FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Kolloquium zur Bachelorthesis
Thema:
„Entwicklung und Implementierung eines Spiels für eine RGB-Matrix, deren Ansteuerung mit Einplatinencomputern erfolgt“

1. **Betreuer:** **Prof. Dr. Christian Baun**
2. **Korreferent:** **Prof. Dr. Thomas Gabel**

Fachbereich 2 Informatik und Ingenieurwissenschaften

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Agenda

- Einleitung
- Stand der Technik
- Design
- Implementierung
- Fazit
- Live Demonstration

Einleitung

- Ziel
  - Entwicklung und Implementieren eines Spiels
  - RGB-Matrixen dienen als Ausgabegerät
- Motivation
  - Wenig Information über den Entwicklungsprozess

Stand der Technik

- Entwicklungssystem
  - Speicher

Stand der Technik

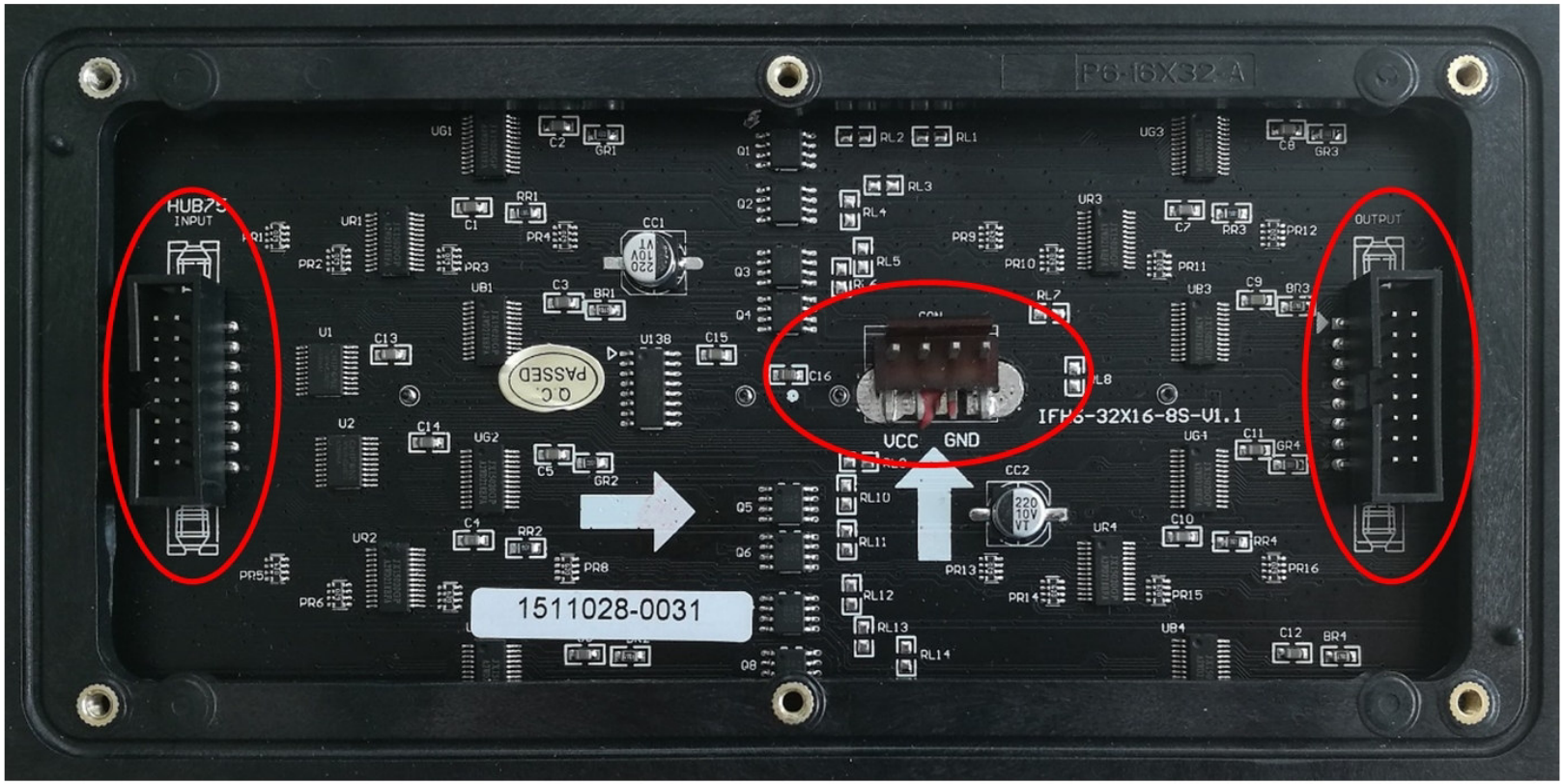
- Entwicklungssystem
  - Speicher
  - Betriebssystem
    - Offiziell Raspbian
    - Bis zu 40 weitere Distributionen

Stand der Technik

- Entwicklungssystem
  - Speicher
  - Betriebssystem
    - Offiziell Raspbian
    - Bis zu 40 weitere Distributionen
- RGB-Matrix
  - 16x32 Bildpunkte

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Stand der Technik

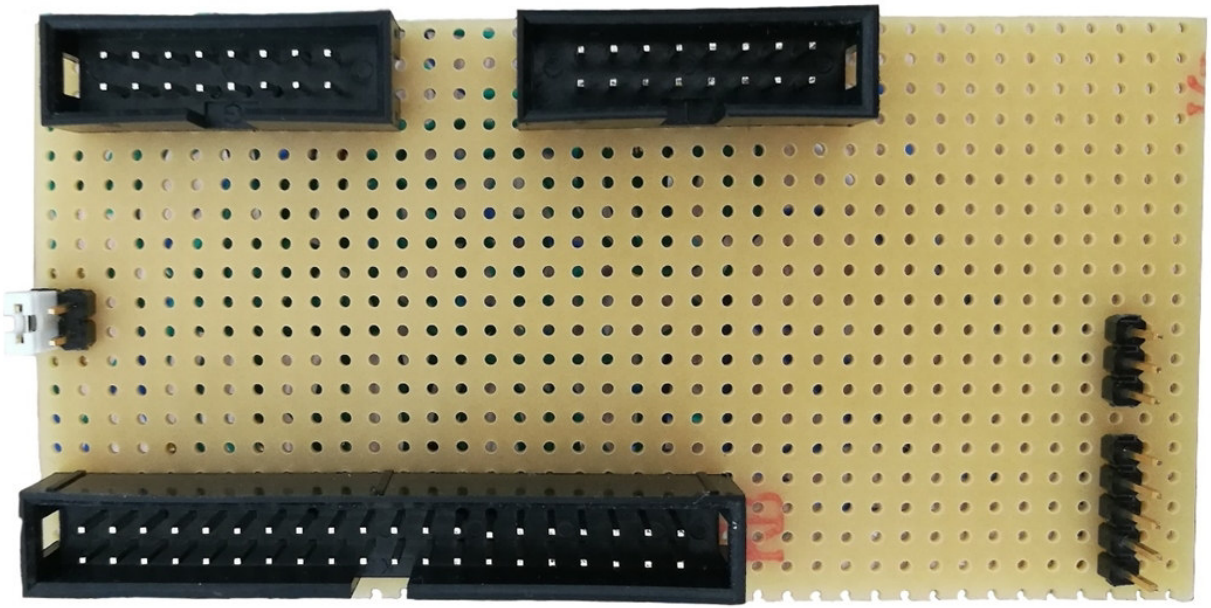


Design

- Anforderungen
- Regeln
- Spieleinstellungen
- Audioeffekte
- Peripheriegeräte
  - Spielcontroller
  - Anschlussplatine

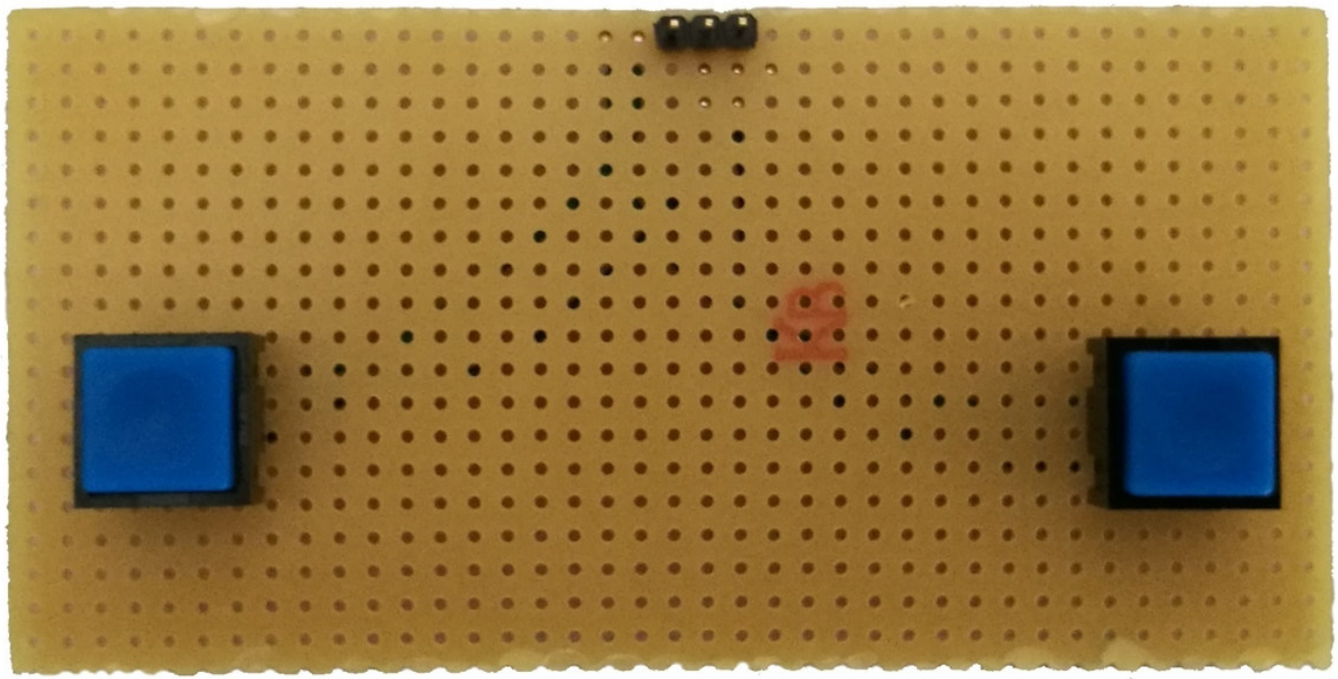
Implementierung

- Peripheriegeräte
  - Anschlussplatine



Implementierung

- Peripheriegeräte
  - Anschlussplatine
  - Spielcontroller



Implementierung

- RGB Matrix ansteuern
  - Programmierschnittstelle (API)
    - Projekt auf Github vom Henner Zeller
  - Audiogerät ausschalten
    - Vermeidung von Störungen
  - Erste Ausgabe

Implementierung

- Externe Audiokarte

Implementierung

- Externe Audiokarte
- Lautsprecher

Implementierung Spiel

- Pakete und Module

```
from rgbmatrix import RGBMatrix, RGBMatrixOptions
from rgbmatrix import graphics
from argparse import ArgumentParser
import time
import random
import RPi.GPIO as GPIO
import threading
import pygame
```

Implementierung Spiel

- Pakete und Module
- Argumente

```
parser = ArgumentParser()
parser.add_argument("-r", "--led-rows", dest="ledrows",
                    help="Anzahl der Reihen einer RGB-Matrix 16,32",
type=int, default=16)
```

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Implementierung Spiel

- Pakete und Module
- Argumente
- Spiel Einstellungen

```
options = RGBMatrixOptions()
options.rows = args.ledrows
options.chain_length = args.seriell
options.parallel = args.parallel
```

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Implementierung Spiel

- Pakete und Module
- Argumente
- Spiel Einstellungen
- Initialisierung der Matrix und der Einstellungen

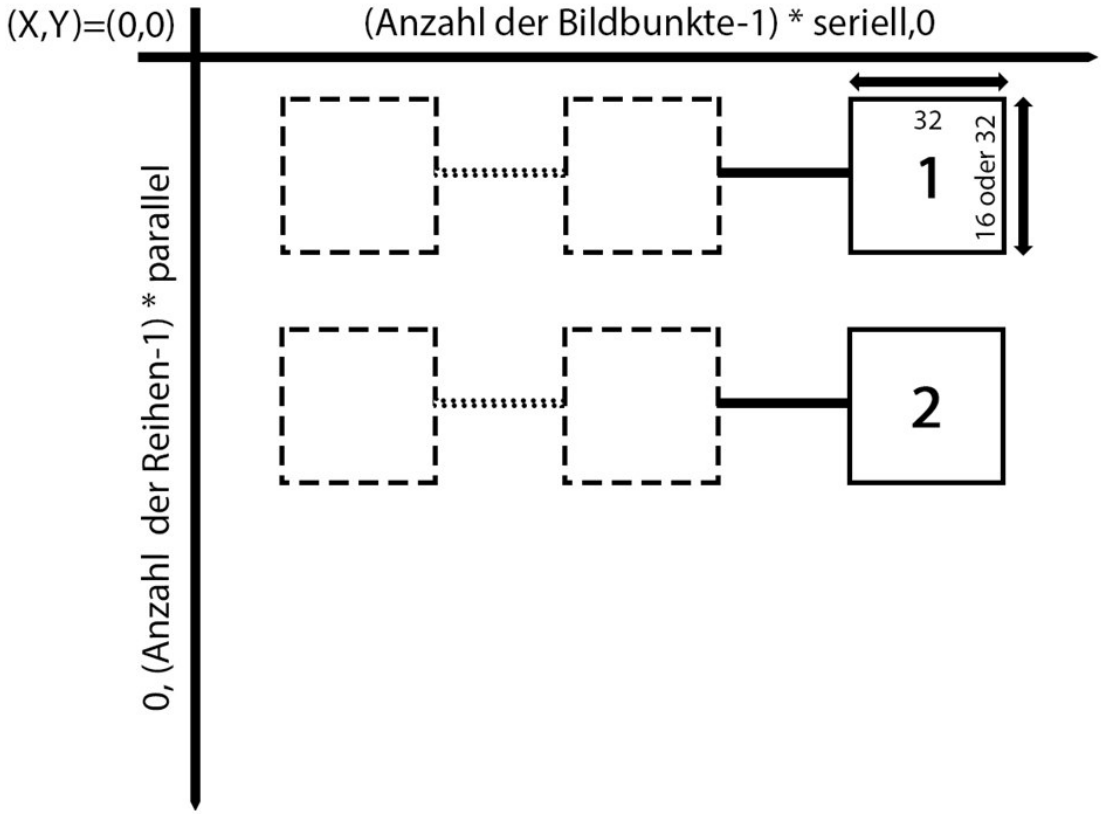
```
matrix = RGBMatrix(options = options)
double_buffer = matrix.CreateFrameCanvas()
```

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Implementierung Spiel

- Pakete und Module
- Argumente
- Spiel Einstellungen
- Initialisierung der Matrix und der Einstellungen
- Positionierung der Objekte

Implementierung Spiel



Dimitri Gubermann | 980065

Implementierung Spiel

- Variablen
  - Farbe, Schriftarten
  - Vertikale und horizontale Anzahl der LEDs
  - Position und Bewegung auf der x- und y-Achse
  - Punktestand

Implementierung Spiel

- Variablen
  - Farbe, Schriftarten
  - Vertikale und horizontale Anzahl der LEDs
  - Position und Bewegung auf der x- und y-Achse
  - Punktestand
- Spielfeld

```
def matchfield(horizontal, vertical):
    graphics.DrawLine(double_buffer, 0, 0, horizontal, 0, blue)
```

Implementierung Spiel

- Spielball

```
def gameball(xAxis, yAxis):
    double_buffer.SetPixel(xAxis, yAxis, 255, 0, 0)
```

Implementierung Spiel

- Spielball
- Ballbewegung

```
ballposx=ballposx+ballmovementx
ballposy=ballposy+ballmovementy
```

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

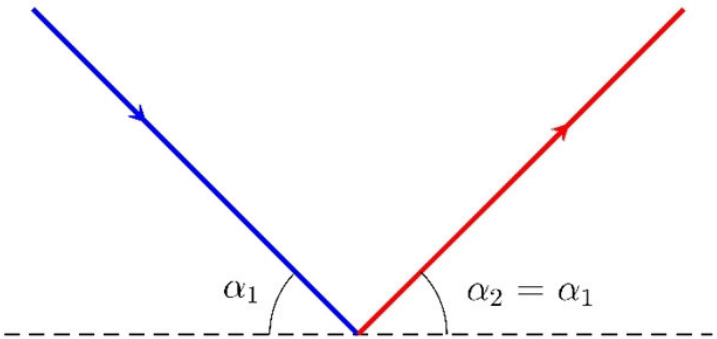
Implementierung Spiel

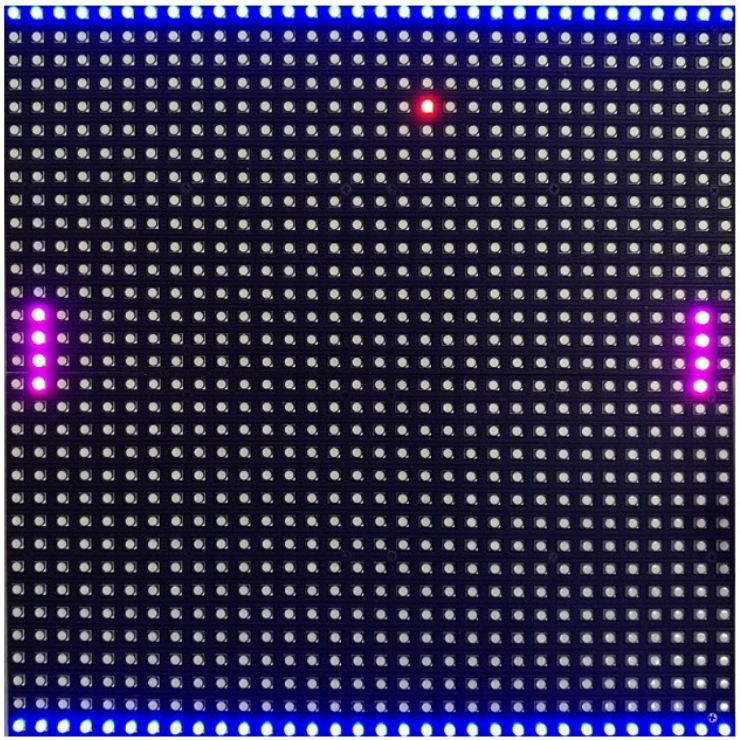
- Spielball
- Ballbewegung
- Spielschläger

```
def gamepaddle1(yAxis,ypos):
   paddlelength=int((yAxis+1))/8
       double_buffer.SetPixel(1, ypos, 255, 0, 255)
       for i in range(1, int(paddlelength)):
       double_buffer.SetPixel(1, ypos+i, 255, 0, 255)
```

Implementierung Spiel

- Spielball
- Ballbewegung
- Spielschläger
- Kollisionserkennung





FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Implementierung Spiel

- Spielball
- Ballbewegung
- Spielschläger
- Kollisionserkennung
- Punkt

Implementierung Spiel

- Spielball
- Ballbewegung
- Spielschläger
- Kollisionserkennung
- Punkt
- Spielstand

```
graphics.DrawText(double_buffer, font, xpos, 8, blue, pointsresult)
```

Implementierung Spiel

- Audioeffekte

```
pygame.init()
def playsoundeffects():
    pygame.mixer.music.load('ping_pong_8bit_plop.ogg‘)
    pygame.mixer.music.play(0)
```

Implementierung Spiel

- Audioeffekte
- Spielende
  - Ausgangsposition

Implementierung Spiel

- Audioeffekte
- Spielende
  - Ausgangsposition
- Bildaufbau

Implementierung Spiel

- Audioeffekte
- Spielende
  - Ausgangsposition
- Bildaufbau

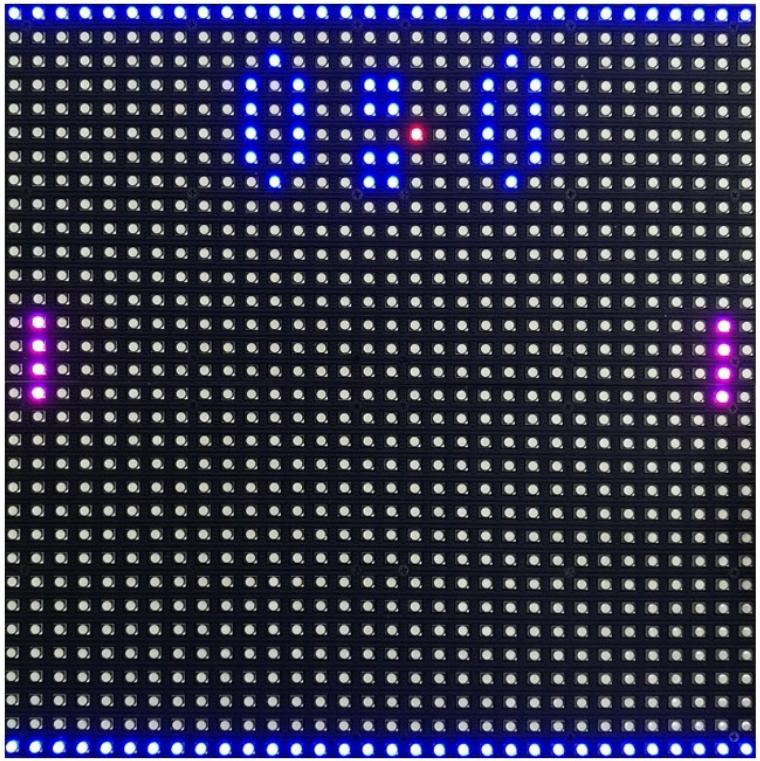
```
while True:
    matrix.Clear()
    matchfield (horizontalleds, verticalled)
    double_buffer = matrix.SwapOnVSync(double_buffer)
    time.sleep(0.04)
```

Implementierung Spiel

- Audioeffekte
- Spielende
  - Ausgangsposition
- Bildaufbau
- Parallele Prozesse

Implementierung Spiel

- Audioeffekte
- Spielende
  - Ausgangsposition
- Bildaufbau
- Parallele Prozesse
- Spiel ausführen



Fazit

- Ziel erreicht
  - Ansteuerung der RGB-Matrix untersucht
  - Anforderungen umgesetzt
  - Spiel ist lauffähig

Fazit

- Ziel erreicht
  - Ansteuerung der RGB-Matrix untersucht
  - Anforderungen umgesetzt
  - Spiel ist lauffähig
- Hindernisse
  - Wenige brauchbare Quellen

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Fazit

- Ziel erreicht
  - Ansteuerung der RGB-Matrix untersucht
  - Anforderungen umgesetzt
  - Spiel ist lauffähig
- Hindernisse
  - Wenige brauchbare Quellen
- Idee
  - RGB-Matrixen als Bildschirmersatz

Quellen

- http://artofcircuits.com/wp-content/uploads/2016/05/Raspberry-Pi3-3.jpg