
STEUERUNG EINES SERVOMOTORS MIT EINEM KEYPAD UND ARDUINO ÜBER BLUETOOTH-KOMMUNIKATION

DAS ZIEL DIESES PROJEKTS IST ES ,EINEN SERVOMOTOR
MIT HILFE EINES KEYPADS ZU STEUERN

PROJEKTZUSAMMENFASSUNG

Das Ziel dieses Projekts ist es, einen Servomotor in einem bestimmten Winkel mithilfe eines Keypads zu steuern, der an einen Arduino Mega 2560 angeschlossen ist. Die Kommunikation erfolgt drahtlos über Bluetooth zu einem Arduino Due, der den Servomotor antreibt. Das Projekt integriert Hardware- und Softwarekomponenten und betont die Bluetooth-Kommunikation, die Arduino-Programmierung und die Verwendung externer Peripheriegeräte wie das Keypad und den Servomotor.

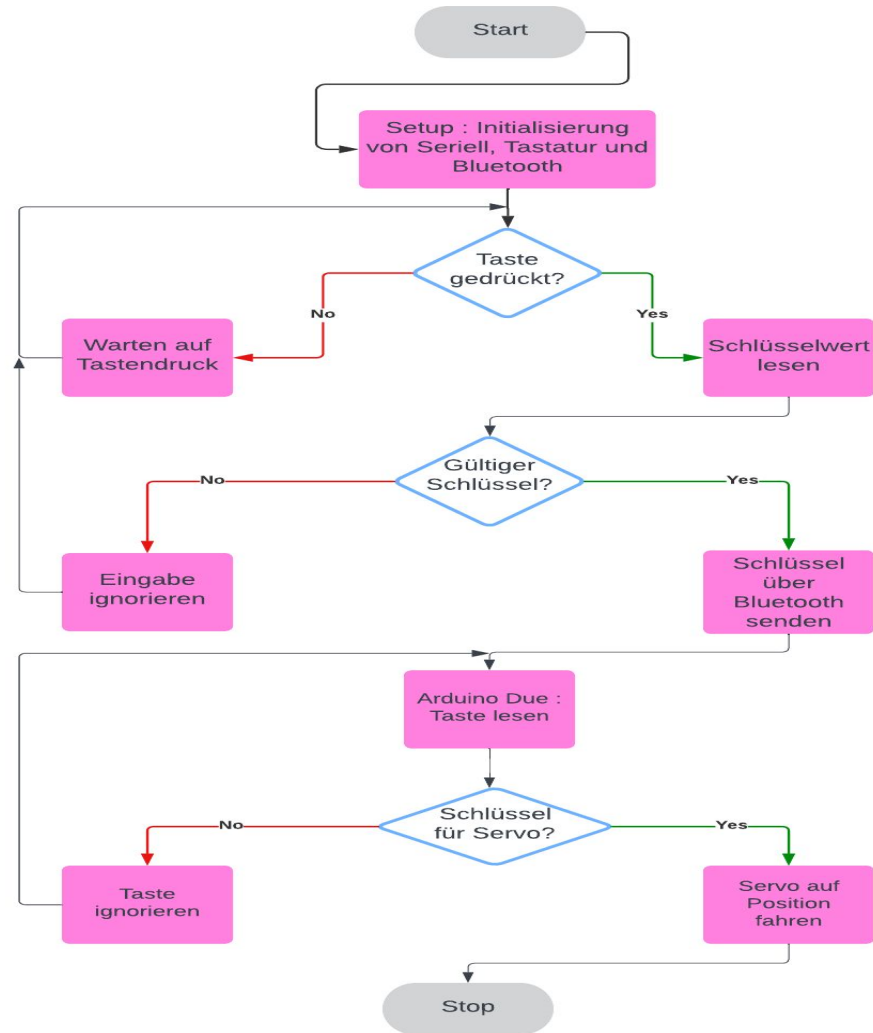
Hardware-Komponenten

- Arduino Mega 2560 (Controller)
- Arduino Due (Device)
- 4x4 Keypad
- Micro Servo Motor (SG90)
- 2 x HC-05 Bluetooth-Module
- Steckbrett
- Jumper-Kabel
- Stromversorgung

Software-Komponenten

- Arduino IDE
- PlatformIO
- Visual Studio Code

Programmablaufplan



PROJEKT SCHRITTE

A. Hardware-Verbindungen

1. Arduino Mega 2560 (CONTROLLER):

- Keypad-Verbindungen:

- Verbindung des Keypad-Reihen und -Spalten mit digitalen Pins am Arduino Mega (D2 bis D9).

- Bluetooth-Modul (HC-05) Verbindungen:

- VCC an 3.3V
- GND an GND
- TX an RX1 (D19)
- RX an TX1 (D18)

- Strom- und Erdungsverbindungen:

- Der 5V-Pin am Arduino Mega mit der positiven Schiene auf dem Steckbrett.
- Die GND-Pins am Arduino Mega mit der Erdungsschiene auf dem Steckbrett.

2. Arduino Due (DEVICE):

- Servo-Motor-Verbindungen:

- Signalkabel an digitalen PWM-Pin (D10)
- Stromkabel an 5V auf dem Steckbrett
- Erdungskabel an GND auf dem Steckbrett

- Bluetooth-Modul (HC-05) Verbindungen:

- VCC an 3.3V
- GND an GND
- TX an RX1 (D19)
- RX an TX1 (D18)

- Strom- und Erdungsverbindungen:

- Den 5V-Pin am Arduino Due mit der positiven Schiene auf dem Steckbrett.
- Die GND-Pins am Arduino Due mit der Erdungsschiene auf dem Steckbrett.

B. Konfiguration der Bluetooth-Module

- Controller-Modul (HC-05 am Arduino Mega) konfigurieren:

1. AT-Befehl und konfigurierung:

- AT+ROLE=1 (Als Controller einstellen)
- AT+CMODE=0 (Mit einem bestimmten Gerät verbinden)
- AT+PAIR=<98d3,34,90fc38> (Mit dem Device-Modul koppeln)
- AT+BIND=<98d3,34,90fc38> (Mit dem Device-Modul binden)
- AT+LINK=<98d3,34,90fc38> (Verbindung herstellen)

- Device-Modul (HC-05 am Arduino Due) konfigurieren:

1. AT-Befehl und Konfigurierung:

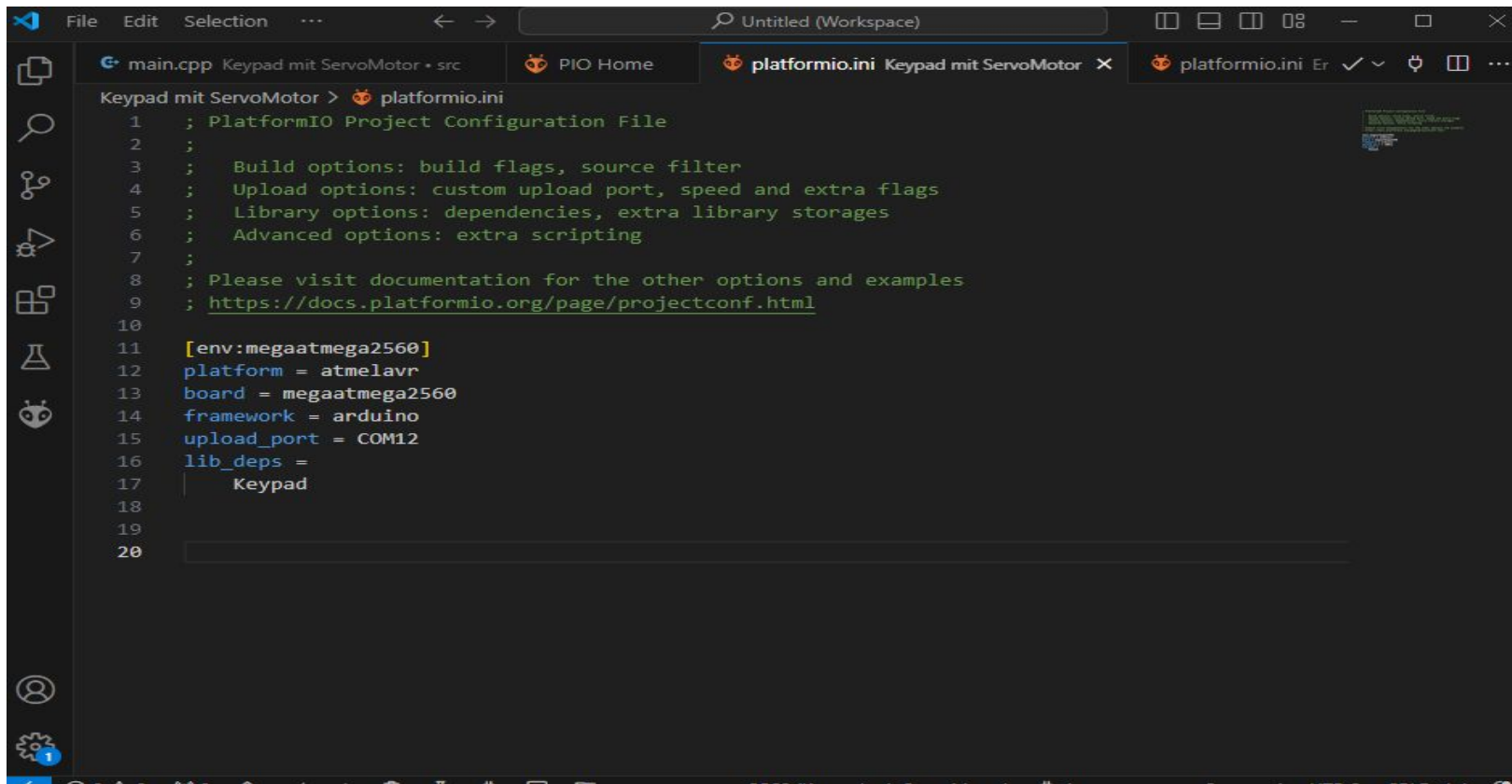
- AT+ROLE=0 (Als Device einstellen)
- AT+ADDR? (Adresse des Device-Moduls abrufen)

C. Software-Implementierung

1. Arduino Mega 2560 (CONTROLLER) Code:

- Keypad-Eingaben lesen.
- Befehle über Bluetooth an den Arduino Due senden.

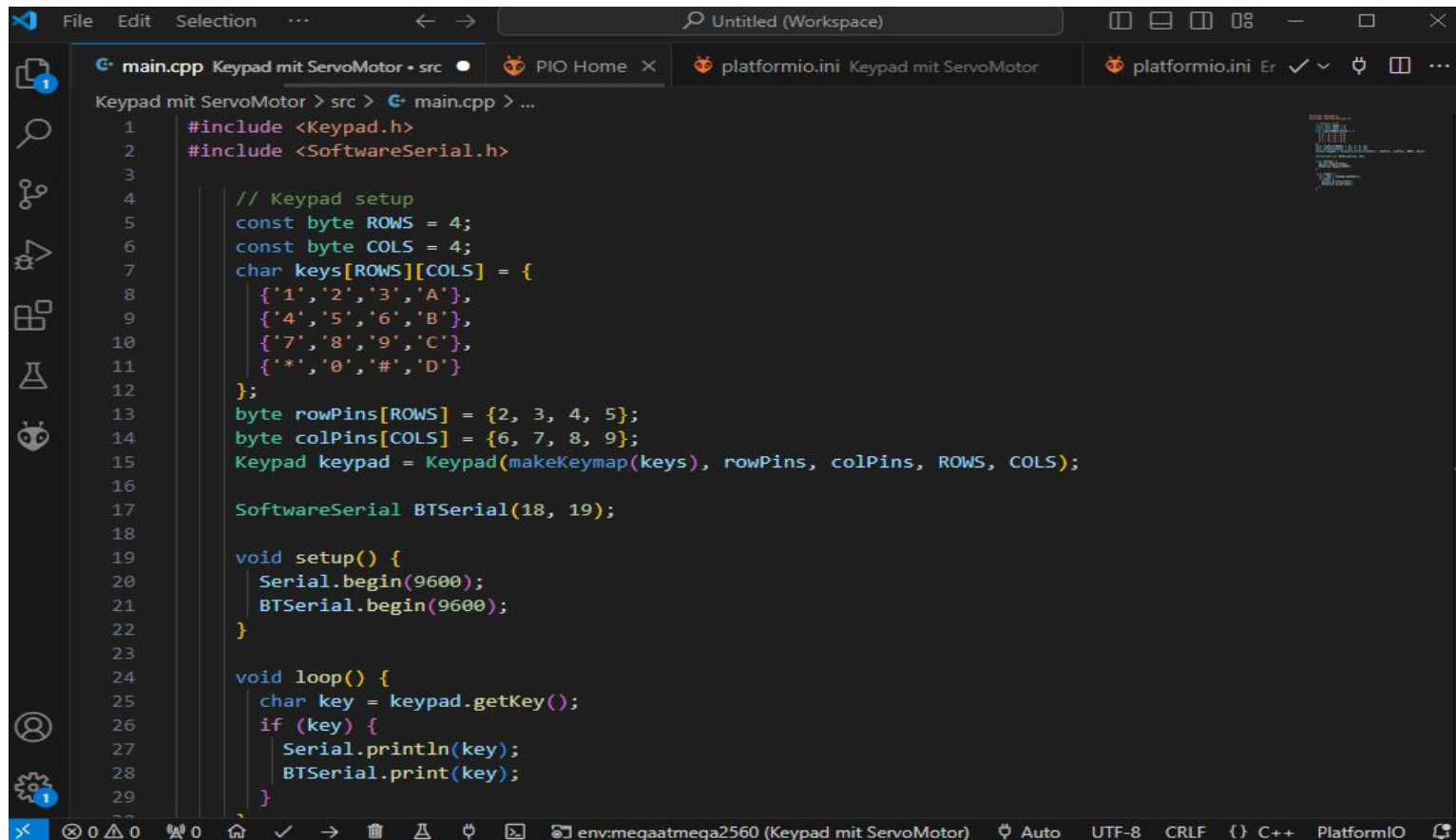
PLATFORMIO.INI



The screenshot shows the PlatformIO IDE interface. The top menu bar includes File, Edit, Selection, and a search bar. The workspace title is 'Untitled (Workspace)'. The left sidebar contains icons for Explorer, Search, Source Control, Run and Debug, Extensions, Test Explorer, and a user profile icon. The main editor area displays the 'platformio.ini' file for the project 'Keypad mit ServoMotor'. The file content is as follows:

```
1 ; PlatformIO Project Configuration File
2 ;
3 ; Build options: build flags, source filter
4 ; Upload options: custom upload port, speed and extra flags
5 ; Library options: dependencies, extra library storages
6 ; Advanced options: extra scripting
7 ;
8 ; Please visit documentation for the other options and examples
9 ; https://docs.platformio.org/page/projectconf.html
10
11 [env:megaatmega2560]
12 platform = atmelavr
13 board = megaatmega2560
14 framework = arduino
15 upload_port = COM12
16 lib_deps =
17     Keypad
18
19
20
```


MAIN.CPP

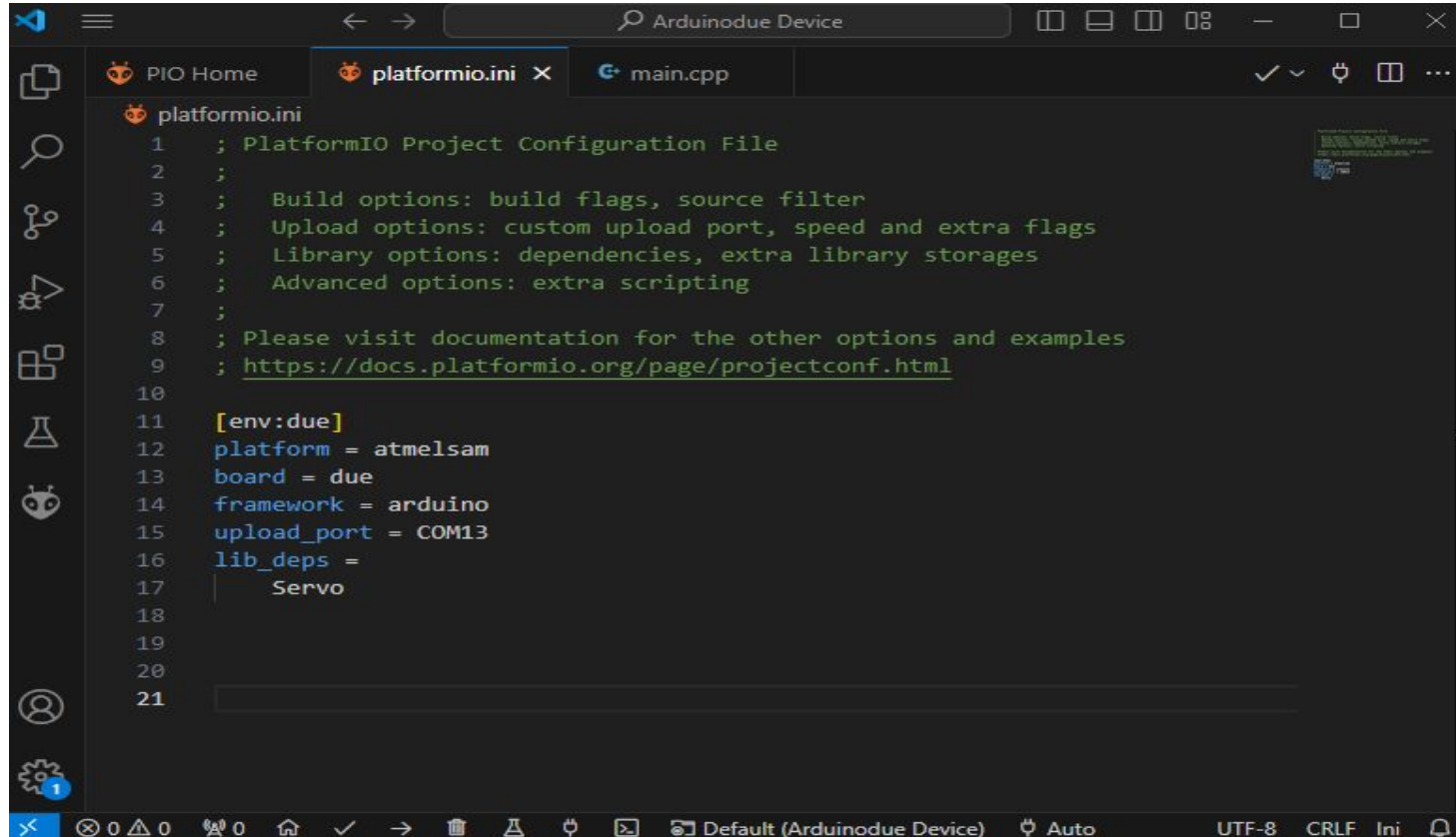


```
1  #include <Keypad.h>
2  #include <SoftwareSerial.h>
3
4  // Keypad setup
5  const byte ROWS = 4;
6  const byte COLS = 4;
7  char keys[ROWS][COLS] = {
8      {'1','2','3','A'},
9      {'4','5','6','B'},
10     {'7','8','9','C'},
11     {'*','0','#','D'}
12 };
13 byte rowPins[ROWS] = {2, 3, 4, 5};
14 byte colPins[COLS] = {6, 7, 8, 9};
15 Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);
16
17 SoftwareSerial BTSerial(18, 19);
18
19 void setup() {
20     Serial.begin(9600);
21     BTSerial.begin(9600);
22 }
23
24 void loop() {
25     char key = keypad.getKey();
26     if (key) {
27         Serial.println(key);
28         BTSerial.print(key);
29     }
```

Arduino Due (DEVICE) Code:

- Befehle über Bluetooth empfangen.
- Den Servomotor basierend auf den empfangenen Befehlen steuern.

PLATFORM.INI



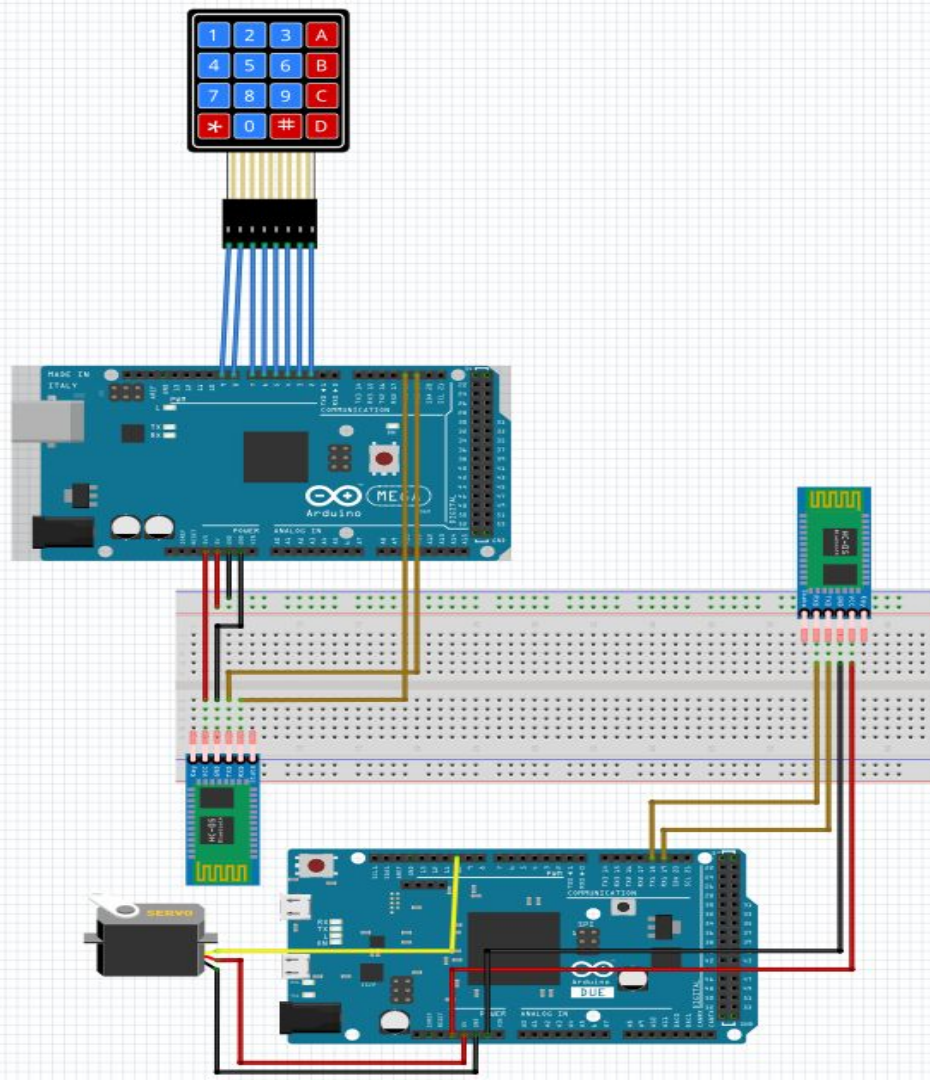
The screenshot shows the Arduino IDE interface with the 'platformio.ini' file open. The file is a PlatformIO Project Configuration File for an Arduino Due board. It includes build options, upload options, library options, and advanced options. The configuration is set for the 'due' board using the 'atmelsam' platform and 'arduino' framework. The upload port is set to 'COM13'. The library dependencies include 'Servo'. The file is saved as 'platformio.ini' and the main.cpp file is also visible in the background.

```
1 ; PlatformIO Project Configuration File
2 ;
3 ; Build options: build flags, source filter
4 ; Upload options: custom upload port, speed and extra flags
5 ; Library options: dependencies, extra library storages
6 ; Advanced options: extra scripting
7 ;
8 ; Please visit documentation for the other options and examples
9 ; https://docs.platformio.org/page/projectconf.html
10
11 [env:due]
12 platform = atmelsam
13 board = due
14 framework = arduino
15 upload_port = COM13
16 lib_deps =
17 | Servo
18
19
20
21
```

src > main.cpp > loop()

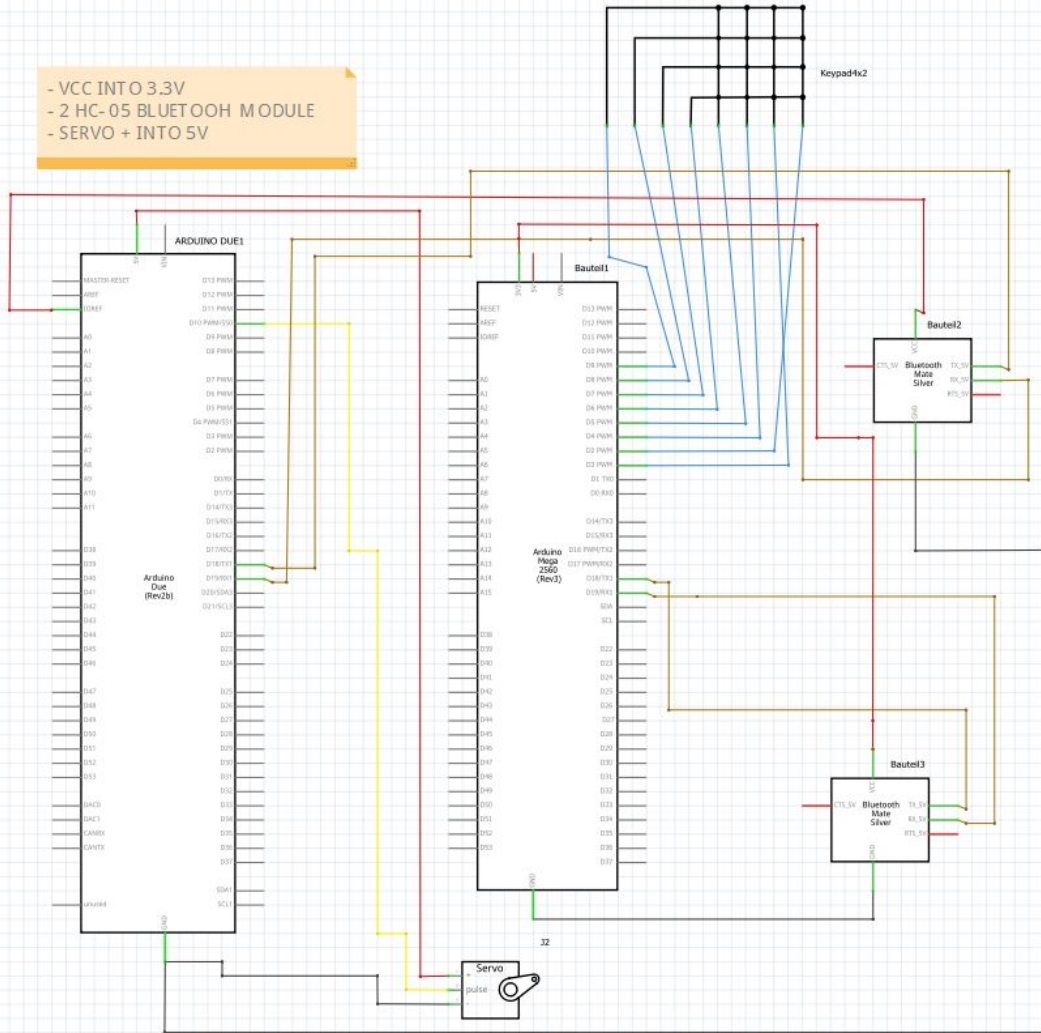
```
1  #include <Servo.h>
2  #include <Arduino.h>
3
4  Servo myServo;
5  int pos = 0;
6  HardwareSerial &BTSerial = Serial1;
7
8  void setup() {
9      Serial.begin(9600);
10     BTSerial.begin(9600);
11     myServo.attach(10);
12 }
13
14 void loop() {
15     if (BTSerial.available()) {
16         char key = BTSerial.read();
17         Serial.println(key);
18         switch (key) {
19             case '1':
20                 pos = 0;
21                 break;
22             case '2':
23                 pos = 15;
24                 break;
25             case '3':
26                 pos = 30;
27                 break;
28             case '4':
29                 pos = 45;
30                 break;
31             case '5':
32                 pos = 60;
33                 break;
34             case '6':
35                 pos = 90;
36                 break;
37             case '7':
38                 pos = 120;
39                 break;
40             case '8':
41                 pos = 150;
42                 break;
43             case '9':
44                 pos = 180;
45                 break;
46         }
47         myServo.write(pos);
48     }
```

STECKPLATINE

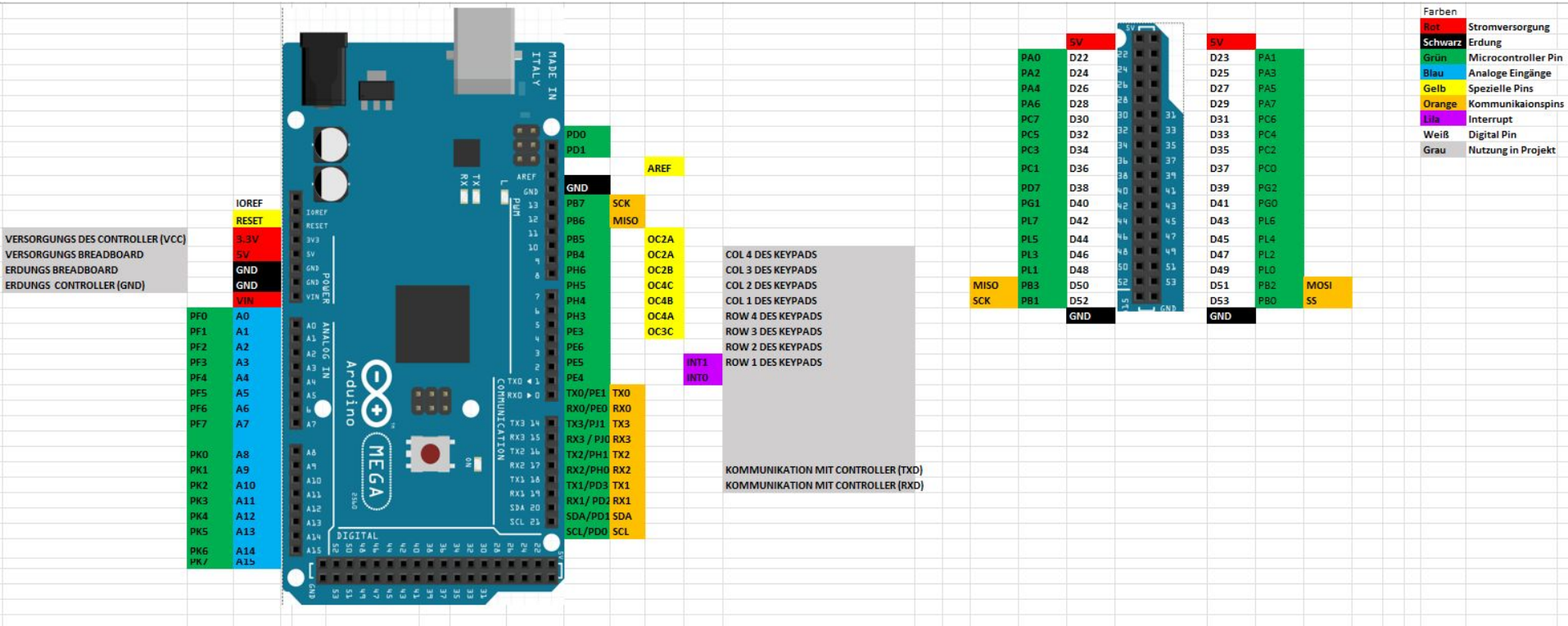


SCHALTPLAN

- VCC INTO 3.3V
- 2 HC-05 BLUETOOTH MODULE
- SERVO + INTO 5V



Arduino-Mega Pinout

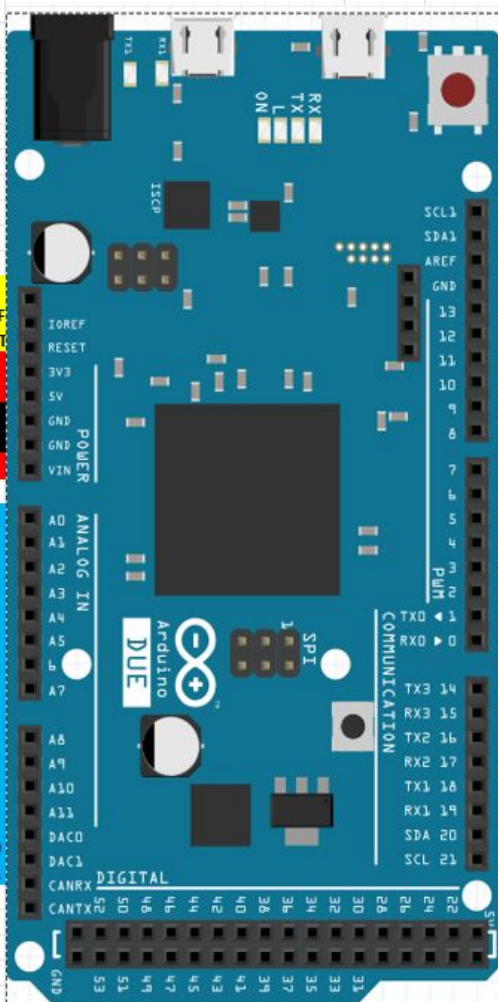


VERSORGUNG DES DEVICE (VCC)
 VERSORGUNG DES SERVOMOTORS
 ERDUNG FÜR SERVO MOTOR
 ERDUNG DES DEVICE

CANRX
 CANTX

PA16 A0
 PA24 A1
 PA23 A2
 PA22 A3
 PA6 A4
 PA4 A5
 PA3 A6
 PA2 A7
 PB17 A8
 PB18 A9
 PB19 A10
 PB20 A11
 PB15 DAC0
 PB16 DAC1
 PA1
 PA0

NC
 IOREF
 RESET
 3V3
 5V
 GND
 GND
 VIN



D21 PA18
 D20 PA17
 AREF
 GND
 D13 PD9 SCK
 D12 PD8 MISO
 D11 PD7 MOSI
 D10 PC29
 D9 PC21
 D8 PC22
 D7 PC23
 D6 PC24
 D5 PC25
 D4 PC26
 D3 PC28
 D2 PB25
 D1 PA9 TX0
 D0 PA8 RX0
 D14 PD4 TX3
 D15 PD5 RX3
 D16 PA13 TX2
 D17 PA12 RX2
 D18 PA11 TX1
 D19 PA10 RX1
 D20 PB12 SDA
 D21 PB13 SCL

STEUERUNG DES SERVOMOTORS

KOMMUNIKATION MIT DEVICE (RXD)
 KOMMUNIKATION MIT DEVICE (TXD)

Farben
 Rot Stromversorgung
 Schwarz Erdung
 Grün Microcontroller Pin
 Blau Analoge Eingänge
 Gelb Spezielle Pins
 Orange Kommunikationspins
 Weiß Digital Pin
 Grau Nutzung in Projekt

5V
 5V
 D22
 D24
 D26
 D28
 D30
 D32
 D34
 D36
 D38
 D40
 D42
 D44
 D46
 D48
 D50
 D52
 GND
 CANRX
 CANTX
 GND
 D23 PA14
 D25 PD0
 D27 PD2
 D29 PD6
 D31 PA7
 D33 PC2
 D35 PC3
 D37 PC6
 D39 PC8
 D41 PC10
 D43 PC12
 D45 PC14
 D47 PC16
 D49 PC18
 D51 PC20
 D53 PC22

Versuchsaufbau

