|  |  |
| --- | --- |
|  |  |
|  | |
| VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ  BRNO UNIVERSITY OF TECHNOLOGY | |
|  | |
| FAKULTA ELEKTROTECHNIKY  A KOMUNIKAČNÍCH TECHNOLOGIÍ  FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION | |
|  | |
| ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY  DEPARTMENT OF CONTROL AND INSTRUMENTATION | |

Projekt ELEKTRONKOVÝCH HODIN

Projekt xxx

Semestrální PRÁCE BSPC

Semestral thesis BSPC

AUTOŘI PRÁCE Martin Šťastný, Ondřej Vybíral

AUTHORS

BRNO 2019

zadání semestrální práce

1. Připojit hodiny k Wi-Fi síti
2. Běh hodin bez použití externích perifériích (externí RTC)
3. Možnost aktualizace času z NTP serveru
4. Nastavení budíčku
5. Webové rozhraní pro změnu parametrů
6. Zabezpečit přístup do webového rozhraní
7. Navrhnout a realizovat hodiny

Obsah

[1 Úvod 1](#_Toc9494897)

[2 Hardware 1](#_Toc9494898)

[3 Software 3](#_Toc9494899)

[4 Závěr 6](#_Toc9494900)

[Seznam příloh 7](#_Toc9494901)

Seznam obrázků

[Obrázek 1: Hlavní obrazovka (webové rozhraní) 4](file:///C:\git_projects\Projekt_BSPC\Nixie_Clock\Podklady\Dokumentace.docx#_Toc9494886)

[Obrázek 2: Inicializace programu 7](file:///C:\git_projects\Projekt_BSPC\Nixie_Clock\Podklady\Dokumentace.docx#_Toc9494887)

[Obrázek 3: Smyčky 8](#_Toc9494888)

# Úvod

Cílem projektu bylo vytvořit hodiny s použitím elektronek. Pro jejich ovládání byl použít mikrokontrolér WeMos D1 Mini ESP8266 WiFi modul. O veškerou logiku se tedy staral zmíněný mikrokontrolér.

* Co bylo cílem
* Co jsme použili

# Hardware

Jako řídicí člen byl použitý zmíněný WeMos D1 Mini ESP8266 WiFi modul. Jak název napovídá obsahuje Wi-Fi modul, který pracuje na frekvenci 2,4 GHz a podporuje standarty 802.11 b/g/n. Maximální rychlost je 72,2 Mbps. Veškeré piny jsou v 3,3 V logice.

Tabulka 1: Specifikace WeMos D1 Mini ESP8266 WiFi modul

|  |  |
| --- | --- |
| Mikrokontrolér | ESP8266EX |
| USB – Sériový Převodník | CH340G |
| Digitální I/O piny | 9 |
| Analogové vstupy | 1 |
| Flash | 4MB |
| Taktovací frekvence | 80MHz/160MHz |
| Rozměry | |
| Délka | 34,2mm |
| Šířka | 25,6mm |
| Výška | 7mm |
| Váha | 8,26g |

Tabulka 2: Piny a jejich využití WeMos D1 Mini ESP8266 WiFi modul

|  |  |  |
| --- | --- | --- |
| **WeMos piny** | **Funkce** | **ESP8266 piny** |
| TX | TXD | TXD |
| RX | RXD | RXD |
| A0 | Analogový vstup | A0 |
| D0 | I/O | GPIO16 |
| D1 | I/O, SCL | GPIO5 |
| D2 | I/O, SDA | GPIO4 |
| D3 | I/O, 10k pull-up | GPIO0 |
| D4 | I/O, 10k pull-up, LED | GPIO2 |
| D5 | I/O, SCK | GPIO14 |
| D6 | I/O, MISO | GPIO12 |
| D7 | I/O, MOSI | GPIO13 |
| D8 | I/O, 10k pull-down, SS | GPIO15 |
| G | Země | GND |
| 5V | 5V |  |
| 3V3 | 3.3V | 3.3V |
| RST | Reset | RST |

* Pár slov k WeMosu
* funkce v HW, (Martinovy kouzla)
* ?možná schéma?

# Software

Vývoj softwaru lze využít vývojové prostředí Arduino IDE (velmi rozšířené u mikrokontroléru vycházející z konceptu Arduina). Toto prostředí je multiplatformní (Linux, Windows, macOS, webové verze), desktopová verze je programována v jazyce Java. Další alternativou je vývoj v prostředí Visual Studio Code vyvíjené společností Microsoft. Toto prostředí je též dostupné pro výše zmíněné tři operační systémy. Pro úspěšné kompilování souboru v tomto rozhraní je třeba doinstalovat rozšíření pro Arduino.

Samotný program využívá knihovny pro připojení k Wi-Fi, práci s časem, HTTP server, multicast DNS, obsluhu posuvných registrů pro rozšíření perifériích, a nastavení časovače skrze přerušení. Viz tab. 3. Jednotlivé diagramy jsou v příloze. Všechny jsou veřejně dostupné na internetu a lze je stáhnout a použít. Reset mikrokontroléru vede k vyrestování nastavení, má tak nahrazovat tlačítko pro tovární nastavení. V knihovně ESP8266WiFi je třeba upravit v souboru ESP8266WiFiGeneric.cpp u funkce hostByName()parametr na timeout 100 (ms). Konkrétně na hostByName(aHostname, aResult, 100).

Tabulka 3: Použité knihovny

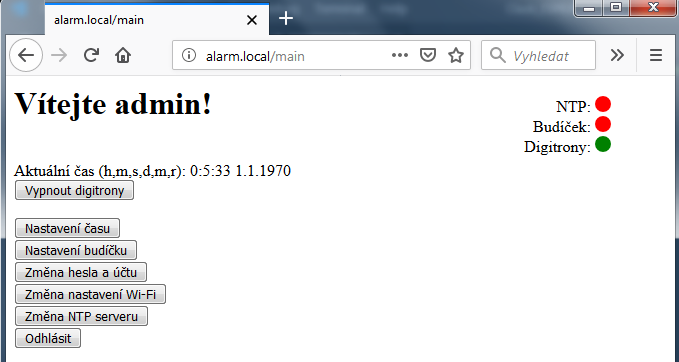
|  |  |
| --- | --- |
| Použité knihovny | Využití |
| TimeLib | Čas |
| ESP8266WiFi | Připojení k Wi-Fi sítím |
| WiFiUdp | Práce s UDP pakety |
| ESP8266WebServer | Provoz http serveru |
| ESP8266mDNS.h | Multicast DNS |
| ShiftRegister74HC595.h | Posuvný registr pro rozšíření vstupů |
| Ticker.h | Časovač skrze přerušení |

* vývojový diagram

Nastavení hodin mělo být, pokud možno snadno dostupné. Bylo zde několik variant, které bylo možné využít. Namátkou je to například specializovaný software, který by jen komunikoval s hodinami přes sériovou linku. My jsme zvolili elegantnější řešení, kdy komunikace probíhá skrze webové rozhraní (přes Wi-Fi). Tím je vyloučena potřeba aplikace a ovládání je tak multiplatformní. Výchozí účet a heslo je: admin, admin. Po přihlášení se uživatel dostane na hlavní stránku (/main), která se každou 1 vteřinu aktualizuje. Viz obr. 1.

Jednotlivé stánky a s jejich příponami jsou v tab. 4. Zabezpečení je u všech stánek řešeno pomocí porovnání zadaného hesla a účtu, které se ukládá to dočasné pracovní proměnné tmpLogin a tmpPassword. To znamená, že pokud je rozhraní odemčené, tak je dostupné pro všechny uživatele, dokud se jeden z nich neodhlásí. Potom vyžaduje znovu po všech uživatelích autorizaci.

Jednotlivé vstupy formulářů pro hodiny a podobně jsou ošetřeny proti nežádoucím vstupům jako jsou znaky: &x@ a jiné.



Obrázek 1: Hlavní obrazovka (webové rozhraní)

Tabulka 4: Seznam použitých stránek

|  |  |  |  |
| --- | --- | --- | --- |
| Funkce pro jednotlivé stránky | Účel | Metoda | Přípona |
| void handleRoot(); | Přihlašovací okno | GET | / |
| void handleMain(); | Hlavní obrazovka | GET | /main |
| void handleNotFound(); | Chybný parametr | - | - |
| void handleLogin(); | Kontrola autorizace | POST | /login |
| void handleSetTime(); | Nastavení času | GET | /settime |
| void handleSetMessage(); | Zpracování dat z času | POST | /setmessage |
| void handleAlarm(); | Nastavení budíčku | GET | /alarm |
| void handleAlarmMessage(); | Zpracování dat z budíčku | POST | /alarmmessage |
| void handleAlarmOff(); | Zpracování dat - vypnutí budíčku | GET | /alarmoff |
| void handleChLogin(); | Změna hesla | GET | /chlogin |
| void handleChLoginMessage(); | Zpracování dat - změna hesla | POST | /chloginmessage |
| void handleLogout(); | Odhlášení | GET | /logout |
| void handleAccess(); | Změna přihlašovacích údajů k Wi-Fi | GET | /access |
| void handleAccessMessage(); | Zpracování dat | POST | /accessmessage |
| void handleNTP(); | Nastavení parametrů NTP | GET | /ntp |
| void handleNTPMessage(); | Zpracování dat z NTP | POST | /ntpmessage |
| void handleDixi(); | Za/vypnutí digitron | GET | /dixi |

Čas se při zapnutí defaultně počítá od 00:00:00 1.1.1970. Program se snaží pravidelně dotazovat NTP (Network Time Protocol) serveru pro aktuální čas. Výchozí server je ntp.cesnet.cz. Interval dotazů na NTP server je ve výchozím nastavení 180 sekund (kvůli demonstraci) v makru NTPSYNC. Časovou zónu je též možné měnit, výchozí je +2.

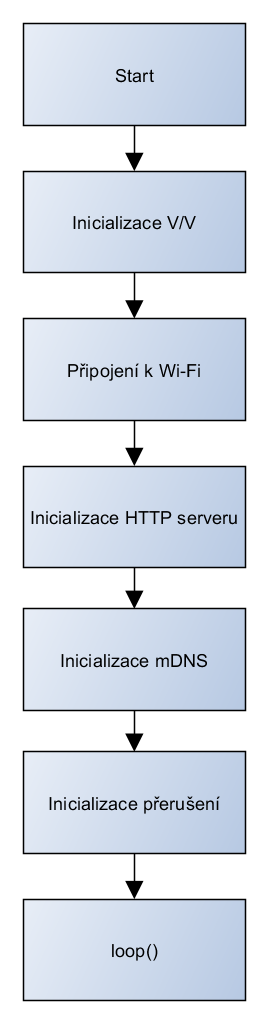
Martinova část

* obrázky rozhraní
* seznam použitých knihoven
* ?Visual Code?
* seznam použitých knihoven
* Popsat feature
  + odhlašování
  + budík
  + NTP – jak často se aktualizuje

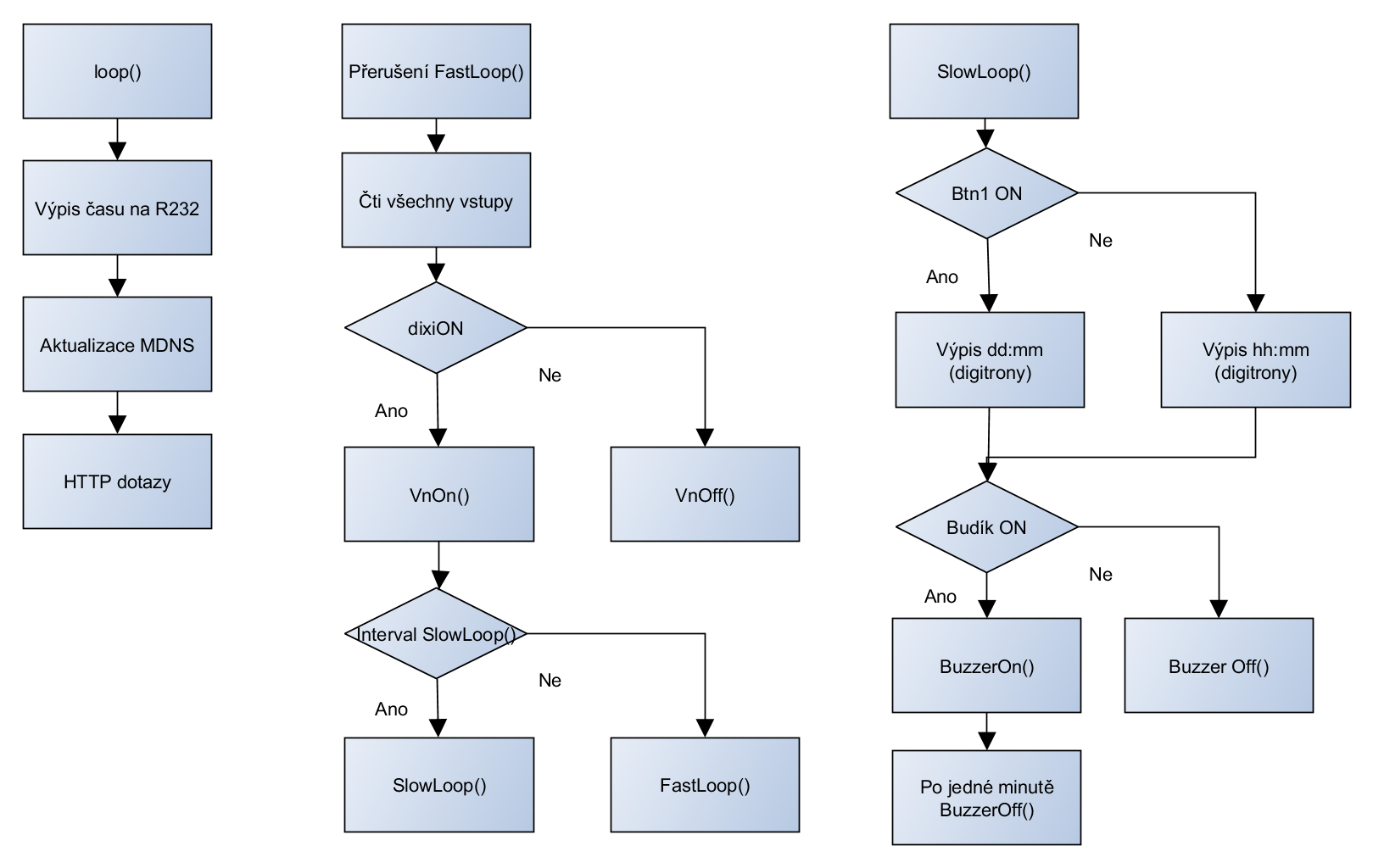
# Závěr

Bude doplněno.

Seznam příloh



Obrázek 2: Inicializace programu



Obrázek 3: Smyčky