SŠPU
STŘEDNÍ ŠKOLA PRŮMYSLOVÁ
A UMĚLECKÁ, OPAVA

ZÁVĚREČNÁ STUDIJNÍ PRÁCE

dokumentace

Digitální hodiny s využitím LED pásku WS2812b

Filip Híreš

Obor:	18-20-M/01 INFORMAČNÍ TECHNOLOGIE se zaměřením na počítačové sítě a programování
Třída:	IT4
Školní rok:	2021/2022

Poděkování

Děkuji panu učiteli Ing. Petru Grussmannovi za cenné rady a pomoc při vývoji projektu, panu Mgr. Marcelu Godovskému za pomoc se součástkami a panu Mgr. Marku Lučnému za rady a pomoc s webovými technologiemi.

Prohlašuji, že jsem závěrečnou práci vypracoval samostatně a uvedl veškeré použité informační zdroje.

Souhlasím, aby tato studijní práce byla použita k výukovým účelům na Střední průmyslové a umělecké škole v Opavě, Praskova 399/8.

V Opavě 31. 12. 2021

podpis autora práce

ANOTACE

Projektem bylo vytvoření digitálních hodin pomocí čipu ESP8266 s wifi modulem a LED pásku WS2812b. Hlavní částí softwaru tvoří technologie pro přenos aktuálního času z NTP klienta a technologie WebSocket pro přenos dat. Programová část využívá jazyku Arduino, která vychází ze syntaxe jazyka C++. S projektem souvisí také webová aplikace tvořena samostatnou HTML stránkou s využitím kaskádových stylů, JavaScriptu i včetně jQuery.

Klíčová slova

ESP8266; WebSocket, NTP klient, LED pásek WS2812b, Wifi, Arduino

OBSAH

ÚVOD **5**
1 VYUŽITÉ TECHNOLOGIE **6**
2 HARDWARE **7**
2.1 SCHÉMA 7
2.2 LED PÁSEK WS2812B 7
3 POUŽITÉ TECHNOLOGIE **9**
3.1 KNIHOVNA ESP8266WIFI 9
3.2 NTP KLIENT 9
3.3 ASYNCHRONNÍ SERVER 10
3.3.1 WebSocket 10
3.4 OTA – „OVER THE AIR“ 11
4 WEBOVÁ APLIKACE **12**
4.1 VÝPIS ČASU NA HTML STRÁNKU 13
4.2 JAS 14
4.3 BARVY A RGB COLOR-PICKER 15
5 VÝSLEDKY ŘEŠENÍ, VÝSTUPY, UŽIVATELSKÝ MANUÁL **16**
ZÁVĚR **17**
SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ **18**
SEZNAM PŘÍLOH **20**

ÚVOD

Rozhodl jsem se pro tento projekt, protože mám rád barvy, barevné efekty a celkové míchání barev, které mi umožňuje právě LED pásek WS2812b.

Cílem projektu však byla komunikace digitálních hodin s jakýmkoliv zařízení, které se umí připojit přes wifi, aby uživatel dostával přesný čas s minimálním opožděním, mohl si libovolně na digitálních hodinách měnit barvu zobrazení času i její intenzitu svítivosti. Protože se jedná o můj první projekt v této oblasti, odhodlal jsem se použít známé technologie, s kterými jsem se v praxi setkal, a o kterých se zmíním v další části dokumentace. Hardwarovou část tvoří čip ESP8266 s wifi modulem a LED pásek WS2812b.

Komunikace ESP8266 s NTP klientem, technologie, odkud čip dostává aktuální čas a samotnými hodinami je napsána v jazyce Arduino. Webovou část, tedy webovou aplikaci tvoří jazyky HTML5, JavaScript (JS) a kaskádové styly (CSS). Komunikace je zajištěna díky technologii WebSocket.

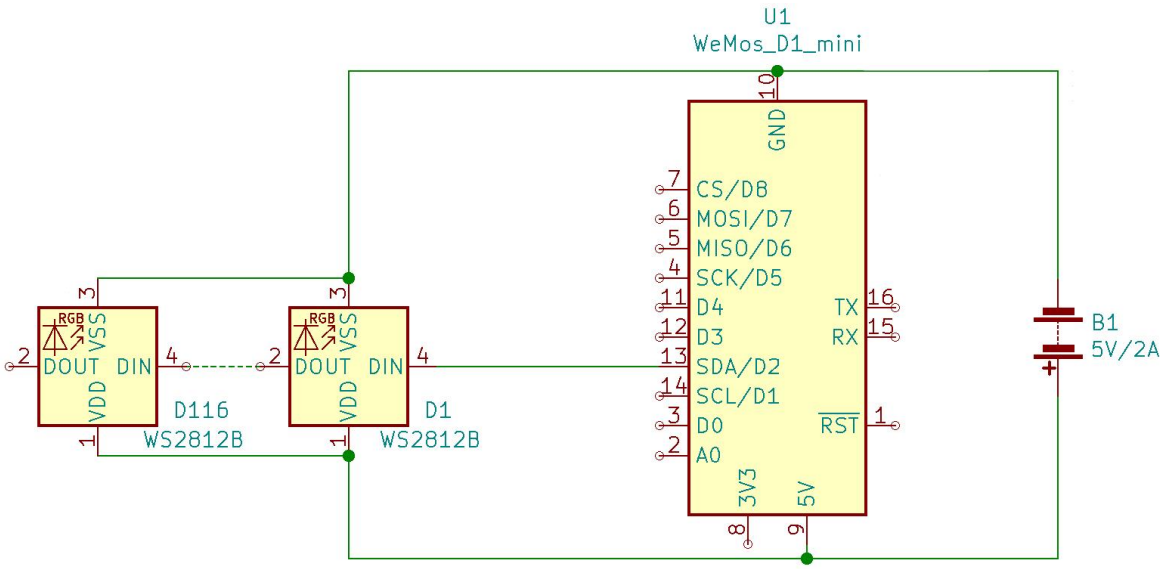
1 VYUŽITÉ TECHNOLOGIE

- Vývojová deska ESP8266 WeMos Mini D1
- LED pásek WS2812b
- Napájecí adaptér 5V/2A
- Visual Studio Code
- Platformio
- Bootstrap4
- CSS
- JavaScript
- jQuery
- Fritzing
- KiCad

2 HARDWARE

2.1 Schéma

Abych si mohl vyzkoušet, zda LED pásek opravdu funguje, zapojil jsem si součástky podle technického schémata. Napájení jsem poté zkusil i přes adapter s parametry 5V/2A.



Obrázek č. 1 Technické schéma

2.2 LED pásek WS2812b

Aby tento projekt byl realizovatelný, potřeboval jsem si pořídit LED pásek, který je datovatelný pro každou diodu zvlášť, to znamená, že na každou diodu je možné použít jinou barvu, jas či příkaz, který má vykonat. Tyto úkony mi umožnil pořízený LED pásek WS2812b.

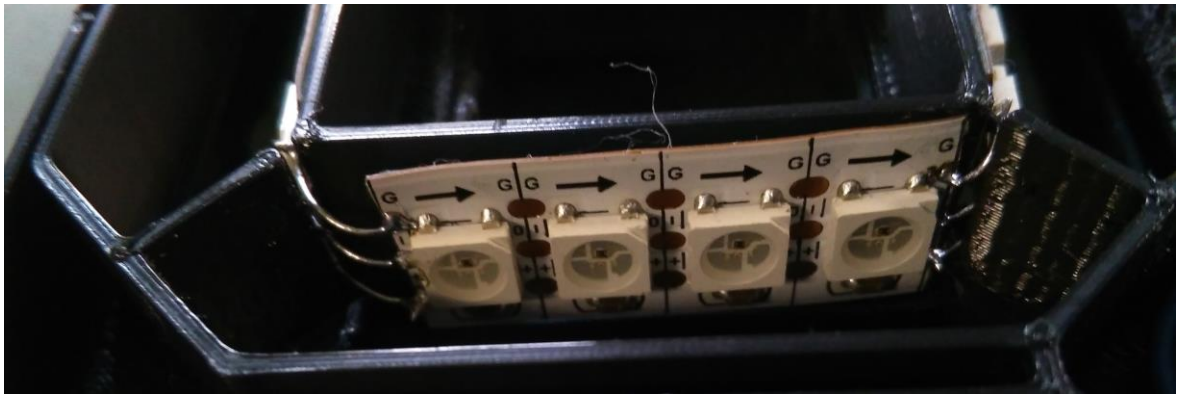
První má představa byla taková, že na každý jednotlivý segment použiji 5 diod. Jenže již vytištěný model hodin mi neumožňoval napojit více než 4 diody. Tím pádem jsem z 1 m pásku se 144 diodami využil 112 diod na všechny segmenty a 4 diody na dvojtečku po 2 diodách. Celkově jsem využil 116 diod a zbylo mi 28 diod rezervních.

Diody jsem nastříhal dle potřeby. Kvůli malým měděným plošným spojům jsem použil pájecí pastu a spoje jsem pocínoval. Následně jsem si nastříhal drátky na potřebnou velikost, nakalafunoval jsem je, pocínoval je a připájel je. Chtěl jsem použít známé barvy pro jednotlivé piny a zachovat tak barevnou integritu kvůli lepšímu přehledu nebo v případě

opravy hodin. Pásek obsahuje 3 piny z obou stran, jedná se o piny pro vstupní napětí 5V, data pin a GND pin. Proto pro vstupní napětí jsem použil drátek s červenou barvou, na data pin jsem využil zelený drátek a pro GND jsem se nakonec našel využití pro barvu modrou.

Když byly hodiny spájené a vložené do vytisknutého rastru, nastal čas pro vyřešení, které piny na desce připojím k pinům na LED pásku. Bylo mi jasné, že 116 diod nemohla deska utáhnout na plný výkon, v případě bílé barvy, proto jsem využil adaptér pro napájení LED pásku s hodnotami 5V/2A.

Následně mě čekala programovací část. Nejdříve jsem musel najít knihovnu, která podporuje adresaci ledek a celkově s nimi umí pracovat. V mém případě se jedná o knihovnu *fastled/FastLED@^3.4.0*. Knihovnu jsem nainstaloval pomocí příkazu pro platformio *pio lib install"fastled/FastLED"* a následně jsem tuto knihovnu inicializoval do souboru *ESPWS_8621_WS_DC.ino*. Abych zkontroloval, zda vše funguje, použil jsem zkušební funkce pro rozsvěcování jednotlivých diod.

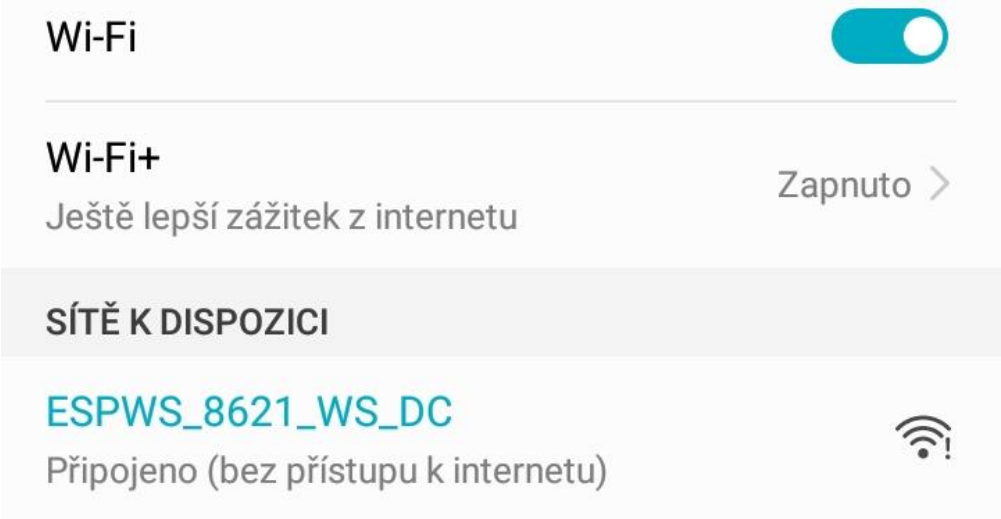


Obrázek č. 2 Detail spájených diod v rastru

3 POUŽITÉ TECHNOLOGIE

3.1 Knihovna ESP8266WiFi

Díky této knihovně jsem se mohl připojit na jakoukoliv wifi síť bez omezení či jakéhokoli zasáhnutí do kódu. Když je v kódu definována SSID a heslo libovolné wifi, na které se má ESP8266 připojit, tak se přes přístupový bod desky (access point) může připojit pomocí dynamické IP adresy, kterou je možné zjistit v konzoli, a zobrazit si tak HTML stránku s naprogramovanými nástroji. Pro větší eleganci je možné tuto IP adresu zadat do prohlížeče a připojit se na HTML stránku aplikace, i když nejsme připojeni přes přístupový bod desky.



Obrázek č. 3 Synchronizace s mobilem

3.2 NTP klient

Prvním krokem nejen pro webovou aplikaci bylo zajištění aktuálního času, které čip ESP8266 bude přijímat a následně odesílat na HTML stránku. Ke zprovoznění NTP klienta jsem upotřebil knihovnu a sice se jedná o knihovnu *arduino-libraries/NTPClient@^3.1.0.* Po zprovoznění knihovny jsem dále potřeboval připojení k Wifi, a proto jsem připojil knihovnu již známou pro Platformio *ESP8266WiFi.* Základem je SSID (název sítě) a heslo k samotné síti. Samozřejmostí je výpis času, já jsem použil čas pro Českou republiku pomocí tohoto příkazu:

```
NTPClient timeClient(ntpUDP, "cz.pool.ntp.org", 3600, 60000);
```

Čas jsem si vypisoval každou sekundu do konzole přes příkaz:
`Serial.printf("%s\n", formattedDate);`

Později jsem čas promítnul i na HTML stránku.

3.3 Asynchronní server

Dalším krokem bylo vypořádat se s úspěšnou komunikací mezi ESP8266 a zařízením, které se na něj připojí. Vyřešil jsem tento problém díky knihovnám *ESPAsyncWebServer* a *ESPAsyncTCP*, podporující asynchronní připojení na server. Tyto knihovny jsem opět inicializoval do hlavního souboru *ESPWS_8621_WS_DC.ino*. Následně už stačilo jen zadat hodnoty SSID a heslo, na kterou wifi se chci připojit. ESP8266 dostává automaticky od DHCP serveru v každé síti dynamickou IP adresu dle lokálního připojení. IP adresa je vypsána do konzole.

3.3.1 WebSocket

Nyní jsem znal IP adresu ESP8266, avšak přišel problém z hlediska připojení a ověření funkčnosti celé komunikace. Z mnoha různých technologií jsem si vybral WebSocket.

WebSocket je technologie, která zasílá celá data v jednotlivých paketech, avšak v případě, že data jsou příliš objemná, rozkouskuje zprávu do více paketů a vše pošle a zapouzdří tak, aby došlá data byla celistvá a úplná.

Abych ověřil, že skutečně WebSocket komunikuje s HTML stránku použil jsem jednoduchý kód. Pokud je propojení s WebSocketem úspěšné, tak klient dostane zprávu s jeho ID a je zaslán ping, zda komunikace probíhá:

```
if(type == WS_EVT_CONNECT){
  Serial.printf("ws[%s][%u] connect\n", server->url(), client->id());
  client->printf("Hello Client %u :)", client->id());
  client->ping();
}
```

Spojení je vypsáno do konzole:

```
Serial.printf("ws[%s][%u] connect\n", server->url(), client->id());
```

Avšak abych mohl posílat více informací zvláště soubory s HTML stránkou, která využívá přidružené soubory JavaScript a kaskádové styly, potřeboval jsem souborový systém. Aby i tyto soubory byly součástí projektu. Proto jsem využil knihovnu *SPIFFSEditor*, která je součástí nabídky platformia. Tato knihovna se stará o celý souborový systém:

```
server.serveStatic("/", SPIFFS, "/").setDefaultFile("index.html");
```

3.4 OTA – „over the air“

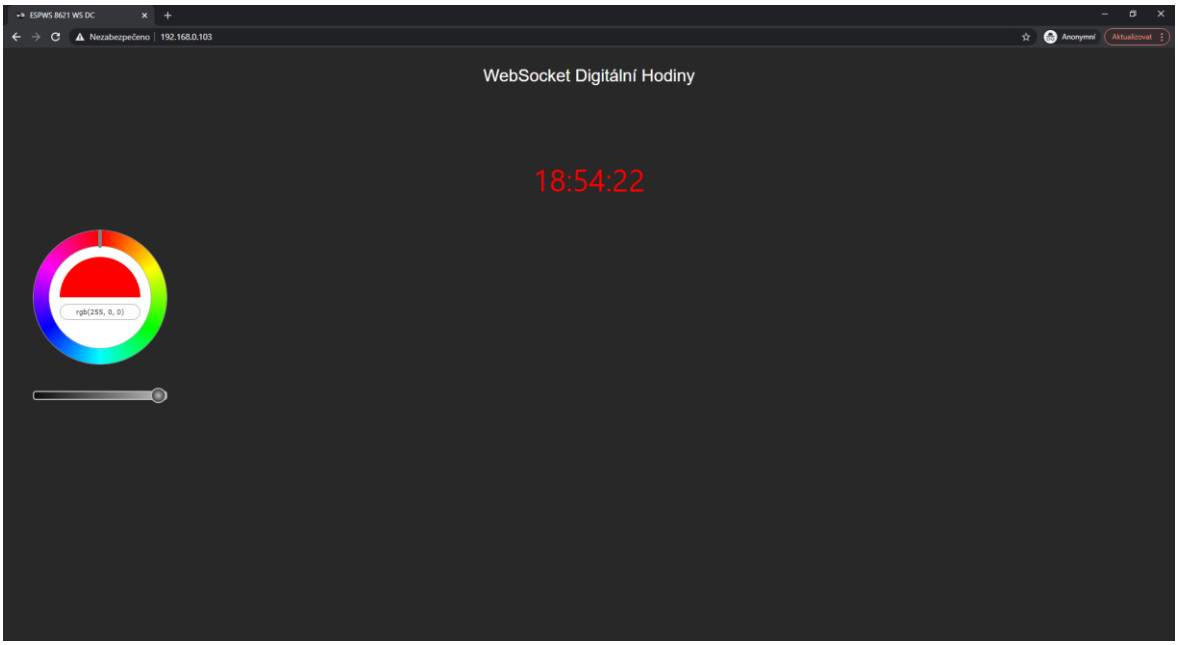
Jedná se o knihovnu, která je součástí balíčků platformia zajišťující šifrovanou bezdrátovou komunikaci v podobě jednoduchých zpráv a je spárovaná s WebSocketem. Výhodou technologie OTA je, že z jednoho centrálního místa může ESP8266 poslat aktualizaci všem uživatelům. Následující ukázka řeší zaslání webové aplikace, aby byla veřejně dostupná v lokální síti a zpětně přijímá událostí z webové stránky.

```
  /* Metoda pro uploadování souboru na server */
server.onFileUpload([](AsyncWebServerRequest *request, const String& filena-
me, size_t index, uint8_t *data, size_t len, bool final){
    if(!index)
      Serial.printf("UploadStart: %s\n", filename.c_str());
      Serial.printf("%s", (const char*)data);
    if(final)
      Serial.printf("UploadEnd: %s (%u)\n", filename.c_str(), index+len);
  });

  /* Metoda pro příjem informací ze stránky index.html do ESP */
  server.onRequestBody([](AsyncWebServerRequest *request, uint8_t *data,
size_t len, size_t index, size_t total){
    if(!index)
      Serial.printf("BodyStart: %u\n", total);
    Serial.printf("%s", (const char*)data);
    if(index + len == total)
      Serial.printf("BodyEnd: %u\n", total);
  });
  server.begin();
}
```

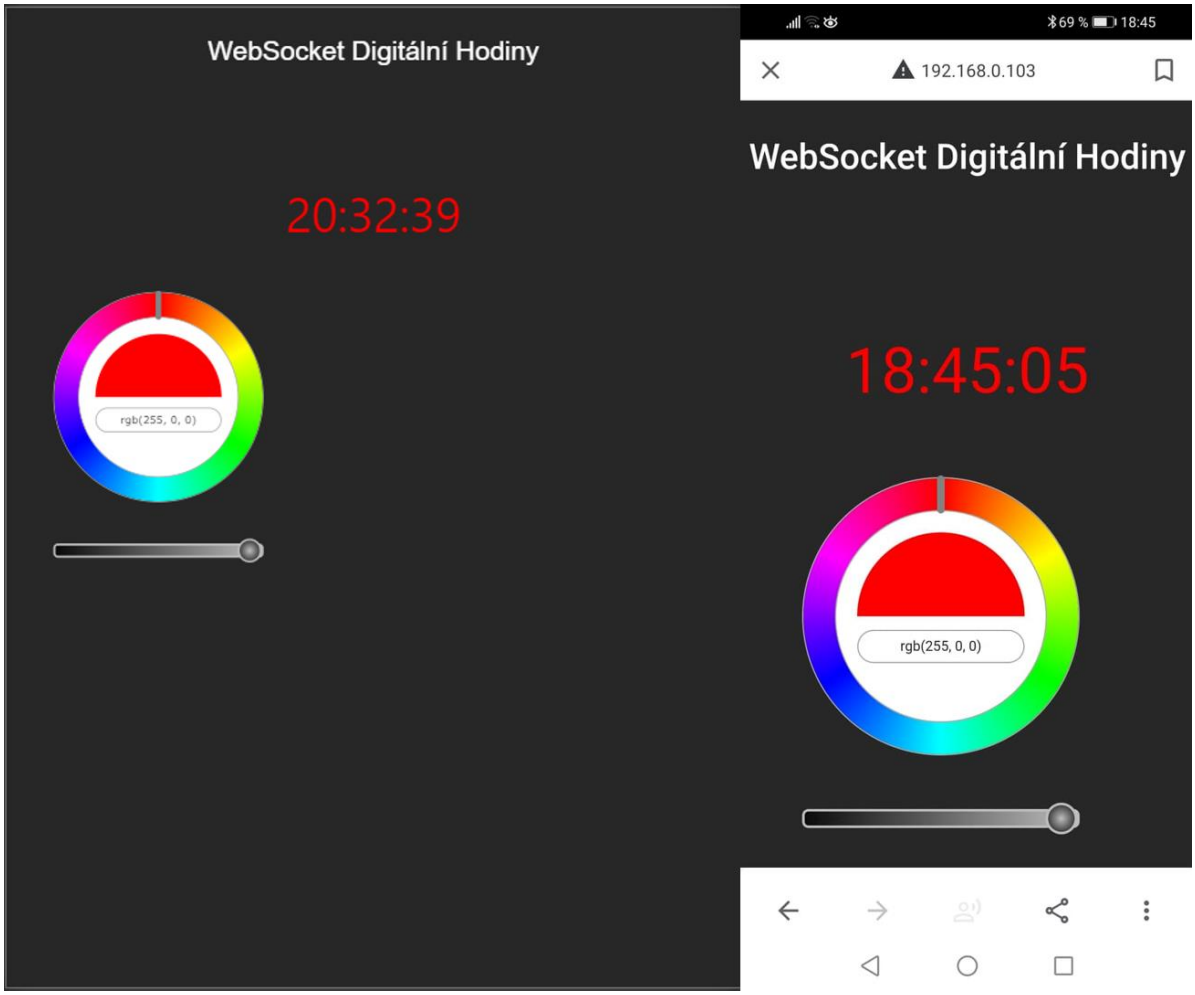
4 WEBOVÁ APLIKACE

Nejpodstatnější funkcí HTML stránky, která je původně určená pro mobilní zařízené, byla možnost rozšířeného ovládání hodin na dálku, vzhledu ovladače a byla uživatelsky příjemná. Ovládání v podobě volby barev ledek přes RGB picker a změnu jasu. HTML stránka zobrazuje aktuální čas, RGB picker a jas. Aby stránka fungovala, potřeboval jsem k tomu ještě další technologie v podobě JavaScriptu (JS), který se stará o přenos událostí z HTML stánky přes WebSocket do čipu, posílání aktuálního času do HTML stránky a posuvník pro jas. Javascriptová knihovna jQuery se stará o RGB color picker. Dále knihovnu JSON, která mi umožnila lepší manipulaci s daty v podobě JSON formátu. Nakonec přišly na řadu kaskádové styly (CSS), které se starají o celkový design stránky a komfortního čtení.

Pomocí souborového systému *SPIFFS* je HTML stránka a další přípojné soubory v HTML stránce součástí projektu. Samotná HTML stránka se nachází v adresáři data s názvem index.html. Lepší přehlednost a responzivitu mi zajistil framework Bootstrap 4, který je součástí HTML stránky.



Obrázek č. 4 Vzhled a responzivita stránky pro velkou obrazovku



Obrázek č. 5 Vzhled a responzivita stránky pro ipad (vlevo) a mobil (vpravo)

4.1 Výpis času na HTML stránku

Když se mi čas pěkně vypisoval do konzole, bylo na čase vložit čas i na webovou stránku. K tomu mi posloužil JavaScript a NTP klient. První část ukázky z kódu je z hlavního souboru *ESPWS_8621_WS_DC.ino*. Každou sekundu je aktualizovaný čas převeden do JSON formátu např.: {“time” : ”13:45:25”}

```
while(!timeClient.update()) {
    timeClient.forceUpdate();
  }
  formattedDate = timeClient.getFormattedTime();
  int splitT = formattedDate.indexOf("T");
  dayStamp = formattedDate.substring(0, splitT);
  timeStamp = formattedDate.substring(splitT+1, formattedDate.length()-1);
  StaticJsonDocument<200> doc;
  doc["time"] = dayStamp.c_str();
```

```
  serializeJson(doc, jsondata);
  events.send(jsondata.c_str());

delay(1000);
```

Druhá část ukázky je ze Javascriptového souboru *functions.js*, která pomocí knihovny JSON rozkouskuje data do proměnné *obj* z podoby {“time” : “13:45:25”} do podoby 13:45:25. Následně pomocí ID *#ntptime* pošle přes WebSocket čas a opět každou sekundu obnovuje čas a přemazává ho.

```
es.onmessage = function(e) {
        const obj = JSON.parse(e.data);
        $('#ntptime').empty().append($('<p/>').text(obj.time).html());
    };
```

Třetí a poslední část ukázky se nachází přímo v souboru *index.html*. Zde do ID je vkládán aktuální čas.

```
<div>
    <p id="ntptime"></p>
</div>
```

4.2 Jas

Podobně jako čas jsem řešil i jas světel. První část ukázky se nachází opět v souboru *ESPWS_8621_WS_DC.ino*. Kód přijímá hodnotu v podobě čísla a do konzole vypisuje hodnoty z posuvníku.

```
Serial.printf("Prijata data: ");
    AwsFrameInfo * info = (AwsFrameInfo*)arg;
    String msg = "";
    if(info->final && info->index == 0 && info->len == len){
Serial.printf("ws[%s][%u] %s-message[%llu]: ", server->url(), client->id(),
(info->opcode == WS_TEXT)?"text":"binary", info->len);
```

Druhá část ukázky je ze souboru *functions.js*. Funkce se stará o posuvník, který přijímá data z HTML stránky pomocí ID a hodnotu jasu posílá přes WebSocket do čipu. Pro ověření funkce jsem nastavil, že se jas (v mém případě průhlednost) promítne i ve výpisu času na webové aplikaci v podobě zprůhlednění.

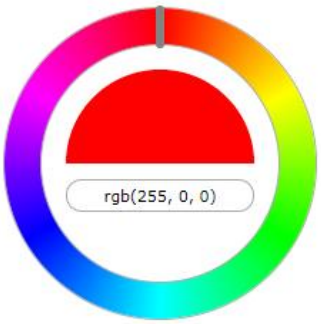
```
/* Jas LED pásku */
let brightness = document.getElementById("brightness");
let outputBrightness = document.getElementById("outputBrightness")

/* Funkce naslouchá posuvníků a zaznamenává jeho hodnoty, které následně
zašle skrz WebSocket */
brightness.addEventListener('change', function() {
    outputBrightness.innerHTML = this.value;
    ws.send(this.value);

/* Ovládání světelnosti na aktuálním čase a Color-pickeru v HTML stránce */
    $("#ntptime").css("opacity", (this.value / 100));
    console.log("Alfa: " + this.value);
})
```

Třetí část ukázky výpisu času se nachází v samotné HTML stránce. Tato část kódu řeší výchozí nastavení posuvníku a vložením správného ID se propojí již výše zmíněné funkce.

```
<div class="slidecontainer">
      <input type="range" min="1" max="100" value="100"
class="sliderBrightness" id="brightness">
      <div id="outputBrightness"></div>
</div>
```

4.3 Barvy a RGB color-picker

Aby aplikace využila i rozmanitosti voleb různých barev LED pásku, rozhodl jsem se vložit do webové aplikace i barevný posuvník pro volbu barvy. K tomu jsem potřeboval z Javascriptu knihovnu jQuery. Při posunutí či kliknutí na libovolnou barvu na RGB color-pickeru, jQuery zašle hodnotu RGB do čipu přes WebSocket. V rámci designu a zkoušky funkčnosti jsem nastavil, aby vybraná barva měnila barvu výpisu času na HTML stránce.



Obrázek č. 6 RGB Color-picker

5 VÝSLEDKY ŘEŠENÍ, VÝSTUPY, UŽIVATELSKÝ MANUÁL

V současné době je LED pásek uložený ve vytištěném plastovém rastru, aby působil skutečně jako digitální hodiny. Jelikož má původní představa o výrobě hodin, kde každý segment obsahuje 5 diod nevyšel, tak jsem použil 4 diody na každý segment a dvojtečka je tvořena 2 páry diod. Z celého 1metrového LED pásku se 144 diodami jsem použil jen 116 diod a zbytek jsem měl rezervních. Následně přišlo dlouhé pájení, které nepůsobí v rastru hodin efektivně, ale jen funkčně. K tomu všemu se může stát, že nějaký spájený spoj povolí, a tak se hodiny budou zobrazovat až do špatně spojeného místa spoje. Nakonec jsem náchylné spoje přilepil tavnou pistolí, aby se kontakty sebe nedotkly a od měděného spoje neodtrhly. Do budoucna by bylo lepší použít stejný LED pásek ovšem s větším rozestupem diod i na úkor jejich počtu.

Webová aplikace funguje a ovládá jas a RGB color-picker. RGB color-picker a posuvník jasu jsou aktivně využity na HTML stránce v podobě změny vybrané barvy či jasu aktuálního času přes CSS. Pro mě tyto funkce slouží jako kontrola funkčnosti, pro jiné to může být zpestřením webové stránky.

WebSocket funguje i s technologií OTA, přijímá data a zpětně je odesílá.

Co bych v budoucnu chtěl zlepšit z hlediska webové aplikace by určitě bylo nahrazení IP adresy díky DNS serveru, který by nahradil IP adresu čipu pro vyhledávač, a tak bych do vyhledávače napsal jen název aplikace, který by byl jedinečný a lépe zapamatovatelný než jen měnící se IP adresa např.: ESPWS_8621 nebo ESPWS_DC.

Další zajímavá vylepšení by mohla být ochrana ESP8266 v podobě naslouchání jiného portu např. na port 8621. Tak by mohla být zajištěna základní opatření vůči neoprávněného přístupu do hodin.

Přemýšlel jsem i o instalaci SSH (secure shell) na čip, abych jej mohl spravovat na dálku, a tak nezasahoval do konstrukce hodin.

Závěr

Cílem projektu bylo vytvořit LED hodiny, které budou zobrazovat aktuální čas a měnit barevnost pásku. Velmi zajímavá je i možnost ovládání hodin pomocí webového rozhraní, díky tomu je i vyřešen problém ovladače.

Neexistuje žádný projekt, který by se nedal vylepšit, a to platí i pro tento projekt. Možností vylepšení je mnoho. V budoucnu bych chtěl ze softwarové části hodiny více zabezpečit např. naslouchání na jiném portu, přidat SSH, a tak programovat čip „na dálku“, dodat i další funkce rozšířené o teplotu, vlhkost, formát hodin na 24hodinový a 12hodinový, možnost výpisu aktuálního data, obohatit hodiny o časovač, rozšíření funkcí v podobě počítadla, a hlavně možností si vybrat z veliké škály grafických efektů. Z hardwarové části bych chtěl vylepšit konstrukci hodin, dodat teplotní čidlo, čidlo vlhkosti či fotorezistor, aby se podle světla v místnosti měnil jas diod, nyní se vše musí nastavit přes webovou aplikaci.

GitHub: https://github.com/hiresfilip/ESPWS_8621_WS_DC

SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ

[1] Inteligentní RGB LED pásek 1m WS2812 NeoPixel (144led/m, 43W/m). Pájeníčko.cz [online]. Copyright © 2021, Pájeníčko s.r.o., Všechna práva vyhrazena [cit. 30.12.2021]. Dostupné z: https://pajenicko.cz/inteligentni-rgb-led-pasek-1m-ws2812-neopixel-144led-m-43w-m?search=ws2812&page=2

[2] The WebSocket API (WebSockets) - Web APIs | MDN. [online]. Copyright © 2005 [cit. 30.12.2021]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API

[3] WebSocket - Wikipedia. [online]. Dostupné z: https://en.wikipedia.org/wiki/WebSocket

[4] FastLED LED animation library for Arduino (formerly FastSPI_LED). FastLED LED animation library for Arduino (formerly FastSPI_LED) [online]. Dostupné z: https://fastled.io/

[5] GitHub - leonvandenbeukel/3D-7-Segment-Digital-Clock: 3D Printed 7 Segment Digital Clock with multi color LEDs. GitHub: Where the world builds software · GitHub [online]. Copyright © 2021 GitHub, Inc. [cit. 30.12.2021]. Dostupné z: https://github.com/leonvandenbeukel/3D-7-Segment-Digital-Clock

[6] S7ripClock - Basic Edition by parallyze - Thingiverse. Thingiverse - Digital Designs for Physical Objects [online]. Dostupné z: https://www.thingiverse.com/thing:4190016

[7] ntp.org: Home of the Network Time Protocol. ntp.org: Home of the Network Time Protocol [online]. Dostupné z: http://www.ntp.org/

[8] Network Time Protocol: Best Practices White Paper - Cisco. Cisco - Networking, Cloud, and Cybersecurity Solutions [online]. Copyright © [cit. 30.12.2021]. Dostupné z: https://www.cisco.com/c/en/us/support/docs/availability/high-availability/19643-ntpm.html

[9] Over-the-air programming - Wikipedia. [online]. Dostupné z: https://en.wikipedia.org/wiki/Over-the-air_programming

[10] Over-the-Air (OTA) - Technology - Thales. Redirecting to https://www.thalesgroup.com/en [online]. Dostupné z: https://www.thalesgroup.com/en/markets/digital-identity-and-security/technology/ota

[11] Yours for the making - Instructables [online]. Copyright ©C [cit. 31.12.2021]. Dostupné z: https://www.instructables.com/Retro-7-Segment-Clock-the-Final-Ones/

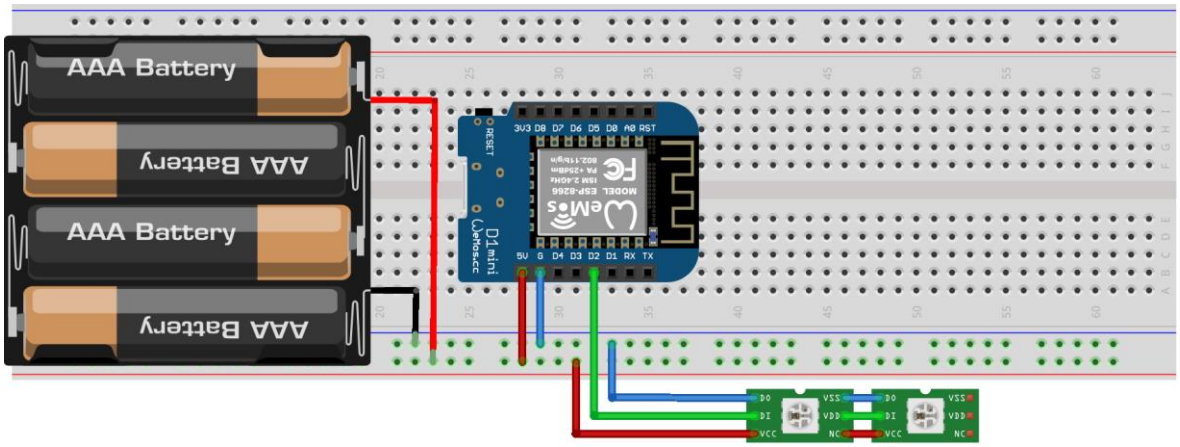
[12] Web Sockets - Zdroják. Zdroják - o tvorbě webových stránek a aplikací [online]. Dostupné z: https://zdrojak.cz/clanky/web-sockets/

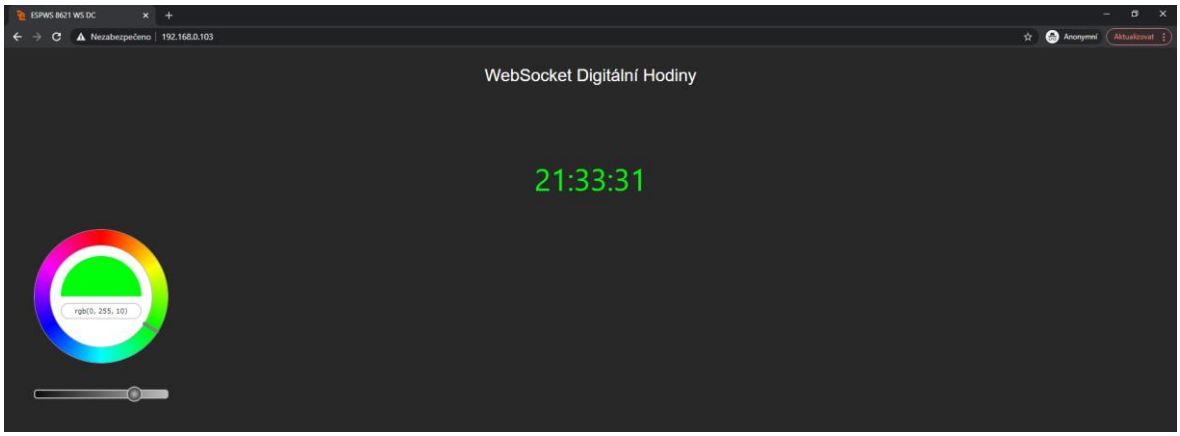
SEZNAM PŘÍLOH

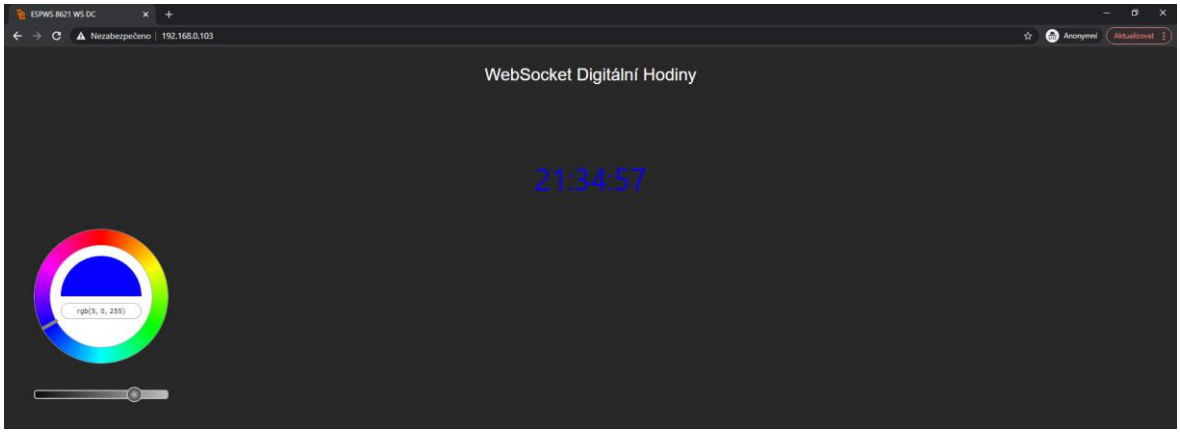
č. 1 Fotodokumentace

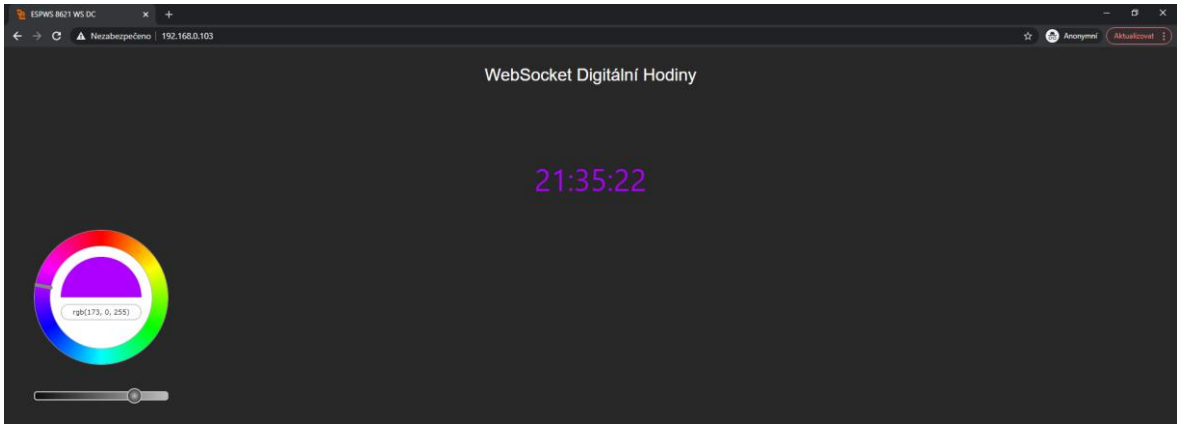
č. 2 Seznam obrázků

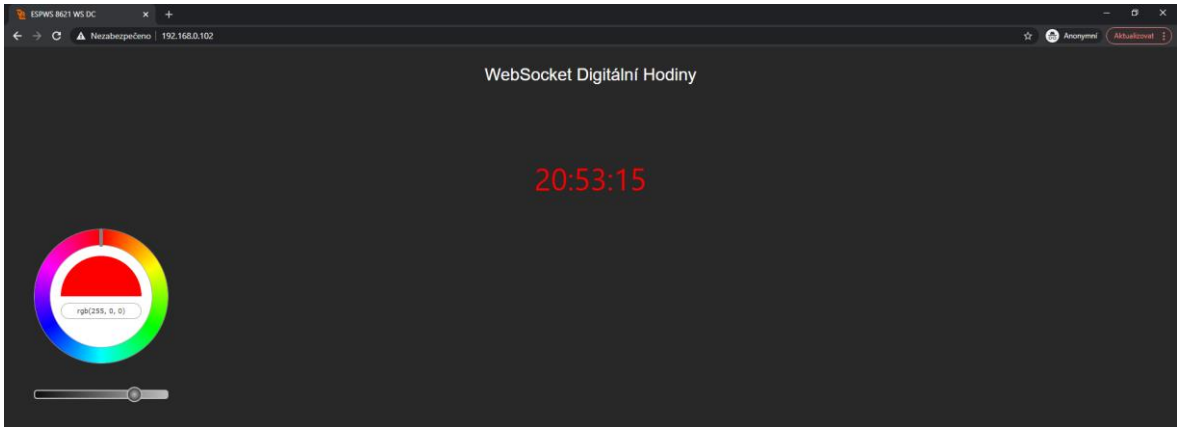
Příloha č. 1: Fotodokumentace

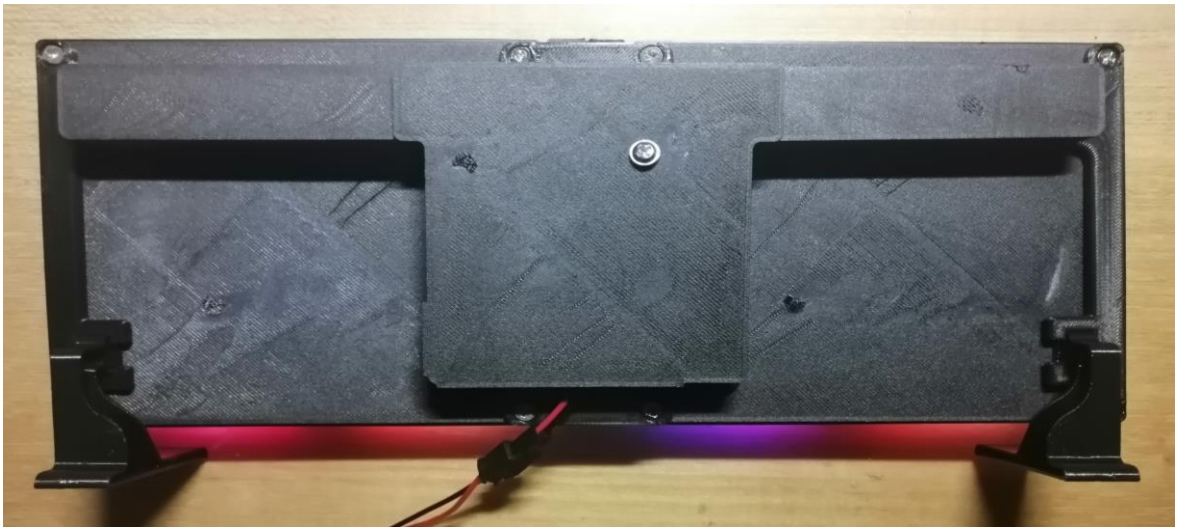


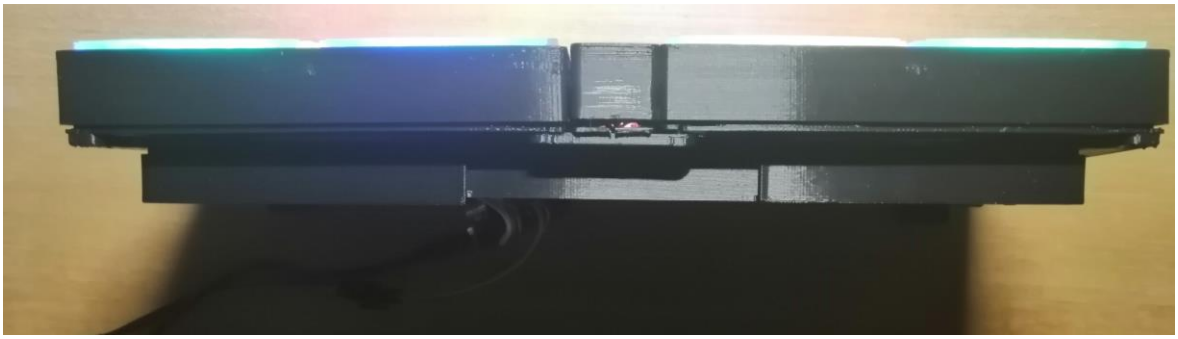


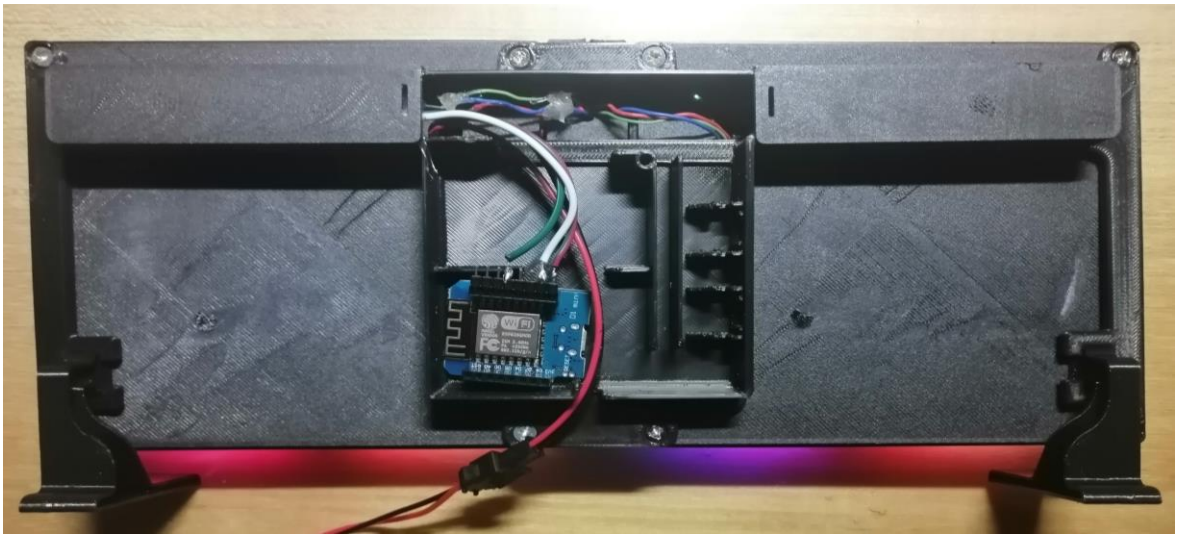


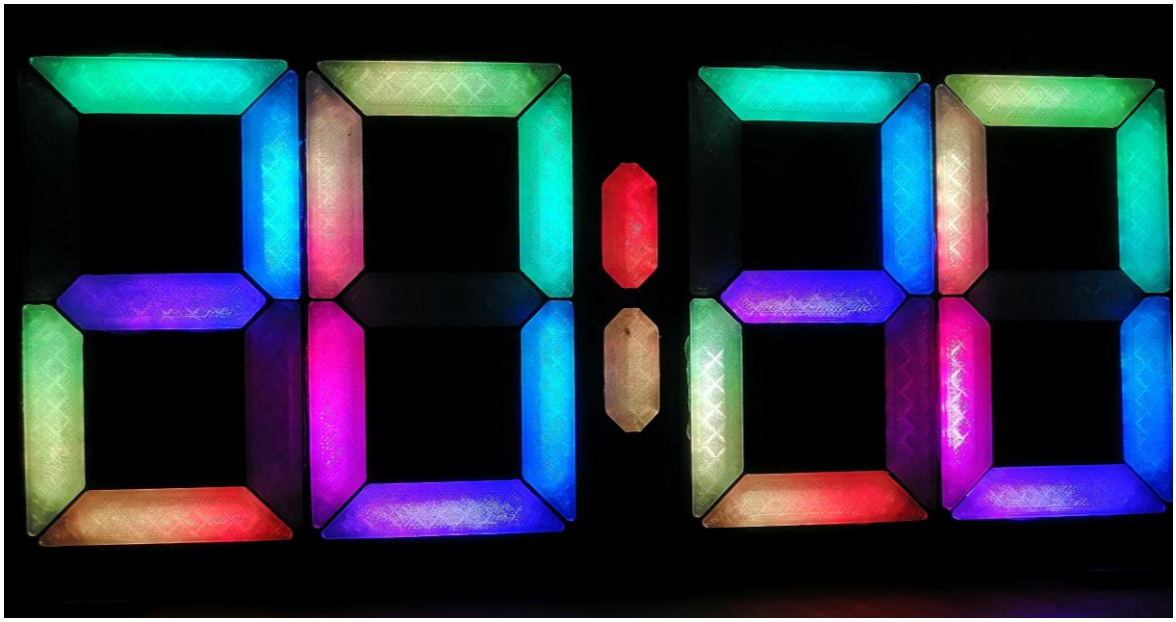

ESPWS 8621 WS DC
Nezabezpečeno
192.168.0.103
Anonymní
Aktualizovat
WebSocket Digitální Hodiny
21:35:22
rgb(173, 0, 255)

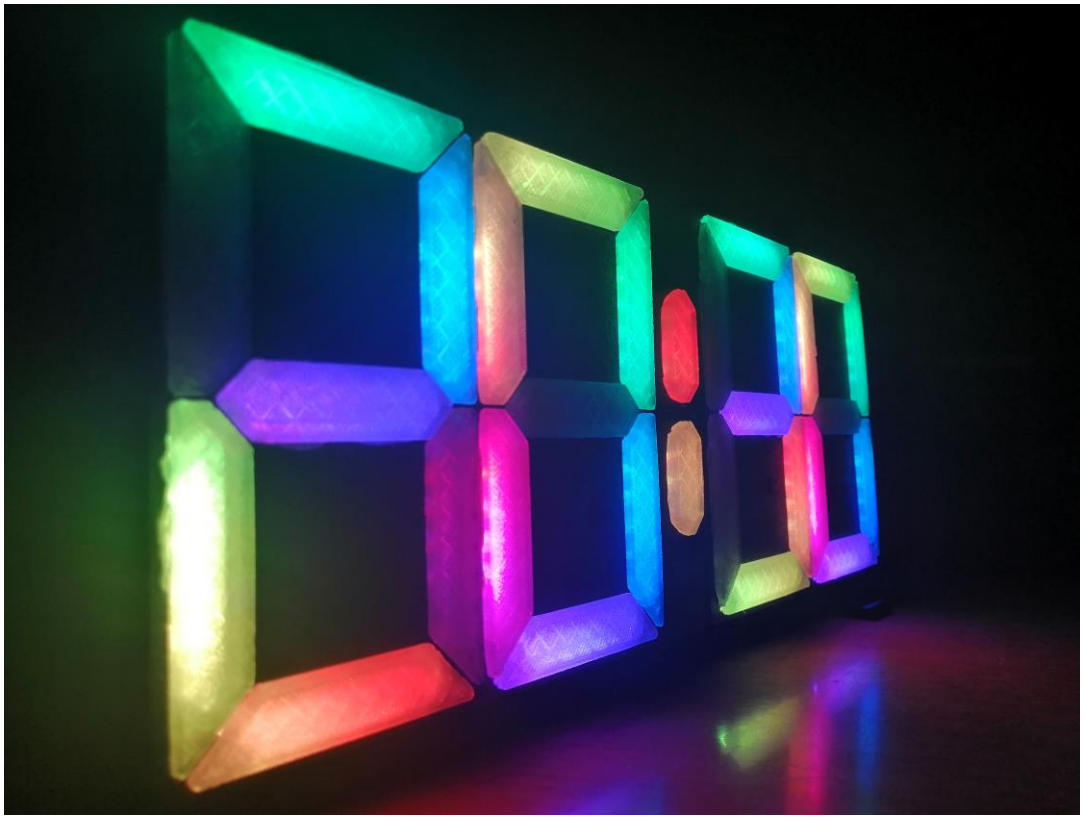

ESPWS 8621 WS DC
Nezabezpečeno
192.168.0.102
Anonymní
Aktualizovat
WebSocket Digitální Hodiny
20:53:15
rgb(255, 0, 0)












20:30

Příloha č. 2: Seznam obrázků

Obrázek č. 1 Technické schéma

Obrázek č. 2 Detail spájených diod v rastru

Obrázek č. 3 Synchronizace s mobilem

Obrázek č. 4 Vzhled a responzivita stránky pro velkou obrazovku

Obrázek č. 5 Vzhled a responzivita stránky pro ipad a mobil

Obrázek č. 6 RGB Color-picker