWM ist der begrenzende Faktor für das Auflösungsvermögen des Globus. Wäre der PWM-Zyklus des WS2803 schneller und könnte man die Daten noch schneller in den Chip spielen, könnten noch mehr Spalten angezeigt werden. Zu berücksichtigen ist, dass die Verwendung des WS2803 im Globe einen Extremfall darstellt, da sich hier die LEDs in X-Richtung bewegen. Somit ist der PWM-Zyklus deutlich sichtbar, was bei einer fix angesteuerten LED nicht auffallen würde.

2.6 SMD RGB-LEDs

Als Leuchtmittel werden RGB-LEDs in der SMD-Bauweise eingesetzt.



2.6.1 Technische Daten

2.6.2 Verkabelung

Bei der Verkabelung wurde Lötlack-Draht mit einem Durchmesser von 0,2mm verwendet. Dadurch war es möglich mit minimalem Gewicht die sehr aufwendige Verkabelung jeder einzelnen LED durchzuführen. Jede der 48 SMD-LEDs musste mit drei Kabeln mit dem WS2803 verbunden werden. Zudem wurden alle Anoden der LED zusammengeschlossen und über einen stabilen Silberdraht mit 5V Versorgungsspannung verbunden. Über den feinen Lötlack-Draht kann somit der WS2803 die jeweilige LED beliebig ansteuern. Die LEDs sind in einem Dreierbündel zusammengefasst (siehe Abb. 2.8), die direkt über eine Stiftleiste mit der Trägerplatine der WS2803-Chips verbunden sind.

2.7 Reed-Kontakt für die Bildsynchroisation

Die Synchronisation des Bildsignals erfolgt mit Hilfe eines Reed-Kontaktes. Ein Reed-Kontakt ist ein Glaskröpfchen, in dem sich zwei kleine Drähte befinden, die nach außen geführt sind. Kommt nun der Reed-Kontakt in die Nähe eines starken Magnetfeldes berühren sich die beiden dünnen Drähte und die Verbindung wird leitend. Das Magnetfeld wird in diesem Fall mit einem Neodym-Magneten erzeugt, der auf der Innenseite des Stahlrings oben angebracht ist. Der Reed-Kontakt ist entsprechend an der rotierenden Fresnel-Scheibe angebracht. So kommt der Reed-Kontakt jeweils einmal pro Umdrehung in die Nähe des Magneten und löst beim Mikrocontroller einen Interrupt aus, indem er den Interrupt-Pin des Mikrocontrollers auf Masse zieht. Problematisch



Abb. 2.8: Verkabelung der LEDs

bei der Verwendung des Reed-Kontaktes ist das Prellen, welches stark ausfällt und von der Software im Mikrocontroller entfernt wird.

Als Alternative wäre hier ein Hall-Sensor denkbar, der schneller schaltet und bei dem das Prellen seltener auftritt als beim Reed-Kontakt.

3 Software-Architektur

3.1 Anforderungen

Bei diesem Projekt gibt es hohe Anforderungen an das System und dessen Architektur. Durch die wenig potente Hardware müssen viele Feinheiten beachtet werden, die sämtliche Teilbereiche des Projektes betreffen.

3.1.1 Speicher

Der Speicher ist eines der Elemente die sich bei diesem Projekt am stärksten auf die Software-Architektur auswirkt. Es ist unmöglich ein komplettes Bild im RAM zu halten und darauf Operationen und Algorithmen anzuwenden, da der Speicherbedarf bei ca. 10kB liegen würde und der Mikrocontroller lediglich 2kB RAM besitzt. Mehr Platz bietet hingegen der ROM.

$$\text{Speicherbedarf} = \text{PWM Auflösung} * 3 * \text{AnzahlLEDsProReihe} * \text{AnzahlReihen}$$

$$\text{Speicherbedarf} = 8\text{Bit} * 3 * 48 * 70$$

$$\text{Speicherbedarf} = 10080\text{Byte}$$

$$\text{Speicherbedarf} = 10\text{kByte}$$

Fragen die beim Optimieren aufgetreten sind und berücksichtigt wurden:

- brauchen wir diese Variable wirklich?
- können einzelne Flags in einer 8-Bit Variable zusammengefasst werden?

3.1.2 Performance

Die Zeit ist ein anderer Aspekt . Da der Mikrocontroller mit 16Mhz Taktfrequenz und 16 MIPS beschränkt leistungsfähig ist, ist auch hier auf die Optimierung sämtlicher Softwarekomponenten zu achten.

Fragen die beim Optimieren aufgetreten sind und berücksichtigt wurden:

- ist der Kopiervorgang wirklich notwendig?
- kann der Kopiervorgänge durch Umbiegen von Pointern ersetzt werden?
- ist ein komplexer Algorithmus durch eine Lookup-Table ersetzbar?
- kann ein Algorithmus auch mit 8-Bit-Arithmetik auskommen?

3.1.3 Übersichtlichkeit, Erweiterbarkeit und Modularisierung

Bei der Entwicklung des Quellcodes, ist es den Autoren ein Anliegen möglichst simpel und verständlich zu programmieren. Die Intention dahinter ist es zum einen Fehler zu vermeiden und zum anderen die Software erweiterbar zu machen. Daher wird sehr viel Energie in die Definition einer einfach zu bedienenden API verwendet. Im Vordergrund steht dabei die Namensgebung, die Kommentare und die Modularisierung.

Fragen die beim Optimieren aufgetreten sind und berücksichtigt wurden:

- kann dieser Code ab gekapselt werden?
- sind Schnittstellen möglichst klein gehalten?
- hat diese Variable einen selbsterklärenden Namen?
- gibt es verschiedene Zustände, die mit einem Zustandsautomaten vereinfacht werden können (Enums)?
- sind ausreichend Kommentare vorhanden?

3.2 Simulation

Es war schon bei Beginn des Projektes klar, zusätzlich zur Hardware, eine Globe-Simulation zum Testen der Algorithmen zu erstellen. Gerade in der Anfangsphase schleichen sich viele Fehler ein. Durch die Mechanik und den Umstand, dass der Mikrocontroller mit ca. dreizehn Umdrehungen pro Sekunde dreht, kann nicht über ein Debug-Interface auf dem Target gedebuggt werden. Nur eine Konsolenausgabe über Bluetooth ist möglich, was bei der Komplexität des Projektes nicht ausreichend ist. Um eine Simulation realisieren zu können, musste der Software-Architektur eine Hardware-Abstraktionsschicht (HAL) beigelegt werden. Durch die HAL ist es nun möglich, die Anwendung autark von der darunter liegenden Schicht laufen zu lassen. Alle hardware-abhängigen Teile sind komplett losgelöst von der Logik des Globe. Im Folgenden werden die beiden HALs für die Simulation auf dem PC und für die reale Anwendung auf dem Mikrocontroller näher beschrieben.

3.3 Host-spezifisch

3.3.1 Display

Für die Simulation auf dem Host wurde als Display die Konsolenausgabe benutzt. Natürlich wäre es möglich eine ansprechende GUI mit OpenGL-Animationen zu realisieren, was aber den Rahmen dieses Projektes deutlich gesprengt hätte. Für die Simulation auf dem Host wurde deshalb als Display die Konsolenausgabe benutzt. Dabei wurden die drei Dimensionen der Kugel auf zwei reduziert. Einzelne Ziffern in einem rechteckigen Block bilden die Zeilen und Spalten der LED-Anzeige ab.

Abb. 3.1: Simulation von Spiel Pong auf dem Display

Falls ein Pixel rote Farbanteile enthält, wird an der entsprechenden Stelle ein *R* eingefügt, bei Blau ein *B*, bei Grün ein *G* oder ein - falls die LED nicht leuchtet.

3.3.2 Synchronisations-Signal

Das Synchronisationssignal wird durch das Drücken der Entertaste generiert.

3.3.3 Übertragung von Befehlen

Das Senden von Befehlen läuft über das Eingeben von ASCII-Zeichen ab. Diese können nach jedem Einzelbild eingegeben, und mit der Entertaste bestätigt werden.

3.4 Target-spezifisch

3.4.1 Display

Das Display besteht hier aus den LEDs die durch die zeitliche Ansteuerung und die Rotation den Effekt eines Kugeldarstellung erzielen.

3.4.2 Synchronisations-Signal

Das Synchronisationsignal wird durch einen Interrupt bereitgestellt, der durch den Reed-Kontakt ausgelöst wird, wenn dieser in die Nähe eines Magnetfeldes kommt.

3.4.3 Übertragung von Befehlen

Die Übertragung von Befehlen erfolgt über das UART-Interface. Dieses ist mit dem Bluetooth-Modul verbunden und kann somit Befehle eines PC oder Handys empfangen und dadurch entsprechende Aktionen ausführen.

3.4.4 Interrupts

Durch Interrupts ist es möglich, die Software in ihrer Performance zu optimieren indem Warte-Schleifen vermieden werden, und um die damit gewonnene Zeit durch wichtige Berechnungen zu füllen. Auch das Auslesen des Reed-Kontaktes ist somit viel effizienter möglich und benötigt durch das Auslösen eines Interrupts weniger Ressourcen als das vergleichbare Polling, das ständige Überprüfen des Zustandes.

3.4.5 SPI-Interface

Das SPI-Interface wird für die WS2803-Chips benötigt. Hier muss eine Initialisierung der Hardware-Peripherie erfolgen. Zudem ist anzumerken, dass nach der Initialisierung zunächst einmal ein Byte (z.B. 0x00) gesendet werden muss, damit es später möglich ist, das "Ich bin fertig mit sendenBit abzufragen. Sonst könnte es zu einem Dead-Lock führen und damit werden keine Daten über das SPI-Interface gesendet.

3.4.6 Bluetooth

Das Bluetooth-Modul muss bei jedem Start initialisiert werden. Diese Initialisierungssequenz, ist in der Software abgebildet.

4 Algorithmen

4.1 Schrift

Ein nicht zu vernachlässigendes Fundament der Software ist die Schrift. Sie ermöglicht es dem Entwickler Wörter auf dem Display darzustellen. Dieser Software-Baustein wurde erst nach der Entwicklung von Pong und Snake und damit unerwartet spät programmiert, der eine Modus Auswahl ohne Schrift unmöglich ist. Dazu kommt, dass auch Zahlen wie High-scores, dem Spieler nicht dargestellt werden konnten. Schrift ist folglich dringend notwendig und gibt dem Globe einen entscheidenden Mehrwert.

4.1.1 Prinzip

Um Wörter und Zahlen anzeigen zu können, werden diese in ihre Bruchteile, nämlich die einzelnen Buchstaben, zerlegt. Die Darstellung der Zeichen ist dem Globe unbekannt. Daher wird dem uC die Anzeige der Zeichen aus der ASCII-Tabelle (siehe 4.1) via Pixel-Map bekannt gemacht. Der Implementierung kommt zu Gute, dass dem uC die ASCII-Tabelle und die zugehörige Chronologie bekannt ist. Die Pixelmaps werden nach der Reihenfolge der einzelnen Zeichen in der ASCII-Tabelle in ein Array gelegt. Somit bilden die Indizes der ASCII-Zeichen das Bindeglied zur zugehörigen Anzeige. Das Ergebnis ist, dass der Nutzer die Funktion mit beliebigen Text, Position und Farbe als Eingabeparameter aufrufen kann. Zusätzlich zur Schrift, soll es

!	"	#	\$	%	&	'	()	*	+
,	-	.	/	0	1	2	3	4	5	6
8	9	:	;	<	=	>	?	@	A	B
D	E	F	G	H	I	J	K	L	M	N
P	Q	R	S	T	U	V	W	X	Y	Z
\]	^	_	`	a	b	c	d	e	f
h	i	j	k	l	m	n	o	p	q	r
t	u	v	w	x	y	z	{		}	~

Abb. 4.1: ASCII-Tabelle

dem Entwickler ermöglicht werden Integer-werte auf dem Globe darzustellen. Hierfür kommt die Funktion Dec-To-ASCII zum Einsatz, die auch in der Vorlesung Computerarchitektur 2 gelehrt wird. Sie wandelt durch Division und Modulo eine Dezimalzahl in einen String um.

4.1.2 Problemlösungen

Der Globe besitzt aktuell nur 48 LED-Zeilen und bietet daher keinen Platz für eine hochauflösende Darstellung der Buchstaben. Daher wurde bei der Konstruktion der Buchstaben darauf



Abb. 4.2: Darstellung der ASCII-Tabelle auf dem Globe

wert gelegt, die Anzeige möglichst klein zu halten. Folglich wurde die Anzeige eines Letters auf eine Größe von 4*5 Pixel festgelegt.

Ein weiteres Problem ist die Speicherknappheit auf dem RAM. Die einzelnen Pixelmaps wurden daher auf dem Flashspeicher untergebracht und via Lookup für die Funktionen verfügbar gemacht.

4.2 Mode-Select-Menu

Das Mode-Select-Menu ist Dreh- und Angelpunkt aller weiteren Algorithmen bzw. Modi. Es dient als grafisches Auswahlmenü für den Ein- und Ausgang der einzelnen Elemente.

4.2.1 Prinzip

Dank der Möglichkeit Schrift auf dem Globe anzuzeigen, gibt es das Mode-Select-Menu. Es listet alle vorhandenen Modi auf und der Anwender kann mit [W] und [A] innerhalb dieser Modii navigieren und mit der Eingabe von [#] den gewählten Modus auszuführen. Das “>“-Zeichen vor einem Modus zeigt den aktuell selektierten Modus an. Nachdem der Anwender einen Modus ausgeführt hat, kommt er mit der erneuten Eingabe von [#] wieder ins Mode-Select-Menu zurück.



Abb. 4.3: Darstellung des Menü auf dem Globe

4.2.2 Optimierung

3 Zeilen

4.3 Pong

Der Vater der Videospiele Pong findet sich in leicht abgewandelter Variante auch auf dem LED-Globe wieder. Das Spiel in seiner Grundidee wurde schon in den 70er Jahren programmiert. Mit Atari wurde es weltweit vermarktet und zu dem Klassiker, der er heute ist. Deshalb darf Pong auf dem LED-Globe nicht fehlen. Pong ist auf der Zeitachse der Studienarbeit das erste Spiel, welches für die Kugel entwickelt wurde. Damit bildet es die Grundlage für viele weitere Spielideen und Erweiterungen.

Die Kugelform des Globes reduziert das Spiel auf einen Spieler, der gegen sich selbst spielt. Diese Variante von Pong nutzt als Zusatz die Endlosigkeit des Kreises aus. Ein ultimatives 360 Grad Erlebnis!

4.3.1 Spielaufbau

Das auf dem LED-Globe realisierte 2D-Spiel Pong besteht aus einem Paddle (ein vertikaler Strich) und einem Ball (ein Punkt). Das Paddle, befindet sich auf einer fixen x-Koordinate im Feld, während die y-Koordinate im Spiel variabel ist. Das Paddle besitzt eine vordefinierte Breite, die wesentlich geringer als die Spielfeldbreite ist. Der Ball kann sich während der Spielzeit sowohl in x-Richtung als auch in y-Richtung frei bewegen und beginnt mit einer vordefinierten

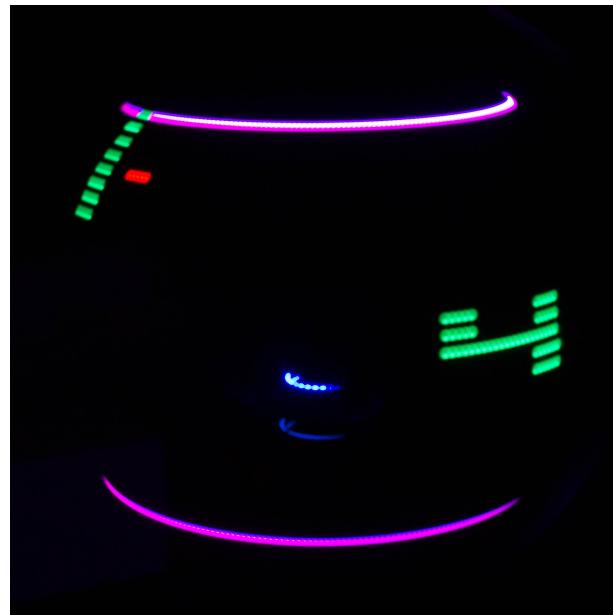


Abb. 4.4: Darstellung von Pong auf dem Globe

Anfangsbewegung.

Die Grenzen in y-Richtung bieten zwei horizontale Linien, welche am Globe oben und unten angeordnet sind. Sie haben den gleichen Abstand vom jeweiligen “Nord-“ bzw. “Südpol“ des Globes. In x-Richtung ist die x-Koordinate des Paddles, sowohl von links kommend, als auch von rechts kommend, die Grenze für das Spielfeld des Balls.

4.3.2 Spielprinzip

Ziel des Spiels ist es den Ball so lange wie möglich im Spielfeld zu halten. Als Bande an denen der Ball abprallt dienen sowohl die horizontalen Grenzen, als auch in der Vertikalen das Paddle. Das Paddle kann vom Spieler nach unten und nach oben gesteuert werden. An den horizontalen Grenzen gilt das Gesetz Einfallswinkel gleich Ausgangswinkel. Am Paddle hängt der Ausfallswinkel des Balls von der Höhe ab, in der der Ball auf das Paddle trifft. Wenn der Ball auf das Paddle oberhalb der Mitte trifft, springt der Ball nach oben ab; Wenn der Ball das Paddle unterhalb der Mitte trifft, springt der Ball nach unten weg. Bei jedem Aufprall des Balls auf dem Paddle erhöht sich die Punktzahl. Passiert der Ball ohne das Paddle zu berühren, endet das Spiel.

4.3.3 Optimierung

Wegen der RAM-Knappheit auf dem Mikrokontroller sind alle Variablen nach ihrem notwendigen Maximalwert gewählt. Für x- und y-Koordinaten wurden der Datentyp `uint8_t` mit einem Byte Speicherplatz verwendet, da die Spielfeldbreite und -höhe aktuell kleiner als 256 sind. Außerdem wurde nur Speicherplatz für Variablen die sich während eines Spieles ändern auf dem RAM reserviert. Die Breite des Paddels, die Geschwindigkeit des Balls oder die Frames pro Berechnung, müssen über Konstanten vorkonfiguriert werden.

4.3.4 Problemlösungen

Das rechteckige Pong-Spielfeld wird auf dem Globe am oberen und unteren "Pol" stark verzerrt. Deshalb werden Ball und Paddle nur sehr schlecht erkannt. Um diese Verzerrung möglichst klein zu halten befinden sich die horizontalen Grenzen in einem definierten Abstand von den Polen. Auf Grund der RAM-Knappheit auf dem Mikrokontroller sind alle Variablen nach ihrem notwendigen Maximalwert gewählt. Für x- und y-Koordinaten wurden der Datentyp `uint8_t` mit einem Byte Speicherplatz verwendet, da das Spielfeldbreite und -höhe sind aktuell kleiner als 256 Pixel sind.

Die Geschwindigkeit des Balles ist über einen Geschwindigkeitsvektor dargestellt. Er lässt einen Geschwindigkeitsbetrag von bis zu 7 Pixeln in eine Richtung zu. Geschwindigkeiten über 3-4 Pixel stellen sich als Sprünge für den Betrachter dar. Deshalb ist eine Begrenzung der Maximalgeschwindigkeit für x- und y-Richtung sinnvoll und notwendig.

4.4 Snake

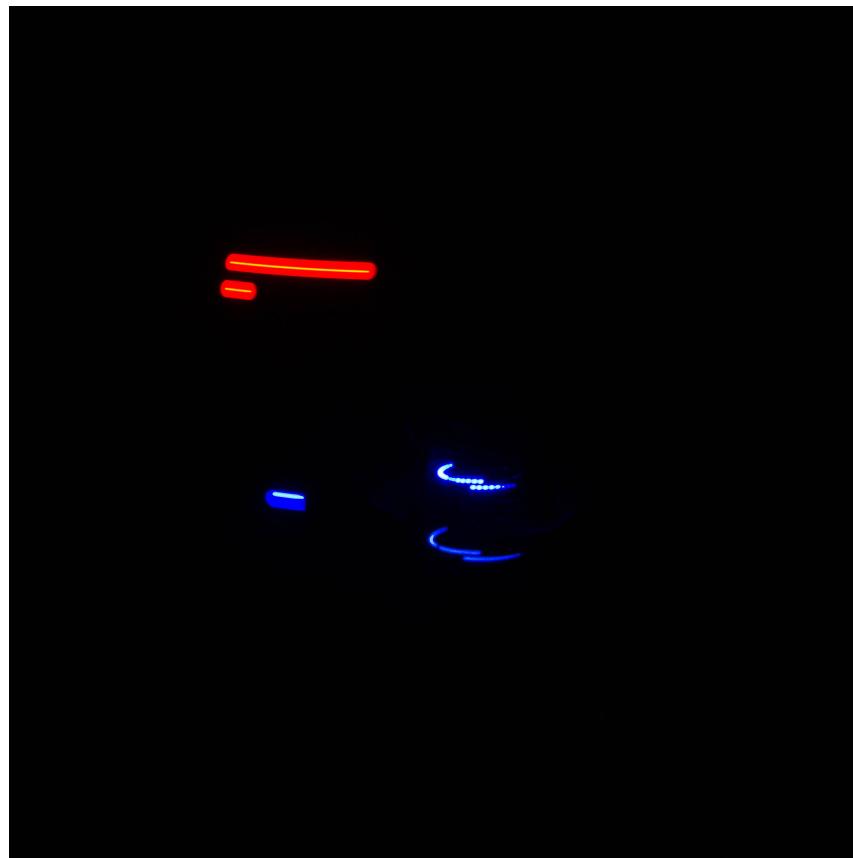


Abb. 4.5: Darstellung von Snake auf dem Globe

Snake - eines der ersten Spiele auf Handys im digitalen Zeitalter gewinnt eine neue Dimension auf der Kugel. Diese bietet die Freiheit im dreidimensionalen Raum bei der es keine Barrieren

gibt. Die Schlange kann sich in jede Richtung fortbewegen und bietet damit ein nettes Gimmick für den LED-Globe.

4.4.1 Spielaufbau

Die Spielfigur bei Snake ist die Schlange. Sie ist aus vielen einzelnen Schlangensegmenten zusammengesetzt. Das Kopfsegment führt die Schlange an, dem alle weiteren Segmente folgen. Zusätzlich befindet sich auf dem Feld der “Köder“, welcher als Fressen für die Schlange gedacht ist.

Das Spielfeld für Snake besteht aus zwei horizontalen Grenzen. Zwischen ihnen wird der Köder erzeugt. Für die Schlange gibt es auf dem Globe keine Grenzen. Sowohl in x- als auch in y-Richtung ist die Welt für sie unbegrenzt.

4.4.2 Spielprinzip

Das Ziel von Snake ist es den Kopf der Schlange zum Futter zu führen. Der Spieler kann die Bewegungsrichtung des Schlangenkopf und den nachfolgenden Segmenten des Schlangenschwanzes steuern. Mit jedem Stück gefressenem Köder, wächst die Schlange um ein Segment. Kollidiert der Schlangenkopf mit seinem Hinterteil ist das Spiel verloren.

4.4.3 Optimierung

Bei der Schlange kam eine verkettete Liste zum Einsatz. Theoretisch könnte die Schlange daher unbegrenzt anwachsen. Eine Begrenzung der Maximalanzahl der Segmente ist erforderlich, da der Arbeitsspeicher begrenzt ist.

4.4.4 Problemlösungen

Der Köder wird nur in einer definierten Entfernung von den “Polen“ generiert. Durch die starke Verzerrung an den Polen, ist es sehr schwer auszumachen in welcher Spalte und Reihe sich Schlange und Köder befinden.

Zusätzlich wurde es notwendig Futter an einer zufälligen Stelle zu generieren. Man kam also nicht umhin einen Algorithmus für eine geeignete Zufallszahlen Erstellung von Start zu Start zu entwickeln. Diese Implementierung ist in [2.3.2](#) genauer beschrieben.

4.5 Flappy Bird

Auch ein modernes Spiel hat den Weg zum LED-Globe gefunden. Auf den heutigen Smartphones hat das Spiel Flappy Bird einen so großen Anklang gefunden. Aufgrund der Suchtgefahr fühlte sich der Erfinder des Spiels, Dong Nguyen, verpflichtet das Spiel aus den Online-Stores zu nehmen. Heute sind, zur Freude der Smartphonebesitzer, viele Cover des Original-Spiels im Umlauf. Selbst Android fügte das Spiel als Eastereggs ins neueste Betriebssystem (Android 5.0 “Lollipop“) ein. Beim Grafikdesign hat Nguyen auf das Kultspiel Super Mario World zurückgegriffen. Ein Vogel

fliegt zwischen den altbekannten grünen Röhren hindurch, während im Hintergrund am blauen Himmel die weißen Wolken vorbeiziehen. Was den Spielreiz bei Flappy Bird ausmacht, sind die wahnsinnigen Glücksgefühle beim Passieren des Vogels durch die engen Passagen.

4.5.1 Spielaufbau

Die Abwandlung von Flappy Bird auf dem Globe steht dem Original in nichts nach. Im Mittelpunkt steht ein kleiner vom Spieler gesteuerter Vogel der am linken Rand des Displays in einer definierten fixen x-Position fliegt. Wie beim großen Bruder trumpft das Spiel mit zweier Paaren von grünen Röhren auf, die jeweils von oben und unten in das Display herein ragen. Zwischen den Röhren kommt es zu einer Lücke, die dem Vogel als Fluchtweg dient. Die Rohr-Paare werden im gleichmäßigen horizontalen Abstand am rechten Bildschirmrand erstellt und unterscheiden sich durch ihre vertikale Länge und die Größe der Lücke zwischen ihnen. Nach der Erstellung bewegen sie sich langsam auf den Vogel zu, bis sie am linken Rand ankommen und wieder entfernt werden. Am unteren Displayrand grenzt eine sich durchweg ändernde Wolkendecke an. Dadurch kommt das Gefühl auf, dass der Vogel in Richtung der Röhren fliegt.



Abb. 4.6: Darstellung von Flappy Bird auf dem Globe

4.5.2 Spielprinzip

Das Ziel des Spiels ist es den Vogel davor zu bewahren zum einen gegen eine Wand zu prallen und zum anderen nicht in den Abgrund zu fallen. Wie in 4.5.1 beschrieben gibt es eine Lücke in einem Rohr-Paar und genau dort soll der Vogel durchfliegen. Dafür kann der Spieler den Vogel in der Vertikalen nach oben beschleunigen und der Vogel fällt durch die Schwerkraft wieder nach unten. Für jedes passierte Lücke erhöht sich die Highscore.

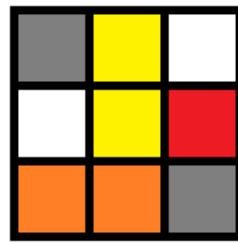


Abb. 4.7: Konzept für die Darstellung des Vogels auf dem Globe

4.5.3 Optimierung

Flappy Bird ist das dritte Spiel auf dem Globe und kann daher auf die Erfahrungen mit den ersten beiden Spiele, Pong und Snake, zurückgreifen. Es konnte auf die bestehende Problematik, wie die Speicherknappheit, die schlechte Performance und die Verzerrung an den “Polen“ schon bei der Konzeptionierung eingegangen werden. Eine erweiterte Optimierung war nicht notwendig.

4.5.4 Problemlösungen

Fun Fact: Bei Flappybird wird jede LED an jeder Position der Drehung zum Leuchten gebracht. Beim ersten Test, wurde die Intensität aller LEDs auf den maximalen Wert gestellt. Nach dem Anwählen von Flappy Bird und dem kurzen Aufleuchten des Globes, war der Spaß vorbei. Das Netzteil konnte den benötigten Strom nicht liefern und ging in den Safe State - Strom aus. Um dieses Problem zu beheben ist der Großteil des Hintergrunds, sprich Himmel und Wolken in ihrer Intensität reduziert. Die entscheidenden Elemente, wie Vogel und Röhren behalten dagegen höhere Intensität.

4.6 Pacman

Ein gelber Fleck, süchtig nach kleinen weißen Pillen, wird in seiner Paranoia von farbigen Geistern verfolgt – Pacman.

Selbst das komplexe Spiel Pacman kann auf dem Globe dargestellt werden. Es übertrifft alle anderen Spiele nicht nur durch eine neuartige Perspektive, sondern auch durch erste computergesteuerte Gegenspieler. Außerdem bietet das große Spielfeld eine Herausforderung hinsichtlich der Speicherkapazität. Doch die Autoren scheut keine Mühe und das Ergebnis ist dieses wunderbare Spiel.

4.6.1 Spieldaten

Pacman besteht aus vier verschiedenen farbigen Geistern, Pacman selbst eine gelbe Kreisfläche und einem Spielfeld, welche in den folgenden Unterpunkten genauer beschrieben werden.

Spielfeld und Pillen

Das Spielfeld besteht, wie in Abbildung 4.8 gezeigt, aus einem Labyrinth mit vielen schwarzen Gängen und blauen Mauern. Diese bestimmte Anordnung der Gänge und der Maßstab entspricht dem Spielfeld des Original-Spiels. In der Mitte befindet sich ein Käfig in dem sich die Geister zu Beginn des Spiels befinden. Dieser Käfig spielt auch während des Spielverlaufs eine wichtige Rolle für die Geister, wie im Kapitel 4.6.2 genauer beschrieben wird. Pacman beginnt unterhalb des Käfigs. Um das ganze Spielfeld auf dem Globe speichern zu können, wurde es in den Flashspeicher geschrieben.

Auf dem Spielfeld verteilt befindet sich Futter für Pacman. Die genaue Verteilung sieht man in 4.8. Zusätzlich zum “normalen“ weißen Futter befindet sich in jedem Quadranten des Spielfelds ein “spezielles“ türkisfarbenes Futter. Der Effekt der beim Fressen des “speziellen“ Futters entsteht, wird im nächsten Kapitel 4.6.2 erläutert. Der Initialzustand des Futters wird wie das Spielfeld, im Flashspeicher untergebracht. Jedoch muss der doch sehr speicherintensive Zwischenstand des Futters, zum Leid des RAMs, im Arbeitsspeicher gehalten werden.

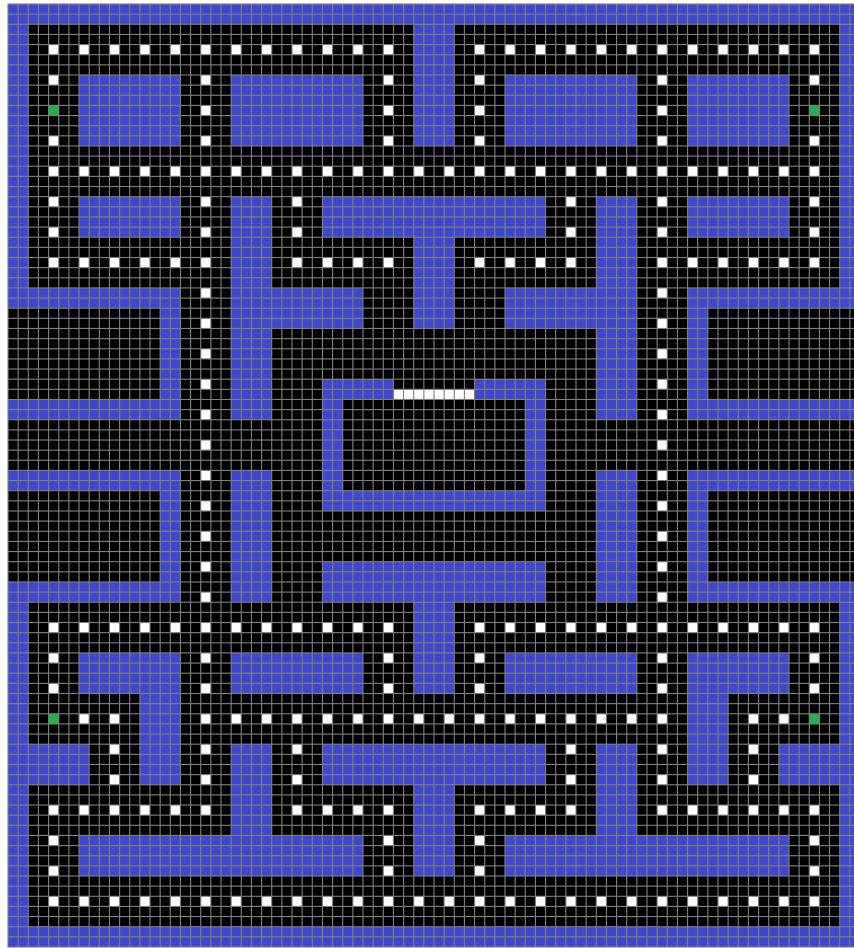


Abb. 4.8: Konzept des Pacman Spielfelds

Geister und Pacman

Die Geister wie auch Pacman sind die Spielfiguren. Sie bewegen sich innerhalb der Gänge und im Spezialfall verschwinden sie im Mittelgang auf einer Seite aus dem Spielfeld und tauchen auf der gegenüberliegenden Seite wieder auf. Die Geister werden von einer Künstlichen Intelligenz gesteuert; Pacman wird dagegen vom Spieler gesteuert.

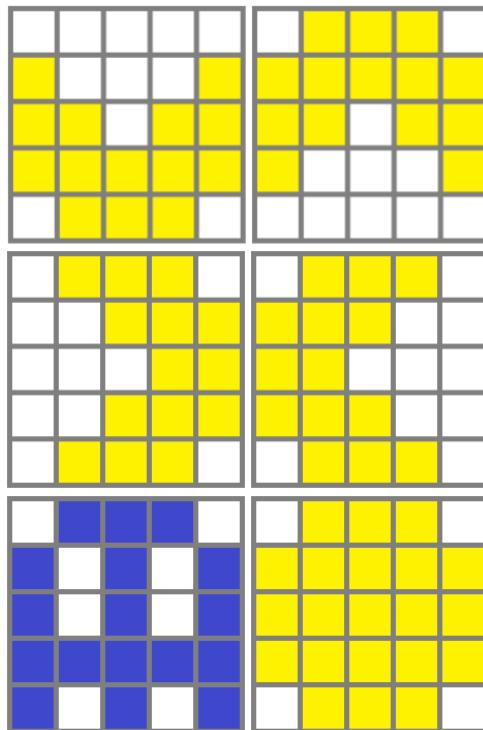


Abb. 4.9: Darstellung der Geister und Pacman

4.6.2 Spielprinzip

Zustandsautomat beschreiben Bei Pacman steht und fällt der Spielspaß mit der Qualität der Künstlichen Intelligenz. Der Unterschied wird deutlich spürbar, wenn die Geister nur zufällig durchs Labyrinth irren.

4.6.3 Optimierung

Da beim Microcontroller nicht nur der RAM-Speicher an seine Grenzen kommt, musste auch beim Flashspeicher an Ressourcen gespart werden. Als erstes wurde daher beim Spielfeld gespart. Das Spielfeld ist wie in Abbildung 4.8 gezeigt an der vertikalen Mittelachse symmetrisch. Daher wurde bei der Speicherung des Spielfelds, wie beim Initialzustand der Pillen dank der vertikalen Achsensymmetrie Ressourcen eingespart.

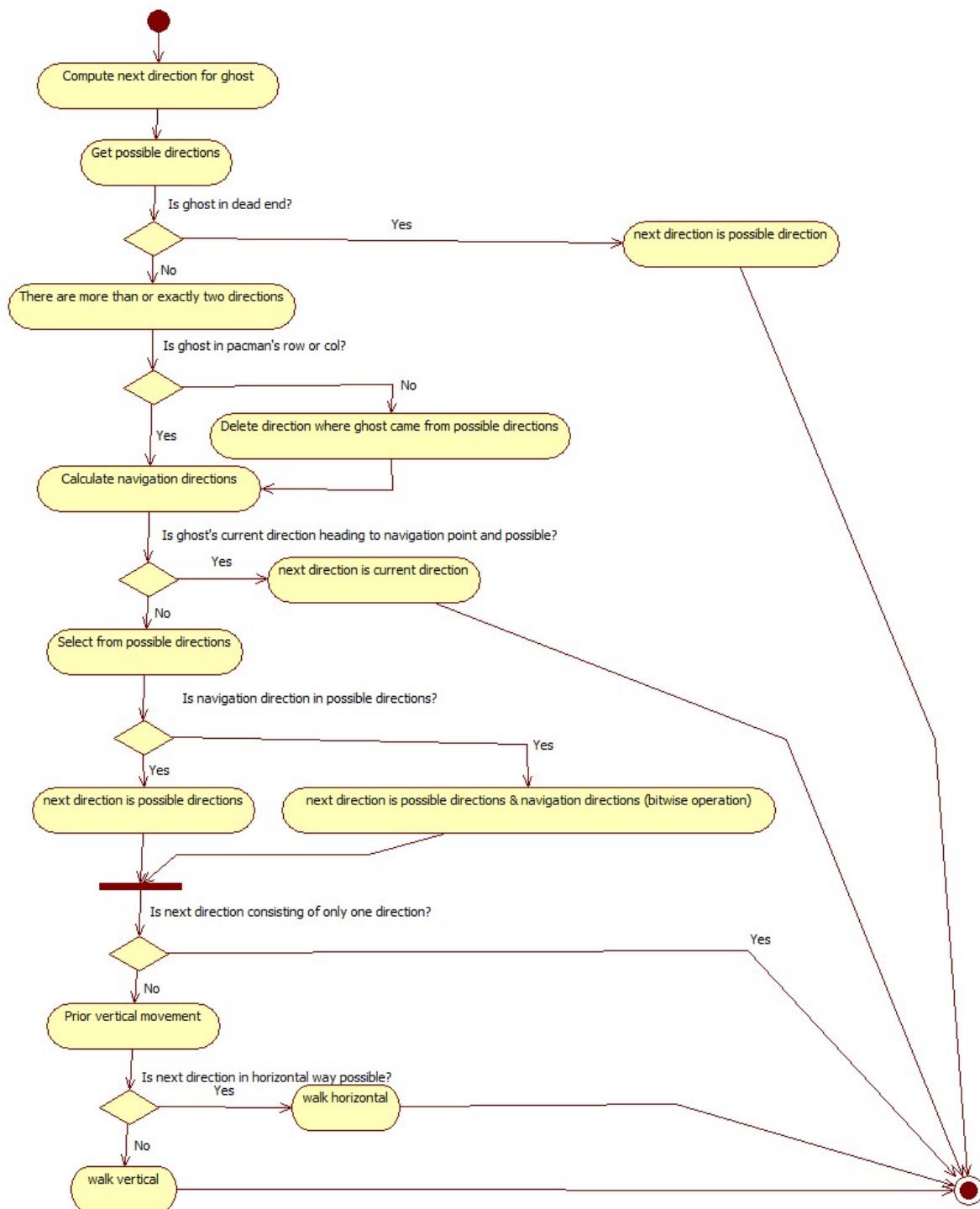


Abb. 4.10: Flussdiagramm der Navigationslogik der Geister

4.6.4 Problemlösungen

Ein Problem das schon im Konzept Sorgen bereitet hat, ist das kleine Display im Vergleich zum großen Spielfeld. Daher ist es unmöglich das ganze Spielfeld darzustellen. Die Lösung ist das Darstellen eines Teilbereichs. In der Mitte des Bereiches ist Pacman und wenn er sich in eine bestimmte Richtung bewegt, folgt die "Kamera". Leider geht dadurch ein Teil der Übersicht verloren, kommt aber der Spieldynamik zu Gute.

4.7 Statische Bilder

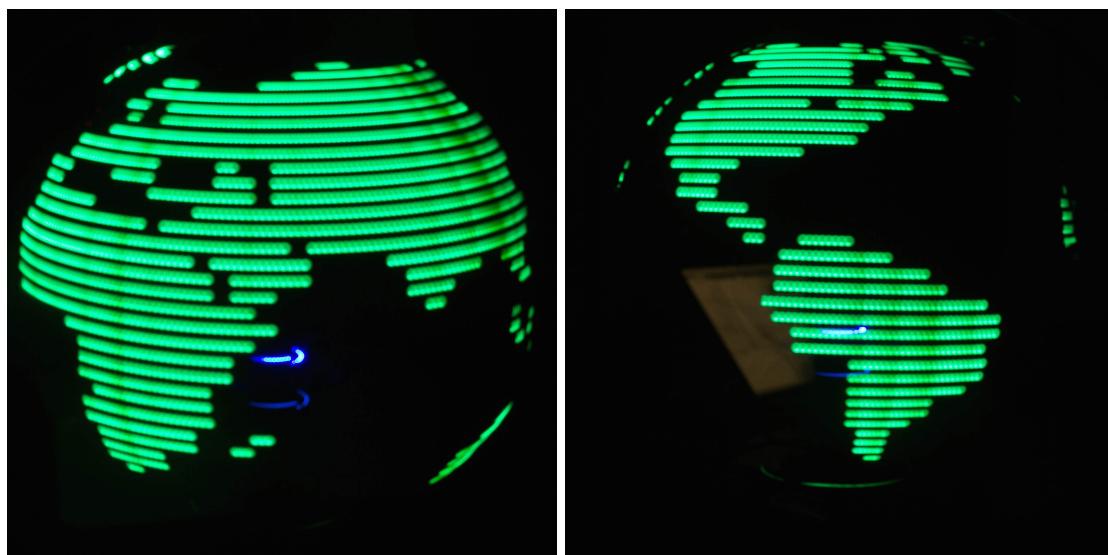


Abb. 4.11: Darstellung einer Weltkarte auf dem Globe

4.7.1 Speicher

Wie liegen die daten im Speicher?

4.7.2 Generierung von Bilder

BMP2C.EXE



Abb. 4.12: Darstellung von Game of Life auf dem Globe - Hier ist ein Pattern zu sehen, das schräg von oben-links nach unten-rechts wandert.

4.7.3 Problemlösungen

4.8 Conway's Game of Life

4.8.1 Beschreibung

Das Spiel des Lebens (Game of Life zukünftig GOL) von Conway hat vier Grundregeln. Es gibt immer eine aktuelle Generation (Array mit "lebenden" und "toten" Pixeln) und eine darauf folgenden Generation. Mit jedem Ausführen des Algorithmus wird die neue Generation berechnet und zur Aktuellen erklärt.

Die Grundregeln zur Berechnung der nachfolgenden Generation lauten:

- ein Pixel mit weniger als zwei lebenden Nachbarn stirbt an Unterbevölkerung
- ein Pixel mit mehr als drei lebenden Nachbarn stirbt an Überbevölkerung
- ein lebendiger Pixel mit zwei oder drei lebenden Nachbarn bleibt am Leben
- ein toter Pixel mit genau drei lebendigen Nachbarn wird zum leben erweckt

Aus diesen Grundregeln kann nun 'Leben' auf dem Array entstehen. Lässt man sich die einzelnen Generationen auf dem LED-Globe anzeigen, sieht dies in etwa wie wachsende Bakterien in einer Petrischale aus.

4.8.2 Optimierungen

4.8.3 Problemlösungen

Um Speicherplatz zu sparen und um GOL überhaupt realisieren zu können, musste auf eine farbige Darstellung verzichtet werden. Jedes Pixel entspricht nun einem Bit in einem der beiden Arrays, die die aktuelle bzw. nächste Generation abspeichern.

Durch die Größe der Array im Vergleich zum verfügbaren Speicherplatz musste dennoch die Komprimierung der Daten weiter optimiert werden. Anstatt die

Bit-array mit Inlinefunktionen für den zugriff -> weniger verschachtelte funktionsaufrufe

4.8.4 Problemlösungen

Das Kopieren der neuen Generation in die Aktuelle war

Selbst die stark optimierte Version des Algorithmus ist zu aufwändig, um eine komplette Generation in einer Umdrehung berechnen zu können. -> berechnung der neuen Generation in Teilschritten -> jedes x-te Frame wird die generation erst neu angezeigt

4.9 Manuelle Farbwahl

Im Globe ist ein Modus implementiert, mit dem man die Farbe der LEDs einstellen kann. Dabei kann der PWM-Wert jeder einzelnen Farbe verändert werden, indem mit den Buchstaben R, G, B die Farben Rot, Blau oder Grün um eine Stufe heller gemacht werden und mit den Buchstaben r, g, b, verdunkelt werden. In Abbildung 4.13 sieht man Rot in den Stufen 1-3 und Blau und Grün ebenfalls in Stufe 3. Durch den hohen Energiebedarf der LEDs ist muss diese Funktion vorsichtig benutzt werden, da sonst die Verbindung mit dem Bluetooth-Modul abbricht, da es zu einem Spannungseinbruch kommt. Prinzipiell wäre es möglich, die Farben in jeweils 255 Schritten einzustellen

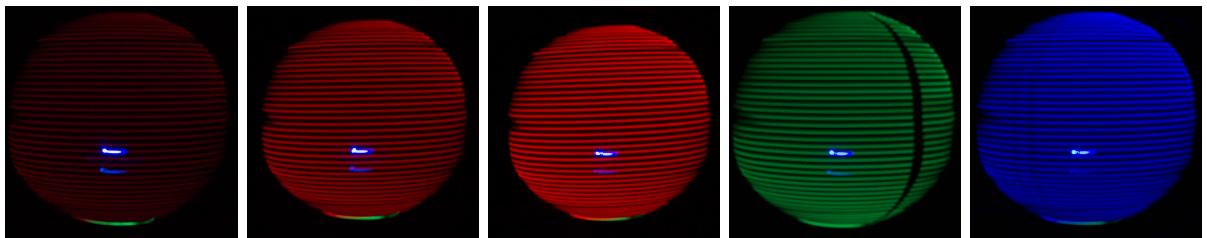


Abb. 4.13: Manuelle Auswahl der Farben

4.10 Color-Fading

4.10.1 HSV-Farbraum

4.10.2 Problemlösungen

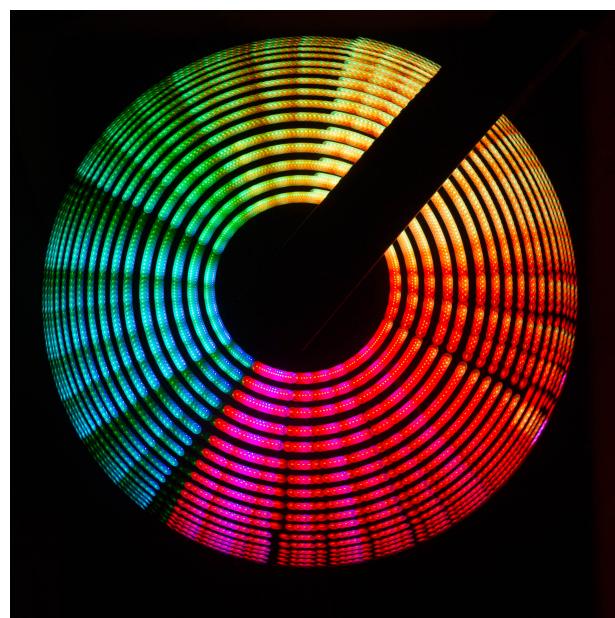


Abb. 4.14: Darstellung von Color-Fading auf dem Globe

5 Mechanik

Die Mechanik des Globe soll in diesem Projekt nicht im Vordergrund stehen. Die Konstruktion soll lediglich einen funktionalen Zweck erfüllen, und als Proof-of-Concept angesehen zu werden. Ein Großteil der verwendeten Materialien war bereits vorhanden, wodurch das Projekt ohne große Investitionen realisieren war.

5.1 Aufbau

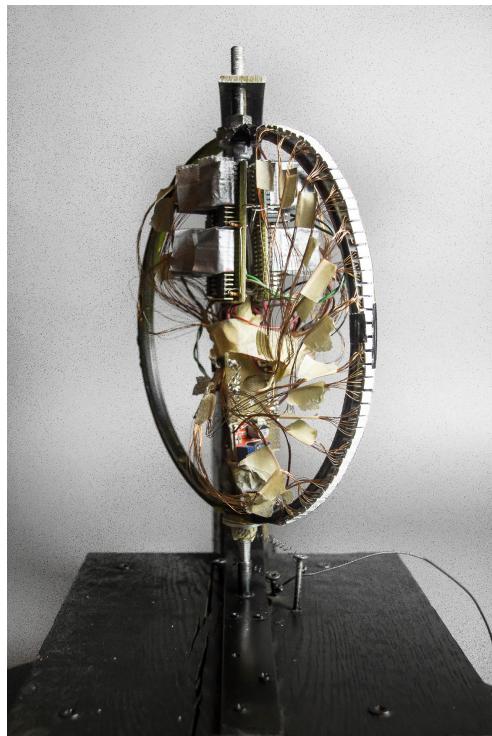


Abb. 5.1: Aufbau des Globe

Der Rahmen des Globe ist aus Holz und einem Stahlring gefertigt. Der Stahlring wurde aus einem Stahlstab gebogen und an beiden Enden eine Öffnung für die aus einer Gewindestange bestehende Achse gebohrt. Zusätzlich wird der Ring durch eine passend rund gesägte Holzplatte stabilisiert um Schwingungen zu minimieren. Um die Leds in einem Halbkreis anordnen zu können wurde ein runder, stabiler Ring benötigt, der aber nicht allzu schwer sein durfte. Hier wird auf eine Friesbee-Scheibe zurückgegriffen, bei der der mittlere Teil des Tellers ausgeschnitten ist und somit nur noch der Rand übrig bleibt. Dieser Ring ist an der Gewindestange mit 4 Muttern befestigt.



Abb. 5.2: Aufbau des Globe (Ansicht von oben)

5.2 Motor

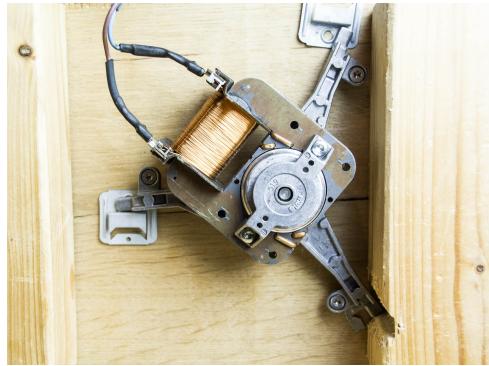


Abb. 5.3: Anbringung des Motors auf der Unterseite

Ein alter Backofenventilator treibt die Konstruktion an, der wegen einer kleinen Umwucht ausgetauscht wurde. Es handelt sich dabei um einen 230V Wechselstrom-Motor mit einer angegebenen Drehzahl von 42,2 U/sec. Durch das Gewicht das auf dem Lager des Motors liegt, ist die resultierende Drehzahl deutlich geringer und beträgt etwa 14 U/sec. Eine genaue Arretierung wird durch die drei Beine, mit denen der Motor mit dem Holzrahmen verbunden ist, erreicht. So ist die Achse gerade und schleift nicht am Loch des Stahlrings.

5.3 Schleifkontakte

Der Schleifkontakt besteht aus Kugelschreiberfedern, die an der Achse des Globe entlangschieben. Da die Übertragung nicht immer unterbrechungsfrei funktioniert, muss eine Pufferung mittels eines Kondensator erfolgen, der die kurzen Spannungsabbrüche abfängt.



Abb. 5.4: Schleifkontakte aus Kugelschreiberfedern

5.4 Alte Revisions

5.4.1 RGB-Streifen mit WS2811-Chips



Abb. 5.5: Die langen PWM-Zyklen der WS2811-LEDs sind sehr deutlich zu erkennen

Bevor die WS2803-Chips zur Ansteuerung der LEDs verwendet wurden, gibt es den Versuch, SMD-LEDs mit integriertem PWM-Controller zu verwenden. Diese LEDs gibt es im Internet in Ketten auf einer flexiblen Leiterbahn zu kaufen, welches den Verkabelungsaufwand auf ein Minimum reduziert. In Abb. 5.5 wird das Problem mit den WS2811 sehr deutlich. Die interne PWM-Frequenz der LEDs ist viel zu langsam und nicht synchron. Durch die schnelle Rotation des Globus sieht man den PWM-Zyklus. Ein Pixel auf dem Globus hat somit eine Länge von etwa 3cm was für die Anwendung nicht ausreicht.

5.4.2 Batteriebetrieben

Die batteriebetriebenen Version des Globe, bei dem zwei Batterieblöcke mit 9V verwendet werden, erweist sich durch den hohen Stromverbrauch als unpraktisch, da die Batterien viel zu schnell leer sind. Deshalb wurde die Lösung mit den Schleifkontakte ausgeführt.

6 Bedienungsanleitung

6.1 Herstellen der Bluetooth-Verbindung

Um eine Verbindung zum Bluetooth-Modul aufzubauen, muss der Globe eingeschaltet sein und sich drehen. Mit dem Authentifizierungscode “1234“ stellt man die Verbindung zum Bluetooth-Modul her. Ein Terminalprogramm wie Putty greift auf die nun vorhandene Serielle Schnittstelle zu. Für die Bluetooth-Verbindung sind folgende Parameter zu verwenden: 19200/8/N/1

6.2 Steuerung des Globe

Die Steuerung des Globes ist dem 1. Modell des Gameboys angelehnt, nur ohne die “Start“-Taste. Zum einen gibt es die einzelnen Richtungstasten für die Navigation im Mode-Select-Menu und in den einzelnen Modi. Zum anderen gibt es die “Select“-Taste mit denen im Mode-Select-Menu der Modus ausgewählt werden kann bzw. um einen angewählten Modus wieder verlassen zu können.

Das Menü ist wie folgt aufgebaut:

- Snake:
Snake spielen...wie gehts genau?
- Wordmap
- Pong

6.3 Erweitern des Globe um eigene Modi

Es ist auch nachträglich möglich, eigene Programme die auf dem Display dargestellt werden, zu entwerfen. Hierfür müssen nur zwei Dateien erstellt (im folgenden Beispiel example.h, example.c), und Änderungen an zwei weiteren Dateien gemacht werden (modes.h, modes.c)

Datei: ./Application/modes/example.c

```
1 #include "example.h"
2
3 void example_init(){
4     // hier werden notwendige Initialisierungen vorgenommen
5     ...
6 }
```

```

7
8     void mode_example(uint8_t index){
9         // Die Variable index gibt die Position der Spalte an,
10        // an der wir uns aktuell befinden
11        ...
12        // Das Spalten-Array leds[] bearbeiten
13        leds[0].r=255; // erste Led soll rot leuchten
14        leds[0].g=0;
15        leds[0].b=0;
16
17        leds[1].r=0;
18        leds[1].g=255; // zweite Led soll rot leuchten
19        leds[1].b=0;
20
21        leds[2].r=0;
22        leds[2].g=0;
23        leds[2].b=255; // dritte Led soll rot leuchten
24        ...
25        // die berechnete Spalte ausgeben
26        display_set_row(leds);
27    }

```

Datei: ./Application/modes/example.h

```

1 #ifndef EXAMPLE_H_
2 #define EXAMPLE_H_
3
4 #include <stdint.h>
5 #include <stdlib.h>
6 #include "../input.h"
7 #include "../display.h"
8
9 void example_init()
10 void mode_example(uint8_t index)
11
12 #endif /* EXAMPLE_H_ */

```

Datei: ./Application/modes/modes.h

```

1 ...
2 #include "example.h" // Include fuer den neuen Modus einfuegen
3 ...
4 // #define NUM_MODES 12
5 #define NUM_MODES 13 // Anzahl der Modi inkrementieren
6 ...
7 // im Enum-Feld einen neuen Eintrag am Ende einfuegen
8 typedef enum modes{
9     SELECT_MODE = 0,

```

```
10     ... ,  
11     EXAMPLE  
12 }Mode;
```

Datei: ./Application/modes/modes.c

```
1 ...  
2 // Am Ende der Modi-Lookup-Tabelle einen neuen Eintrag erstellen  
3 mode_fktm modes_look_up_table[] = {  
4     mode_select_mode,  
5     ... ,  
6     mode_example  
7 };  
8  
9 // Einen neuen Eintrag fuer die Anzeige im Menue einfuegen  
10 const char * modes_names_look_up_table[] = {  
11     "Sel\u20acMode",  
12     ... ,  
13     "Example"  
14 };  
15  
16 // Die Initfunktion aufrufen lassen  
17 void modes_init(){  
18     ...  
19     //init modes  
20     ...  
21     example_init();  
22 }
```

7 Fazit

Durch die enormen Beschränkungen der Hardware-Plattform stellt das Projekt eine sehr hohe Herausforderung dar. Voraussetzung ist ein tiefes Verständnis des Mikrocontrollers, um Timings und andere Probleme die im Laufe der Realisierung aufgetreten sind zu lösen. Die Konstruktion der Mechanik war sehr anspruchsvoll. Das beschränkte Budget wurde durch sehr viel Kreativität kompensiert. Die Entscheidung eine HAL zu implementieren stellte sich als sehr gut heraus. Ohne diesen Weg ist das Erstellen der Algorythmen und Spiele im späteren Verlauf des Projektes sehr schwierig bis unmöglich.

Natürlich steht auch die Frage nach dem Sinn und Nutzen des Projektes im Raum. Wo gibt es Anwendungsbereiche für den Globe? Marketing ist das Stichwort. Technisch ausgerichtete Messen bieten eine Plattform für den Globe - als Eyecatcher für Messestände, genauso wie Lightshows in Discos, Clubs und Bars. Der Phantasie sind keine Grenzen gesetzt. Vorschläge sind den Autoren willkommen.