Grundlagen der Informatik II:

Projektdokumentation: „SNAKE“

Namen und Matrikelnummer:

Ismet Akman: 4141105

Murathan Güler: 4446161

Abgabedatum:

29.07.2018

Inhaltsverzeichnis

Bildverzeichnis

1 Beschreibung des Projekts

In diesem Projekt soll das berühmte Retrospiel „Snake“, mit Hilfe des Arduinos programmiert werden. Der Score soll durch Python in einer GUI angezeigt werden. Die Kommunikation zwischen Arduino und Python findet per Wlan Shield statt. Hierbei soll Python der Server und Arduino der Client sein. In das Spiel startet man als ein Pixel großer Punkt, welches auf der LED-Matrix aufleuchtet und im Verlaufe des Spieles sich in eine „Schlange“ entwickeln soll. Wenn die Schlange sich vorwärts bewegt, somit keiner der Buttons gedrückt wird, wächst sie nicht. Wenn man jedoch auf einen der Buttons drückt, bewegt sich die Schlange dementsprechend nach links oder nach rechts und wächst um einen Pixel. Das Spiel verliert man erst, wenn die Schlange mit sich selber kollidiert oder wenn die Schlange gegen die „Wand“ läuft. Die Niederlage wird durch das ertönen des Buzzers und durch das verschwinden der Pixel deutlich gemacht.

2 Ansatz

Zuallererst haben wir uns überlegt, mit welchem Prinzip wir das Spiel aufbauen und gestalten wollen. Da es viele Möglichkeiten hierfür gibt, haben wir uns zuallererst mit einigen verschiedenen Methoden auseinander gesetzt. Nachdem wir uns für eine Variante entschieden haben, fingen wir an uns über die Funktionen der benötigten Materialien zu informieren, somit wir das ganze System besser verstehen lernen und die Bauteile richtig einbauen und verschalten können. Daraufhin recherchierten wir jedes einzelne wichtige Bauteil, welches Bestandteil der Schaltung sein sollte. Somit setzten wir uns mit den zwei Hauptbestandteilen unseres Aufbaus auseinander, welches die 8x8 LED-Matrix und der Shiftregister sind. Um die Funktionen der LED-Matrix zu verstehen versuchten wir sie im allgemeinen anzusteuern. Nachdem wir die Ansteuerung der LED-Matrix und das Prinzip des shiftens vom Shifter verstanden haben, konnten wir uns dem vollständigen Aufbau des Spieles widmen. Die weiteren Vorgehensweisen waren es, den Code für das Spiel in Arduino und den Score in Python zu codieren. Anschließend soll die Verbindung zwischen beiden Programmen per Wlan Shield hergestellt werden.

3 Projektmaterialien und Aufbau des Schaltplans

3.1 Projektmaterialien

- Arduino Uno Board (1x)
- Wlan Shield (1x)
- Breadboard (1x)
- 8x8 LED-Matrix (1x)
- Button small (2x)
- Buzzer (1x)
- Resistor (10x)
- Shiftregister 74HC595 (1x)
- Jumper Wire (39x)

3.2 Aufbau des Schaltplans

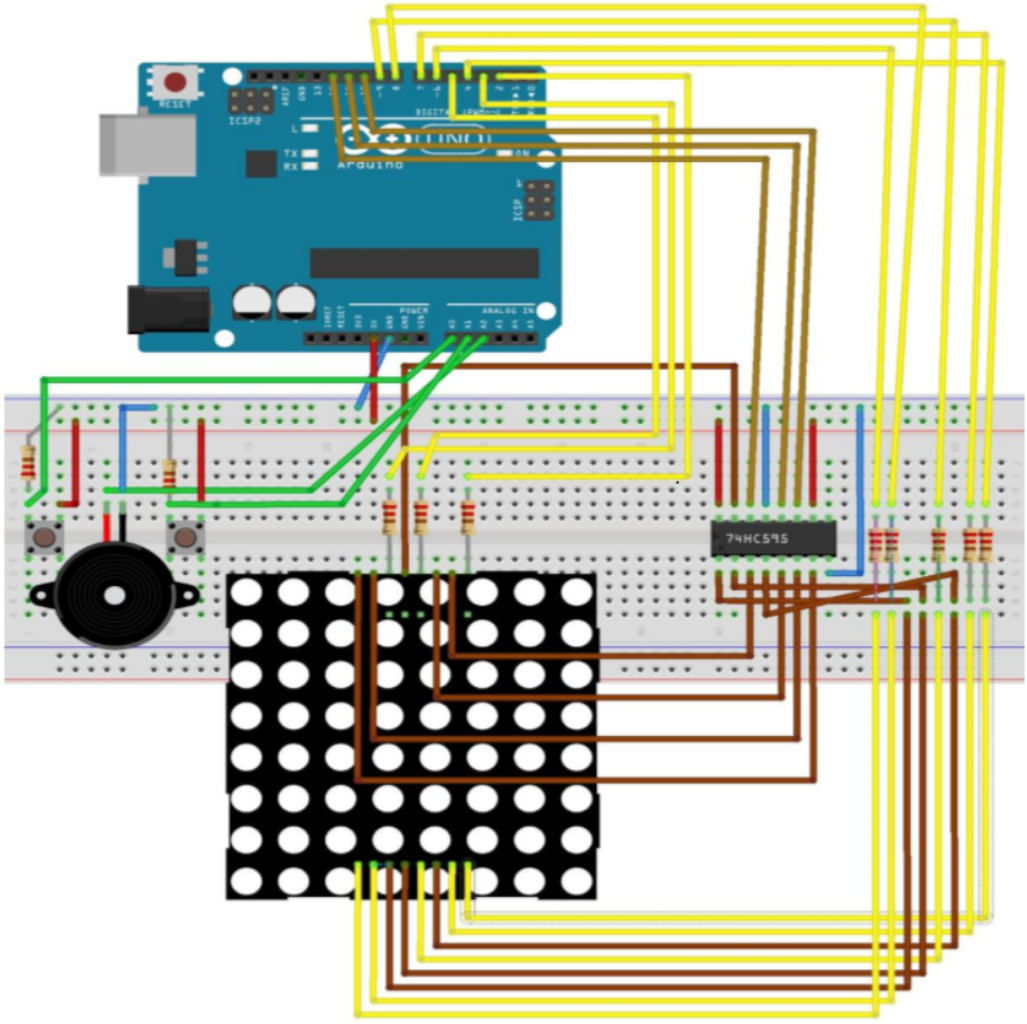

Abbildung 1: „Aufbau des Schaltplans“

4 Codebeschreibungen

4.1 Codebeschreibung Arduino

Als erstes binden wir die benötigten Dateien für die Serververbindung mit „#include“ in den Code ein. Als nächstes initialisieren wir die Größe der Variablen und die Variablen der Pins für das Shiftregister. Die Variablen werden als Shiftregister clock input(sh_reg_clk_inp), serial data input(serial_data) und storage register clock input(store_reg_clk_inp) definiert. Die LED-Matrix selbst und die Pins die mit der LED-Matrix verbunden sind, werden ebenfalls definiert. Außerdem werden hier noch die Pins der beiden Buttons definiert. Die darauf folgenden drei Befehle sorgen dafür, dass die Schlange in einer zufälligen Richtung startet, sich in einem Intervall von 1000ms bewegt und eine Größe von einem Pixel hat. Die darauffolgenden beiden Arrays „int Zeile“ und „int Spalte“ dienen dazu, dass genug Zwischenspeicher vorhanden ist und stellen jeden Schritt der Schlange dar.
In der Funktion „void knopfdruck()“ werden beide Knöpfe so definiert, dass wenn man z.B. den rechten Knopf drückt, eine 300 ms lange Pause entsteht, so dass kein weiterer INPUT mehr eingehen kann. Als Resultat auf den Knopfdruck verändert die Schlange ihre Richtung nach rechts und ihre Länge wird um einen Pixel vergrößert. Wenn die Richtung der Schlange größer als drei ist, wird sie wieder auf Null gesetzt und die Schlange läuft nach oben (0=Hoch, 1=Rechts, 2=Runter, 3=Links). Der linke Knopf ist dementsprechend prinzipiell analog zum rechten,nur dass hier die Schlange nach links beim betätigen des linken Knopfes läuft und dass die Richtung nur auf Null gesetzt wird, wenn sie kleiner als Null ist.
Des weiteren sorgt die Funktion „void shifter“ dafür, dass die dementsprechenden Bits mit Hilfe der Values definiert werden. Anschließend werden die Bits gesetzt und zuletzt wird durch das Umschalten auf HIGH ein Bit geshiftet, wobei ein Bit an den Ausgang übergeben wird. Dieser Vorgang konnte geschehen, da der Shifter zu Beginn einen Anfangszustand (sh_reg_clk_inp) und den Zwischenspeicher (store_reg_clk_inp) auf LOW hat. Wenn sh_reg_clk_inp auf HIGH gesetzt wird,wird der anliegende vorher definierte Bit in den Shifter rein geschoben. Anschließend wird store_reg_clk_inp auf HIGH gesetzt, wodurch das Bit geshiftet wird.
Der Befehl „void buzzer()“ sorgt dafür, dass der Buzzer für eine Sekunde ertönt, um die Niederlage deutlich zu machen.

In der Funktion „void setup ()“ werden die sh_reg_clk_inp, serial_data, store_reg_clk_inp, der buzzer und die mit der LED-Matrix direkt verbundenen Pins als OUTPUT definiert. Die beiden Buttons „linkerknopf“ und „rechterknopf“ hingegen werden als INPUT definiert, da sie Signale empfangen sollen, je nachdem ob eines der Buttons gedrückt wird oder nicht. Die kommenden zwei Arrays legen den Startpunkt fest. Ein weiterer wichtiger Bestandteil der Funktion ist, das Einrichten und das Starten des Serial Ports. Zudem werden die Daten für die Verbindung mit dem Wlan Shield im „esp_server.begin(&....., „Host“, „Passwort“, „Port“)“ eingetragen.
Die letzte Funktion „void loop()“ beinhaltet das Spielgeschehen. Zuerst wird die Spielzeit in Millisekunden registriert. Hierbei können mehrere Fälle auftreten. Im ersten Fall geht die Schlange nach oben, sobald man gegen eine Wand läuft, schaltet sich die Matrix aus und der Buzzer ertönt. Im zweiten Fall läuft die Schlange nach rechts und auch hier ertönt der Buzzer nur wenn die Schlange gegen eine Wand läuft. Im dritten Fall ist es genau die selbe Situation, abgesehen davon, dass die Schlange nach unten läuft. Im letzten Fall läuft die Schlange nach links, der Buzzer ertönt wieder nur, wenn die Schlange gegen eine Wand läuft. Der nächste Befehl sorgt dafür, dass die Schlange nicht größer wird, wenn sie sich gerade aus bewegt. Dies geschieht, weil die Pixel hinter der Schlange gelöscht und erneut auf LOW gesetzt werden. Im weiteren Verlauf des Codes durchläuft eine „for-schleife“, die Pixel der Schlange. Wenn der Kopf der Schlange in der neuen Zeile und Spalte mit einem der vorherigen Teile der Schlange kollidiert wird die Matrix ausgeschaltet und der Buzzer ertönt. Zuletzt wird überprüft, ob es einen Knopfdruck gibt und die Daten werden zum Shifter übertragen.

4.2 Codebeschreibung WLAN-Shield

Zuallererst soll eine Verbindung zum Hotspot hergestellt werden. Hierfür verwenden wir die Funktion void EspServer::begin. Mithilfe dieser Funktion wird die Verbindung zum Hotspot erstellt. Als Hotspot wird ein Handy verwendet, welches die SSID:“Mura“ und das Passwort:“*********“ hat und den Port:“30303“ besitzt. Mit dem nächsten Befehl flush_in_buff() kann man den Buffer lesen und die Werte später im Code benutzen. In der nächsten Funktion connect_wifi() wird das richtige Protokoll benutzt, damit die Verbindung erfolgt. Als letzte wichtige Funktion dient setup_server(), womit ein Server gebaut wird, dass später für den Client benutzt wird. Eine feste Verbindung zwischen dem Client und des Servers ist eingerichtet, damit man spielen kann. Um die Verbindung herzustellen wurden in einer Datei (serverTCP.h) die ganzen Funktionen aufgelistet, welche in der Datei (ServerTCP.cpp) definiert werden.

4.3 Codebeschreibung Python

Im Code snakeSchnittstelle.py werden zuerst zwei Module importiert, welche notwendig sind, um die Verbindung zum Wlan und die GUI herzustellen. Anschließend definieren wir die Funktion „class Arduino(object)“, in der die Verbindung zwischen Arduino und Python hergestellt wird. Die Funktion beinhaltet die Erstellung der GUI, die Verbindung zum Arduino per Socket und das lesen des Buffers. Desweiteren wird mit der Funktion „def periodic_socket_check(self)“ dafür gesorgt, dass die Werte die eingehen periodisch überprüft werden. Damit der Inhalt für die GUI erstellt wird, wird „class SnakeWindow(object)“ definiert. Die Verbindung zum Wlan Shield erfolgt durch die Eingabe vom „host“ und vom „port“, welche sich in der Funktion „class SnakeWindow(object)“ befindet. Die GUI, welche mithilfe des Moduls „tkinter“ aufgebaut wird, wird in der Funktion „def setup_window(self) initialisiert. Ein wichtiger Bestandteil ist die Erstellung der Labels, wo der Score definiert und angezeigt wird. Dies erfolgt in der Funktion „def setup_content(self), hier wird unter anderem das Aussehen der Labels angepasst. Damit die GUI nicht geschlossen wird, werden die beiden Befehle „window = SnakeWindow()“ und „window.run()“ durchgeführt.

Im folgenden Screenshot ist ein Beispiel der GUI, welches einen erzielten Score anzeigt:

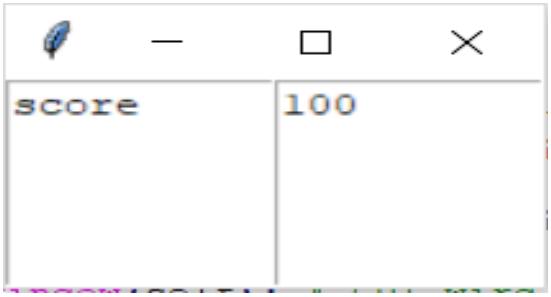

Abbildung 2: „Beispiel der GUI“