Fakulta elektrotechnická
Katedra elektromechaniky a výkonové elektroniky

Bakalářská práce

Bezdrátový programátor procesorů Atmel AVR

Autor práce: Petr Pošvic
Vedoucí práce: Ing. Jaroslav Freisleben, Ph.D.

Plzeň 2016

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta elektrotechnická
Akademický rok: **2015/2016**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Petr POŠVIC**
Osobní číslo:	**E13B0108K**
Studijní program:	**B2644 Aplikovaná elektrotechnika**
Studijní obor:	**Aplikovaná elektrotechnika**
Název tématu:	**Bezdrátový programátor procesorů Atmel AVR**
Zadávající katedra:	**Katedra elektromechaniky a výkonové elektroniky**

Zásady pro vypracování:

1. Seznamte se se současnými typy programátorů mikrokontrolérů Atmel AVR.
2. Porovnejte parametry programovacích protokolů.
3. Seznamte se s dostupnými RF moduly a porovnejte jejich vlastnosti.
4. Navrhněte a realizujte bezdrátový programátor založený na Vámi vybraném RF modulu.
5. Vytvořené technické a programové prostředky podrobně popište.

Rozsah grafických prací:	**podle doporučení vedoucího**
Rozsah kvalifikační práce:	**30 - 40 stran**
Forma zpracování bakalářské práce:	**tištěná/elektronická**

Seznam odborné literatury:

1. **Pinker, J.: Mikroprocesory a mikropočítače.**
2. **Burkhard, M.: C pro mikrokontroléry.**
3. **Firemní literatura a katalogy výrobců daných součástek a obvodů.**
4. **Elektronické informační zdroje.**

Vedoucí bakalářské práce:	**Ing. Jaroslav Freisleben** Regionální inovační centrum elektrotechniky
Datum zadání bakalářské práce:	**15. října 2015**
Termín odevzdání bakalářské práce:	**2. června 2016**

Doc. Ing. Jiří Hammerbauer, Ph.D.
děkan

L.S.

Prof. Ing. Václav Kůs, CSc.
vedoucí katedry

V Plzni dne 15. října 2015

Abstrakt

Práce popisuje současné programátory mikroprocesorů Atmel AVR a poskytuje výčet jednotlivých programovacích protokolů s jejich výhodami a nevýhodami. Dále se práce zabývá realizací bezdrátového programátoru platformy Arduino v podobě shieldu. Je zde popsán výběr bezdrátového modulu. Pak je zde popsána volba vhodného firmware. Je zde vysvětlen způsob nahrání firmware esp-link do WiFi modulu ESP8266 a nastavení tohoto firmware. Dále je zde popsán způsob realizace dálkového programování za použití příkazové řádky a implementace dálkového programátoru do Arduino IDE jako nové třídy na operačním systému Linux. Práce také popisuje návrh samotného hardware včetně schémat a návrhu desky plošných spojů.

Klíčová slova

Atmel, Arduino, Arduino shield, AVR, bezdrátové programování, ESP8266, WiFi

Abstract

Pošvic, Petr. *Wireless programmer of microprocessores Atmel AVR* [*Bezdrátový programátor procesorů Atmel AVR*]. Pilsen, 2016. Bachelor thesis (in Czech). University of West Bohemia. Faculty of Electrical Engineering. Department of Electromechanics and Power Electronics. Supervisor: Jaroslav Freisleben

---

The thesis describes the present microprocessors Atmel AVR programmers and provides a list of various programming protocols with their advantages and disadvantages. Furthermore, the thesis deals with the implementation of wireless programmer on Arduino platform in the form of expanding shield. There is described a selection of wireless module. Then there is how to select the appropriate firmware. There is explained the way to upload the firmware esp-link to a WiFi module ESP8266 and setting of this firmware. Then there is described a method of implementation of remote programming using a command line and implementation of remote programmer to Arduino IDE as a new class on operating system Linux. The thesis also describes the design of the hardware itself, including schematics and PCB design.

Keywords

Atmel, Arduino, Arduino shield, AVR, wireless programming, ESP8266, WiFi

Prohlášení

Předkládám tímto k posouzení a obhajobě bakalářskou práci, zpracovanou na závěr studia na Fakultě elektrotechnické Západočeské univerzity v Plzni.

Prohlašuji, že jsem svou závěrečnou práci vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 270 trestního zákona č. 40/2009 Sb.

Také prohlašuji, že veškerý software, použitý při řešení této bakalářské práce, je legální.

V Plzni dne 31. května 2016

Petr Pošvic

. . . . . .

Podpis

Obsah

Seznam obrázků

Seznam tabulek

Seznam symbolů a zkratek

AP	Access point mode - Mód WiFi zařízení, ke kterému se klienti připojují
AT commands	Jazyk vyvinutý pro komunikaci s modemy
CRC	Cyclic redundancy check - Cyklický redundantní součet
DPS	Deska plošných spojů
EEPROM	Electrically Erasable Programmable Read Only Memory - Paměť sloužící jako úložiště aplikace
FLASH	Paměť obsahující programový kód
I2C	Inter-Integrated Circuit - Sériová sběrnice vyvinutá firmou Philips
IDE	Integrated Development Environment - Vývojové prostředí
IoT	Internet of Things - Internet věcí
ISP	In-System Programming - možnost programovat procesor přímo v obvodu
M2M	Machine to Machine - obecně komunikace mezi přístroji
MISO	Master In, Slave Out - Brána rozhraní SPI
MOSI	Master Out, Slave In - Brána rozhraní SPI
MQTT	Message Queuing Telemetry Transport - ISO standard (ISO/IEC 20922) pro zprávový protokol
PC	Personal computer - Osobní počítač
REST	Representational state transfer - protokol pro práci s informacemi na serveru pomocí jednoduchých HTTP volání.
RX	Receive - Brána pro příjem dat
SPI	Serial Peripheral Interface - Sériová periferní sběrnice
SSID	Service Set Identifier - je v informatice identifikátor bezdrátové sítě Wi-Fi
STA	Station mode - Mód WiFi zařízení, které se připojuje k AP
TCP/IP	Transmission Control Protocol/Internet Protocol - Primární přenosový protokol/protokol síťové vrstvy
TTL	Transistor-transistor-logic - Tranzistorově-tranzistorová logika
TWI	Two-wire Interface - Označení I2C používané firmou Atmel
TX	Transmit - Brána pro odesílání dat

UART	Universal Asynchronous Receiver Transmitter - Rozhraní pro asynchronní sériovou komunikaci
USART	Universal Synchronous Asynchronous Receiver Transmitter - Rozhraní pro synchronní nebo asynchronní sériovou komunikaci
USB	Universal Serial Bus - Univerzální sériová sběrnice
WPA	WiFi Protected Access - Chráněný přístup k Wi-Fi - Zabezpečení WiFi připojení
WEP	Wired Equivalent Privacy- Soukromí ekvivalentní drátovým sítím - Zastaralé zabezpečení WiFi připojení

Kapitola 1

Úvod

Práce shrnuje v kapitole 2 nejdůležitější dostupné programátory Atmel AVR procesorů. Jsou zde uvedeny jejich výhody, nevýhody, výrobci a ceny.

V další kapitole 3 je uveden výčet možných metod programování Atmel AVR procesorů včetně podporovaných protokolů a možných typů programátorů.

Kapitola 4 uvádí popis tří RF modulů dostupných na trhu, které připadali v úvahu pro realizaci dálkového programátoru. Nakonec jsem vybral modul ESP8266 pro jeho nízkou cenu a rozsáhlou podporu open-source komunity.

A v poslední kapitole 5 je popsáno řešení a návrh dálkového (bezdrátového) programátoru platformy Arduino, která je osazena právě mikroprocesorem Atmel AVR. Programátor je řešen za pomoci bezdrátového mostu sériového rozhraní s podporou programovacího protokolu STK500v1. Firmware, který se stará o tento přenos je nahrán do FLASH paměti modulu ESP8266, který je umístěn v patici navrženého shieldu (expandéru) Arduina.

Mimo firmware od vývojáře *JeeLabs* [6], který jsem použil ve svojí implementaci, se tématu dálkového programování Arduina věnuje ještě *Joshua Newell* [11], který řeší programátor taktéž v podobě shieldu, ale pro obousměrnou komunikaci zvolil bluetooth modul HC-05.

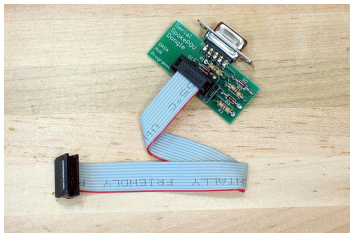
Kapitola 2

Typy programátorů Atmel AVR

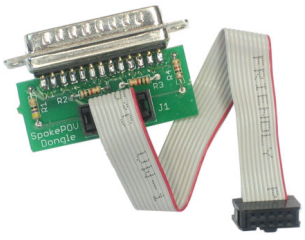
Na trhu je mnoho typů programátorů. Zmíním zde pouze ty nejdůležitější.

2.1 Jednoduché / Základní / Bit-bang programátory

Jednoduché / Základní / Bit-bang programátory jsou jednoduché a levné programátory, které se připojují přímo k sériovému (obr. 2.1) nebo paralelnímu (obr. 2.2) portu počítače. Program v počítači se pak stará o posílání dat do paměti čipu.



Obrázek 2.1: Sériový programátor



Obrázek 2.2: Paralelní programátor

Některé z těchto programátorů jsou osazeny buffer čipem, který zajišťuje odpojení programátoru při spuštění programu, a tak je možné využít piny pro programování na něco jiného.

Bez tohoto čipu je potřeba pro využití programovacích pinů programátor fyzicky odpojit.

Výhody: Jsou levné (10$ - 20$). Protože se software stará o přenos dat, nehrozí potřeba upgradu a ani nehrozí možnost nekompatibility programátoru.

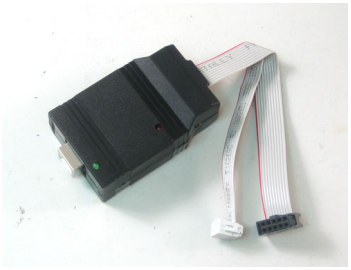
Nevýhody: Je potřeba PC se sériovým nebo paralelním portem. Při práci na větším napětí než 5 V hrozí spálení portu, protože existuje více standardů rozhraní programátorů.

Výrobce: např. Adafruit

2.2 ISP programátory

Tyto programátory se připojují buď do USB, a nebo do sériového portu. Jsou o poznání sofistikovanější než předešlý typ. Nejběžnějším typem je AVRISP a AVRISPv2.

AVRISPv1 (obr. 2.3) se připojuje k sériovému portu počítače a na druhé straně má 6-pinový a 10-pinový konektor pro ISP programování (viz dále)



Obrázek 2.3: Programátor AVRISPv1

AVRISPv2 (obr. 2.4) se připojuje k USB, ale na druhé straně je pouze 6-pinový konektor pro ISP.

Existuje mnoho variant a ještě více domácích provedení programátorů.

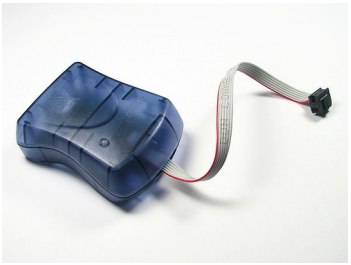
Výrobce: Digikey.com a Mouser.com

Cena: 36$

2.3 Vývojové desky

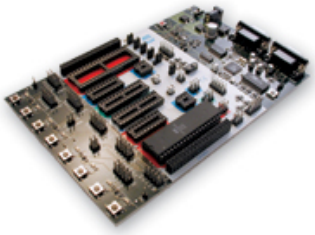
Na trhu je k dispozici mnoho vývojových desek. Zmiňuji zde dvě nejpopulárnější a oficiálně podporované Atmelem.

První je vývojové prostředí STK500 (obr. 2.5), které podporuje všechny verze Atmel procesorů. Je vybavena různými paticemi pro jednotlivé varianty čipů. Má nastavitelné



Obrázek 2.4: Programátor AVRISPv2

taktovtování. Tento programátor je přímo podporován softwarem AVRStudio (programovací prostředí Atmelu).



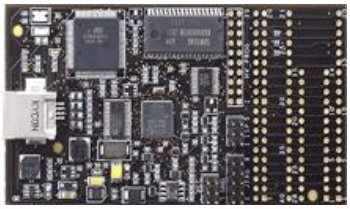
Obrázek 2.5: Programátor STK500

Bohužel je k PC připojitelný jen přes sériový port a nedisponuje USB rozhraním. I když je to přímo programátor, přesto v tomto případě probíhá komunikace za pomocí PC a to přes protokol STK500 (viz. dále).

Výrobce: Digikey.com a Mouser.com

Cena: 80$

Další je AVR Dragon (obr. 2.6), což je emulátor i programátor. Používá se častěji jako ISP programátor než jako vývojová deska. Je vybaven oběma konektory pro ISP programování (6-pin a 10-pin).



Obrázek 2.6: Programátor AVR Dragon

2.4 Bootloadery (zavaděče)

V posledních letech přidali návrháři mikroprocesorů čipům schopnost si programovat vlastní flash paměť (self-programming). To znamená, že program vypálený do čipu může měnit vlastní program. Tento přístup má nevýhodu v tom, že program může přepsat sám sebe, poškodit se, a tak způsobit nefunkčnost mikroprocesoru. V určitých případech je ale tento přístup výhodou.

Bootloader (zavaděč) je krátký program vypálený do čipu, který umožňuje komunikaci přes USB nebo seriové rozhraní, je schopen stáhnout nový firmware bez použití programátoru. Update firmware na mobilních telefonech, MP3 přehrávačích či televizích je řešen touto cestou.

Arduino používá bootloader, který je nahrán do čipu již při výrobě. Bootloader je chráněn, takže nemůže přepsat sám sebe. Nahrání programu pak probíhá pouze připojením Arduina k PC pomocí USB a softwarového rozhraní Arduino IDE, které rozpozná zařízení a realizuje upload.

Nevýhodou bootloaderů je, že zabírají kus paměti, která by jinak mohla být využita pro samotnou funkci mikroprocesoru. Další nevýhodou je, že pro vypálení bootloaderu je potřeba ISP programátor (problém vejce a slepice). Bootloadery neumožňují změnit *fuses* (pojistky) na mikroprocesoru. Pojistky jsou konfigurační bity mikroprocesoru, kterými lze na mikroprocesoru nastavit adresování, taktování, atd.

Kapitola 3

Programovací protokoly a jejich parametry

3.1 ISP (In System Programming)

Podporováno: Většina AVR mikroprocesorů
Programátory: AVRISP v1 a v2, JTAG v2, STK500, STK600, Dragon, AVRISP klony, AVRONE, AVR910

ISP je nejběžnější metoda používaná k programování flash, EEPROM, fuse (pojistek) pro celou škálu AVR mikroprocesorů. ISP programování může probíhat na velmi vysokých taktovacích frekvencích (cílový procesor by měl běžet minimalně na čtyřnásobku rychlosti ISP programátoru). Tato metoda je nejčastěji využívána v hobby aplikacích.

3.2 JTAG

Podporováno: —
Programátory: JTAG-ICE, JTAG-ICE v2, Dragon, klony JTAG-ICE, AVRONE, STK600 (pouze programování)

JTAG je *debugovací* (ladící) systém a ne programovací metoda. JRAG umožňuje naprogramovat mikroprocesor, který tuto metodu podporuje.

JTAG je nástroj pro ladění AVR mikroprocesoru, který umožňuje změnu a čtení stavu procesoru za běhu v obvodu. Umožňuje uživateli kdykoliv zastavit běh programu, měnit vnitřní registry AVR, atd.

3.3 DebugWire

Podporováno: menší AVR mikroprocesory
Programátory: JTAG-ICE v2, Dragon, AVRONE

Podobně jako JTAG se jedná o ladící nástroj a ne o programovací rozhraní, ale opět umožňuje nahrání programu na AVR mikroprocesory, které tuto metodu podporují. DebugWire rozhraní používá pouze jeden pin AVR mikroprocesoru (RESET) pro veškerou komunikaci, což se velmi hodí u AVR procesorů, které disponují málo piny (řada TINY).

3.4 Bootloader

Podporováno: většina nových AVR mikroprocesorů
Programátory: —

Opět se technicky nejdná o programovací metodu. Bootloader je malý program umístěný v uživatelsky nastavitelné paměti přímo na flash AVR mikroprocesoru. Využívá možnosti u novějších AVR programovat sebe sama daty nahrávnými z externího zdroje. Zdrojová data pro bootloader mohou být umístěna kdekoliv (SD karta, flash, PC, ...), ale nejčastěji se používá komunikace mezi PC a RS-232 seriovým portem mikroprocesoru.

Bootloadery jsou omezeny pamětí, neboť se dělí o paměť mikroprocesoru se samotným programem, který chceme do AVR nahrát.

Neumí měnit *fuses* (pojistky). Pojistky jsou konfigurační bity, kterými lze na mikroprocesoru nastavit adresování, taktování, atd.

3.5 HVPP (High Voltage Parallel Programming)

Podporováno: většina AVR mikroprocesorů bez označení TINY
Programátory: STK500, STK600, Dragon, AVRONE

HVPP je metoda, která se používá velice zřídka, ale přesto se běžně používá k „oživení“ mikroprocesorů, u kterých byly špatně nastaveny pojistky při programování jinou metodou.

Během HVPP metody je na RESET pin programovaného procesoru přivedeno nezvykle vysoké napětí 12V, které aktivuje vnitřní programovací paralelní obvody mikroprocesoru. RESET pin je jediný pin (u procesorů podporujících HVPP), který může být bezpečně zatížen takovouto úrovní napětí.

3.6 HVSP (High Voltage Serial Programming)

Podporováno: většina AVR mikroprocesorů s označením TINY
Programátory: STK500STK600, Dragon, AVRONE

HVSP (High Voltage Serial Programming) je podobná metoda jako HVPP. Tato metoda využívá sériového rozhraní. Používá se u mikroprocesorů, které nemají dost pinů pro paralelní programování.

3.7 PDI

Podporováno: AVR mikroprocesory s označením XMEGA
Programátory: STK600, Dragon, AVRONE, JTAG v2, AVRISPv1

PDI je nová programovací metoda založená na DebugWire protokolu pro mikroprocesory řady XMEGA. U žádných jiných 8-bitových procesorů než této řady se nevyužívá.

3.8 TPI

Podporováno: 6-pinové TINY AVR mikroprocesory (ATINY10, ...)
Programátory: STK600, Dragon, AVRISPv2

TPI se používá k programování novějších mikroprocesorů řady TINY. Stejně jako DebugWire používá RESET pin pro komunikaci, ale protože procesory TINY nemají implementované debugovací rozhraní, používá se TPI protokol na třech pinech v half-duplex režimu. RESET je potřeba napájet 12V – toto je zatím podporováno pouze u novějších programátorů STK600.

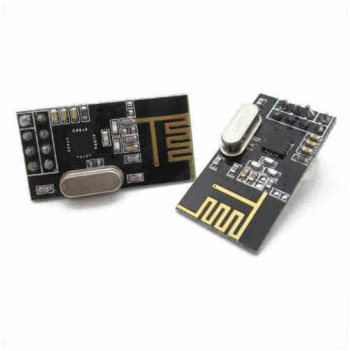
Kapitola 4

Současné RF moduly na trhu

Na trhu je k dispozici nepřeberné množství RF modulů v široké škále kvality a ceny. Výběr jsem nakonec zúžil na následující tři moduly: nRF24L01+, HC-05 a ESP8266. Moduly disponují obousměrnou komunikací, dostatečnou kapacitou přenosu a v neposlední řadě také nízkou cenou, neboť při realizaci dálkového programátoru AVR mikroprocesoru jsem chtěl dosáhnout co největší dostupnosti výsledného produktu. Všechny zde zmíněné moduly jsem zkoušel, ale pro realizaci dalkového programátoru nakonec připadaly v úvahu pouze HC-05 (4.2) a ESP8266 (5.2) pro snadné řešení resetu potřebného k inicializaci bootloaderu Arduina.

4.1 nRF24L01+

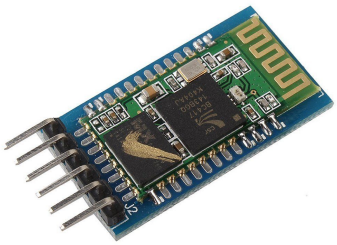
- Maximální přenosová rychlost je 2 Mbps
- ULP (Ultra low power)
- Využívá pásmo ISM (Industrial, Scientific and Medical)
- Napájení 1,9 V až 3,6 V
- Špičkové proudy RX/TX jsou menší než 14 mA
- Power-down mód spotřebovává v řádech μA
- SPI rozhraní
- 125 kanálová komunikace
- CRC detekce chyb
- Vestavěný napěťový regulátor
- Dosah asi 50 m
- Cena 1$ za kus



Obrázek 4.1: Proprietární 2,4 GHz modul nRF24L01+

4.2 HC-05

- Napájení 3.6 V až 6 V
- Proudy při vyhledávání zařízení dosahují 30 mA
- Proudy při spárování jen 10 mA
- Může být přímo připojen na sériové rozhraní mikrokontroléru
- Dosah 10 m maximálně
- Nastavování modulu probíhá přes sériové rozhraní pomocí AT příkazů
- Cena kolem 3$



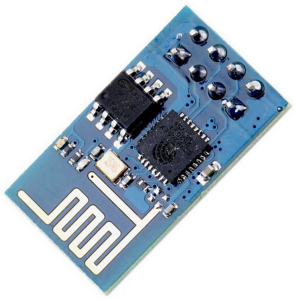
Obrázek 4.2: Bluetooth modul HC-05

4.3 ESP8266

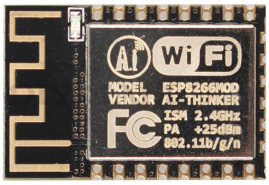
Modulů ESP8266 je na trhu celá řada (od verze 01 až po verzi 12 viz obrázek 4.4). Vzájemě se moduly liší v možnosti přístupu k pinům procesoru, velikostí vnitřní paměti

flash a možností připojení externí antény. Podrobnějším informacím k modulu se věnuji v kapitole 5.2. Právě tento modul jsem vybral pro realizaci dálkového programátoru.

- 802.11 b/g/n
- Wi-Fi Direct (P2P), soft-AP
- Integrovaný TCP/IP protocol
- Spotřeba při power-down módu je míň jak 10μA
- Integrovaný 32-bit CPU
- SPI rozhraní
- UART rozhraní
- Dosah až 25 m
- Cena 2$



Obrázek 4.3: WiFi modul ESP8266-01



Obrázek 4.4: WiFi modul ESP8266-12

Kapitola 5

Realizace dálkového programátoru

V této kapitole popíšu samotnou realizaci dálkového programátoru. První část se zabývá návrhem hardware a popisem funkcí samotné desky plošných spojů. Ve druhé části bude popsaná instalace vlastního firmware do modulu. A v poslední části popíšu samotné dálkové programování.

5.1 Arduino

Dálkový programátor jsem se rozhodl realizovat na platformě Arduino.

Arduino [9] je obchodní název pro malý jednodeskový počítač osazený osmibitovým procesorem z rodiny AVR od firmy Atmel a dalšími podpůrnými obvody. Jedná se o značně rozšířenou, uživatelsky oblíbenou a otevřenou platformu.

Každá deska má většinu I/O pinů přístupných přes standardizované patice, do kterých lze snadno připojit rozšiřující obvody, kterým se říká *shieldy*. V této podobě (tj. v podobě shieldu, který lze přímo zapojit do Arduina) následně navrhuji samotný hardware.

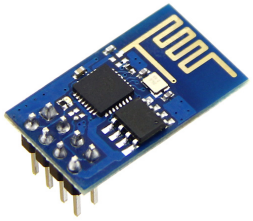
Obrázek 5.1: Arduino UNO

Hlavní procesor, který je uživatelsky programovatelný, již má *bootloader* (kód, který se po spuštění postará o základní nastavení mikrokontroléru, jako jsou interní časovače, nastavení rozhraní USART a další) a nastavené potřebné *fuses* bajty (těmi se nízkoúrovňově nastavují některé vlastnosti čipu). Díky tomu se uživatel nemusí starat o detaily a své programy píše v jazyce podobném C/C++.

Ačkoliv je Arduino připojeno k počítači pomocí rozhraní USB, je softwarově simulována sériová komunikace přes linku RS-232.

5.2 WiFi modul ESP8266

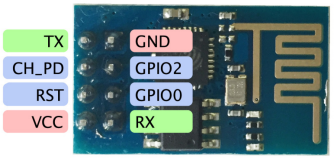
Pro realizaci bezdrátového připojení jsem nakonec vybral modul ESP8266 ve verzi ESP-01 [5]. Tento malý a extrémně levný modul vyráběný firmou Espressif umí vytvářet jednoduché TCP/IP spojení. Nízká cena, možnost nepoužívat další externí komponenty a malá velikost učinily tento modul jedním z nejoblíbenějších v komunitě „bastlířů“ a komunitě zabývající se *IoT* (*Internet of Things*). Modul dokonce získal cenu *Internet of Things 2015/16 Awards* [5].



Obrázek 5.2: Wifi modul ESP-01

5.2.1 Podrobnější popis modulu ESP8266

- 32-bit RISC CPU: Tensilica Xtensa LX106 (taktováno na 80 MHz)
- 64 kB instrukční RAM, 96 kB data RAM
- Externí QSPI flash - 512 kB až 4 MB (podporováno až 16MB)
- IEEE 802.11 b/g/n Wi-Fi
- WEP a WPA/WPA2 kódování
- SPI, I2C,
- UART



Obrázek 5.3: Přiřazení pinů na modulu ESP8266-01

Tabulka 5.1: Popis pinů modulu ESP8266

TX	transmiter line
CH_PD	chip power down
RST	reset
VCC	+3,3V
GND	0V
GPIO2	IO pin 2
GPIO0	IO pin 0
RX	reciever line

5.3 Firmware pro modul ESP8266

Díky oblíbenosti ESP8266 a *opensource* platformě existuje na internetu nepřeberné množství *firmwarů* pro různé účely.

Například:

- Espressif - základní firmware poskytovaný výrobcem
- rozšířený v0.95 AT Firmware - založeno na základním firmwaru a SDK 0.93
- NodeMCU - firmware pro spouštění Lua skriptů
- MicroPython - firmware umožňující spouštět Python 3 skripty
- ESP8266Basic - interpretr jazyka Basic
- espduino - klient MQTT
- a tak dále

Já zvolil *esp-link* firmware od autora JeeLabs [6], který umožňuje vytvořit transparentní most mezi WiFi a sériovým rozhraním a navíc řeší i automatický reset Arduina pro aktivaci bootloaderu a tudíž následného naprogramování mikrokontroléru. Kromě toho pokrývá ještě několik zajímavých uplatnění jako je MQTT a REST API, což umožňuje značně rozšířit škálu bezdrátové komunikace Arduina (ať už s jinými Arduiny, Raspberry Pi i třeba s mobilními telefony).

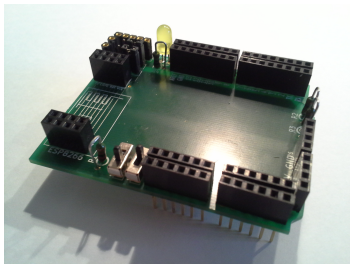
Firmware má tyto funkce:

- transparentní most mezi WiFi a sériovým rozhraním
- možnost programování mikroprocesoru AVR (stejně tak Arduina a dokonce i jiného ESP8266 modulu) přes WiFi
- vestavěný *STK500v1* AVR programátor
- MQTT klient pro větší možnost komunikace mikrokontroléru s internetem (publish / subscribe)
- HTTP REST API rozhraní (např. pro předávání dat měřených hodnot na servery k tomu určené)
- konzole pro debugování

5.4 Návrh hardware shieldu pro Arduino

Návrh zařízení (shieldu) je velice jednoduchý (viz příloha A.1) z důvodu udržení co nejnižší pořizovací ceny, aby byl shield pro případné uživatele cenově snadno dostupný. Ze stejného důvodu není zařízení vybaveno USB/TTL převodníkem, protože po prvotní instalaci firmwaru není již převodník pro běžný provoz potřebný, a tak stačí uživateli pouze jeden, kterým může esp-link firmware nahrát do více takových shieldů.

Shield zprostředkovává určitý uživatelský komfort pro přepínání stavů důležitých pinů modulu ESP8266 při běžném provozu a při instalaci firmwaru. Dále zajišťuje správné propojení jednotlivých komponent (sériového rozhraní Arduina, napájení, atd.), pak umožňuje monitorování modulu ESP8266 bez nutnosti jeho odpojení na vedlejší patici a také umožňuje přesměrování sériového rozhraní modulu ESP8266 na dva nezávislé piny, které jdou v rámci jakéhokoliv prototypu propojit dráty buď s jiným zařízením a nebo s jinými piny Arduina (např. pro potřeby virtuálního debugovacího portu).

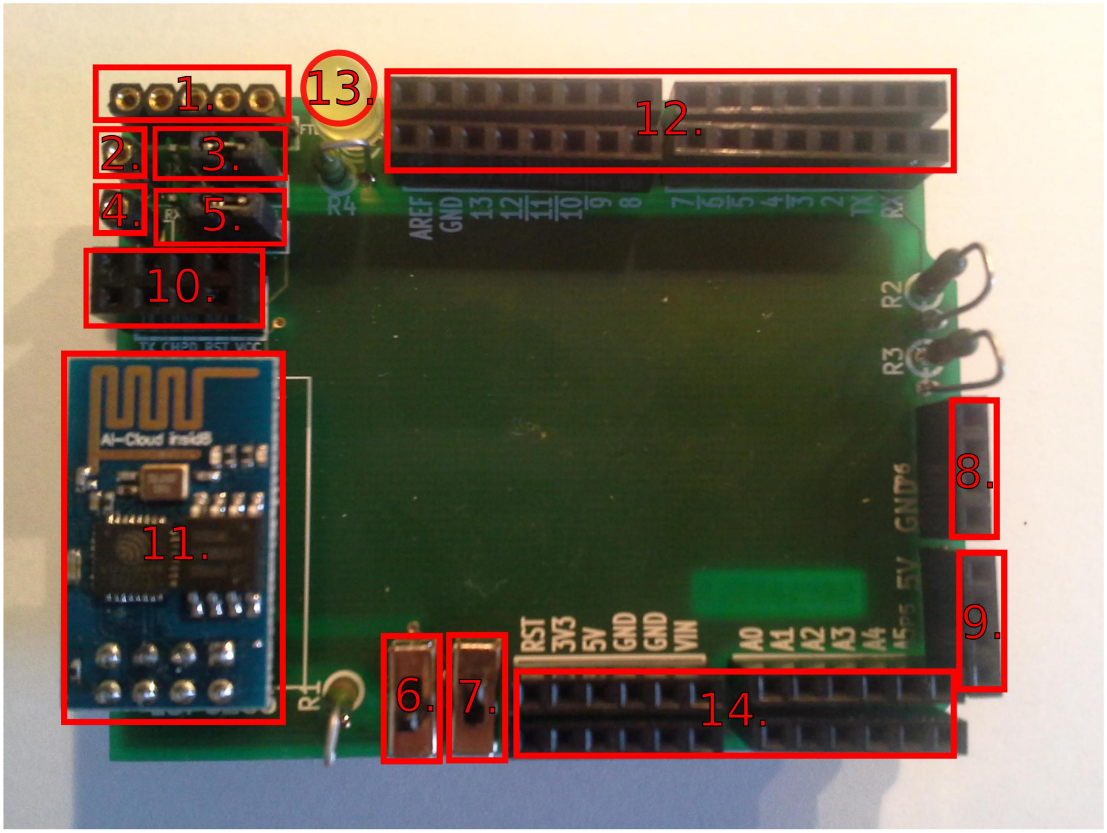


Obrázek 5.4: Osazený finální produkt

5.4.1 Popis shieldu a jeho funkcí

Obrázek 5.5 znázorňuje polohu jednotlivých komponent na shieldu a následující výčet popisuje jejich funkci (čísla ve výčtu korespondují s čísly na obrázku 5.5):

1. Konektor pro připojení USB/RS-232 TTL převodníku (viz obrázek 5.6 a 5.7).
2. viz bod 3.
3. Jumper, který v poloze ON propojuje TX pin z ESP8266 na RX pin Arduina a v poloze OFF propojuje tentýž pin se jednopinovou paticí označenou v této tabulce číslem 2. Tato funkce je zde z důvodu možnosti přesměrovat sériové rozhraní modulu do jiné elektroniky, popř. na další virtuální sériové rozhraní Arduina.
4. viz bod 5.
5. Jumper, který v poloze ON propojuje RX pin z ESP8266 na TX pin Arduina a v poloze OFF propojuje tentýž pin s jednopinovou paticí označenou v této tabulce číslem 4. Tato funkce je zde z důvodu možnosti přesměrovat sériové rozhraní modulu do jiné elektroniky, popř. na další virtuální sériové rozhraní Arduina.
6. ON / OFF přepínač pro možnost vypnutí modulu ESP8266.
7. RST / FLASH přepínač kde lze určit, jestli má být modul ESP8266 v režimu flash firmwaru (poloha FLASH, viz kapitola 5.5), a nebo jestli má mít modul správu nad Arduino RESET pinem pro možnost dálkového programování.
8. Konektor 0 V pro možnost napájení prototypové elektroniky.
9. Konektor 5 V pro možnost napájení prototypové elektroniky.
10. Patice jako v následujícím bodu 11. (pro možnost na Arduinu nezávislého ovládání modulu.
11. WiFi modul ESP8266 zasazený do patice.
12. Běžné rozhraní Arduina (vnější řada pro zachování možnosti expanze o další shieldy a vnitřní řada pro možnost připojení prototypové elektroniky).
13. Standardní LED připojená na běžný Arduino pin D13 používaná jako signalizace.
14. Stejně jako bod 12.



Obrázek 5.5: Poloha jednotlivých komponent shieldu

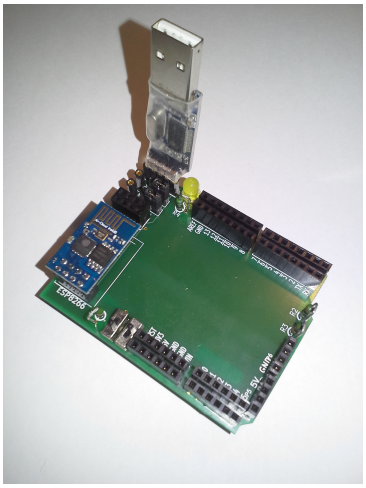
5.5 Instalace esp-link na ESP8266

Pro instalaci firmwaru je modul vybaven paticí pro připojení levného a dostupného modulu USB/RS-232 TTL. Modul je na obrázku 5.6. Instalace se provádí na samotném shieldu (bez připojeného Arduina). Napájení obvodů je zprostředkováno pomocí USB kabelu přímo z USB/RS-232 TTL modulu, který propojíme prodlužovacím USB kabelem s PC.

Obrázek 5.6: Modul USB/RS-232 TTL

5.5.1 Zapojení pro instalaci firmwaru

Pro instalaci firmwaru je potřeba přepnout přepínač 6 do polohy `ON` (to umožní provoz modulu ESP8266) a přepínač 7 do polohy `FLASH` (viz obrázek 5.5). Přepínač 7 připojí pin GPIO0 modulu ESP8266 na GND a tím modul přejde do módu instalace. Poloha obou jumperů nemá na instalaci vliv.



Obrázek 5.7: Modul USB/TTL připojený k shieldu

5.5.2 Instalace na operačním systému linux

Operační systém Linux jsem zvolil pro snadnou obsluhu na systémové úrovni, opensource formátu a snadné dostupnosti.

Veškeré příkazy se zadávají z příkazové řádky. Všechny funkce příkazů podrobně nerozepisuji, neboť je to mimo možnosti rozsahu této práce. Vše je případně k dohledání v nápovědě v systému [8].

Pro instalaci je potřeba Python 2.7 nebo novější.

Předně instalujeme pomocí příkazu `pip` poslední stabilní verzi *esptool.py* (to je Python skript, který umí komunikovat s bootloaderem v modulu ESP8266). Příkaz v následující podobě nainstaluje do počítače určený Python balíček.

```
$ pip install esptool
```

Dále pomocí sekvence příkazů nainstalujeme *esp-link* firmware na ESP8266. První dvojice příkazů `curl` a `tar` zkopíruje a rozbalí z github repozitáře poslední verzi firmwaru. Druhý příkaz nás přesune do vzniklého adresáře. Poslední příkaz nainstaluje firmware na modul ESP8266 (Tedy za předpokladu, že je zařízení připojeno k portu `/dev/ttyUSB0`).

```
$ curl -L https://github.com/jeelabs/esp-link/releases/download/
v2.2.beta2/esp-link-v2.2.beta2.tgz | tar xzf -
```

```
  $ cd esp-link-v2.2.beta2

  $ esptool.py --port /dev/ttyUSB0 --baud 460800 write_flash -fs
4m -ff 40m 0x00000 boot_v1.5.bin 0x1000 user1.bin 0x7E000 blank.bin
```

Následně by měla proběhnout instalace a výpis, který nás informuje o zápisu jednotlivých bloků. Bude vypadat takto:

```
Connecting...
Erasing flash...
Wrote 3072 bytes at 0x00000000 in 0.1 seconds (285.6 kbit/s)...
Erasing flash...
Wrote 304128 bytes at 0x00001000 in 8.5 seconds (285.6 kbit/s)...
Erasing flash...
Wrote 4096 bytes at 0x0007e000 in 0.1 seconds (306.6 kbit/s)...

Leaving...
```

5.6 Nastavení esp-link firmwaru

Po úspěšné instalaci můžeme přejít k nastavení samotného firmwaru esp-link, který lze provádět následujícími dvěma způsoby. Za prvé se shieldem nepřipojeným k Arduinu a napájeným přes USB/TTL modul přes USB kabel (obrázek 5.7). Za druhé se shildem, již zasazeným do patice Arduina, odpojeným USB/TTL modulem a Arduinem napájeným buď ze sítě adaptérem, nebo USB kabelem (obrázek 5.9).

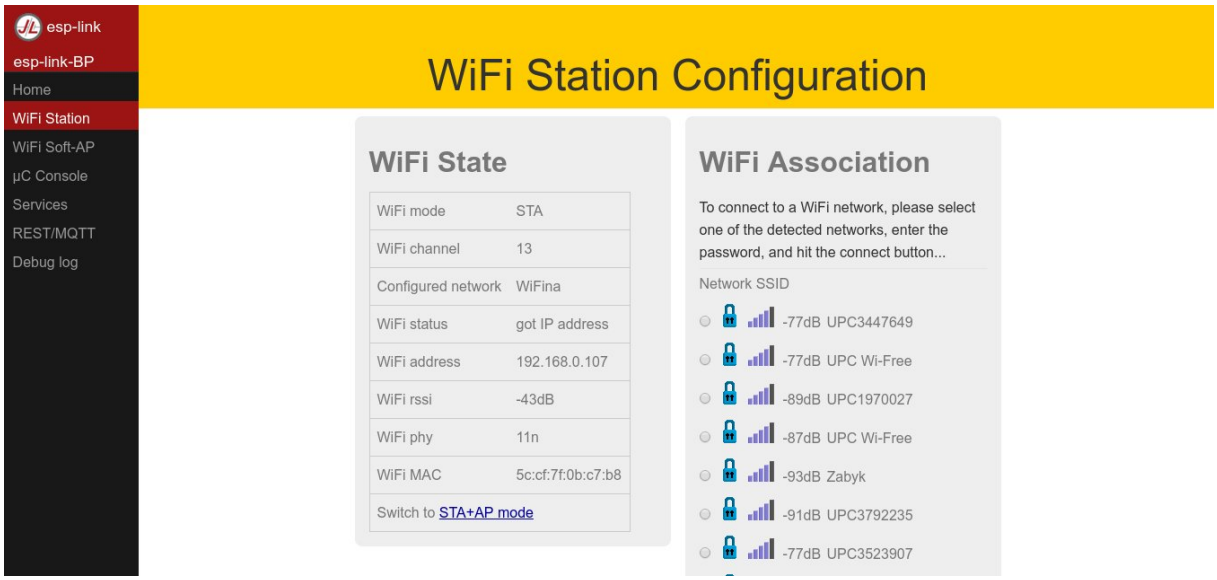
Pro užívání shieldu jako dálkového programátoru i pro nastavení firmwaru je potřeba, aby byl přepínač 6 nastaven do polohy `ON` a přepínač 7 do polohy `RST` (obrázek 5.5). Poloha obou jumperů musí být nastavena na `ON`.

Firmware esp-link se chová tak, že pokud nenajde domovskou WiFi síť, přepne se do režimu `AP` (`Access point`), aby bylo možno se na něj připojit a nakonfigurovat ho. Stejně tak se chová po první instalaci, protože žádná domovská síť doposud není nastavena.

Na modul v režimu AP se připojíme přímo počítačem nebo telefonem. Přístupový bod (AP) vytvořený firmwarem esp-link bude mít SSID ve tvaru `ESP_012ABC` a do běžného prohlížeče zadáme `http://192.168.4.1/`.

Firmware má webové rozhraní pro snadnější konfiguraci. Na zobrazené stránce (obrázek 5.8) vybereme SSID sítě, do které chceme, aby byl modul ESP8266 připojen, a do příslušného pole zadáme heslo (je-li potřeba).

Po kliknutí na tlačítko `Connect` se nastavení uloží a modul ESP8266 automaticky přejde z AP módu do `STA` módu (`stacionary mode`), ve kterém se připojí na námi navolenou WiFi síť. Od této chvíle již není modul dostupný na adrese `http://192.168.4.1/`, ale na nové lokální adrese v námi vybrané síti, která bude ve tvaru `http://192.168.0.xxx/`, kde



Obrázek 5.8: Úvodní stránka webového rozhraní pro konfiguraci WiFi připojení

číslo `xxx` musíme dohledat například v Client Listu v nastavení routeru. Je samozřejmě potřeba na modul přistupovat z WiFi sítě, kterou jsme modulu zvolili v nastavení. Tzn. pokud jsme prováděli nastavení modulu z PC, ze kterého chceme modul programovat, musíme se odpojit z AP vytvořeného modulem (i když ten stejně zanikne nastavením připojení) a připojit se na námi zvolené SSID.

Proběhlo-li vše správně, můžeme provést další nastavení z adresy `http://192.168.0.xxx/`. Ve webovém rozhraní firmwaru nahraném na modulu je mnoho možností:

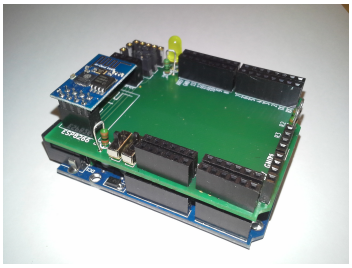
- Změna Hostname
- Konfigurace pinů (který pin simuluje RESET, ...). Pro tento hardware nemusíme konfiguraci měnit (pro ESP8266-01 jsou již piny přednastaveny)
- Je zde možnost dálkového resetu modulu i připojeného Arduina
- Přístup do konzole (pro debugging) i s možností odesílání dat
- Nastavení MQTT (server, heslo, atd.)
- Nastavení REST API
- a další

5.7 Dálkové programování připojeného Arduina

Vše je nyní nastaveno a můžeme přistoupit k samotnému dálkovému naprogramování Arduina.

Pro užívání shieldu jako dálkového programátoru i pro nastavení firmwaru je potřeba, aby byl přepínač 6 nastaven do polohy `ON` a přepínač 7 do polohy `RST` (obrázek 5.5). Poloha obou jumperů musí být nastavena na `ON`.

Shield zasadíme do patice Arduina (obrázek 5.9) a připojíme k napájení. Doporučuji připojit Arduino na 9 V (jak uvádí výrobce), ale vše fungovalo i na 5 V s Arduinem připojeným přes USB. Po zapnutí se modul automaticky připojí do nastavené sítě a Arduino začne provádět program, který je v něm nahraný. Shield nijak neovlivňuje a ani nepoužívá piny samotného Arduina a tak zůstává veškerá kompatibilita s jinými shieldy zachována.



Obrázek 5.9: Shield zasazený do patice Arduina

5.7.1 Odeslání .hex souboru pomocí Arvdude

`Avrdude` vzniknul jako proprietární projekt ISP programátoru a posléze byl uvolněn jako opensource. Jedná se program spouštěný z příkazové řádky, který podporuje širokou škálu programátorů. Od levných ISP (připojovaných k sériovému nebo paralelnímu portu počítače), přes ISP programátory osazenými buffer procesorem, až po složitější desky typu Atmel STK500 nebo Atmel JTAG ICE v2. Podporuje .hex soubory (Intel) i samotné binární soubory. Je možné použít autodetekci vstupního formátu. Veškeré možnostni Avrdude [7] zde nebudu uvádět, neboť je mimo rozsah této práce.

Samotné nahrání zkompilovaného programu v podobě .hex souboru (lze ho získat přímo z menu Arduino IDE) provedeme z konzole příkazem:

```
$ avrdude -DV -patmega328p -Pnet:192.168.0.107:23 -carduino -b115200
-Uflash:w:soubor.hex:i
```

Parametr `-D` zakazuje vymazání paměti flash před programováním. Parametr `-V` zakazuje verifikaci nahraného programu. Dále `-patmega328p` určuje typ AVR mikroprocesoru (v případě Arduina se jedná o ATMEGA328), pak `-Pnet:192.168.0.107:23` specifikuje port, který se použije k programování (je potřeba zadat správnou adresu, na které se modul ESP8266 připojuje). Parametrem `-carduino` specifikujeme typ programátoru (zde se jedná o bootloader Arduina). Parametrem `-b115200` nastavíme přenosovou rychlost a na závěr parametrem `-Uflash:w:soubor.hex:i` určíme, že chceme zapsat zkompilovaný *soubor.hex* do paměti flash.

Výstup Avrdude bude vypadat následovně:

```
ioctl("TIOCMGET"): Inappropriate ioctl for device
ioctl("TIOCMGET"): Inappropriate ioctl for device

avrdude: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.03s

avrdude: Device signature = 0x1e950f
avrdude: reading input file "soubor.hex"
avrdude: writing flash (1066 bytes):

Writing | ################################################## | 100% 1.64s

avrdude: 1066 bytes of flash written

avrdude: safemode: Fuses OK (H:00, E:00, L:00)
ioctl("TIOCMGET"): Inappropriate ioctl for device

avrdude done.  Thank you.
```

Okamžitě po ukončení zápisu do flash paměti začne Arduino provádět námi zaslaný program.

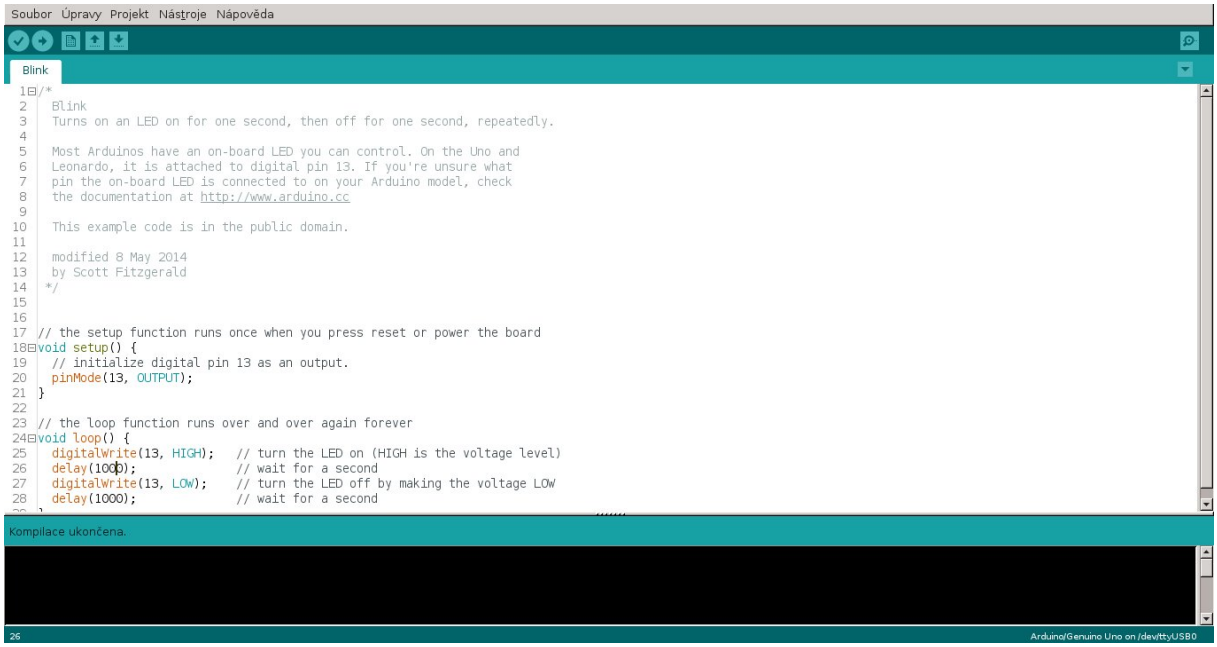
5.7.2 Implementace dálkového programátoru do Arduino IDE

V této kapitole bude popsán postup pro implementaci dálkového programátoru do Arduino IDE. Arduino IDE je prostředí (ukázka je na obrázku 5.10) pro programování Arduina napsané v JAVA a založené na dalších opensource programech, jako je Avrdude. Toto prostředí využívá většina komunity kolem Arduino platformy. Programy se píší v jazyce podobném C jménem Wire. Vzhledem k tomu, že je prostředí poměrně snadno konfigurovatelné, není problém přidat specifikaci dálkového programátoru do konfiguračního souboru, který je umístěn (na systému Linux) v adresáři Arduino IDE `/hardware/arduino/avr` pod názvem `programmers.txt`. Přidáním následujících řádků do tohoto souboru vytvoříme pro dálkový programátor novou třídu:

```
espprogrammer.name=ESP Programmer - local
espprogrammer.communication=serial
espprogrammer.protocol=arduino
espprogrammer.speed=115200
espprogrammer.program.protocol=stk500v1
```

```
espprogrammer.program.speed=115200
espprogrammer.program.tool=avrdude
espprogrammer.program.extra_params=-DV -Pnet:192.168.0.107:23 -b115200
```

Spuštěním Arduino IDE se vytvoří v menu `Nástroje > Programátor` možnost vybrat položku `ESP Programmer - local`. Zvolením této položky můžeme pak náš program do Arduina nahrát volbou `Projekt > Nahrát pomocí programátoru`. Vytvoření třídy nedělá vlastně nic jiného, než že určuje s jakými parametry se program Avrdude spustí (analogie s použitím příkazu `avrdude` v kapitole 5.7.1)



Obrázek 5.10: Ukázka prostředí Arduino IDE

Kapitola 6

Závěr

Navrženým postupem byl realizován bezdrátový programátor Atmel AVR procesorů na platformě Arduino v podobě shieldu. Shield se umí připojit k námi nastavené WiFi síti a vytvoří transparentní sériový most mezi PC a Arduinem, které chceme naprogramovat. Vytvořený most lze použít k samotnému programování, ale také ke konzolovému přístupu k Arduinu v rámci zadávání dat, či čtení stavů, popřípadě k debugování námi programovaného software. Navíc je ve firmware možnost obsluhy REST a MQTT v rámci M2M (machine to machine) komunikace. Výhoda tohoto přístupu tkví v tom, že zařízení můžeme naprogramovat přímo v místě jeho aplikace a nemusíme ho nepohodlně připojovat kabelem k PC (což ani v některých průmyslových aplikacích není možné).

Použití vidím v široké škále odvětví od IoT, přes projekty typu SmartCity, automatizace, domácí aplikace, zkrátka všude tam, kde je potřeba provést upgrade softwaru a zařízení není snadno přístupné.

Arduino jsem zkoušel vzdáleně programovat i přes veřejnou IP a až na nižší stabilitu spojení (vyžadovalo to proces odeslání programu několikrát opakovat, jednalo se totiž o meziměstskou vzdálenost asi 350km) se naprogramování cílového Arduina povedlo.

Budoucí vývoj by se mohl zaměřit na novější typy modulů ESP8266, které vykazují menší spotřebu a některé z těchto modulů disponují externí anténou, která by mohla stabilitu přenosu zlepšit. V této mnou navržené hardwarové konfiguraci se odeslání programu do Arduina daří asi na 90%, což by mohlo být ošetřeno nadstavbou programu Avrdude, která by v případě neúspěšného odeslání proces opakovala.

Literatura

[1] LADYADA. *AVR Tutorial - Choosing a programmer* [Citace: 29. 5. 2016]. Dostupné z: http://www.ladyada.net/learn/avr/programmers.html

[2] AVR FREAKS. *AVR Programming Methods* [Citace: 29. 5. 2016]. Dostupné z: http://www.avrfreaks.net/forum/tut-hard-avr-programming-methods?name=PNphpBB2&file=viewtopic&t=38691

[3] IDEASTUDIO. *Serial Port Bluetooth Module (Master / Slave) : HC-05* [Citace: 29. 5. 2016]. Dostupné z: http://wiki.iteadstudio.com/Serial_Port_Bluetooth_Module_(Master/Slave)_:_HC-05

[4] NORDIC SEMICONDUCTOR. *nRF24L01* [Citace: 29. 5. 2016]. Dostupné z: http://www.nordicsemi.com/eng/Products/2.4GHz-RF/nRF24L01

[5] ESPRESSIF. *ESP8266EX* [Citace: 29. 5. 2016]. Dostupné z: https://espressif.com/en/products/ hardware/esp8266ex/resources

[6] JEELABS. *esp-link* [Citace: 29. 5. 2016]. Dostupné z: https://github.com/jeelabs/esp-link

[7] AVRDUDE MAN. *avrdude(1) - Linux man page* [Citace: 29. 5. 2016]. Dostupné z: http://linux.die.net/man/1/avrdude

[8] SURREY *UNIX Tutorial for Beginners* [Citace: 29. 5. 2016]. Dostupné z: http://www.ee.surrey.ac.uk/Teaching/Unix/

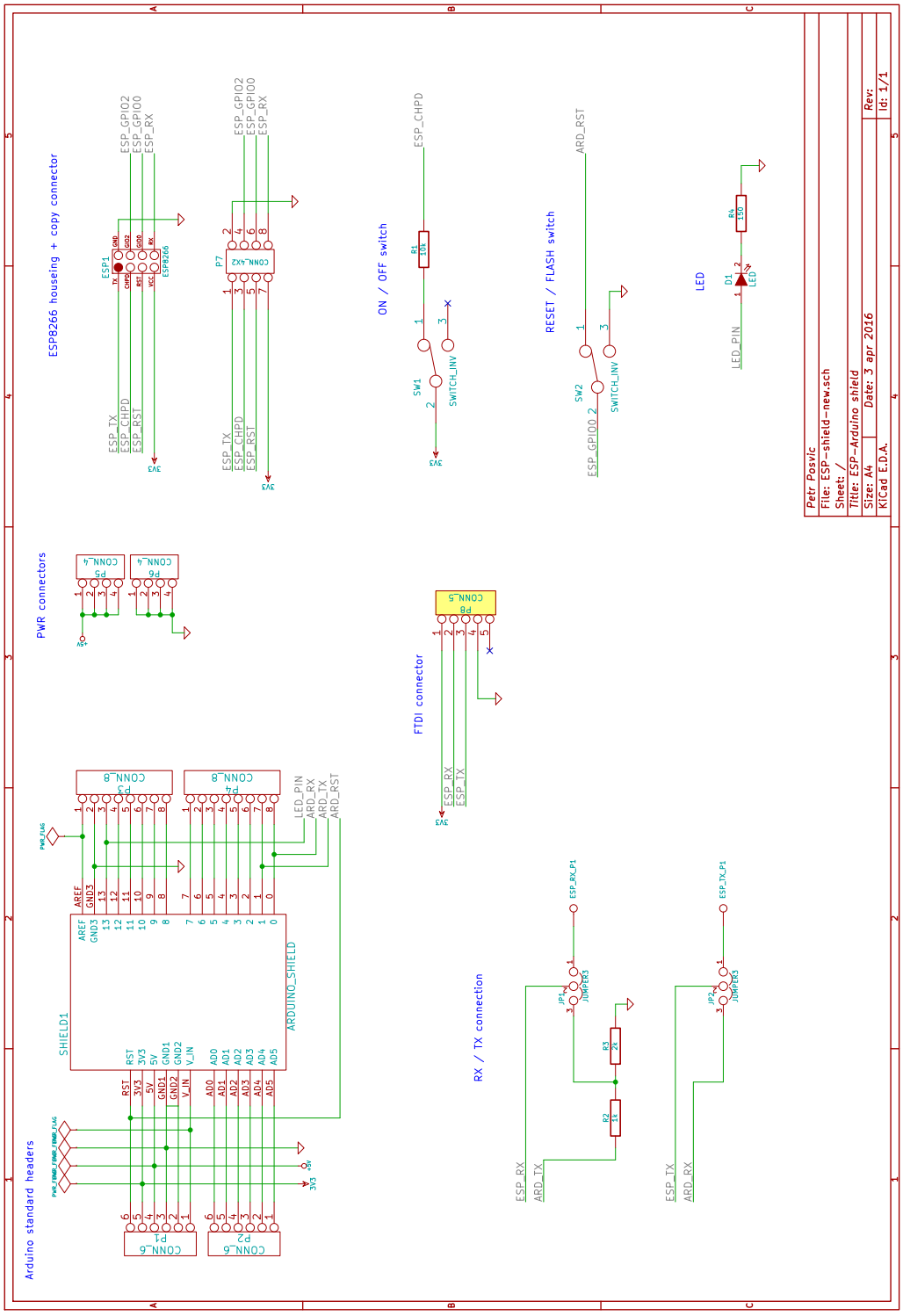
[9] ARDUINO *Arduino* [Citace: 29. 5. 2016]. Dostupné z: https://www.arduino.cc/

[10] WIKIPEDIA *Hayes command set* [Citace: 29. 5. 2016]. Dostupné z: https://en.wikipedia.org/wiki/Hayes_command_set

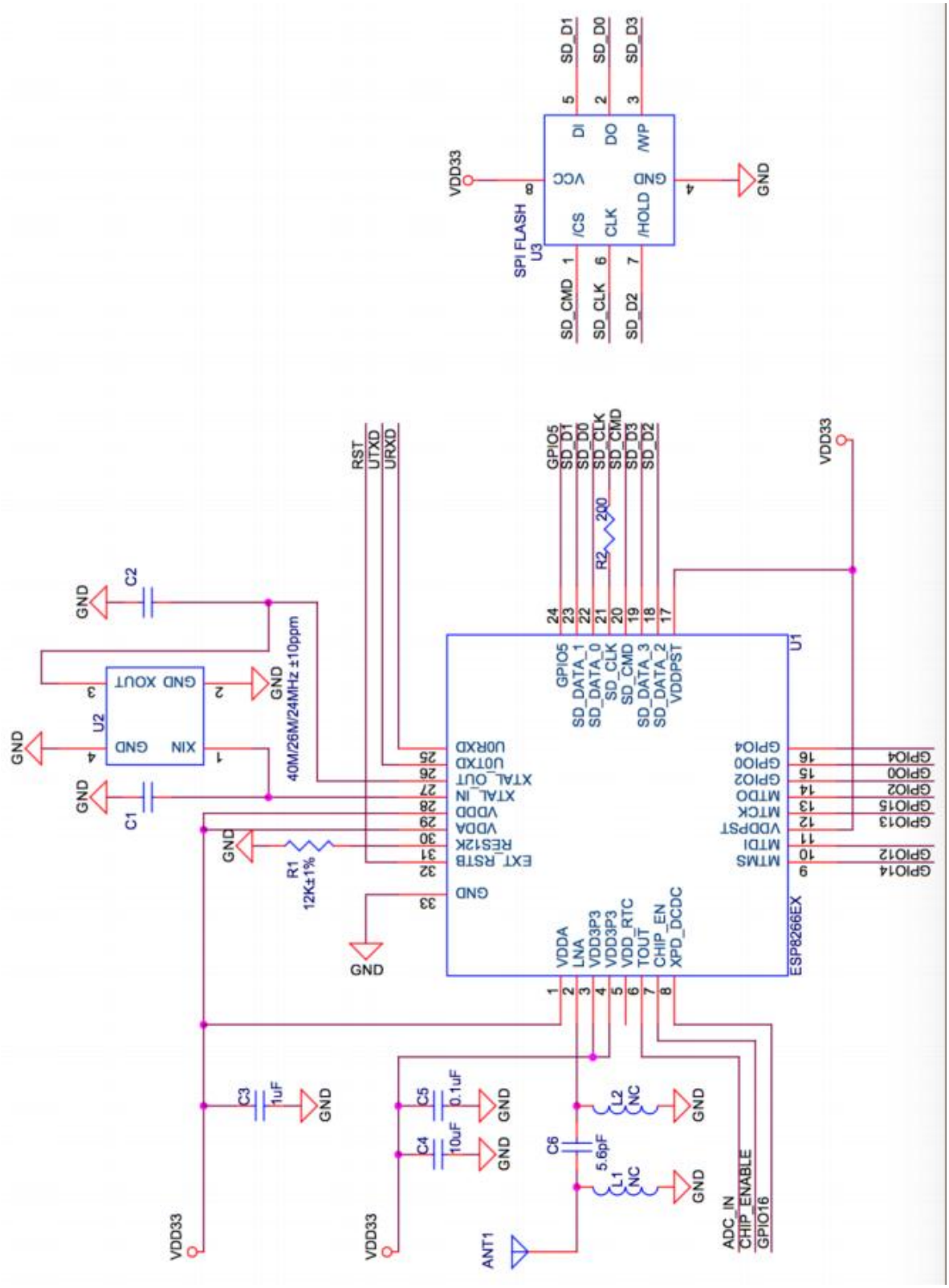
[11] Newell Joshua *DIY Arduino Bluetooth Programming Shield* [Citace: 29. 5. 2016]. Dostupné z: http://makezine.com/projects/diy-arduino-bluetooth-programming-shield/

Příloha A

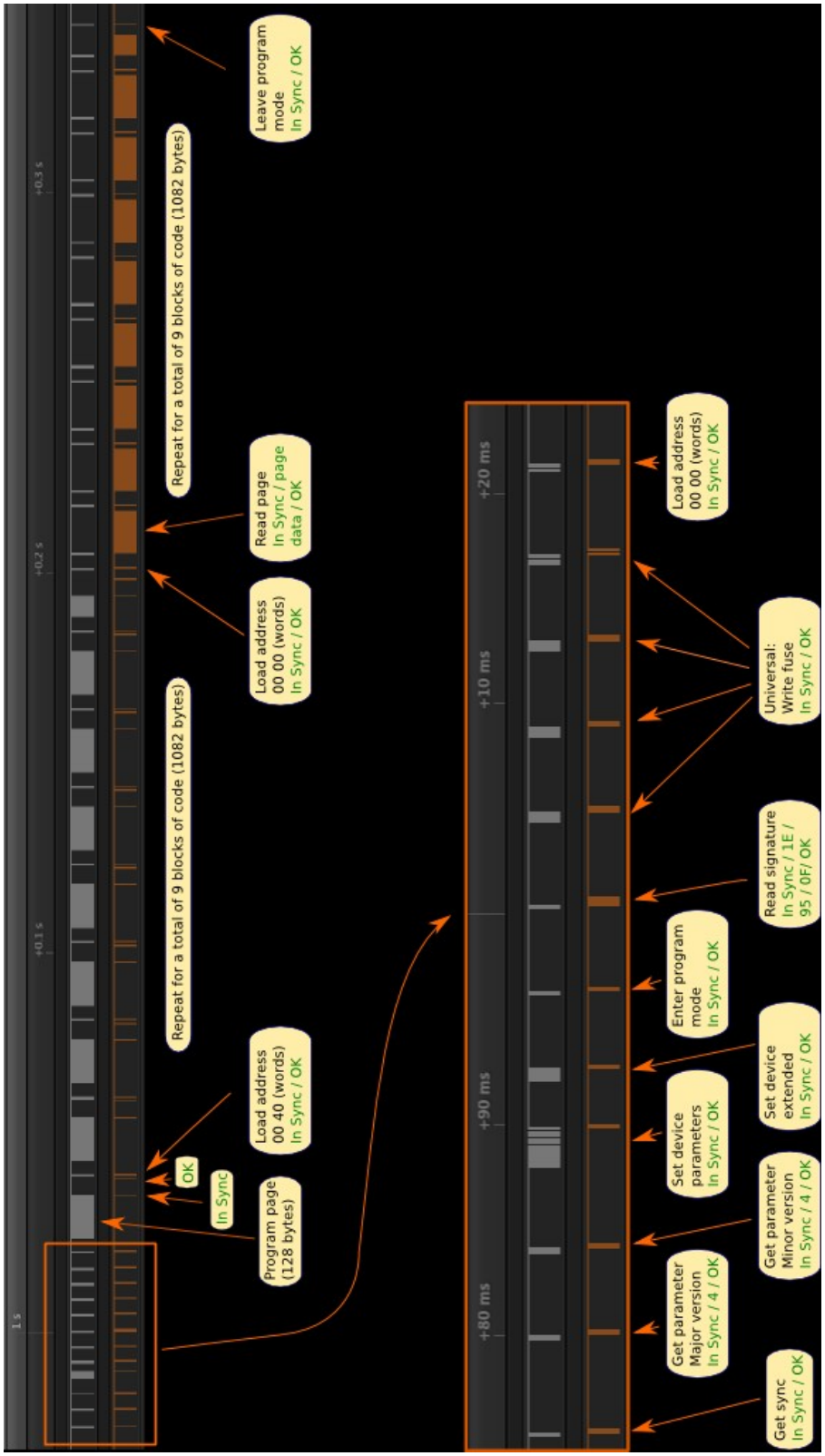
Schéma zapojení



Obrázek A.1: Schéma zapojení shieldu pro Arduino s možností dálkového programování



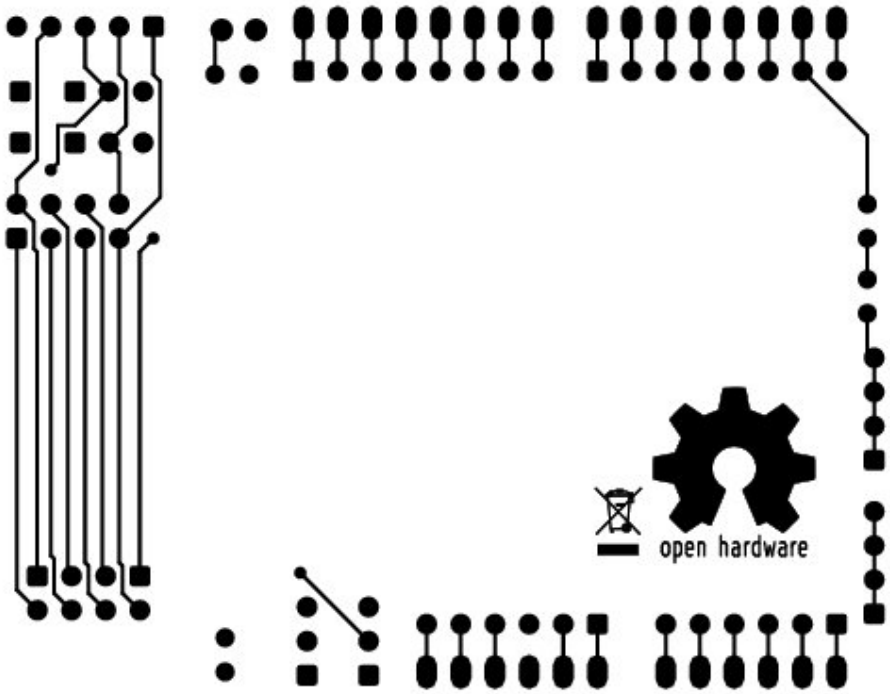
Obrázek A.2: Schéma modulu ESP8266 (Espressif)



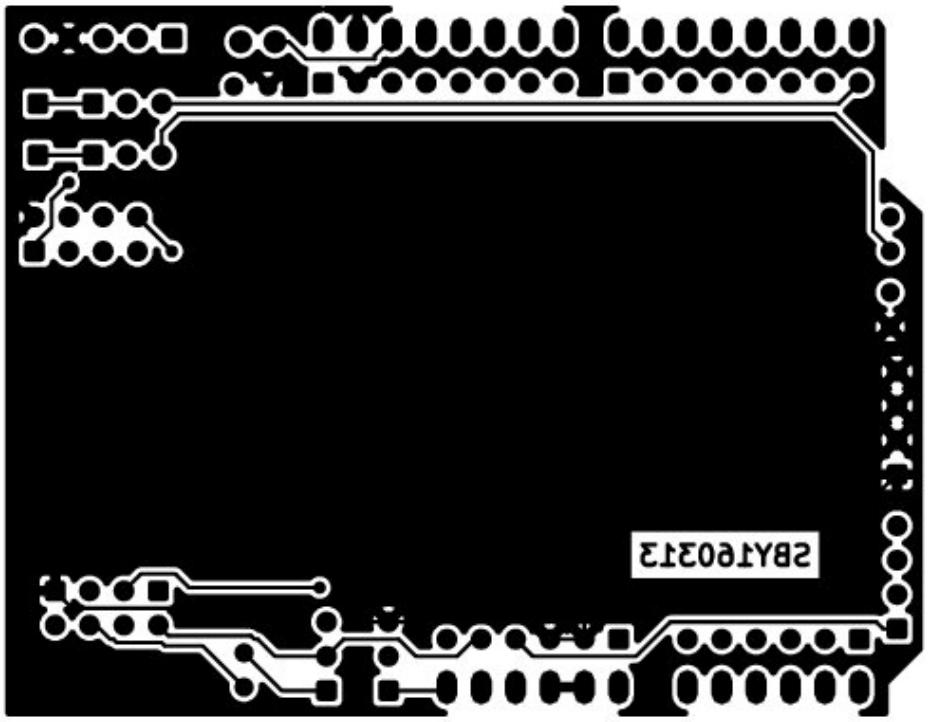
Obrázek A.3: Diagram komunikace protokolu STK500

Příloha B

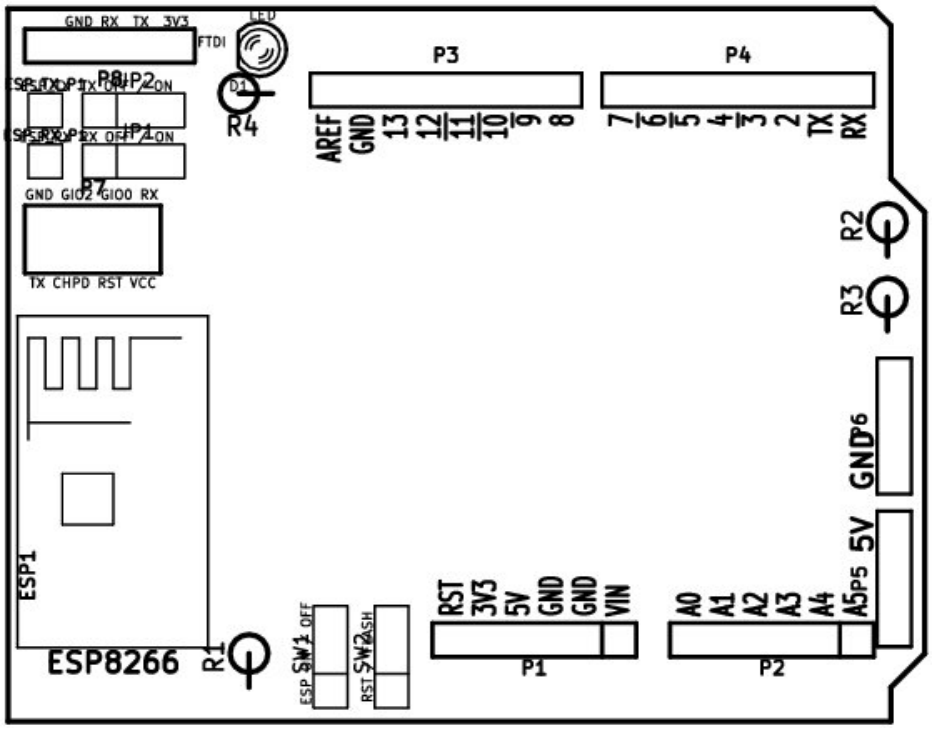
Deska plošných spojů



Obrázek B.1: Přední strana tištěného spoje



Obrázek B.2: Zadní strana tištěného spoje



Obrázek B.3: Servisní potisk tištěného spoje