

Das Projekt

In dem hier vorgestellten Projekt soll die einfache Ansteuerung einer bunten RGB-LED realisiert werden. Die Steuerung soll über eine Webseite oder über eine Smartphone-App möglich sein. Die Verbindung kann sowohl über einen Router, und damit aus der Ferne über das Internet, als auch über den Access Point des Moduls hergestellt werden. Es ändert sich jeweils nur die zu adressierende IP.

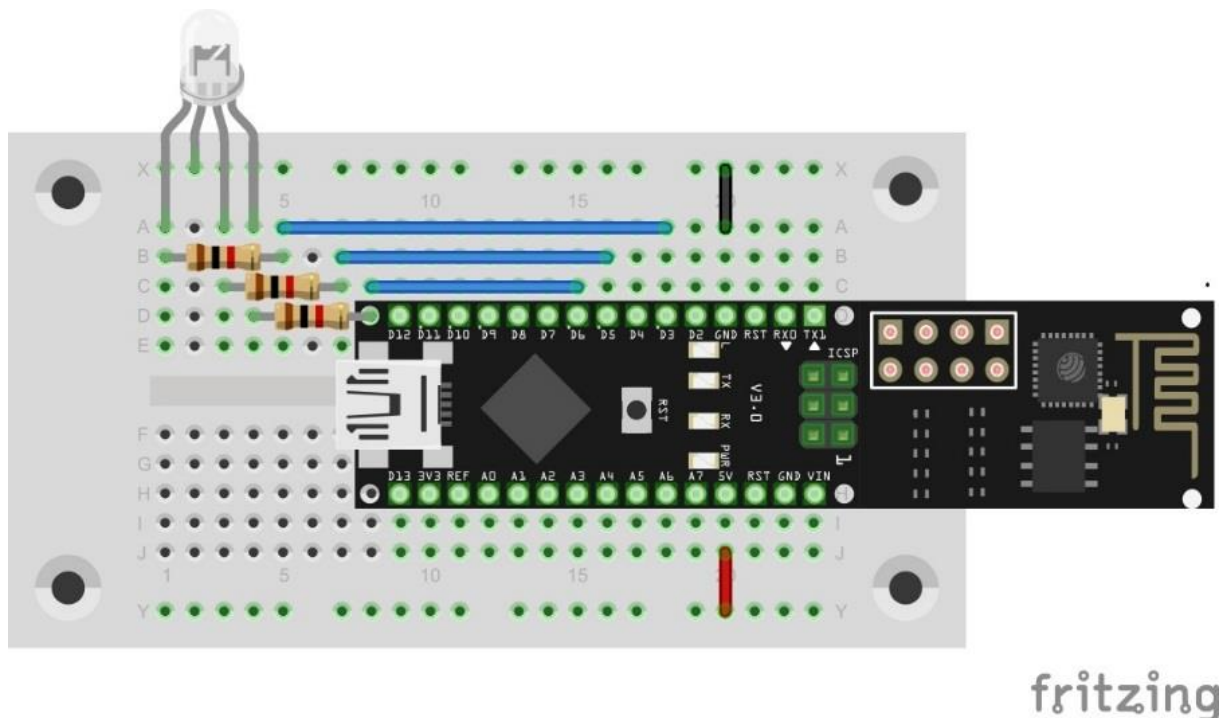


Bild: Der Aufbau der Schaltung

Die Programmierung erfolgt über die Arduino IDE, die sich jeder kostenlos von der Seite <http://iot.fkainka.de/> herunterladen kann. Als Board kann einfach das Arduino-Nano-Board ausgewählt werden (ATmega328p Version). Bevor das Programm hochgeladen wird, müssen die WLAN-Daten am Anfang des Programms eingetragen werden.

Nach dem Upload verbindet sich das Board automatisch mit dem Router und die LED D3 leuchtet zur Bestätigung. Über den seriellen Monitor der Arduino-Umgebung wird nun die vom Router an das Modul vergebene IP ausgegeben. Wenn sich der PC im gleichen Netzwerk befindet, kann man einfach über die IP des Moduls auf die im Quelltext hinterlegte Webseite zugreifen.

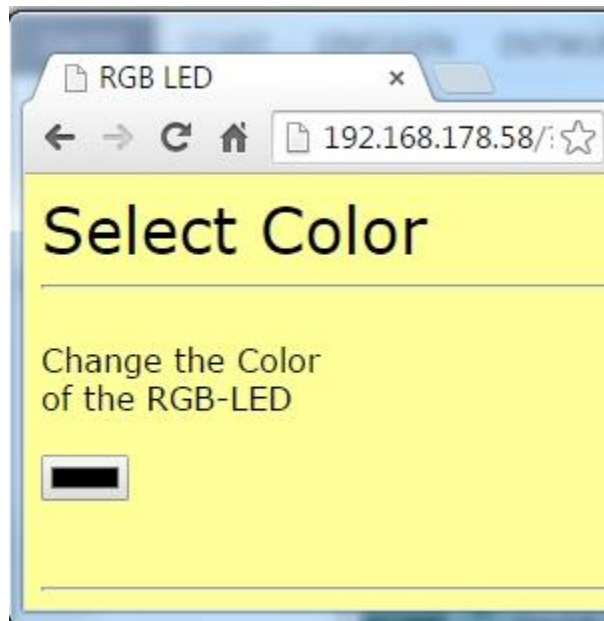


Bild: Die Webseite in einem Browser auf dem PC

Auf der Webseite wird man aufgefordert, eine Farbe zu wählen. Als Eingabeelement kommt hier das HTML5-Element ColorPicker zum Einsatz. Nach dem Auswählen wird die Farbe in der URL verschlüsselt übermittelt und kann einfach vom Board ausgewertet werden.

Gleichzeitig zu der Verbindung zum Router hat das Modul auch sein eigenes Netzwerk mit dem Namen NanoESP erstellt, mit dem man sich nun z.B. mit einem Smartphone verbinden kann. Das Passwort und der Name des Netzwerks wurden vorher im Arduino-Programm festgelegt. Die Vergabe einer IP an ein neues Gerät funktioniert vollautomatisch. Die IP des Moduls in seinem selbst erstellten Netzwerk lautet 192.168.4.1. Nach dem Verbinden mit einem Smartphone kann man über diese IP die auch für Smartphones optimierte Webseite des Moduls aufrufen und über das ColorPicker-Element die Farbe auswählen.

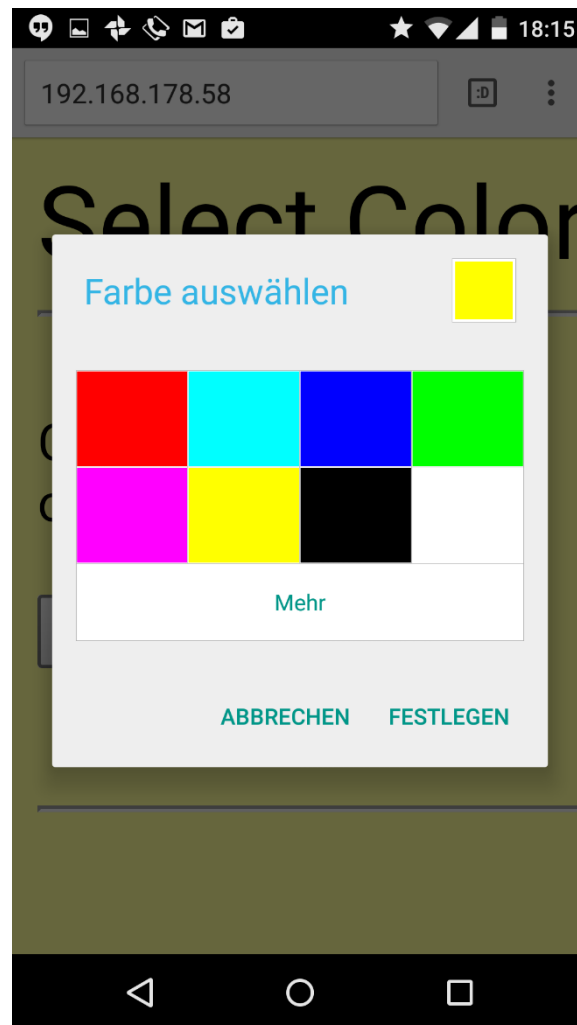


Bild: Das ColorPicker-Element im Android-5.0-Browser

Das Projekt lässt sich gut zu einer bunten Lampe erweitern, die auch aus der Ferne bedient werden kann. Mithilfe einiger weniger zusätzlicher Bauteile kann man die Konstruktion z.B. zu einer Designerlampe im Haus bis hin zu einer beleuchteten Präsentation auf einer Messe erweitern. Die Steuerung kann über die verschiedensten Geräte bedient werden, z.B. auch mit einer Smartphone-App. Mehr Informationen sowie Bauanleitung und Quelltext finden Sie auf der Internetseite:

<http://iot.fkainka.de>



Bild: Lampe

Doch das Pretzel-Board kann noch mehr. Zu den bis jetzt mit dem Board realisierten Projekten gehören unter anderem eine Uhr, die sich über das Internet automatisch stellt, eine Universalfernbedienung mit Webinterface, eine Messstation mit Cloud-Anbindung, Ansteuerung über Twitter und vieles mehr. Die Arduino-Oberfläche sorgt dafür, dass alle Programme leicht verständlich und einsteigerfreundlich bleiben.

Auszug aus dem Arduino-Programm:

```
#include <NanoESP.h>
#include <SoftwareSerial.h>
/*
TCP-Server for RGB-Control
Change SSID and PASSWORD.
RGB-Led (common cathode) on Pins 3,5,6
*/

#define SSID "[Your SSID]"
#define PASSWORD "[Your Password]"

#define LED_WLAN 13

#define RED 3
#define GREEN 5
#define BLUE 6
#define GND 4

#define DEBUG true
```

```
//Website
const char site[] PROGMEM = {
  "<HTML><HEAD>\n<link rel=\"icon\" href=\"data:;base64,iVBORwOKGgo=\">\n<meta
name=\"viewport\" content=\"width=device-width, initial-scale=2.0, user-
scalable=yes\">\n<title>\nRGB LED\n</title>\n</HEAD>\n\n<BODY bgcolor=\"#FFFF99\"
text=\"#000000\">\n<FONT size=\"6\" FACE=\"Verdana\">\nSelect
Color\n</FONT>\n\n<HR>\n<BR>\n<FONT size=\"3\" FACE=\"Verdana\">\nChange the
Color<BR>\nof the RGB-LED\n<BR>\n<BR>\n<form method=\"GET\">\n\n<input type=\"color\"
name=\"rgb\"
onchange=\"this.form.submit()\"><BR>\n</form>\n<BR>\n<HR>\n\n</font>\n</HTML>\n0"
};
```

```
NanoESP esp8266 = NanoESP();
```

```
void setup() {
  Serial.begin(19200);
  esp8266.init();
```

```
  pinMode(RED, OUTPUT);
  pinMode(GREEN, OUTPUT);
  pinMode(BLUE, OUTPUT);
```

```
  pinMode(GND, OUTPUT);
  digitalWrite(GND, LOW);
```

```
//Dual Wifi Mode
  esp8266.configWifi(STATION, SSID, PASSWORD);
  esp8266.configWifi(ACCESSPOINT, "NanoESP", "");
```

```
//Start UDP and TCP Server
  if ( esp8266.startUdpServer(0, "192.168.255.255", 90, 90)) debug("UDP Server Aktiv"); else
  debug("UDP Server Error");
  if ( esp8266.startTcpServer(80)) debug("TCP Server Aktiv"); else debug("TCP Server Error");

  debug(esp8266.getIp());
  digitalWrite(LED_WLAN, HIGH);
}
```

```
void loop() {
  int client = esp8266.getId();
```

```
  if ( client >= 0 )
  {
    debug("Request id:" + String(client));
```

```
    if (esp8266.findUntil("?rgb=", "\n"))
    {
      debug("RGB Data");
      String hexstring = esp8266.readStringUntil(' ');
      long number = (long) strtol( &hexstring[3], NULL, 16); //Convert String to Hex
http://stackoverflow.com/questions/23576827/arduino-convert-a-string-hex-ffffffff-into-3-int
```

```
// Split them up into r, g, b values
int r = number >> 16;
int g = number >> 8 & 0xFF;
int b = number & 0xFF;

analogWrite(RED, r);
analogWrite(GREEN, g);
analogWrite(BLUE, b);
}

if (client > 0) // if client > 0 must be TCP-Request (because client 0 = UDP Server)
{
    if (esp8266.sendDataClose(client, createWebsite())) debug("Website send OK"); else
debug("Website send Error");
}
}
}

String createWebsite()
{
    String xBuffer;

    for (int i = 0; i <= sizeof(site); i++)
    {
        char myChar = pgm_read_byte_near(site + i);
        xBuffer += myChar;
    }

    return xBuffer;
}

//-----Debug Functions-----

void debug(String Msg)
{
    if (DEBUG)
    {
        Serial.println(Msg);
    }
}
```