1. STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

Obor č. 18: Informatika

Wifi hodiny s teplotním čidlem

Michal Dušák
Jihočeský kraj

České Budějovice a 2023

STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

Obor č. 18: Informatika

Wifi hodiny s teplotním čidlem
Wifi clock with temperature sensor

Autoři: Michal Dušák
Škola: Střední škola spojů a informatiky, Bydlinského 2474, 390 11 Tábor 2
Kraj: Jihočeský kraj
Konzultant: Ing. Milan Adámek

České Budějovice a 2023

Prohlášení

Prohlašuji, že jsem svou práci SOČ vypracoval/a samostatně a použil/a jsem pouze prameny a literaturu uvedené v seznamu bibliografických záznamů.

Prohlašuji, že tištěná verze a elektronická verze soutěžní práce SOČ jsou shodné.

Nemám závažný důvod proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších předpisů.

V České Budějovice dne 21.3.2024 ……

Michal Dušák

Poděkování

Chtěl bych poděkovat Ing. Milanu Adámkovi za veškerou jeho podporu čas a úsilí které do mě vložil. Díky jeho pomoci jsem byl schopen vyřešit všechny překážky které se při tvorbě projektu vyskytli. Chtěl bych mu také poděkovat za materiální pomoc bez které by projekt bylo složitější zrealizovat. Poděkování bych chtěl také směřovat směrem k Petru Martínkovi za pomoc s 3D tiskem.

Anotace

Ve své práci SOČ jsem se zabýval tvorbou hodin za pomoci adresovatelných led pásek ovládaných mikrokontrolerem esp32. V této dokumentaci bude čtenář seznámen se součástkami, které byli použity v projektu a jejich fungování.

Záměrem práce bylo vytvořit hodiny dle mích požadavků které na trhu nejsou nebo jsou za příliš vysokou cenu.

Klíčová slova

Digitální hodiny, esp 32, RGB pásky, teplotní čidlo

Annotation

In my SOC work, I have been working on the creation of clocks using addressable led strips controlled by the esp32 microcontroller. In this documentation, the reader will be introduced to the components which were used in the project and how they work.

The intention of the work was to create a clock according to my requirements which are not available on the market or are priced too high.

Keyword

Digital clock, esp 32, RGB strips, temperature sensor

Obsah

1 Úvod

Moderní vzdělávací prostředí neustále hledá nové způsoby, jak integrovat technologii do vyučovacích metod a prostředků. V tomto ohledu jsou časové pomůcky, jako jsou hodiny, klíčovými prvky organizace času a efektivního vedení výuky. Tradiční hodiny však často nedokážou plně reflektovat potřeby současného vzdělávacího prostředí.

S tímto vědomím se zaměřujeme na vývoj inovativních hodin, které spojují pokročilé technologické prvky s praktickým využitím ve školní třídě. Naše hodiny jsou řízeny RTC modulem (Real-Time Clock), který zajišťuje přesný časový údaj nezávisle na externích vlivech. Dále jsou vybaveny teplotním čidlem, které umožňuje monitorovat a zobrazovat teplotní podmínky v třídní místnosti.

Hlavní vizuální aspekt těchto hodin spočívá v čtyřech sedmi segmentových modulech, které jsou osazeny RGB pásky. Tento design umožňuje vytvářet dynamické a atraktivní vizuální zobrazení času a dalších informací, které jsou užitečné pro studenty i učitele.

Řízení těchto hodin je realizováno pomocí platformy ESP32, která poskytuje dostatečný výkon a flexibilitu pro implementaci různých funkcí a interakcí. Tím umožňuje personalizaci a rozšíření funkcí hodin dle potřeb konkrétního vyučovacího prostředí.

Naše hodiny jsou navrženy speciálně pro školní třídy, s důrazem na praktičnost. Jejich vývoj byl motivován absencí podobných produktů na trhu, které by plně vyhovovaly potřebám moderních vyučovacích metod a prostředků.

2 Volba součástek

Bylo nutné si zvolit správně vybrat součástky které bodu splňovat požadované parametry ale zároveň ponechají cenu výsledného produktu přijatelnou. Vybral jsem součástky, které mají dobře zpracovanou dokumentaci pro jednoduší následné využití v projektu. Jeden z parametrů byl také spolehlivost, kdy jsem se snažil najít součástky které mají konzistentní výsledky a jsou na ně dobré výsledky od lidí kteří je již použili.

2.1 Mikrokontroler ESP 32

ESP-WROOM-32 je silný Wi-Fi+BT+BLE MCU modul, který je vhodný pro širokou škálu aplikací, od nízkonapěťových senzorů až po náročnější úkoly, jako je komprese hlasového záznamu, streamování hudby nebo dekódování MP3.

Jádrem tohoto produktu je čip ESP32-D0WDQ. Integrovaný čip je navržen tak, aby se dal škálovat a přizpůsobit. Dvě CPU jádra se dají jednotlivě ovládat, a frekvence je nastavitelná od 80MHz po 240MHz.

ESP32 integruje bohatou sadu periferií, od kapacitních dotykových snímačů, Hallových snímačů, zesilovačů s nízkým šumem, rozhraní SD karet, Ethernet, I2S a I2C.

Integrace Bluetooth, Bluetooth LE a Wi-Fi zajistí, širokou konektivitu a možnost bezdrátové komunikace.

Wi-Fi umožňuje velký dosah a přímé připojení k internetu přes Wi-Fi router, zatímco použití Bluetooth umožní uživateli pohodlné připojení k telefonu nebo vysílat nízkonapěťové signály pro detekci.

ESP32 podporuje přenosovou rychlost až 150Mbps a 20.5 dBm výstupní výkon antény zajistí dlouhý dosah. Čip nabízí vysoký výkon a je vhodný pro elektronickou integraci.

Operační systém vybraný pro ESP32 je freeRTOS with LwIP; TLS 1. s hardwarovou akcelerací je také integrován.

(Vývojová deska ESP32 2,4GHz Dual-Mode Wi-Fi + Bluetooth)

Obrázek 1: Mikrokontroler esp 32

2.2 Modul RTC s DS3231

2.2.1 Úvod

Čidlo DS3231 je vysoce přesný a stabilní real-time hodinový modul (RTC), který je vhodný pro mnoho aplikací, které vyžadují přesné časování. Díky své přesnosti a nízké spotřebě energie je DS3231 populární volbou pro zařízení, která u kterých využít jeho funkci uchování času i v okamžiku kdy je odpojen od elektřiny pro tento případ je vybaven vlastním zdrojem elektřiny který zjišťuje Baterie LIR2032 a udržuje modul neustále v provozu pro udržení nastaveného času aktuálního.

2.2.2 Princip fungování

DS3231 využívá krystalový oscilátor s teplotní kompenzací pro udržení přesného času. Obsahuje interní obvodové bloky, které umožňují měření času, dní, měsíců a roku v reálném čase. Dále je vybaven funkcemi pro sledování a kompenzaci teplotních vlivů na přesnost času.

2.2.3 Elektrická specifikace

Napájecí napětí: 2.3V až 5.5V

Průměrný proud v klidu: 2µA (max.)

Přesnost času: ±2 ppm (±0.16 sekund za den) při 25°C

Baterie LIR2032: k zajištění pojistky při výpadku napájení.

2.2.4 Komunikace

DS3231 podporuje komunikaci přes I²C rozhraní, což umožňuje snadnou integraci s mikrokontrolery a dalšími zařízeními. Tato komunikace umožňuje nastavení času, čtení hodnoty času a kalendáře a nastavení alarmů.

Obrázek 1 Modul RTC DS3231

2.3 Teplotní čidlo DS18B20

2.3.1 Úvod

Čidlo DS18B20 je digitální teploměr, který je vysoce přesný a snadno použitelný do různých elektronických zařízení. Je vhodný pro měření teploty v širokém rozsahu aplikací, včetně průmyslových procesů, klimatizace, počítačů, domácí automatizace a dalších.

2.3.2 Princip fungování

DS18B20 je tzv. jednodrátové digitální čidlo teploty, což znamená, že komunikuje přes jediný datový vodič. Princip fungování je založen na digitálním převodu teploty pomocí vnitřního analogově-digitálního převodníku (ADC). Čidlo obsahuje interní EEPROM paměť, ve které jsou uloženy tovární kalibrační údaje a jedinečná 64bitová adresa.

2.3.3 Elektrická specifikace

Napájecí napětí: 3.0V až 5.5V

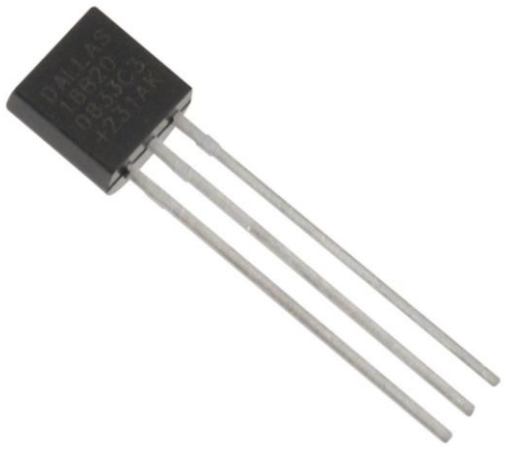
Průměrný proud v klidu: 1µA (max.)

Rozsah teplotního měření: -55°C až +125°C

Přesnost měření teploty: ±0.5°C (-10°C až +85°C)

2.3.4 Komunikace

DS18B20 komunikuje pomocí jednoduchého sériového protokolu nazvaného "1-Wire". Tento protokol umožňuje, aby více čidel bylo připojeno k jedinému datovému vodiči, což usnadňuje jejich propojení a snižuje požadavky na kabeláž.



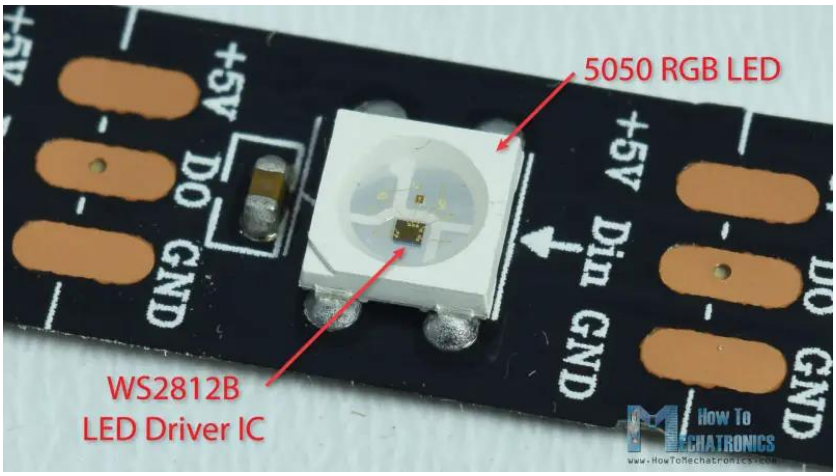
Obrázek 2 Teplotní čidlo DS18B20

2.4 RGB pásek WS2812B

RGB pásek WS2812B je LED pásek, který umožňuje zobrazení barevných efektů a animací. Díky svému jednoduchému propojení a vysoké flexibilitě se WS2812B často používá v osvětlovacích projektech, v interaktivních instalacích a v DIY projektech.

2.4.1 Jak fungují LED WS2812B

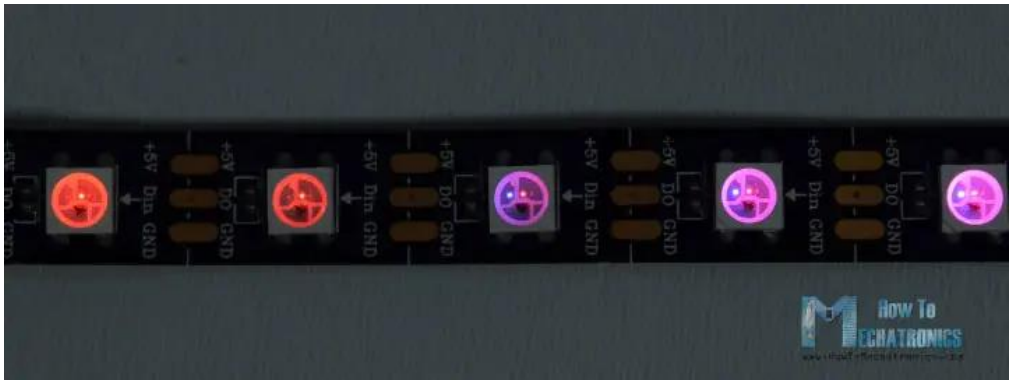
Začněme bližším pohledem na LED pásek. Skládá se z LED typu 5050 RGB, ve kterých je integrován velmi kompaktní LED ovladač WS2812B LED.



Obrázek 3 RGB LED

2.4.2 WS2812B LED ovladač IC 5050 RGB LED

V závislosti na intenzitě tří jednotlivých červených, zelených a modrých LED diod můžeme simulovat jakoukoli barvu, kterou chceme.



Obrázek 4 RGB LED demonstrace adresovatelnosti

(How To Control WS2812B Individually Addressable LEDs using Arduino)

2.4.3 Elektrická specifikace

Napájecí napětí: 3.5V až 5.3V

Počet LED na metr: 30, 60, nebo více (záleží na konkrétním modelu)

Maximální proud na LED: 20 mA

Komunikační rychlost: Až 800 kHz

2.4.4 Komunikace

WS2812B používá jednoduchý sériový protokol, ve kterém se data pro nastavení barev a jasu posílají po jednotlivých LED. Tento protokol umožňuje řízení více LED proužků z jediného řídicího zařízení.

2.5 Wifi

Standart Wifi se stal jedním z nejrozšířenějších na světě. Wifi má hned několik standartu:

802.11n se stává Wi-Fi 4 byl vydán v roce 2003. Wi-Fi 4 je první technologie Wi-Fi, která pracuje současně ve frekvenčních pásmech 2,4 GHz i 5 GHz a může dosahovat přenosových rychlostí až 600 Mbit/s.

802.11ac se stává Wi-Fi 5 byla vydána v roce 2013. Přináší rozšíření radiového pásma (až 160 MHz) a modulační technologii vyššího řádu (256-QAM). Může dosahovat přenosových rychlostí až 1,73 Gbit/s. Je však důležité poznamenat, že Wi-Fi 5 802.11ac podporuje pouze zařízení v pásmu 5 GHz.

802.11ax se stává Wi-Fi 6 čtyřnásobného zvýšení šířky pásma a čtyřnásobného nárůstu počtu souběžných uživatelů. Kromě toho může tento protokol pracovat ve frekvenčním pásmu 2,4 GHz i 5 GHz.

Pokud jde o rychlost sítě, dopad na průměrné domácí uživatele není až tak velký. V současnosti je horní hranice domácího širokopásmového připojení ve většině měst 1 Gbit.

Samotné Wifi je standardizovaná bezdrátová technologie pro připojení zařízení k bezdrátové lokální síti (WLAN – Wireless Local Area Network). Používá rádiové vlny k přenosu dat mezi zařízeními, která jsou vybavena Wifi moduly. Tato technologie využívá standardy IEEE 802.11, které určují způsob, jak se data přenášejí vzduchem.

Proces komunikace mezi zařízeními pomocí Wifi zahrnuje následující kroky:

Vytvoření sítě: Prvním krokem je vytvoření bezdrátové sítě pomocí bezdrátového směrovače nebo přístupového bodu. Tímto způsobem se umožní připojení více zařízení k síti.

Přenos signálu: Zařízení s Wifi schopnostmi mohou posílat a přijímat data pomocí rádiových vln. Tyto signály jsou obvykle vysílány v rádiovém frekvenčním pásmu.

Modulace a demodulace: Při komunikaci jsou data modulována na rádiové signály a na druhé straně demodulována zpět na původní data.

Přenosové rychlosti: Wifi podporuje různé rychlosti přenosu dat, které závisí na použitém standardu. Novější standardy jako 802.11ac a 802.11ax nabízejí vyšší rychlosti a spolehlivější přenos dat než starší standardy.

Zabezpečení sítě: Aby se zabránilo neoprávněnému přístupu a útokům, je důležité zabezpečit Wifi síť. K tomu slouží různé bezpečnostní protokoly, jako například WPA2.

Správa sítě: Bezdrátové sítě vyžadují správu a řízení, aby fungovaly efektivně a bezpečně. To zahrnuje správu připojení, řízení signálu a monitorování bezpečnosti.

Celkově Wifi umožňuje bezdrátové připojení k internetu a lokálním sítím, což je klíčové pro komunikaci a přístup k informacím ve většině domácností, kanceláří a veřejných prostorách já jsem ho v mém projektu použil pro komunikaci mezi mikrokontrolere esp 32 a zařízením které si zvolíte pro ovládání např.: telefon, PC.

2.6 Komunikace mezi ESP 32 a zařízením

V okamžiku komunikace mezi mikrokontrolerem EPS 32 a ovládacím zařízením je ESP 32 v pozici serveru a ovládací zařízení v pozici klienta. Po připojení k esp 32 pomocí wifi si uživatel naskenuje qr kód kde je vepsaná příslušná ip adresa na které je uživatelské rozhraní po zadání parametrů jsou data odeslány zpět do esp 32 kde jsou zpracována. Samotná komunikace probíhá pomocí paketů:

Každý packet obvykle obsahuje několik základních částí:

Hlavička (Header): Obsahuje informace potřebné pro doručení a správné zpracování paketu, jako jsou zdrojová a cílová adresa, typ paketu, kontrolní součet atd.

Data: Samotná uživatelská data, která jsou přenášena.

Případně patřičný závěr (Footer): V některých protokolech může být přidán závěr, který obsahuje kontrolní součet nebo jiné údaje potřebné pro kontrolu integrity dat.

Jak funguje přenos packetů:

Rozdělení dat: Data jsou rozdělena na menší části, pokud jsou příliš velká pro přenos v jednom kuse. Toto rozdělení může být prováděno na různých úrovních komunikačního protokolu (například na úrovni transportního nebo síťového protokolu).

Přidání hlavičky a případně závěru: Každému packetu je přidána hlavička obsahující potřebné informace pro směrování a zpracování paketu cílovým zařízením. V některých případech může být také přidán závěr obsahující kontrolní součet.

Přenos po síti: Pakety jsou přenášeny po síti prostřednictvím různých zařízení (například směrovačů a přepínačů), které je směrují na základě informací v jejich hlavičkách.

Příjem a rekonstrukce dat: Na cílovém zařízení jsou přijímány pakety a rekonstruována do původní podoby. To zahrnuje odstranění hlaviček a případných závěrů a seskupení dat do původního pořadí.

Kontrola integrity dat: Na cílovém zařízení může být provedena kontrola integrity dat pomocí kontrolních součtů nebo jiných metod, aby se zjistily případné chyby přenosu a zajistila správnost dat.

Celkově lze říci, že packetizace dat slouží k rozdělení většího objemu dat ta menší celky, které se v cílovém bodě opětovně složí.

Obrázek 5 Struktura packetu

3 Konstrukce zařízení

3.1 Napájení

Pro napájení bude použit síťový zdroj, který mění střídaví proud na stejnosměrný a snižuje napětí z 230 volt na 5 volt pomocí transformátoru a spínaného zdroje.

3.2 Vstupní data

Do mikrokontroleru ESP 32 přichází data z dvou zařízení z teplotního čidla a z RTC modulu.

Komunikace mezi čidlem DS18B20 a mikrokontrolerem ESP32 probíhá pomocí jednoduchého sériového protokolu nazvaného "1-Wire". DS18B20 je jednodrátové digitální čidlo teploty, což znamená, že využívá pouze jeden datový vodič pro komunikaci.

Prostřednictvím knihovny "OneWire" a "DallasTemperature" je možné implementovat komunikaci mezi čidlem DS18B20 a ESP32 v prostředí Arduino IDE.

Po žádosti o měření teploty čidlo reaguje a vrátí naměřenou hodnotu. Tato hodnota je získána pomocí požadavku ze strany esp 32pro možnost připojení vícero DS18B20 je každá hodnota opatřena indexem kde index určuje konkrétní čidlo.

Díky jednoduché implementaci a využití knihoven OneWire a DallasTemperature je komunikace mezi čidlem DS18B20 a ESP32 efektivní a spolehlivá. Použití protokolu 1-Wire zjednodušuje použití čidla.

Komunikace mezi modulem RTC s DS3231 a mikrokontrolerem ESP32 je realizována prostřednictvím sériového komunikačního protokolu, konkrétně přes I²C (Inter-Integrated Circuit). I2C jsou data přenášena dovnitř zprávy. Zprávy jsou rozděleny do rámců dat. Každá zpráva má rámec adresy, který obsahuje binární adresu otroka, a jeden nebo více datových rámců, které obsahují přenášená data. Zpráva také obsahuje podmínky spuštění a zastavení, čtení / zápisu bitů a ACK / NACK bitů mezi každým datovým rámcem:

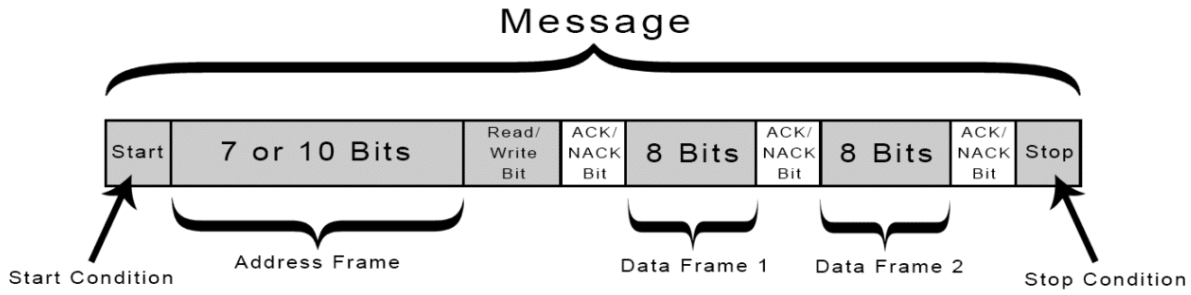
Stav startu: Linka SDA přechází z vysoké úrovně napětí na nízkou úroveň napětí před linka SCL přechází z vysoké na nízkou.

Stav zastavení: Linka SDA přechází z nízké úrovně napětí na vysokou úroveň napětí po linka SCL přechází z nízké na vysokou.

Rámeček adresy: 7 nebo 10bitová sekvence jedinečná pro každého otroka, který identifikuje otroka, když s ním chce mistr mluvit.

Bit čtení / zápisu: Jeden bit určující, zda master odesílá data do otroka (nízká úroveň napětí) nebo požaduje data z něj (vysoká úroveň napětí).

ACK / NACK Bit: Po každém snímku ve zprávě následuje bit potvrzení / nepoznání. Pokud byl úspěšně přijat rámec adresy nebo datový rámec, je odesílateli z přijímajícího zařízení vrácen bit ACK.



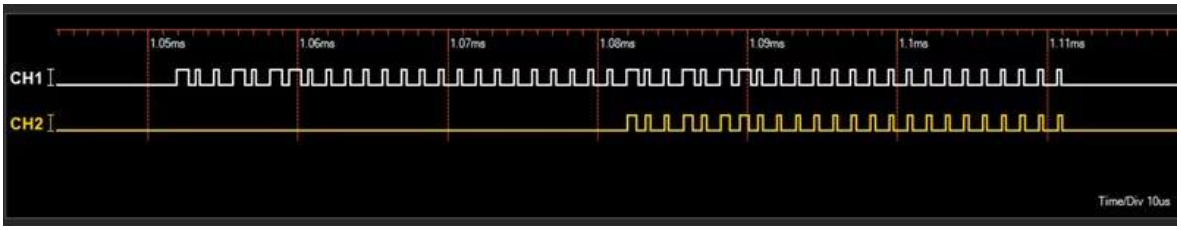
Obrázek 6 Schéma zprávy v protokolu I2C

Pro implementaci této komunikace na ESP32 je třeba inicializovat I²C rozhraní a provést správné připojení modulu DS3231 k ESP32. Poté lze získat časové údaje z modulu DS3231 pomocí příslušných I²C příkazů a funkcí dostupných v prostředí Arduino IDE.“

Po inicializaci I²C rozhraní je možné komunikovat s modulem DS3231 pomocí jeho adresy na sběrnici. Pro čtení času a datumu je používána specifická komunikační adresa DS3231. Pomocí knihoven a funkcí dostupných v prostředí Arduino je možné snadno získat aktuální čas a datum z modulu DS3231.

3.3 Výstupní data

Výstupní zařízení je pouze jedno, a to LED pásek s řadičem WS2812B který ovládá, která LED bude zrovna svíti a jakou barvou komunikace s LED páskem probíhá v protokolu WS2812B



Obrázek 7 Schéma průběhu signálu protokolu WS2812B

„Binární signál CH1 z mikrokontroleru obsahující 48 bitů pro ovládání 2 WS2812Bs. Pouze 24 z těchto bitů je předáno jako CH2.

Abychom porozuměli těmto LED diodám, projdeme, jak tento adresovatelný protokol LED funguje. Každá samostatná červená, zelená a modrá LED v jedné jednotce WS2812B je nastavena tak, aby zářila při 256 úrovních jasu, což je indikováno 8bitovou binární sekvencí nastavenou od 0 do 255. Při kombinaci vyžaduje každá jednotka LED tři sady osmi bitů jasu nebo 24 bitů informací pro plnou kontrolu. Zde je rychlý průvodce procesem:

1. A mikrokontroler přenáší tuto sekvenci osmi zelených bitů, osmi červených bitů a osmi modrých bitů na první LED v řadě.

2. Když více LED jsou k dispozici datová sekvence, která řídí druhou LED, začíná bezprostředně za první zelenou, červenou a modrou. Sekvence pokračuje v tomto vzoru, dokud neosvětlí každou přítomnou LED.

3. První LED vezme informace pro celý řetězec LED a poté předá stejná data bez sekvence, kterou aplikovala na sebe, transformace druhé LED na první komponentu v seznamu (pokud ví).

4. Tato LED jednotka „nové číslo jedna“ pokračuje v předávání informací, dokud nezbývají žádné binární LED sekvence.

Výše uvedený obrázek ukazuje vstupní signál mikrokontroleru do jednotky WS2812B jako CH1, kde dlouhé impulsy označují vysoké signály a krátké impulsy označují nízké signály. Toto vyjde na 24bitovou sekvenci 100101100000000000000000, což znamená hodnotu „10010110“ pro zelenou — 150 při převodu na desetinnou čárku — a nuly pro zbývajících 16 bitů. Výsledná barva je střední jas čistě zelená hodnota, bez smíchání červené nebo modré.

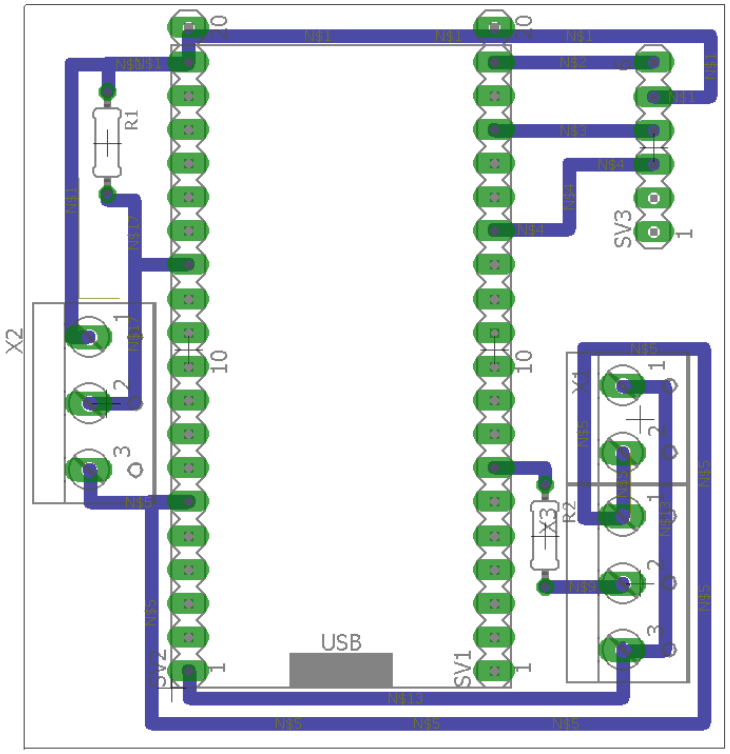
Obě LED jednotky WS2812 jsou zde stejné barvy, takže sekvence CH1 se opakuje a předává stejná data do CH2. Druhá sekvence (a výsledná barva) se může lišit v závislosti na požadovaném efektu uživatele.“

(*WS2812B LED Protocol and LED Microcontrollers)*

3.4 Deska plošných spojů

3.4.1 Návrh

Pro účely mého projektu mi přišlo nejlepší využít jednostrannou desku o tloušťce 1,5 mm z důvodu jednoduchosti a možnosti osadit desky i bez specializovaného zařízení jsem se přiklonil k osazení vývodovými součástkami. Nevýhodou je větší velikou a stím spjatá větší nároky na velikost desky ale z důvodu dostatku místa v prostoru kde je deska osazena jsem nebyl tlačen k zmenšování desky.



Obrázek 8 Schéma desky

3.4.2 Prototypování

Zapojení jsem si musel nejdřív vyzkoušet k tomu účelu jsem použil nepájivé pole. Na kterém jsem mohl libovolně měnit uspořádání a zapojení vodičů a součástek bez nutnosti vyrábění nového tištěného spoje. Což velice zjednodušuje a urychluje prototypování nehledě na snížení finanční náročnosti vývoje která se snižuje pouze na cenu nepájivého pole.

3.5 Tělo hodin

Pro tvorbu těla hodin mi přišlo nejlepší volba 3D tisk. Hned z několika důvodů mezi hlavní důvody patří jednoduchost výroby a cenová dostupnost. Pro modelování těla hodin jsem pou-

žil Fusion 360 je to velice uživatelsky přívětiví program. Tělo hodin se skládá ze čtyř téměř totožných dílů a středového dílu ke všem těmto dílům jsou vršky který spodní díl uzavírají. Čitelnost displeje zajišťují poloprůhledné části ve vrchním díle.

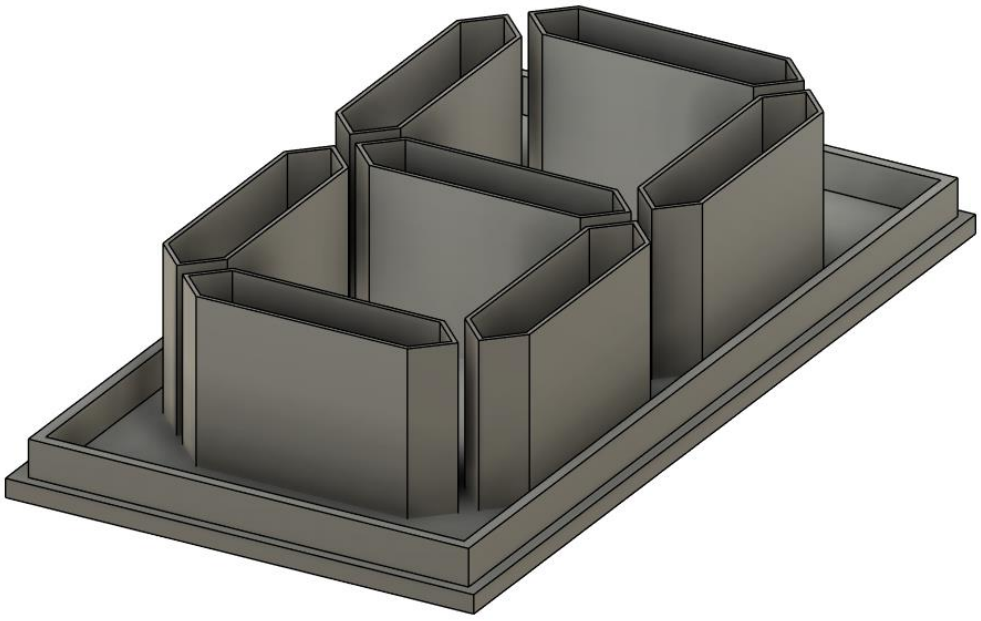
3.5.1 Spodní díl

Spodní díl má sedm segmentů pro vložení LED pásku a z boku jsou zámky pro spojení s ostatními díly. Při vytváření 3D modelu se vyskytlo dva problémy, které jsem musel řešit jako první jsem musel vyřešit problém s velikostí zámků které jsem musel přizpůsobit tak aby bez většího odporu do sebe zajeli a nemusel jsem vyvíjet na tělo hodin z důvodu možnosti prasknutí v případě kdy by jednotlivé zámky správně nepasovali. Proto jsem si vytiskl upravený model těla hodin pouze se zámky tak abych mohl testovat pouze jeden aspekt modelu.

Obrázek 9 3D model zámků

Jako druhý problém jsem musel řešit nevyhovujíc uložení LED pásků, které jsem do jednotlivých segmentů nejdříve zapouštěl a v ohraničení segmentu jsem vytvořil obdélníkovou dírku s nechaným materiálem v horní části otvoru. Tento způsob se ukázal jako nedokonalí při pájení se k pinům na Led pásku dostával rozteklí filament a při manipulaci s vodiči mi překážel nechaný materiál nad otvorem v ohraničení. V druhé verzi jsem LED pásky již nezapouštěl a nechal jsem je v úrovni těla hodin. Materiál nad otvorem jsem odstranil tak aby se nechali vodiče pohodlně vkládat ze shora. Při testu jsem zjistil že nezapuštěn pásky jsou sice těžší na rovné usazení, ale jsou ovšem mnohem lehčí na napájení a piny nepřichází do kontaktu s rozteklým filamentem při pájení, ale odstranění materiálu nad otvory v ohraničení segmentu je sice lehčí manipulace, ale vodiče se všemožně kroutí a nezůstávají na požadovaných místech. Ve třetí finální verzi jsme zužitkoval poznatky z obou předchozích verzí a LED pásek jsem nezapouštěl a materiál nad prostupem jsem zanechal z důvodu jednoduššího usměrnění kabeláže.

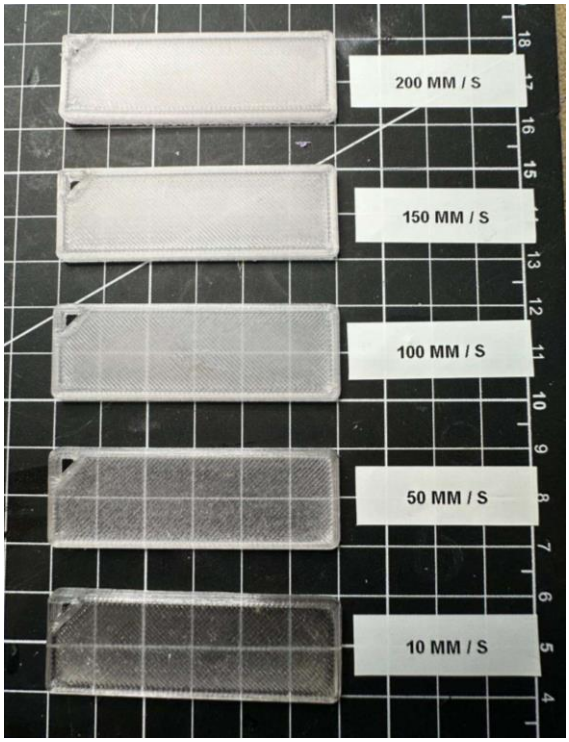
Obrázek 10 3D model těla hodin



Obrázek 11 3D model vršek hodin

3.5.2 Průsvitné segmenty

V každém se čtyř dílů je sedm průsvitných segmentů, které fungují jako takový display. Při nasvícení segmentu LED páskem, který je pod ním začne světlo rozrážet a vytvářet iluzi toho že svítí. Pro dosáhnutí průsvitnosti jsem použil průhledné PETG. Průhlednost Následného výtisku je ovšem velice ovlivněna rychlostí tisku.



Obrázek 12 Průhlednost 3D tisku podle rychlosti tisku

4 Software

Pro programování jsem zvolil Programovací rozhraní Arduino IDE, které je jedním z nejpoužívanějších programovacích rozhraní pro ESP 32. V Arduino IDE používá variaci jazyku C++. Pro použití

4.1 Knihovny

Knihovny v Arduino IDE slouží k poskytnutí předpřipraveného funkce pro zařízení, což usnadňuje vývoj programů pro různé desky. Tyto knihovny obsahují předdefinované funkce a třídy, které umožňují komunikaci s různými senzory, zařízeními a dalšími periferiemi. Slouží také k zjednodušení práce s různými komunikačními protokoly, jako je například I2C, SPI nebo sériová komunikace.

Vývojáři mohou vytvářet vlastní knihovny, které obsahují specifické funkce nebo ovladače pro jejich projekty a sdílet je s komunitou Arduino. Tím se umožňuje rychlý vývoj projektů a využití širokého spektra dostupného hardwaru bez nutnosti psaní kódu zcela od začátku.

Kromě toho, že knihovny ulehčují vývoj, také pomáhají při řešení problémů, protože jsou obvykle dobře dokumentovány a mají širokou podporu komunity, což znamená, že existuje mnoho zdrojů a příkladů, které lze použít k porozumění a implementaci konkrétní funkcionality.

4.1.1 Arduino.h

Knihovna Arduino.h je jednou z klíčových knihoven pro vývoj softwaru pro desky ESP 32 v Arduino IDE. Tato knihovna poskytuje základní funkce a definice potřebné pro práci s ESP 32 a jeho periferiemi.

Zde jsou některé z hlavních funkcí poskytovaných knihovnou Arduino.h:

Základní funkce I/O: Knihovna Arduino.h obsahuje funkce pro práci s digitálními a analogovými piny desky Arduino. To zahrnuje funkce pro čtení a zápis digitálních signálů (digitalRead() a digitalWrite()) a analogových hodnot (analogRead() a analogWrite()).

Časové funkce: Pro práci s časem a časovými prodlevami poskytuje knihovna Arduino.h funkce jako delay(), delayMicroseconds() a millis(). Tyto funkce umožňují programátorům řídit časování svých programů a akcí.

Komunikace sériové linky: Pro komunikaci s periferiemi, jako jsou počítačové terminály, displeje a další mikrokontroléry, knihovna Arduino.h obsahuje funkce pro práci s sériovou linkou, například Serial.begin(), Serial.print() a Serial.read().

Interrupce: Knihovna poskytuje funkce pro práci s hardwarovými a softwarovými přerušeními. Přerušení jsou užitečná pro reagování na události v reálném čase bez blokování hlavního programu.

Další funkce a definice: Kromě výše uvedených funkcí poskytuje Arduino.h mnoho dalších funkcí a definic, které usnadňují programování pro desky ESP 32. To zahrnuje matematické funkce, práci s čísly s plovoucí řádovou čárkou, funkce pro práci s řetězci a další.

Celkově lze říci, že knihovna Arduino.h poskytuje programátorům ESP 32 všechny základní nástroje potřebné k vytvoření programů pro jejich desky, a to bez nutnosti psát nízko úrovňový kód pro komunikaci s hardwarem. Díky této knihovně je programování s ESP 32 snadné a přístupné i pro začátečníky.

4.1.2 WiFi.h

Knihovna WiFi.h je klíčovou knihovnou pro práci s bezdrátovým připojením WiFi na desky Arduino, zejména na desky s integrovaným modulem WiFi, jako je například ESP 32 WiFi Rev2 nebo ESP8266. Tato knihovna umožňuje programátorům vytvářet aplikace, které mohou komunikovat přes WiFi s jinými zařízeními, síťovými službami nebo internetem.

Zde jsou některé z hlavních funkcí a definic poskytovaných knihovnou WiFi.h:

Připojení k WiFi síti: Knihovna WiFi.h obsahuje funkce pro připojení desky Arduino k existující WiFi síti. To zahrnuje funkce jako WiFi.begin() pro zahájení procesu připojení a WiFi.status() pro kontrolu stavu připojení.

Konfigurace WiFi: Programátoři mohou pomocí funkcí v knihovně WiFi.h nastavovat různé parametry WiFi, jako je například SSID a heslo sítě, IP adresy a další.

Ovládání přenosových protokolů: Knihovna umožňuje ovládat přenosové protokoly jako TCP a UDP pro komunikaci s dalšími zařízeními nebo síťovými službami.

Správa spojení: Programátoři mohou spravovat spojení s WiFi sítí pomocí funkcí pro připojení, odpojení a obnovení spojení.

Informace o síti: Knihovna umožňuje získávat informace o připojení k WiFi síti, jako je například IP adresa desky ESP 32 v síti, signál síťového spojení a další.

Bezpečnostní funkce: WiFi.h také poskytuje funkce pro zajištění bezpečnosti připojení, včetně podpory šifrování a ověřování.

4.1.3 WebServer.h

Knihovna WebServer.h je důležitým nástrojem pro vývoj webových aplikací na desce ESP 32. Tato knihovna umožňuje programátorům vytvářet webové servery přímo na svých deskách ESP 32, což umožňuje komunikaci s jinými zařízeními přes webové rozhraní.

Zde jsou některé z hlavních funkcí a definic poskytovaných knihovnou WebServer.h:

Vytvoření webového serveru: Knihovna WebServer.h poskytuje funkce pro vytvoření a správu webového serveru přímo na desce ESP 32. Programátoři mohou definovat cesty URL a přidružené funkce k obsloužení požadavků na tyto cesty.

Routování požadavků: Pomocí knihovny WebServer.h mohou programátoři definovat různé obslužné funkce pro různé cesty URL. To umožňuje efektivní routování požadavků a poskytování různých odpovědí na základě URL adresy.

Zpracování HTTP požadavků: Knihovna umožňuje zpracovávat různé typy HTTP požadavků, jako jsou GET, POST, PUT, DELETE atd. Programátoři mohou definovat funkce pro zpracování těchto požadavků a provádění příslušných akcí.

Odpovědi na požadavky: WebServer.h poskytuje funkce pro generování odpovědí na HTTP požadavky. To zahrnuje možnost odesílat HTML stránky, JSON data nebo jiné typy obsahu v odpovědi na požadavek.

Správa spojení: Programátoři mohou pomocí knihovny WebServer.h spravovat spojení s klienty a sledovat stav spojení.

4.1.4 FastLED.h

Knihovna FastLED.h je klíčovým nástrojem pro programátory využívající desky ESP 32 k řízení adresovatelných LED diod. Tato knihovna umožňuje snadné a efektivní ovládání barevných efektů a animací na LED proudech.

Zde jsou některé hlavní funkce a definice poskytované knihovnou FastLED.h:

Inicializace LED proudu: Pomocí FastLED.h mohou programátoři inicializovat LED proudy a nastavit jejich počet a typ (jako WS2812B, WS2811, APA102 atd.).

Nastavení barev LED: Knihovna umožňuje programátorům snadno nastavit barvy jednotlivých LED diod pomocí RGB (červená, zelená, modrá) nebo HSV (odstín, sytost, hodnota).

Animace a efekty: S FastLED.h mohou programátoři vytvářet různé barevné animace a efekty, jako jsou blikání, průchod, plynulé přechody, plameny a mnoho dalšího.

Optimalizace výkonu: Knihovna poskytuje optimalizace pro rychlý a účinný výkon, což umožňuje plynulý chod animací i při ovládání velkého množství LED diod.

Podpora různých platforem: FastLED podporuje širokou škálu platform, včetně desek Arduino, ESP8266, ESP32 a dalších.

Uživatelská rozhraní: FastLED umožňuje tvorbu uživatelských rozhraní pro ovládání animací a efektů, což usnadňuje interakci s LED proudy.

Celkově lze říci, že knihovna FastLED.h je nepostradatelným nástrojem pro tvorbu poutavých světelných efektů a animací s adresovatelnými LED diodami na deskách ESP 32. Díky své jednoduchosti použití a široké škále funkcí je oblíbenou volbou pro tvůrce světelných instalací, uměleckých projektů a dalších interaktivních aplikací.

4.1.5 OneWire.h

Knihovna OneWire.h je klíčovým nástrojem pro komunikaci s digitálními zařízeními pomocí protokolu OneWire na deskách ESP 32. Tato knihovna poskytuje programátorům snadný způsob, jak číst a zapisovat data z jednoduchých senzorů a zařízení, které používají komunikační protokol OneWire.

Zde jsou některé hlavní funkce a definice poskytované knihovnou OneWire.h:

Inicializace komunikace: Pomocí OneWire.h mohou programátoři inicializovat komunikaci s zařízením používajícím protokol OneWire a nastavit pin, na kterém je datová linka připojena.

Čtení a zápis dat: Knihovna umožňuje programátorům číst a zapisovat data z a do zařízení komunikujících pomocí protokolu OneWire.

Adresování zařízení: OneWire.h poskytuje funkce pro adresování jednotlivých zařízení na sběrnici OneWire, což umožňuje komunikaci s více zařízeními připojenými k jednomu pinu.

Podpora různých senzorů: Knihovna OneWire.h je široce používána pro čtení dat z různých typů senzorů, jako jsou teplotní senzory (například DS18B20), čidla vlhkosti, čidla tlaku a další.

Jednoduché použití: OneWire.h je navržena tak, aby byla snadno použitelná i pro začátečníky, a poskytuje jednoduché funkce pro komunikaci s OneWire zařízeními.

Toto je několik knihoven, které jsem použil a pokus o rychlé seznámení s některými. Ve Své práci jsem použil osm knihoven krom těch, které jsem vyjmenoval tak také: DallasTemperature.h, RTClib.h, Wire.h.

4.2 Kód pro ESP 32

Zde okomentuji vybrané úseky z kódu, který jsem sepsal pro mikrokontroler ESP 32 v jazyce C++ a pro uživatelské prostředí jsem použil jazyk JavaScript, CSS a značkovací jazyk HTML.

4.2.1 Vytvoření Wifi

Pro komunikaci mezi ESP 32 a zařízením kterým bude uživatel nastavovat hodiny jsem zvolil komunikaci pomocí Wi-Fi. Hlavní důvod, proč jsem nepoužil Bluetooth je že v případě Wi-Fi stačí pouze se připojit na ip adresu a nemusíte stahovat žádnou aplikaci. Proto je v mém konkrétním případě lepší řešením.

4.2.2 Uživatelské rozhraní

Uživatelské rozhraní je napsané v HTML, JavasScript, CSS. Je responzivní, takže i při otevření například na počítači tak funguje bez problému i když je původně psané pro mobilní telefon.

Obrázek 13 uživatelské rozhraní

```
function getCurrentTime() {
  var now = new Date();
  var hours = now.getHours().toString().padStart(2, '0');
  var minutes = now.getMinutes().toString().padStart(2, '0');
  var seconds = now.getSeconds().toString().padStart(2, '0');
  return hours + ":" + minutes + ":" + seconds;
}
```

Obrázek 14 Kód pro získání aktuálního času

Tento kód je funkce v JavaScriptu nazvaná getCurrentTime(), která slouží k získání aktuálního času a jeho formátování do formátu "hh:mm:ss" (hodiny:minuty:sekundy).

Zde je vysvětlení kódu:

function getCurrentTime() {: Tímto začíná deklarace funkce getCurrentTime(), která není závislá na žádných vstupních argumentech.

var now = new Date();: Vytváří novou instanci objektu Date, která obsahuje aktuální datum a čas.

var hours = now.getHours().toString().padStart(2, '0');: Získává hodiny z aktuálního času (now.getHours()), převede je na řetězec (toString()) a pokud je délka řetězce menší než 2, přidá na začátek nuly (padStart(2, '0')). Tím se zajistí, že hodiny budou vždy dvoumístné.

var minutes = now.getMinutes().toString().padStart(2, '0');: Stejně jako u hodin, tento řádek získává minuty z aktuálního času (now.getMinutes()), převede je na řetězec a přidá na začátek nuly, pokud je délka řetězce menší než 2.

var seconds = now.getSeconds().toString().padStart(2, '0');: Stejný postup jako u hodin a minut pro získání a formátování sekund.

return hours + ":" + minutes + ":" + seconds;: Vrací řetězec s aktuálním časem ve formátu "hh:mm:ss", kde hours, minutes a seconds jsou dvoumístné řetězce reprezentující aktuální hodiny, minuty a sekundy, oddělené dvojtečkami.

Tímto způsobem funkce getCurrentTime() vrátí aktuální čas ve formátu "hh:mm:ss", připravený k použití v kódu.

```
function toggleTimeDisplay() {
  var checkbox = document.getElementById("showTimeCheckbox");
  var timeDisplay = document.getElementById("settime");

  if (checkbox.checked) {
    timeDisplay.style.display = "inline-block";
    timeDisplay.value = getCurrentTime();

    setInterval(function() {
      timeDisplay.value = getCurrentTime();
    }, 1000);
  } else {
    timeDisplay.style.display = "none";
    timeDisplay.value = "00:00:00";
  }
}
```

Obrázek 15 Kód pro zobrazení času

Tato funkce zajišťuje zobrazení a aktualizaci aktuálního času s id "settime".

```
var checkbox = document.getElementById("showTimeCheckbox");
var timeDisplay = document.getElementById("settime");
```

Obrázek 16 Získání odkazů na prvky DOM

Tento segment získává odkazy na HTML prvky, konkrétně zaškrtávací políčko s id "showTimeCheckbox" a element s id "settime", který slouží k zobrazení aktuálního času.

```
if (checkbox.checked) {
```

Obrázek 17 Podmínka pro zobrazení aktuálního času

Tento segment kontroluje, zda je zaškrtávací políčko zaškrtnuté. Pokud ano, provádí se následující kód pro zobrazení a aktualizaci času.

```
timeDisplay.style.display = "inline-block";
timeDisplay.value = getCurrentTime();
```

Obrázek 18 Zobrazení aktuálního času

V tomto segmentu se nastavuje styl zobrazení elementu na "inline-block", aby byl viditelný na stránce, a aktualizuje se jeho hodnota na aktuální čas získaný z funkce getCurrentTime().

```
setInterval(function() {
  timeDisplay.value = getCurrentTime();
}, 1000);
```

Obrázek 19 Aktualizace času s intervalovou funkcí

Tento segment spouští intervalovou funkci, která každou sekundu aktualizuje hodnotu elementu s aktuálním časem.

```
} else {
  timeDisplay.style.display = "none";
  timeDisplay.value = "00:00:00";
}
```

Obrázek 20 Skrytí zobrazení aktuálního času

V tomto segmentu se provádí kód pro případ, že zaškrtávací políčko není zaškrtnuté. Element s časem je skrytý nastavením jeho stylu na "none", a jeho hodnota je nastavena na "00:00:00", tedy na začátek dne.

4.2.3 Řídící kód

```
void handleMessage() {
  // Získání odeslané zprávy

  jas = server.arg("hodnota1");
  color = server.arg("color");
  hod = server.arg("settime");
  display_time = server.arg("display-time");
  display_temperature = server.arg("display-temperature");
  odpocet = server.arg("setcout");
  cas = server.arg("settime");
  pocet_blik = server.arg("display-pocetblik");
  aktualnicas = server.arg("time5");
```

Obrázek 21 Transport dat

V tomto segmentu jsou získány hodnoty z odeslaných parametrů přiřazeny k odpovídajícím stringům pro další použití v jiných částech kódu.

```
void one() {
  leds[0] = CRGB(0, 0, 0);
  leds[1] = CRGB(0, 0, 0);
  leds[2] = CRGB(0, 0, 0);
  leds[3] = CRGB(0, 0, 0);
  leds[4] = CRGB(0, 0, 0);
  leds[5] = CRGB(0, 0, 0);
  leds[6] = CRGB(0, 0, 0);
  leds[7] = CRGB(0, 0, 0);
  leds[8] = CRGB(0, 0, 0);
  leds[9] = CRGB(color1 * jasf, color2 * jasf, color3 * jasf);
  leds[10] = CRGB(color1 * jasf, color2 * jasf, color3 * jasf);
  leds[11] = CRGB(color1 * jasf, color2 * jasf, color3 * jasf);
  leds[12] = CRGB(color1 * jasf, color2 * jasf, color3 * jasf);
  leds[13] = CRGB(color1 * jasf, color2 * jasf, color3 * jasf);
  leds[14] = CRGB(color1 * jasf, color2 * jasf, color3 * jasf);
  leds[15] = CRGB(0, 0, 0);
  leds[16] = CRGB(0, 0, 0);
  leds[17] = CRGB(0, 0, 0);
  leds[18] = CRGB(0, 0, 0);
  leds[19] = CRGB(0, 0, 0);
  leds[20] = CRGB(0, 0, 0);

  FastLED.show();
}
```

Obrázek 22 Funkce pro vypsání čísla

Zde je nadefinována globální funkce se jménem one(). Tato funkce nastavuje barvy pro jednotlivé LED na LED pásku. Čísla od 0 do 20 označují jednotlivé LED na pásku. LED jsou bud vypnuté (0, 0, 0) nebo definované barvy color1, color2 a color3 násobené hodnotou jasf která představuje jas. FastLED.show(); spouští aktualizaci zobrazení LED pásku, čímž se promítnou změny provedené v předchozím segmentu na fyzický LED pásek.

Obrázek 23 Střídání dat na hodinácvh

```
unsigned long currentMillis = millis();
unsigned long currentMillis2 = millis();
```

Obrázek 24 Nastavení aktuálního času

Tento segment získá aktuální čas v milisekundách pomocí funkce millis().Aktuální čas je uložen do proměnných currentMillis a currentMillis2.

```
interval_1 = display_time.toInt() * 1000;
if (interval_1 < 1000) {
  interval_1 = 10000;
}
interval_2 = display_temperature.toInt() * 1000;
if (interval_2 < 1000) {
  interval_2 = 5000;
}
```

Obrázek 25 Nastavení intervalů

Nastavení intervalů pro odpočet a zobrazení teploty. Tento segment převede hodnoty zobrazeného času a teploty na milisekundy a nastaví minimální intervaly, pokud jsou hodnoty menší než 1000 milisekund tak je nastaví na 5000 a 10000.

```
if (countdown != 0) {
  if (currentMillis2 - previousMillis >= interval) {
    previousMillis = currentMillis2;  // Uložení aktuálního času

    if (countdown > 0) {
      countdown--;                // Snížení odpočtu
      Serial.println(countdown);  // Výpis do sériové linky
      int seccountdown = countdown % 60;
      int mincountdown = (countdown - seccountdown) / 60;
      int secjedcout = seccountdown % 10;
      int secdescout = (seccountdown - secjedcout) / 10;
      int minjedcount = mincountdown % 10;
      int mindescout = (mincountdown - minjedcount) / 10;
      if (countdown == 1) {
        int pocetb = pocet_blik.toInt();
        for (A; A <= pocetb; A++) {
          off();
          off2();
          off2();
          off3();
          off4();
          teckaoff();
          delay(200);
          eight();
          eight2();
          eight3();
          eight4();
          tecka();
          delay(200);
        }
      }
```

Obrázek 26 Odpočet

Část kódu, která provádí odpočet. Tento segment provádí odpočet. Pokud countdown není rovno nule a uplynul určitý čas (currentMillis2 - previousMillis >= interval), sníží se countdown. Dále se vypíše zbývající čas a provede se příslušné akce v závislosti na hodnotě countdown.

5 Cenová kalkulace pro velkoformátové hodiny

Cenu hodin jsem se snažil dostat co nejníže ale stále zachovat vysokou kvalitu. Do ceny jsem nezapočítával filament, protože jsou v jednotlivých druzích a značkách takové rozdíly že je to zavádějící. Dále jsem uvádím cenu, za kterou jsem součástky nakupoval já s největší pravděpodobností se dají pořídit jak dráž, tak levněji. Nezapočítávám také dopravu a balné. Dále v ceně není zahrnut cín, kalafuna a dráty.

Název	typ	počet	Cena Kč	Celkem Kč
WS2812B Individually Addressable RGB Led Strip	délka 2m 60LED na 1m	1	180,89	180,89
DS18b20		1	46,12	46,12
DS3231		1	50,76	50,76
Rezistor	470 Ω	1	0,8	0,8
Rezistor	4,7 kΩ	1	0,4	0,4
ESP32-WROOM-32		1	72,4	72,4
šroubovaná svorkovnice		2	23,8	47,6
dutinová lišta		3	6	18
Zdroj	5V 6A	1	232,8	232,8
Celková cena				649,77

Závěr

Záměrem této práce bylo vytvořit chytré velkoformátové hodiny které budou finančně dostupné budou nabízet chytré funkce. Součástí hodin je i Komfortní uživatelská HTML stránka, kde si uživatel může nastavovat parametry hodin.

Vývoj hodin prošel hned několika vývojovými fázemi poslední stávající splňuje všechny nastavené cíle a požadavky. Hodiny fungují podle požadavků. Správně přijímají a zpracovávají data z uživatelského rozhraní. Na data správně reagují.

Seznam použitých zdrojů

Vývojová deska ESP32 2,4GHz Dual-Mode Wi-Fi + Bluetooth [online]. [cit. 2024-03-22]. Dostupné https://dratek.cz/arduino/1581-esp-32s-esp32-esp8266-development-board-2.4ghz-dual-mode-wifi-bluetooth-antenna-module.html

How To Control WS2812B Individually Addressable LEDs using Arduino [online]. [cit. 2024-03-23]. Dostupné https://howtomechatronics.com/tutorials/arduino/how-to-control-ws2812b-individually-addressable-leds-using-arduino/

VYZNEJTE SE VE WI-FI STANDARDECH. Online. 2022. Dostupné z: https://www.adaptech.cz/news/vyznejte-se-ve-wi-fi-standardech [cit. 2024-03-23].

VYZNEJTE SE VE WI-FI STANDARDECH [online]. [cit. 2024-03-23]. Dostupné z: https://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/

WS2812B LED Protocol and LED Microcontrollers. Online. 2019. Dostupné z: https://www.arrow.com/en/research-and-events/articles/protocol-for-the-ws2812b-programmable-led. [cit. 2024-03-23].