VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2016 Bc. Pavel Frkal

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

NÁVRH A REALIZACE MANAGOVATELNÉHO POE INJEKTORU
DESIGN AND IMPLEMENTATION OF MANAGED POE INJECTOR

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Pavel Frkal**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Ondřej Krajsa, Ph.D.**
SUPERVISOR

BRNO 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Telekomunikační a informační technika**
Ústav telekomunikací

Student: Bc. Pavel Frkal ***ID:*** 89728
Ročník: 2 ***Akademický rok:*** 2015/16

NÁZEV TÉMATU:

Návrh a realizace managovatelného PoE injektoru

POKYNY PRO VYPRACOVÁNÍ:

Navrhněte a realizujte pasivní PoE injektor pro zařízení Mikrotik. Při návrhu uvažujte 24 napájených portů a umístění v 19" rozvaděči. Zařízení bude možno obsluhovat pomocí webového rozhraní, kdy bude umožněno zapínání/vypínání napájení jednotlivých portů. Analyzujte možnost měření spotřeby na jednotlivých portech.

DOPORUČENÁ LITERATURA:

[1] MARGOLIS, Michael. Arduino Cookbook. 2nd ed. O'Reilly, 2011. ISBN 1449313876.

[2] WILCHER, Don. Arduino Electronics Blueprints. Packt Publishing, 2015. ISBN 9781784393601.

Termín zadání: 1.2.2016 ***Termín odevzdání:*** 25.5.2016

Vedoucí práce: Ing. Ondřej Krajsa, Ph.D.
Konzultant diplomové práce:

doc. Ing. Jiří Mišurec, CSc., *předseda oborové rady*

ABSTRAKT

Práce se zabývá možnostmi napájení síťových zařízení přes stávající strukturovanou kabeláž. Rozebírá možnosti napájení, jejich limity a možnosti měření poskytovaného výkonu. Výstupem je konstrukce 24-portového pasivního injektoru pro vzdálené napájení síťových zařízení. Je vhodný pro zařízení, která nemají vestavěnou podporu PoE napájení, jelikož nevyžadují zásadní úpravy na straně napájených zařízení. Napájení je možné ovládat přes webové rozhraní a rovněž je ve stejném rozhraní přístupná spotřeba jednotlivých portů.

KLÍČOVÁ SLOVA

PoE, Power over Ethernet, injektor, ehternet, vzdálené napájení, napájení po síťovém kabelu, vzdálený management, 802.3af, 802.3.at, měření spotřeby, Arduino, AVR.

ABSTRACT

This study deals with power supply for network devices using existing structured cabling. Various power supplies are analysed including their limitations and also deals with measuring of the consumed energy. The outcome is 24-port passive PoE injector for remote power supply of network equipment. It's suitable namely for remote devices which lacks built-in PoE support as these do not need major modifications (using passive injection). Power options including current measurement may be done via administrative web interface.

KEYWORDS

PoE, Power over Ethernet, injector, ethernet, remote power, power over ethernet cable, remote management, 802.3af, 802.3.at, consumption measurement, Arduino, AVR.

FRKAL, Pavel *Návrh a realizace managovatelného PoE injektoru*: diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2016. 48 s. Vedoucí práce byl Ing. Ondřej Krajsa, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Návrh a realizace managovatelného PoE injektoru“ jsem vypracoval samostatně pod vedením vedoucího diplomové 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 diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové 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 § 152 trestního zákona č. 140/1961 Sb.

Brno ....

(podpis autora)

Děkuji vedoucímu diplomové práce Ing. Ondřeji Krajsovi, Ph.D. za trpělivost, podporu a cenné konstruktivní návrhy při vedení této diplomové práce. Zároveň děkuji Ing. Pavlu Hanákovi, Ph.D, za cenné připomínky k použítému CAD softwaru Eagle.

SIX research centre
sensor, information and communication systems

Faculty of Electrical Engineering
and Communication

Brno University of Technology
Technicka 12, CZ-61600 Brno, Czechia

http://www.six.feec.vutbr.cz

Výzkum popsaný v této diplomové práci byl realizovaný v laboratořích podpořených projektem Centrum senzorických, informačních a komunikačních systémů (SIX); registrační číslo CZ.1.05/2.1.00/03.0072, operačního programu Výzkum a vývoj pro inovace.

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

2007–13
OP Výzkum a vývoj pro inovace

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

Tato diplomová práce se věnuje konstrukci managovatelného PoE (Power over Ethernet) injektoru. Účelem tohoto zařízení je poskytnout napájení pro síťové prvky bez nutnosti instalace dodatečného silového rozvodu – po stávajících kabelech strukturované kabeláže (UTP/U-FTP/SF-UTP). Pasivní je v tomto případě myšleno to, že koncové zařízení nemusí aktivně komunikovat s injektorem, aby mu bylo napájení poskytnuto (jak je tom např. u standardů 802.3af/at [7]) V mnoha případech představuje možnost využití stávající kabeláže značné finanční úspory, případně i zjednodušení mechanické kostrukce zařízení pro trvalou instalaci v exteriéru.

Navrhovaný injektor umožní napájení 24 zařízení s celkovým špičkovým příkonem až 150 W (případně až 250 W při osazení výkonnějším napájecím zdrojem). Zařízení je konstruováno pro montáž do 19" rozvaděčové skříně – tedy vhodně kombinovatelné se stávající strukturovanou kabeláží (vyvedené patch panely) tak aktivními síťpvými prvky (přepínače či přímo směrovače). Z možných zapojení bylo vybráno trvale aktivní vedené po signálových vodičích – pro možnost aplikace na 1000BASE-T ethernetu, který využívá všechny čtyři kroucené páry pro síťovou komunikaci. Aby nebylo nutno stávající napájená zařízení modifikovat, je napájení stále přítomné na kabelu, takže na straně napájeného zařízení lze použít pouze pasivní PoE extraktor, který se předřadí před vstup do zařízení. Plánované využití je pto nápajení směrovačů zn. Mikrotik v rámci laboratoří VUT.

1 POŽADAVKY PRO REALIZACI

1.1 Napájení síťových zařízení

Pro funkci každého elektronického zařízení je nutné dodat zařízení potřebnou elektrickou energii. Pro zařízení poskytující telektomunikační služby tomu není jinak. Telefon se jakožto první osobní služba od počátku potýkal s problematickým dodáním energie, jelikož rozšířenost silové elektrické sítě byla relativně velmi malá. Řešením býval povětšinou lokální zdroj energie (klička s mechanickým elektrogenerátorem nebo baterie), ovšem s narůstajícími požadavky uživatelů na komfort a nezávislost na výpadcích lokálních dodávek bylo nakonec přistoupeno k dálkovému napájení. Již v této době hrála nemalou roli cena, kterou by bylo nutno investovat do instalace dodatečné či rozšířené kabeláže, která by zajišťovala toto napájení autonomně. Proto bylo zvoleno napájení stejnosměrné, po stávajícím dvoudrátu – tedy mimo samotné přenosové pásmo telefonu. Toto řešení pro svou jednoduchost a úsporné provedení zvítězilo na více než celé století. Dnešní telekomunikační sítě jsou převážně tvořeny různými verzemi ethernetu [6], který díky své otevřenosti překonal ostatní síťové technologie. Je na snadě, že i zde se potřeba napájení rovněž objevuje. Nejčastějšími aplikacemi jsou telefony využívající technologii VoIP, IP kamery, WiFi přístupové body a směrovače, případně tiskové servery nebo i jiná zařízení. Naprostá většina těchto zařízení pracuje na nízkém stejnosměrném napájení (dáno již z podstaty přítomnosti mikroprocesoru). Toto napájení je vesměs řešeno adaptérem do elektrické sítě, což je pro většinu aplikací vyhovující řešení. Pro jednodušší zařízení je výhodné využít napájení po datovém kabelu, který z hlediska elektrického představuje dostatečné parametry [1]. Krom elegantního řešení umožníme provoz zařízení i v místech, kde by zajištění napájení z rozvodné sítě byl problém.

V mnoha případech se setkáváme s potřebou napájet již existující zařízení a požadavkem minimalizace zásahů jak do samotného zařízení, tak kabeláže. Pro tyto účely velmi výhodně užijeme navrhované zařízení, které na straně napájeného zařízení vyžaduje pouze pasivní PoE extraktor.

1.1.1 Možnosti napájení různých typů kabeláže

Ethernet se stal díky své variabilitě a otevřenosti nejběžnější síťovou technologií dnešní doby. Je proto na snadě, že umožňuje použití celé škály kabeláže. Pro naprostou většinu dnes používané metalické kabeláže je značná část parametrů obdobná, výjimku tvoří zastarlý tlustý a tenký koax a nově standardizovaná média

[1] Dnes nejběžněji používaná kabeláž Cat5e má vodiče třídy AWG 24 (0,205 mm^2), Cat6 dokonce AWG 23 (0,259 mm^2) [18].

pro ehternet rychlostí 25, 40 a 100 Gbit/s.

1.1.2 Standardy kabeláže ethernetu

Médium 10/100BASE-T představuje do nedávna nejrozšířenější technologii co se týče sítí LAN. Pro tento standard je použita čtveřice kroucených dvojlinek – tzv. UTP (Unshielded Twisted Pair). K vysílání i příjmu je použit vždy jeden pár kroucených vodičů – dle standardu TIA/EIA-568 [20] používají páry 2 a 3 a to na kabelážích UTP kategorie 3 a výše. Přehledná tabulka 1.1 ukazuje jednotlivé kategorie dle standardizace ISO/IEC 11801.

Název:	Šírka pásma	Rozhraní	pozn.
Cat.3	16 MHz	10BASE-T 100BASE-T4	Dle standardu EIA/TIA-568. Dnes již nevyráběný.
Cat.4	20 MHz	16 Mbit/s Token Ring	Dnes již nevyužit.
Cat.5	100 MHz	100BASE-TX 1000BASE-T	Nejběžnější kabel, často přímo splňuje přísnější normu Cat5e.
Cat.5e	100 MHz	100BASE-TX 1000BASE-T	Rozšířená (Enhanced) Cat5. Rozdíl je v testování, fyzicky stejné požadavky jako Cat5.
Cat.6	250 MHz	10GBASE-T	Nejnižší staard pro 10Gbit/s, přidává mechanické vymezení jednotlivých párů.
Cat.6A	500 MHz	10GBASE-T	Jako Cat6, navíc přidává stínění.
Cat.7	600 MHz	10GBASE-T	Plně stíněný kabel – dle ISO/IEC 11801 2nd Ed. (2002)
Cat.7A	1000 MHz	10GBASE-T	Zvýšená šířka pásma. ISO/IEC 11801 2nd Ed. Am. 2. (2008)
Cat.8/8.1	1600-2000 MHz	40GBASE-T	Ve vývoji (ANSI/TIA-568-C.2-1, ISO/IEC 11801 3rd Ed.)
Cat.8.2	1600-2000 MHz	40GBASE-T	Ve vývoji (ISO/IEC 11801 3rd Ed.)

Tab. 1.1: Kategorie kabeláží dle ISO/IEC 11801.

Z konstrukce jednotlivých druhů kabeláže vyplývají požadavky a možnosti napájení pro jednotlivé standardy.

1.1.3 Standardy napájení ethernetu

Organizace IEEE vydala zatím 2 standardy pro napájení po ethernetu a to 802.3af-2003 a 802.3at-2009 [2]. Jak již z označení vyplývá jsou integrální součástí standardů pro ethernet. Oba předpokládají kabeláž kategorie 5, avšak s omezeními (především výkonovými) lze použít i kabel kategorie 3. Krom detailního popisu logických částí obvodů zabezpečujících automatické nastavení parametrů a samotného ovládání (jež nejsou součástí navrhovaného řešení) obsahuje tyto stadardy v zásadní konvence samotného napájení.

Základní terminologie

Většína ve standardech používaných termínů je běžně používána v oblasti sítí, pro základní orientaci jsou pak důležité především tyto dva:

- PSE – Power Sourcing Equipment – zařízení (jedno či více), které napájení přivádí.
- PD – Powered Device – napájené zařízení.

Vzhledem k podobnosti těchto dvou termínů je pro jednoznačnost nejsou často využity, avšak pokud čtenář nahlédne do citované literatury velmi pravděpodobně se s nimi potká.

Hlavní prvky standardů

Samotný rozdíl mezi 802.3af a 802.3at spočívá jednak v doplnění specifických funkcionalit na logické úrovní ovládání a díky vyššímu povolenému napětí umožňuje využití až 25,5 W (v porovnání s 15,4 W pro starší ale rozšířenější 802.3af). Zkrácený přehled elektrických parametrů těchto standardů ukazuje tabulka 1.2. Jak lze vidět na obrázku 1.1 jsou standardizovány 2 možnosti zapojení při přímém propojení aktivních síťových prvků (napájecí element je součástí přepínače/směrovače). Jsou označeny písmeny A – napájení vedené po signálových párech a B – vedené po volných párech. Jedinou možností pro zapojení injektoru [3] je pak zapojení mimo aktivní síťový prvek. V zásadě je buď použito volných párů nebo je stejnosměrné napájení přivedeno přes střední vývod vinutí oddělovacího tranformátoru signálových párů. Tyto oddělovací tránsformátory jsou vyžadovány pro koncová zářízení (dle specifikace) pro galvanické oddělení síťových zařízení. K napájení je nejčastěji použito dvou párů, při použití všech čtyř párů totiž vzájenné galvanické propojení není nezanedbatelné a je potřeba počítat s významným nárůste přeslechů, jež

[2] Označovaný také PoE+ či PoE plus

[3] Dle standardu označovaného jako Midspan Power Insertion Equipment

významným způsobem ovlivňují základní pásmo signálu. Standard 802.3at toto výslovně zakazuje.

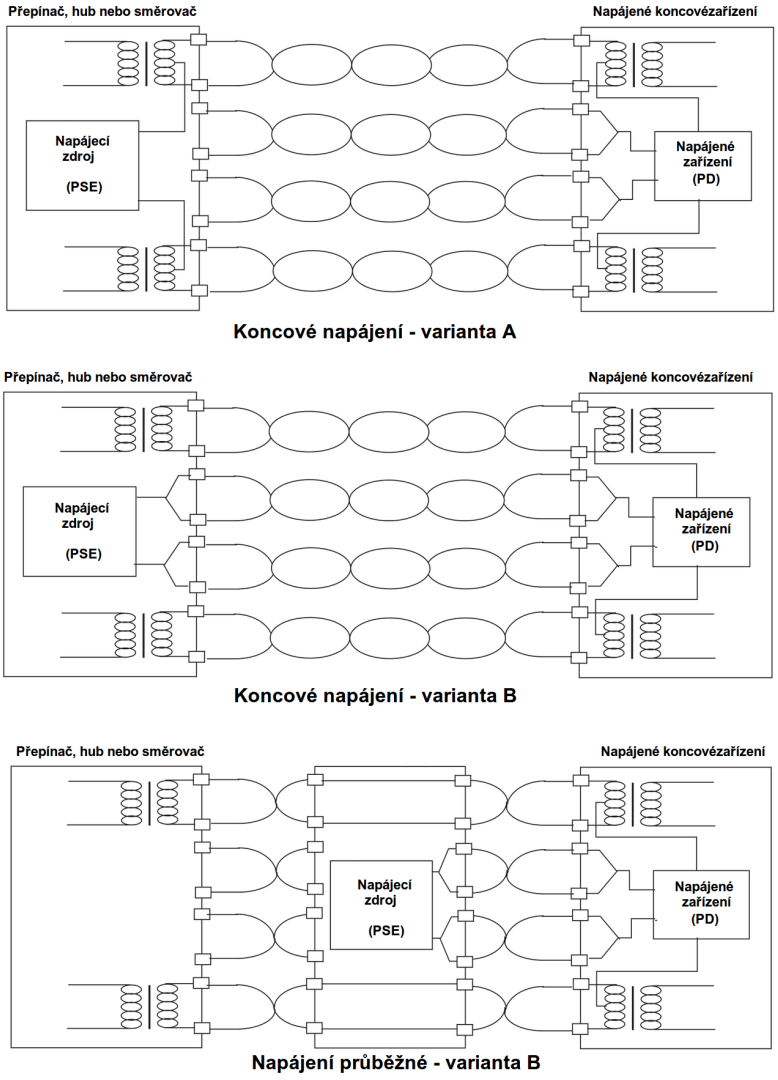
Parametr:	802.3af	802.3at typ 2
Výkon dostupný na napájeném zařízení (PD)	12,95 W	25,50 W
Maximální dodaný výkon (PSE)	15,40 W	30,0 W
Napěťový rozsah napaječe (PSE)	44,0–57,0 V	50,0–57,0 V
Rozsah napětí napájeného zařízení (PD)	37,0–57,0 V	42,5–57,0 V
Maximální proud	350 mA	600 mA per mode
Maximální odpor páru	20 Ω (kategorie 3)	$12,5$ Ω (kategorie 5)
Management výkonu	tři úrovně výkonu dohodnuté při vstupním vyjednání	čtyři úrovně výkonu dohodnuté při vstupním vyjednávání nebo po krocích 0,1 W průběžně upravovaných
Snížení maximální provozní teploty okolí kabeláže	žádné	o 5°C s provozem na dvou párech
Podporované režimy	mód A (napájení na konci), Mode B napájení injektorem)	mód A, mód B

Tab. 1.2: Prametry napájení a užitečný výkon dle standardů 802.3af a 802.3at.

Popsané standardy však představují jen část běžně používaných zapojení. Významná část napájených zařízení však využívá pasivní (z pohledu aktivnosti) napájení.

1.1.4 Pasivní napájení

Ačkoli je dodržování standardů základem pro zajištění kompatibility je v oblasti napájení po ethernetu velmi hojně využíváno pasivní napájení. Pasivní je v tomto ohledu chápáno jako přímé připojení napájení (ať již přímo na volné páry tak i na signálové páry) – tzn. napřímo – bez logiky v obvodech. Je tomu zřejmně proto, že zmiňovaná doporučení jsou příliš složitá a často představují netriviální úpravu



Obr. 1.1: Možnosti zapojení napájení přes ethernet. Upraveno z [7].

napájených zařízení a v případě embeded zařízení [4] dokonce nezanedbatelné zásahy do firmware zařízení. Pro použití proprietálních specifikací rovněž hovoří absolutní limity stanovené pro napájecí napětí i proud. V praxi je totiž často potřeba napájet nízkoodběrová zařízení konstruovaná na nižší maximální povolené napětí – u bezdrátových směrovačů a přístupových bodů jde často o výkon kolem 5 W a napětí 12–24 V na relativně krátké vzdálenostii (z rozvaděče uvnitř budovy na její střechu), takže celková výkonová ztráta na vedení je značně nižší než standardy povolená. Často se také jedná o samostatně vedený kabel, takže je možno počítat s vyšším proudem díky předpokládanému lepšímu chlazení. V praxi se používají na krátkou vzdálenost špičkově i proudy kolem 1 A při napětí 12-24 V. V navrhnutém injektoru je použito právě 24 V. Ačkoliv pasivní napájení není součástí standardů IEEE, je snahou se jim co nejvíce přiblížit. Proto je přihlédnuto k standardní polaritě napájení i použitých párů. Protože požadavkem na navrhovaný injektor je možnost provozu 1000BASE-T, je nutno použít k napájení signálové vodičei. Zde díky chybějícím normativním parametrům došlo k různosti zapojení. Toto představuje poměrně vysoké riziko v podobě poškození napájeného zařízení díky přepólování. Z tohoto důvodu bylo zvoleno napájení pouze po dvou párech – a to 4-5 a 7-8. Tím pak zminimalizujeme možnost nežádoucího vlivu při použití křížené kabeláže. Křížený kabel [5] by totiž prohodil polaritu napájení na straně napájeného zařízení (PD). Toto je ve standardech ošetřeno zapojením Gretzova můstku ihned za oddělovací transformátor. Toto však nelze zaručit u zařízení, pro něž je navrhovaný injektor určen. Kromě tohoto opatření je navíc použito vratných elektronických pojistek (PTC) pro každý napájený kanál. Tabulka 1.3 uvádí nejčastěji používané polarity napájení po jednotlivých párech.

1.1.5 Požadavky na spotřebu

Plánovaným využitím navrhovaného PoE injektoru je napájení modulárních routerboardů zn. Mikrotik. Tyto směrovače jsou velmi variabilní a proto je injektor navrhován pro pokrytí nemalých výkonových špiček. Pro maximální přípustné zatížení byl pro ověření proveden výpočet ohřevu vodičů při průchodu limitně velkým proudem po doby uvedené níže. Z výpočtu vyplývá, že vodiče kategorie 5 lze při uvažování okolní teploty 25°C a maximální dovolené teploty 70°C (tedy maximální ohřev o 45°C) relativně dlouhou dobu (řádově minuty) používat i pro proudy kolem 1 A. Z tohoto omezení pak vyplývá i dimenzování měření odběru, konstrukce DPS i dimenzování jištění. Viz tabulka 1.4. Takový odběr by měl pokrýt i špičkově vyšší proudové odběry napájených směrovačů. Pro obdobný proud by pak rovněž jsou

[4] Zařízení minimalizovaná na minimální požadavky

[5] Využívá standardu ANSI/TIA/EIA-568-B na jedné straně a ANSI/TIA/EIA-568-A na druhé.

Vodič:	Zapojení A (MDI-X)	Zapojení A (MDI)	Zapojení B
1	-U_{nap}	+U_{nap}	-
2	-U_{nap}	+U_{nap}	-
3	+U_{nap}	-U_{nap}	-
4	-	-	+U_{nap}
5	-	-	+U_{nap}
6	+U_{nap}	-U_{nap}	-
7	-	-	-U_{nap}
8	-	-	-U_{nap}

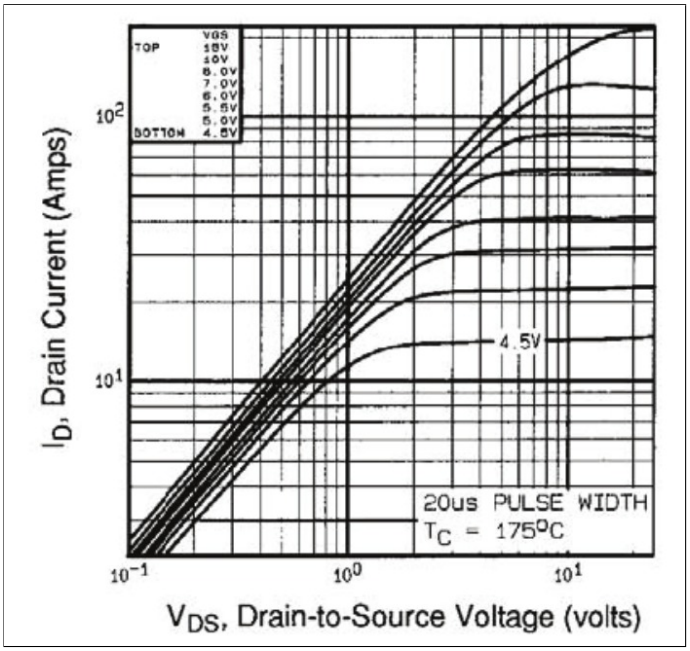
Tab. 1.3: Zapojení vodičů

dimenzovány spínací polem řízené tranzistory. Vzhledem k relativně nízké ceně byly zvoleny předimenzovaně – jak co se týče maximální výkonové ztráty tak povoleného trvalého proudu. Byl zvolen typ IRFZ44[13], jeden z běžně vyráběných polem řízených tranzistorů (MOSFET). Vzhledem vzhledem k očekávanému zatížení se při katalogových hodnotách R^{DSON} i dle výstupní charakteristiky (viz **??**) a velikosti výstupního napětí I/O použitého mikrokontroléru nepředpokládá nutnost chladit jednotlivé tranzistory. Pokud by to bylo potřeba je deska plošných spojů kostruována tak, aby bylo možno použít chladiče tvaru U pro každý z nich. V krajních případech, kdy by nestačil požadovaný výkon přenesený po dvojici párů by bylo navíc možno využít všechny 4 páry kabelu UTP a tím umožnit přenos téměř dvojnásobného výkonu [6]. Jak již bylo zmíněno, navrhované zařízení však z důvodu kompatibility a rovněž požadovaných dodaných výkonů koncových zařízení s tímto nepočítá.

1.2 Správa zařízení

Pro správu injektoru bylo zvoleno vytvoření webového rozhraní přímo v rámci řídícího mikrokontroleru, který ovládá samotné napájení, zajišťuje měření spotřeby a teploty okolí spínacích tranzistorů a je tedy srdcem celého zařízení. Ovládací program sestává z jednoduchého webového serveru implementujícího základní prvky protokolu HTTP a samotného kódu spínání napájení a měření spotřeby na jednotlivých portech. Spínací tranzistory jsou spínány uložením vysoké hodnoty na výstupní porty, měření je pak zajištěno přes multiplexovaný analogový spínač použitím 3x3 bi-

[6] V takovém případě by bylo nutno počítat zvýšené zahřátí svazku vodičů v kabelu UTP.



Obr. 1.2: Výstupní charakteristiky tranzistoru IRFZ44 (převzato z [13].

tové adresace [7]. Jako analogový multiplexer byl zvolen obvod 4051[8] (technologie CMOS).

1.2.1 Vývojové prostředí Arduino

Vývojové prostředí Arduino v sobě skrývá širokou škálu vývojových desek od nejmenších 8-bitových procesorů řady AVR s několika kilobajty flash paměti až po výkonné 32-bitové procesory s jádrem ARM. Jeho hlavní výhodou je velmi snadná instalace, přítomnost programátoru na většině vývojových desek a rovněž přítomnost bootloaderu umožňujícím přímé programování přes USB rozhraní bez nutnosti použití externího programátoru (ISP). Škálovatelnost desek a jejich maximální hardwarová kompatibilita umožňuje otestovat projekt na základní vývojové desce a následné ho rozšířit na desku s výkonnějším procesorem bez zásadních změn v kódu ani hardwarové konstrukce.

Celé prostředí je postaveno na jazyku C, je zde však mnoho předdefinovaných maker, která programování značně usnadní i začátečníkům. Pro ověření modelu byla použita deska Arduino/Genuino Uno – viz 1.3 se síťovým rozhraním na bázi chipsetu ENC28J60. Jedná se o kombinaci 8-bitového mikroprocesoru architektury AVR (ATMega328 32kB Flash paměti, 2kB RAM, 1kB EEPROM) a jednoduchého

[7] Bylo by samozřejmně možné použít spěrnici 5-ti bitovou, avšak adekvátní analogový přepínač pro toto řešení by byl neúměrně nákladný.

Proud [A]	$t_{\Delta}t = +45°C$:x[s]
0,1	38832,1
0,2	9708,0
0,3	4314,7
0,4	2427,0
0,5	1553,3
0,6	1078,7
0,7	792,5
0,8	606,8
0,9	479,4
1,0	388,3

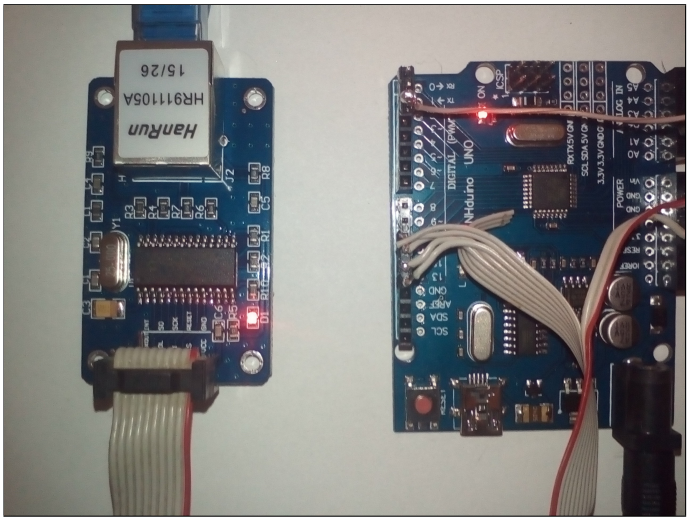
Tab. 1.4: Výpočet maximálního času protékaného proudu pro zahřátí vodiče Cat5e o 45° C

síťového rozhraní s chipsetem podporujícím pouze PHY a MAC [8] vrstvy ethernetu. Samotná implementace TCP/IP protokolu je pak provedena na úrovni programu mikrokontroleru. Tato konfigurace posloužila dobře pro ověření funkčnosti avšak vzhledem k nízkému počtu vstupně/výstupních portů byla ve výsledném provedení nahrazena výkonnější deskou. Konkrétně se jedná o desku Arduino/Genuino MEGA s procesorem Atmel ATmega2560 –viz 1.4 s 256 kB flash paměti, 8 kB RAM, 4 kB pamětí EEPROM a s až 54 dostupnými I/O porty [9]. Pro síťové rozhraní pak byl použit síťový kontrolér od výrobce Wiznet W5100. Tento chip má oproti dříve zmiňovanému ENC28J60 navíc kompletně integrovánu síťovou a transportní vrstvu (umožňuje až 4 paralelní TCP spojení - viz [15]) což umožní výrazně výpočetně odtížit primárnímu mikrokontroléru. Tím je rovněž umožněno připojení mikro SD karty, na kterou bylo možno umístit pomocné soubory webového rozhraní a v případě rozšíření rovněž umožnit zápis systémového logu.

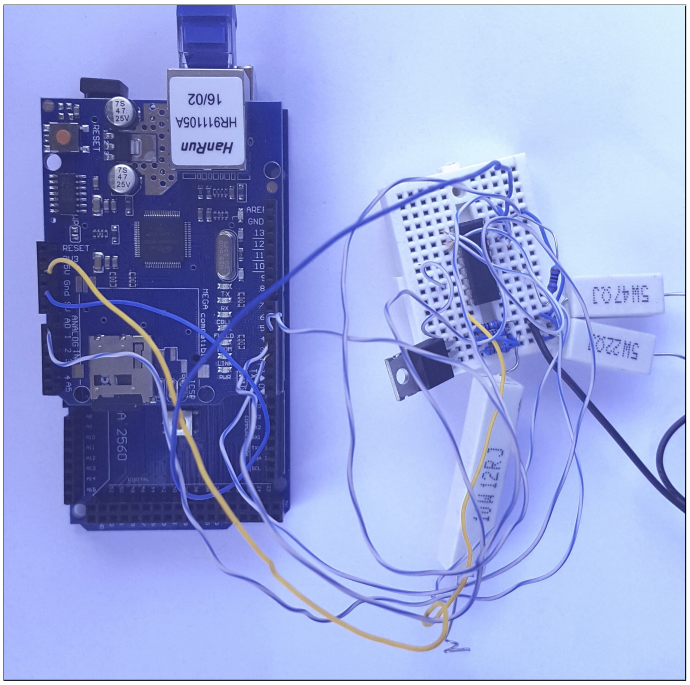
1.2.2 Spínání zátěže

Jedním ze základních požadavků na injektor je jeho manažovatelnost – tedy především možnost zapínat nebo vypínat napájení jednotlivých portů. Ačkoliv by se mohlo zdát, že síťová zařízení je potřeba napájet nepřetržitě, existuje řada situací, kdy ovládání napájení může být užitečné – například vzdálený reset či omezení provozní doby WiFi sítě. Protože použitý mikrokontrolér nabízí na výstupu jednotlivých posrtů limit až 40 mA na port, celkově však maximálně 200 mA na všech

[8] Tedy vrstvy fyzické, linkové avšak nikoli síťové

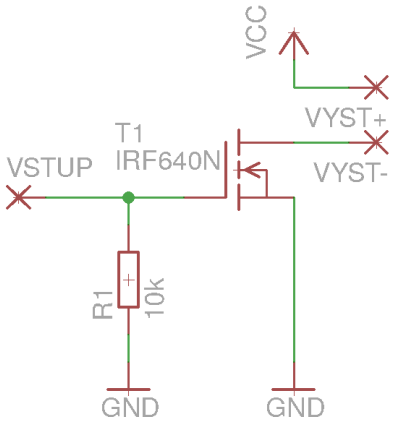


Obr. 1.3: Vývojová deska NHdruino UNO (Arduino/Genuino kompatibilní).



Obr. 1.4: Vývojová deska NHdruino MEGA s Ethernet shieldem (Arduino/Genuino kompatibilní).

portech, bylo by v případě reléového spínání potřeba pro každý kanál použít další vnější tranzistor. Na druhou stranu úroveň logické jedničky představuje ideální napětí pro přímé připojení unipolárního tranzistoru (MOSFET). Vzhledem ke kladné úrovni výstupních portů byl zvolen tranzistor s N-kanálem, kterému k saturaci stačí napětí něco kolem 4 V. Přímému připojení je pak pouze přidán odpor mezi bránu a zem z důvodu vysokých vnitřních odporů jak samotného unipolárního tranzistoru tak především výstuního hradla mikrokontroleru [9]. Toto lze vidět na schématu 1.5. Jak již bylo zmíněno díky přijatelné ceně byl zvolen typ tranzistoru s dostatečnou rezervou [13] pro špičkové výkyvy spotřeby.



Obr. 1.5: Zapojení spínacího obvodu.

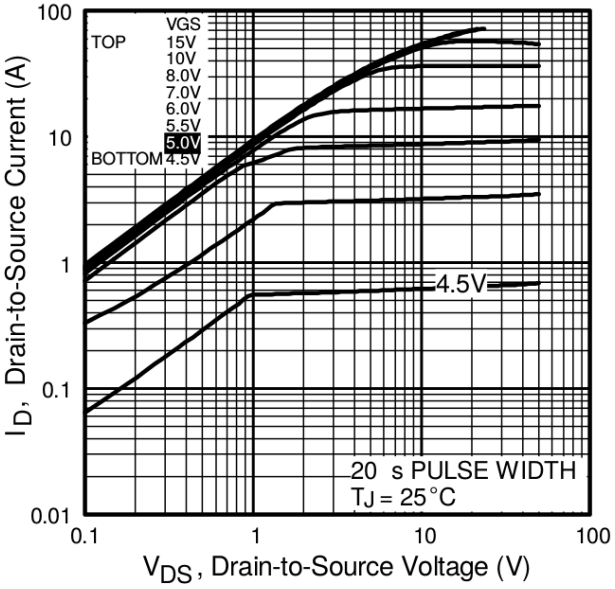
1.2.3 Měření spotřeby napájených zařízení

Vzhledem k minimalizaci výkonových ztrát byla uvažována varianta měření úbytku napětí přímo na přechodu DS spínacího tranzistoru. Proto byl původně záměrně vybrán tranzistor s vyšší hodnotou R_{DS_ON} [12]. Tato variata se ovšem během testování projevila jako příliš nepřesná. Téměř lineární výstupní charakteristika tranzistoru se projevila daleko více teplotně závislá, než uvádí výrobce, dále i při konstatní teplotě nebyla zcela lineární (výrobce nejspíše záměrně používá logaritmická měřítka aby zakryl nelinearitu – viz obrázek 1.6) a v neposlední řadě jednotlivé kusy tranzistorů vykazovaly značné rozdíly v samotném R_{DS_ON} [10]. Z tohoto důvodu je v konečném řešení použit rezistor 1.7 v obvodu zdrojové elektrody (source), na kterém je

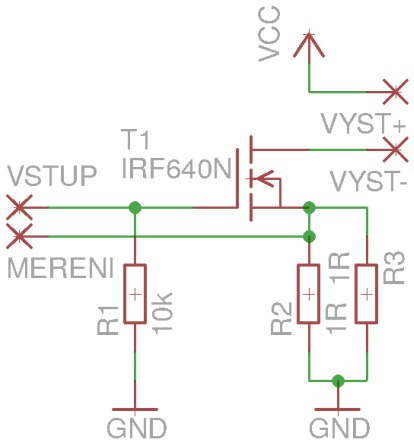
[9] Bez tohoto odporu by se snadno naindukovalo na spoji dostatečný náboj pro nechtěné otevření tranzistoru.

[10] Hlavní parametr unipolárního tranzistoru, který představuje zmíněnou téměř lineární závislost U_{DS} na protékaném proudu.

měřen úbytek napětí na protékaném proudu (tedy nepřímé měření). Hodnota tohoto rezistoru 1 Ω byla volena tak, aby bylo možno přímo využít vnitřní ADC převodníky mikrokontroleru, které mají rozlišení 10 bitů. Pro cílový mikrokontrolér je navíc při použití vnitřního referenčního napětí 1,1 V vhodně využit celý rozsah. Pro použití dostatečně přesných (1 %) rezistorů byla zvolena dvojice s dvojnásobným odporem zapojena paralelně (maximální povolená výkonová ztráta 0,6 W na kus). Toto zapojení navíc samo o sobě představuje i ochranný obvod – při průtoku špičkového proudu v řádu nekolika ampér totiž úbytek napětí na rezistorech kompěnzuje naopětí U_{GS}, takže se tranzistor při velkém nadproudu sám uzavře.



Obr. 1.6: Katalogové výstupní charakteristiky použitého tranzisotru [12].



Obr. 1.7: Schéma zapojení spínacího obvodu s měřením.

2 REALIZACE POE INJEKTORU

2.1 Elektrické schéma

Zapojení vychází z již nastíněného řešení spínání a bude upraveno dle nutnosti užití jiného mikrokontroléru a rovněž multiplexeru na straně měření napájení. Zapojení mikrokontroléru vychází z návrhové platformy, která je jen mírně upravená od doporučeného zapojení stanoveného výrobcem v datasheetu [9]. Na vstupně/výstupní porty mikrokontroléru jsou přímo připojeny jednotlivé spínací unipolární tranzistory pro každý port. Brána každého tranzistoru je ošetřena odporem 10 $k\Omega$ aby byl sveden případný naindukovaný elektrický náboj. Výstupní impedance I/O vývodů mikrokontroléru je totiž velmi vysoká při nízké logické úrovni na výstupu portu a i při samotném testování jsem ověřil, že běžné elektromagnetické pole prostředí stačilo k tomu, aby se tranzistor otevřel. V návrhovém prototypu byly využity výstupy odpovídající I/O pinům 22-45 vývojové desky. Pro měření odběru byl vybrán vstup prvního digitálně-analogového převodníku (ADC), tedy pin označený A0 vývojové desky. Pro měření odběru proudu na všech portech je pak použito trojice osmiportových analogových multiplexů 4051 [8]. Ty jsou připojeny paralelně na 3-bitovou sběrnici zapojenou na I/O piny 11-13 vývojové desky, výběr jednotlivého multiplexu je pak řízen I/O piny 5-7. Zvolené řešení sice nevyužívá celou šířku sběrnice tvořené dohromady 6-ti piny (připojujeme 24 kanálů na 6 bitů sběrníce – s teoretickou šířkou $2^6 = 64$, přičemž by stačilo 5 bitů - $2^5 = 32$), avšak vzhledem k parametrům mikrokontroléru a minimální ceně těchto multiplexů je toto řešení vyhovující.

Samotný napájecí zdroj je použit komerční, jelikož nabídka profesionálních přístrojových napaječů je široká a daná aplikace neklade na samotný zdroj specifické požadavky. Injektor je osazen zrojem 24 V/150 W firmy Mean Well – typ RS-150-24 [14]. Jeho mechanická konstrukce (výška pouhých 38 mm) jej předurčuje k montáži do rackové přístrojové skříně výšky 1U[1]. Tento zdroj by měl předpokládanému využití dostačovat svým výkonem, co se týče napájených směrovačů je toto napětí limitní (některé modely mají maximální limit pro napájeni 30 V). Osazený zdroj by měl pokrýt očekávané krátkodobé špičkám (např. při hromadném bootování směrovačů po výpadku napájení). Trvalý odběr by však neměl přesáhnout zhruba polovinu maximálního. V případě potřeby by bylo možné použít napájecí zdroj výkonnější – konstrukčně je samotný injektor dimenzován minimálně pro celkových 240 W (tedy proud 1 A/port).

[1] 1U (Rack Unit) odpovídá 1,75", tedy 44,4 mm.

2.2 DPS a mechanická konstrukce

Návrh desky plošných spojů vychází z umístění zařízení do 19" rackové montáže. Na přední panel bylo třeba vyvést konektry RJ45 pro všechny napájené porty (2x24 – vždy jeden vstupní a jeden výstupní – napájený). Dále bylo potřba na předním panel umístit další ethernetový port pro vlastní management a také štěrbina pro umístění paměťové karty micro SD. Během konstrukce bylo rovněž myšleno na umístění vývojové desky tak, aby bylo možno aktualizovat firmware pomocí technologie ISP (programování mikrokontroléru přímo v obvodu) pomocí na vývojové desce přítomného programátoru s vyvedeným rozhraním USB.

Při návrhu DPS bylo nutno přihlížet k mnoha faktorům, které ovlivňují vedení signálových vodičů, fyzické dostupnosti konektorů a rozumné rozmístění pro využití prostoru. Použitý síťový kontrolér (W5100) určil svým pouzdrem minimální konstrukční třidu VI. Díky tomu bylo možno navrhnout desku jako dvouvrstvou, což odpovídá požadavkům na nízkou cenu zařízení.

Po rozmístění konektorů pro vývojovou desku, síťové rozhraní a samotné napájecí obvody bylo nutno provést ruční natažení všech spojů. Během návrhu byla tetována funkce autorouteru programu Eagle. Po rozumném rozmístění všech komponent a rozvedení silových vodičů byl spuštěn proces s minimem požadavků a pouze 5-ti požadovanými variantami. Výsledkem po více než 12-ti hodinovém běhu na 4 jádrech procesoru Intel(R) Core(TM) i7 (2670QM CPU @ 2.20GHz) obsahovala nejoptimálnější varianta přes 300 prokovů s tím, že vyžadovala propojení dalších více než 100 spojů. Ručně navržená deska obsahuje oproti tomu necelých 200 prokovů.

Vzhledem k tomu, že nebylo možné konstrukčně striktně dodržet vedení symetrických párů mezi konektory a oddělovacími transformátory [2], bylo nutno tyto spoje vést co nejkratší a dodržet pokud možno stejné délky vodičů pro jednotlivé páry i v rámci každého páru. Pro alespoň částečné přiblížení se charakteristické impedanci 100 Ω, je navíc každý pár veden co nejdéle tak, že médium desky tvoří dielektrikum mezi vodiči. Toto řešení by mělo dostačovat pro dodržení parametrů ehternetu [6].

Deska plošných spojů je přišroubována na distanční sloupky ukotvené ve spodní části skříně. Přední panel obsahuje otvory pro konektory – zejména síťové, USB konektor vývojové desky a štěrbina pro paměťovou kartu. Zadní strana pak obsahuje síťový (EURO) konektor a hlavní vypínač. Použitá skříň je typ RE 4031 od firmy Revatech (<http://revatech.cz/src/katalog1/det18.htm>.

[2] V oblasti síťových zařízení se v podstatě výhradně využívají vícevrstvé desky s využitím vnitřních vrstev jako potenciálových ploch.

2.3 Firmware

2.3.1 Struktura programu

Kód mikrokontroléru se zkládá ze tří základních částí – částí definicí parametrů, struktury inicializační a části obslužné smyčky. Definiční část obashuje část připojení jednotlivých knihoven, namapování jednotlivých I/O pinů k jednotlivým funkcím a také obsahuje definice jednotlivých obslužných funkcí webového serveru. Část inicializační obsahuje funkce pro zahájení či obnovšní činnosti po zapnutí/restartu zařízení. Ačkoli část obslužné smyčky je nejdůležitější, obsahuje však jen cyklické volání webového serveru. Vývojový diagram celého programu je vidět na obrázku 2.1

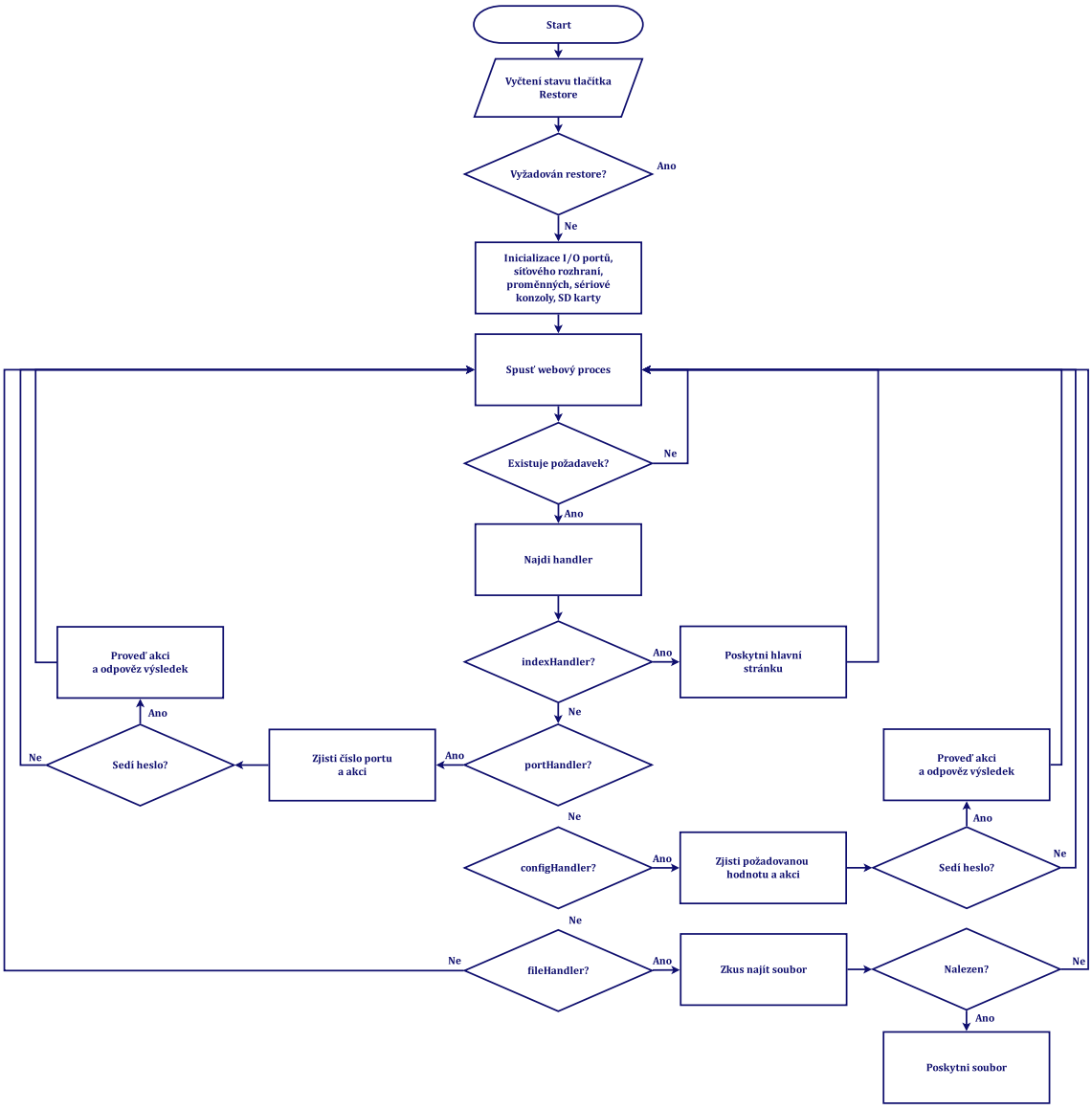
Definiční část

K funkčnosti všech částí zařízení je potřeba následujících knihoven:

- SPI – pro obsluhu komunikační sběrnice SPI.
- Ethernet – ovladač síťového rozhraní Wiznet W5100.
- Flash – knihovna obsahující makra k snadnému namapování používaných konstant do programové paměti (k ušetření zabrané paměti RAM).
- SD – knihovna zajišťující komunikaci s paměťovou kartou.
- EEPROM – knihovna obluhující zápis a čtení z/do paměti EEPROM.
- TinyWebServer – knihovna poskytující služby webového serveru.

Všechny zmíněné knihovny jsou s výjimkou TinyWebServeru distribuovány současně s vývojovým prostředím (IDE) Arduino a jsou šířeny pod GPL licencí [11]. Základní dostupná dokumentace ja pak šířena pod licencí Creative Commons Attribution Share-Alike. Tyto licence umožňují použití jak ke studijním tak i komerčním účelům. K ladění vyvíjeného programu je využívána standardní sériová konzole, která je součástí programátoru přímo na vývojové desce a je dostupná přes rozhraní USB. Sériová konzole je však iniciována až po úspěšném připojení pomocí sběrnice USB. Množství vypisovaných ladících informací je minimalizováno na základní funkce, pro rozšířený výpis je možno nastavit proměnnou DEBUG na pravdivostní hodnotu true.

Síťová karta je iniciována se statickou konfigurací uloženou v paměti EEPROM. výchozí IP adresa je nastavena na 192.168.1.177, v sekci nastavení iwebového rozhraní je pak možné ji změnit. Použitý kontrolér umožňuje i automatickou konfiguraci pomocí protokolu DHCP, to však v případě manažovatelného rozhraní by se však jednalo spíše o nežádoucí vlastnost. Adresa fyzického rozhraní MAC je v programu stanovena pevně, pro sériovou výrobu by bylo potřeba provést registraci výrobce u organizace IEEE a přidělovat každému ivyrobenému kusu unikátní adresu.



Obr. 2.1: Vývojový diagram obslužného programu PoE injektoru.

Inicializační část

Tato část začíná inicializací sériové konzoly. Ta je inicializována pouze v případě připojeného kabelu USB. V případě, že je již zařízení v provozu a teprve následně je kabelem USB připojen počítač dojde k resetu celého zařízení. Hned po inicializaci konzoly je pak kontrolován stav mikrospínače RESTORE. Pokud je ve stisknutém stavu během dvou testů opakovaných po sekundě, je v paměti EEPROM přepsána kompletní konfigurace do úvodního nastavení. Toto umožňuje v případě chybné konfigurace reset zařízení do výchozího stavu aby mohlo být znovu nastaveno. Po uplynutí této doby a případném resetu jsou pak konfigurační data načtena z paměti EEPROM do RAM.

Dále jsou nastaveny režimy jednotlivých vstupně-výstupních (I/O) portů. Kód zde počítá odkazuje na definiční část, tedy případné fyzické změny kkonstrukce nemusí být zde reflektovány. Následně je aktivována paměťová karta a je připojen souborový systém karty. Pokud by bylo potřeba kartu vrůběhu běhu zařízení vyměnit, bylo by potřeba přidat rutinu volanou ze smyčky, což by mohlo znatelně prodloužit-průměrný čas odezvy. Nakonec je nakonfigurováno a spuštěno síťové rozhraní – IP adrese nastavena je nastavena dle adresy uložené v EEPROM.

Část smyčky

Jak již bylo zmíněno tato část obsahuje volání samotného webového procesu. Vzhledem k přítomnosti watchdogů v mikrokontroléru by bylo možnou použít přerušení a tím umožnit přechod do úsporného režimu s aktivním přerušením. K tomu byla na desce plošných spojů zachován pájecí můstek (INT), který stačí přemostit a poté po úpravě firmwaru využít tuto funkci. Vzhledem k tomu, že od mikrokontroléru nepožadujeme další úlohy a úspora spotřeby by ve srovnání se zbytkovým odběrem napájecího zdroje, síťového kontroléru a dalších součástek byla nezaznamenatelná.

2.3.2 Webové rozhraní

Jako web server byl použit otevřený projekt TinyWebServer [21] pro jeho jednoduchost a minimální požadavky na prostředky mikrokontroléru (zabírá necelých 512 bytů RAM a pouze 10 kB programové paměti). Kromě standradní procedury pro poskytnutí obsahu souboru přes protokol HTTP (sendFileName) byly implementovány rutiny pro poskytnutí domovské stránky (indexHandler), obsluhu portů (portHandler) a konfiguraci zařízení (configHandler). Tyto rutiny poté využívají další procedury (i rekurzivně) k provedení vyžadovaných akcí tak, aby byla minimalizována duplicita kódu a kód byl čitelnější. Vzhledem k rozsáhlosti zde nebudou podrobně rozebírány, čtenář má možnost nahlédnout do přílohy A.5.

Pro snadné ovládání bylo vytvořené snadno ovladatelné webové rozhraní. Obsahuje stavové menu (Status), kde je možné ověřit stav portů, menu ovládání jednotlivých portů, menu nastavení a také menu nápovědy. Grafická část je postavena na moderním projektu bootstrap [3]. Tento otevřený projekt představuje širokou platformu poskytující uniformní vzhled webových rozhraní napříč všemi prohlížeči (včetně mobilních) a přitom obsahuje velkou škálu možností přizpůsobení konkrétnímu projektu. Pro ověřovací projekt byla zvolena možnost stažení potřebných souborů se styly a funkcemi javascriptu přímo ze stránek vývojářů, čímž se minimalizovala potřeba uložení větších souborů ve flash paměti mikrokontroleru. Pro konečné řešení pak bylo zvoleno uložení aktuální verze na paměťové kartě. Tato konfigurace umožňuje případný update velké částí webového rozhraní bez nutnosti přeprogramování mikrokontroleru.

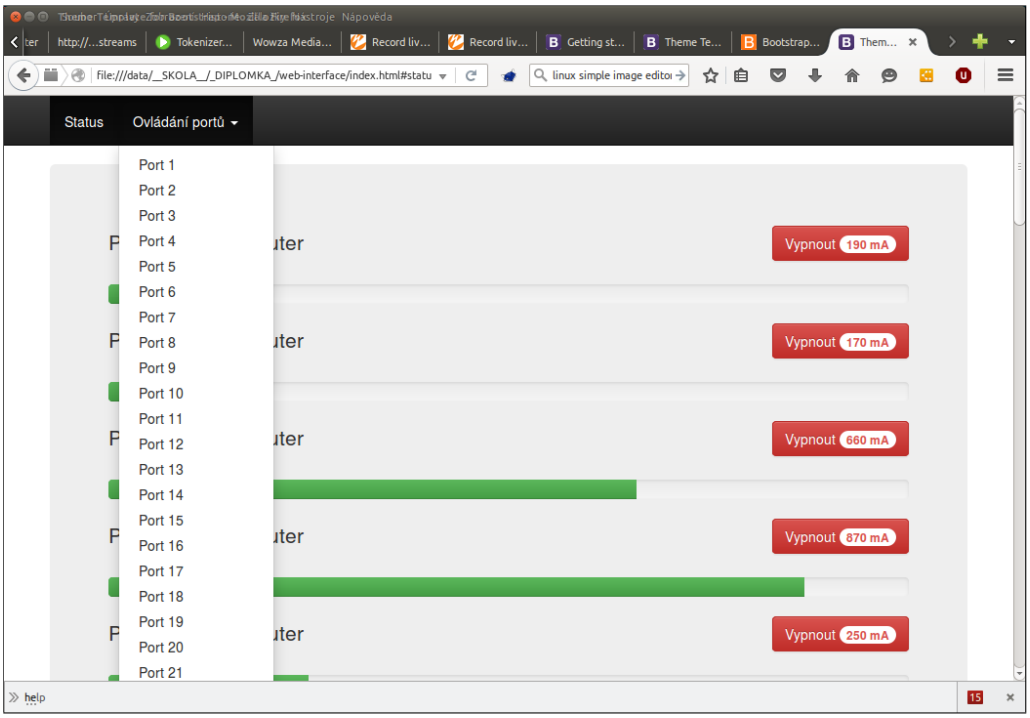
V menu status je možno přehledněi – na jedné stránce zkontrolovat stav všech portů. Rozbalovací menu "Ovládání portů" pak umožňuje výběř jednotlivých portů se zobrazením aktuálně odebíraného proudu a možností přepnutí stavu. Menu je programově ošetřeno tak, že nabízí uživateli vždy jen aktuálně platné volby a neobtěžuje tak uživatele nepěřehledným výčtem všech možností. Zajištění těchto funkcionalit je přenecháno na webovém prohlížečí, kde se dá předpokládat nadbytečný výkon v porovnání s velmi omezeným výkonem mikrokontroléru. Náhled rozhraní je vidět ze snímků obrazovky 2.2 a 2.3.

Celý kód firmwaru se nachází na přiloženém kompaktním disku, výňatek lze najít jako přílohu A.5.

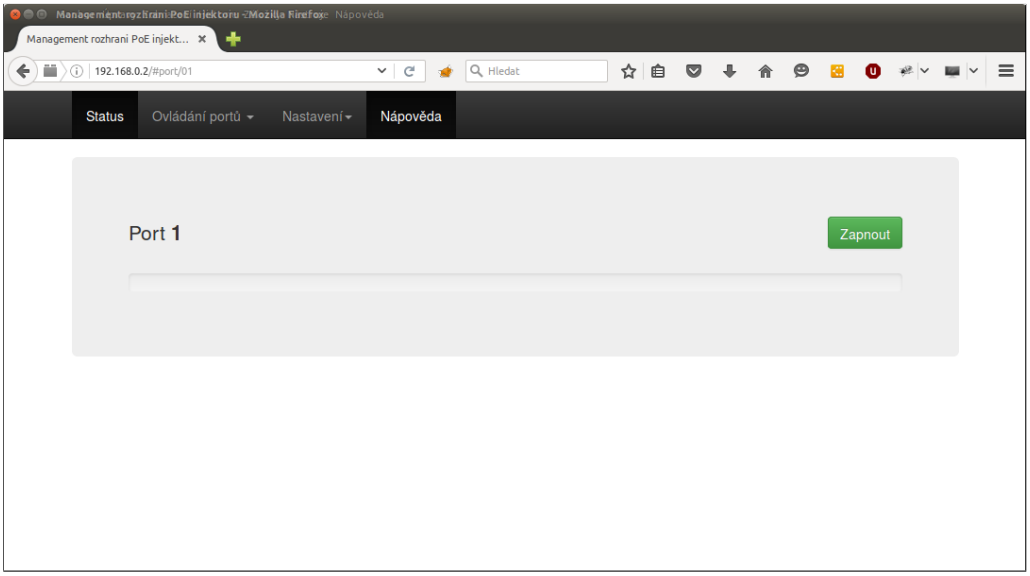
2.4 Testování

Vzhledem k tomu, že ke dni odevzdání diplomové práce nebyly dodány všechny potřebné součástky nebylo možné otestovat zařízení vcelku. Průběžné testování spočívalo v testování na kontatkním poli. Ověřováno bylo již výše zmíněné zapojení měřících obvodů spotřeby, kde byl měřen úbytek na přechodech tranzistorů, později na pomocných rezistorech – a hodnoty změřené DAC kontrolérem byly porovnávány s hodnotami zjištěnými přímým měřením proudu. Chyby měření byly v naprosté většině do 5 %, což bylo shledáno akceptovatelné pro orientační měření. Při testování občas docházelo k chybným měřením, kdy příčina byla softwarového charakteru, kdy následné měření již poskytlo správnou hodnotu.

Po osazení DPS bude kromě běžného oživení ověřit (detailně proměřit a otestovat) splnění podmínek dle doporučení 802.3 a to nejlépe pro všechny podporované typy médií – tedy 10/100/1000BASE-T, případně i 100BASE-TX. Před uvedením do produkčního prostředí bude potřeba otestovat i ochranu proti přetížení



Obr. 2.2: Náhled hlavního menu webového rozhraní.



Obr. 2.3: Náhled ovládání jednotlivých portů webového rozhraní.

(zkrat), což je v praxi velmi běžný jev (všechna rozhraní splňující standrad 802.3 musí být proti libovolnému propojení/zkratu odolná a splňovat bezpečnostní parametry pro zařízení/vedení malého napětí).

3 ZÁVĚR

Zadané téma bylo zpracováno na úrovni ověření konceptu a po upravení návrhu o zjištěné nedostatky byl návrh optimalizován. Bylo ověřeno konkrétní řešení jak samotného managementu injektoru, tak možnosti měření spotřeby. Byl napsán firmware, který splňuje požadavky plné funkčnosti (jak spínání napájení jednotlivých portů, tak měření odběru), i když jeho vylepšení a rozšíření funkcionality je umožněno i díky umístění USB konektoru na předním panelu zařízení, které umožňuje nahrání nové verze bez nutnosti demontáže zařízení z racku. Pro výrobu konečného zařízení byla zkonstruována kompletní deska plošných spojů umožňující montáž do typizované přístrojové skříně určené pro montáž do standardních 19" rozvaděčů. Během vývoje zařízení bylo otestováno několik možných konstrukčních řešení spínání napájení. Bylo ověřeno, že původně zamýšlené měření spotřeby pomocí měření úbytku napětí na kanále použitého spínacího tranzistoru není vhodné ani k orientačnímu měření spotřeby a bylo tedy přistoupeno k nepřímému měření protékaného proudu na rezistoru zapojeném v sérii s napájeným zařízením. Navržený injektor umožňuje napájení až 24 zařízení přičemž je možné vzdáleně ovládat stav každého jednotlivého portu pomocí uživatelsky přívětivého webového rozhraní. Pomocí toho rozhraní je také možno konfigurovat další nastavení injektoru a monitorovat spotřebu na jednotlivých portech. Samotná realizace je v době odevzdání této práce dokončována.

LITERATURA

[1] BARNETT, D., GROTH, D., MCBEE, J. *Cabling: The Complete Guide to Network Wiring*. Third Edition. SYBEX Inc., Alamdeda, CA, USA, 2004. ISBN 0-7821-4331-8.

[2] DEERING, S., HINDEN, R. *Internet Protocol, Version 6 (IPv6) – Specification*. RFC online. [citováno 2015-11-06] Xerox PARC, Ipsilon Networks (USA): Network Working Group, 1995. Dostupné z: <http://tools.ietf.org/html/rfc1883>.

[3] FENKART, H., LAUKE, P. H., OTTO, M., REBER, C., THILO, J., THORNTON, J., XHMIKOS R. a kol. *iBootstrap framework*. Online dokumentiace. [citováno 2015-12-10] V3.3.6 (USA), 2015. Dostupné z: <https://github.com/twbs/bootstrap/releases/download/v3.3.6/bootstrap-3.3.6-dist.zip>.

[4] GALIT M. *All You Need To Know About Power over Ethernet (PoE) and the IEEE 802.3af Standard*. Online dokument. [citováno 2015-11-30] PowerDsine (USA), 2004. Dostupné z: <https://portal.chippc.com/support/downloads/files/PoE_and_IEEE802_3af.pdf>.

[5] HUSTON, G. *IPv4 Address Report*. Online dokument. [citováno 2015-11-30] (USA), 2013. Dostupné z: <http://www.potaroo.net/tools/ipv4/index.html>.

[6] IEEE Computer Society. *IEEE Standard for Ethernet*. IEEE Standard. [citováno 2015-11-23] The Institute of Electrical and Electronics Engineers, Inc., New York, NY (USA), IEEE, 2012. Dostupné z: <http://www.trincoll.edu/Academics/MajorsAndMinors/Engineering/Documents/IEEEStandardforEthernet.pdf> ISBN 973-07381-7312-2.

[7] IEEE Computer Society. *Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications Amendment: Data Terminal Equipment (DTE) Power via Media Dependent Interface (MDI)*. IEEE Standard. [citováno 2015-10-15] The Institute of Electrical and Electronics Engineers, Inc., New York, NY (USA), IEEE, 2003. Dostupné z: <http://www.lan-power.com/pdf/802.3af-2003.pdf> ISBN 978-0-7381-6684-1 STDPD97126.

[8] kolektiv *HEF4051B 8-channel analog multiplexer/demultiplexer*. Rev. 12 — 25 March 2016 [citováno 2016-04-30] NXP Semiconductorsi N.V., 2016. Dostupné z: <http://www.nxp.com/documents/data_sheet/HEF4051B.pdf>.

[9] kolektiv *Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V 8-bit Atmel Microcontroller with 16/32/64KB In-System Programmable Flash.* 2549Q–AVR–02/2014 [citováno 2015-11-03] Atmel Corporation, California (USA), 2014. Dostupné z: <http://www.atmel.com/images/atmel-2549-8-bit-avr-microcontroller-atmega640-1280-1281-2560-2561_datasheet.pdf>.

[10] kolektiv *ATmega48A/PA/88A/PA/168A/PA/328/P - ATMEL 8-BIT MICROCONTROLLER WITH 4/8/16/32KBYTES IN-SYSTEM PROGRAMMABLE FLASH.* Atmel-8271J-AVR- ATmega-Datasheet_11/2015 [citováno 2015-11-03] Atmel Corporation, California (USA), 2015. Dostupné z: <http://www.atmel.com/images/atmel-8271-8-bit-avr-microcontroller-atmega48a-48pa-88a-88pa-168a-168pa-328-328p_datasheet_complete.pdf>.

[11] kolektiv *iGNU Operating System – Licenses.* S11-0517-Rev. B, 21-Mar-11 [citováno 2016-04-06] Free Software Foundation, 2017. Dostupné z: <http://www.vishay.com/docs/91291/91291.pdf>.

[12] kolektiv *IRF640N, IRF640NS, IRF640NL.* Rev. 6 — 22 May 2015 [citováno 2015-10-26] NXP Semiconductors N.V., 2015. Dostupné z: <http://www.nxp.com/documents/data_sheet/74HC_HCT4067.pdf>.

[13] kolektiv *IRFZ44, SiHFZ44.* S11-0517-Rev. B, 21-Mar-11 [citováno 2016-04-12] Vishay Siliconix, 2008. Dostupné z: <http://www.vishay.com/docs/91291/91291.pdf>.

[14] kolektiv *RS-150 series 150W Single Output Switching Power Supply.* RS-150-SPEC 2015-07-08 [citováno 2016-04-12] Mean Well, 2015. Dostupné z: <http://www.meanwell.com/mw_search/RS-150/RS-150-spec.pdf>.

[15] kolektiv *W5100 Datasheet.* Technická specifikace online, Jan 30th 2008 [citováno 2016-05-03] Version 1.1.6 WIZnet Co., Inc., 2008. Dostupné z: <https://www.sparkfun.com/datasheets/DevTools/Arduino/W5100_Datasheet_v1_1_6.pdf>.

[16] kolektiv *WÜRTH ELEKTRONIK 749020011A.* Datasheet online, 02 Aug 2013 [citováno 2015-10-06] WÜRTH ELEKTRONIK, 2014. Dostupné z: <http://www.farnell.com/datasheets/1803237.pdf>.

[17] kolektiv *WÜRTH ELEKTRONIK HORIZONTAL SHIELDED 2 x 6 STACKED PORT WITH EMI PANEL FINGER 8P8C MODULAR JACK TAB*

UP DOWN. Technická specifikace online, 02 Aug 2013 [citováno 2015-10-06] WÜRTH ELEKTRONIK, 2014. Dostupné z: <http://www.farnell.com/datasheets/1803237.pdf>.

[18] LUND, Mark W. *Wire Gauge and Current Limits Including Skin Depth and Strength*. Dokument online. [citováno 2015-11-30] PowerStream Technology, 2015. Dostupné z: <http://www.powerstream.com/Wire_Size.htm>.

[19] MARGOLIS, Michael. *Arduino Cookbook*. 2nd ed. O'Reilly, 2011. ISBN 1449313876.

[20] OLIVIERO, A., WOODWARD, B. *Cabling: The Complete Guide to Copper and Fiber-Optic Networking*. 5th edition John Wiley Sons, 2009 ISBN 0470550058.

[21] PREDESCU, O. *Arduino Tiny Web Server*. Dokument online Ovidiu Predescu, 2010-2013. Dostupné z: <https://webweavertech.com/ovidiu/weblog/archives/000484.html>.

[22] SCHWÖBEL, U. *S/STP cable*. Online dokument. [citováno 2015-11-30] 2015. Dostupné z: <https://upload.wikimedia.org/wikipedia/commons/e/ef/S-STP-cable.svg>.

[23] WILCHER, D. *Arduino Electronics Blueprints*. Packt Publishing, 2015. ISBN 9781784393601.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

1000BASE-T gigabitový ethernet vedený metalickým kabelem – 1000 Mbit/s ethernet over copper cable

AVR označení pro 8bitové a některé 32bitové RISC mikrokontrolery fy. Atmel

DEMUX demultiplexer

DPS deska lošných spojů

FET polem řízený tranzistor – Field Effect Transistor

MOSFET polem řízený tranzistor vyrobený oxidační metodou – Metal Oxide Semiconductor Field Effect Transistor

MUX multiplexer

PCB deska lošných spojů – angl – angl.

PoE napájení po ethernetové kabeláži – Power Over Ethernet

SF-UTP nestíněná kroucená dvojlinka – Unshielded Twisted Pair

U-FTP nestíněná kroucená dvojlinka – Unshielded Twisted Pair

UTP nestíněná kroucená dvojlinka – Unshielded Twisted Pair

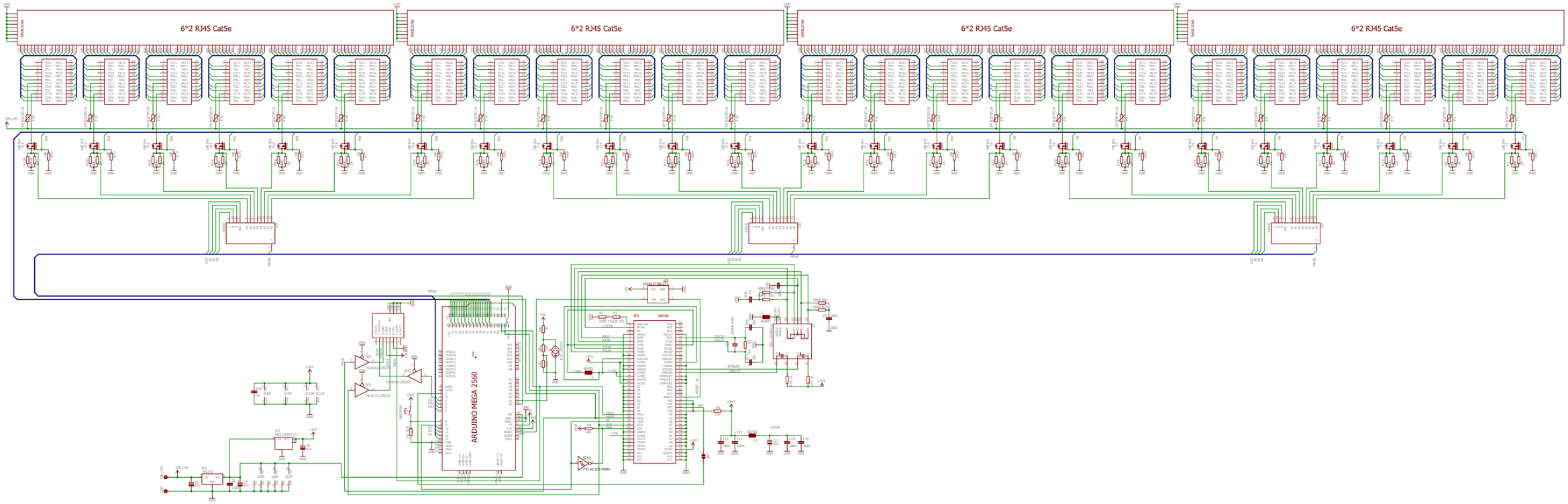
SEZNAM PŘÍLOH

A PŘÍLOHY

A.1 Soupiska součástek a materiálu PoE injektoru

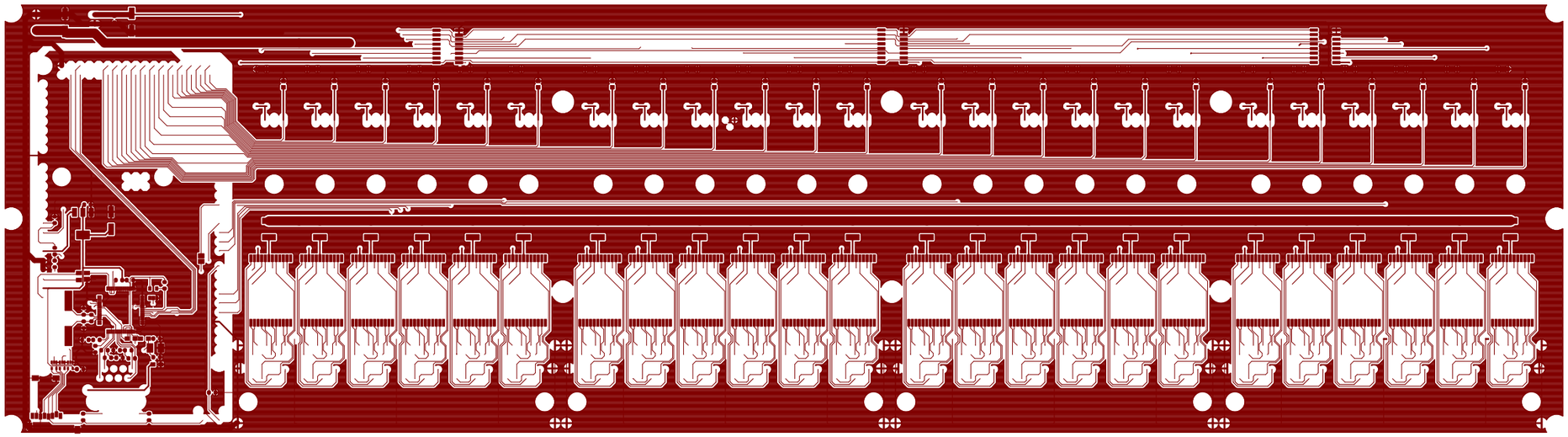
1 ks	Arduino/Genuino MEGA2560	-
3 ks	BLM21	SMD 0805
1 ks	CAT811TTBI-CT3	SMD SOT143
24 ks	IRFZ44	TO220
3 ks	Kondenzator elektrolyticky 47 uF/30 V	SMD
1 ks	Kondenzator keramicky 1 uF	SMD 0603
8 ks	Kondenzator keramicky 100 nF	SMD 0603
2 ks	Kondenzator keramicky 22 pF	SMD 0603
1 ks	Krystal 25 MHz	SMD 5*13,5 mm
1 ks	LDO 7812	TO220
1 ks	LM335	TO92
1 ks	MC33269ST-3.3T3	SOT223
1 ks	Mikrospinac do DPS 90	B3F-31XX
24 ks	Odddlovaci transformator HX5008NL	-
24 ks	PTC elektronicka pojistka PPTC075F/24	SMD 1812
1 ks	RB1-125BAG1A	MagJack RJ45
48 ks	Rezistor dratovy 2R	0207
1 ks	Rezistor 49,9 R 1 %	SMD 0603
1 ks	Rezistor 300R 1%	SMD 0603
2 ks	Rezistor 1k	SMD 0603
1 ks	Rezistor 2 k	SMD 0603
1 ks	Rezistor 2,2 k	SMD 0603
1 ks	Rezistor 7,8 k	SMD 0603
26 ks	Rezistor 10k	SMD 0603
1 ks	Rezistor 12k 1%	SMD 0603
1 ks	Rezistor 1M	SMD 0603
1 ks	Slot na microSD kartu SDCARD-15TW-8821	SMD
1 ks	Svorky do DPS pro vodic 1,5-2,5 mm^2	roztec 5 mm
1 ks	Wiznet W5100	SMD
4 ks	2X6 RJ45 Cat5E (WUE615096243321)	-
3 ks	4051D	SO16
3 ks	74LVC1G125DCK	SC70-5
1 ks	74LVC1G14DBV	SOT23-5

Příloha A.2 Elektrické schéma PoE injektoru



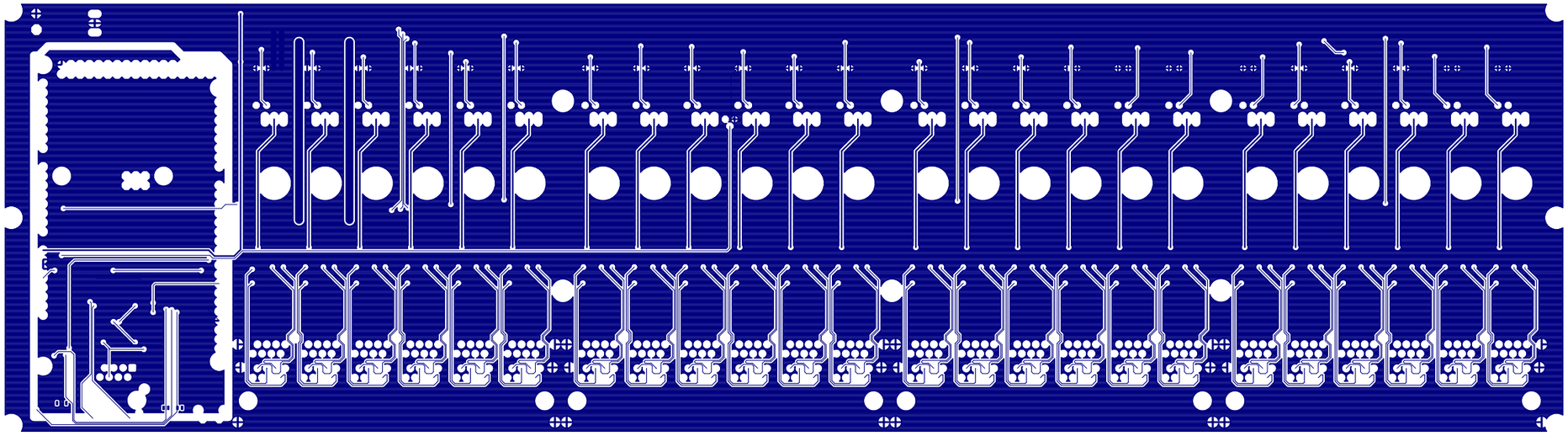
	PoE_Injektor
	5/23/2016 4:13 AM
	Sheet: 1/1

Příloha A.3 Deska plošných spojů PoE injektoru – strana součástek



PoE_Injektor
5/23/2016 4:19 AM

Příloha A.4 Deska plošných spojů PoE injektoru strana spojů



PoE_Injektor
5/23/2016 4:19 AM

A.5 Program mikrokontroléru

```
#include <SPI.h>
#include <Ethernet.h>
#include <Flash.h>
#include <SD.h>
#include <EEPROM.h>
#include <TinyWebServer.h>

// Functions:
boolean file_handler(TinyWebServer& web_server);
boolean indexHandler(TinyWebServer& web_server);
boolean portHandler(TinyWebServer& web_server);
boolean configHandler(TinyWebServer& web_server);

// Global variables:
boolean DEBUG = false;
boolean has_filesystem = true;
Sd2Card card;
SdVolume volume;
SdFile root;
SdFile file;
static int eepromMac = 0;
static int eepromIP = 6;
static int eepromPortStatus = 100;
static int eepromPortPower = 200;
static int eepromPortMeasAddr = 300;
static int eepromPortMeas = 400;
static uint8_t mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
byte ip[] = { 10, 0, 0, 200 };
//TODO fill from EEPROM
boolean portStatus[] = {true, false, true, false, true, false, true, false, true,
    false, true, false, true, false, true, false, true, false, true, false, true,
    false, true, false};
static byte portPower[] = {22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35,
    36, 37, 38, 39, 40, 41, 42, 43, 44, 45};
static byte portReadAddr[] = {11, 12, 13 };        // Port measurement address (3
    bits address witchin multiplexer, 3bits negative chipselect (multiplexer), 2
    unused
static byte portReadChipSelect[] = {5, 6, 7}; // Port measurement chip select
static byte portRead[] = {B00001100, B00101100, B01001100, B01101100, B10001100,
    B10101100, B11001100, B11101100, B00010100, B00110100, B01010100, B01110100,
    B10010100, B10110100, B11010100, B11110100, B00011000, B00111000, B01011000,
    B01111000, B10011000, B10111000, B11011000, B11111000};
static byte portMeasurement = 0;
const int sdChipSelect = 4;            // SD card chipSelect

TinyWebServer::PathHandler handlers[] = {
  {"/", TinyWebServer::GET, &indexHandler },
  {"/port/" "*", TinyWebServer::GET, &portHandler },
  {"/" "*", TinyWebServer::GET, &file_handler },
  {NULL},
};

boolean file_handler(TinyWebServer& web_server) {

  if(!has_filesystem) {
    web_server.send_error_code(500);
```

```
    web_server << F("Internal␣Server␣Error");
    return true;
  }

  char* filename = TinyWebServer::get_file_from_path(web_server.get_path());

  if(!filename) {
        web_server.send_error_code(400);
        web_server << F("Bad␣Request");
        return true;
  }

  sendFileName(web_server, filename);
  free(filename);
  return true;
}

void sendFileName(TinyWebServer& web_server, const char* filename) {

  TinyWebServer::MimeType mime_type
    = TinyWebServer::get_mime_type_from_filename(filename);
  if (file.open(&root, filename, O_READ)) {
    web_server.send_error_code(200);
    web_server.send_content_type(mime_type);
    web_server.end_headers();

    Serial << F("Read␣file␣"); Serial.println(filename);
    web_server.send_file(file);
    file.close();
  } else {
    web_server.send_error_code(404);
    web_server.send_content_type("text/plain");
    web_server.end_headers();

    Serial << F("Could␣not␣find␣file:␣"); Serial.println(filename);
    web_server << F("404␣-␣File␣not␣found␣") << filename << "\n";
  }
}

boolean portHandler(TinyWebServer& web_server) {
  String portURL = TinyWebServer::get_file_from_path(web_server.get_path());
  byte QSStart = portURL.indexOf('?');
  if (QSStart < 0) {
    QSStart = 255;
  }
  byte trailSlash = portURL.indexOf('/');
  byte portNumber = portURL.substring(0,QSStart).toInt() - 1; // Ports indexed
      starting zero
  if (portNumber >= 0 && portNumber < sizeof(portPower)) {
    web_server.send_error_code(200);
    web_server.end_headers();
    if (DEBUG) {
      web_server << F("portNumber␣=␣");
      web_server << portNumber;
      web_server << F("\nQSStart␣=␣");
      web_server << QSStart;
      web_server << F("<br␣/>\n");
    }
```

```
  if (portURL.substring(QSStart+1,QSStart+6) == "ZAPNI") {
    web_server << F("Zapinam␣port:␣");
    // web_server << portNumber + 1;
    setPortStatusUp(portNumber);
    //web_server << F("<br />\n");
  }
  else if (portURL.substring(QSStart+1,QSStart+6) == "VYPNI"){
    web_server << F("Vyp n m␣port:␣");
   //  web_server << portNumber + 1;
    setPortStatusDown(portNumber);
    // web_server << F("<br />\n");
  }
    web_server << F("<div␣style=\"height:␣60px;\"><p␣style=\"float:␣left;\">Port␣<b
        >");
    web_server << portNumber + 1;
    web_server << F("</b></p><button␣style=\"float:␣right;\"␣type=\"button\"␣
        onclick=\"window.location.href='#port/");
    if (portNumber < 10) { web_server << "0";}
    web_server << portNumber + 1;
    web_server << getPortStatus[portNumber];
    if (getPortStatus[portNumber] > 0)
      { web_server << F("?vypni'\"␣class=\"btn␣btn-danger\">Vypnout␣<span␣class=\"
          badge\">"); }
    else { web_server << F("?zapni'\"␣class=\"btn␣btn-success\">Zapnout␣<span␣class
        =\"badge\"␣style=\"display:␣none;\">");  }
    web_server << getPortCurrent(portNumber);
    web_server << F("␣mA</span></button></p></div><div␣class=\"progress\"><div␣
        class=\"progress-bar␣progress-bar-success\"␣style=\"width:␣");
    web_server << getPortCurrent(portNumber)/10;
    F("%\"></div></div>");

  } else {
    web_server.send_error_code(404);
    web_server.send_content_type("text/plain");
    web_server.end_headers();

    web_server << F("404␣-␣Port␣not␣found:␣") << portNumber << "\n";
    web_server << F("Port␣URL:␣") << portURL << "\n";

}
  return true;
}

boolean getPortStatus(byte& portNumber)
{
  boolean s;
  s = EEPROM.read(eepromPortStatus + portNumber);
  return s;
}

void setPortStatusUp(byte& portNumber)
{
  EEPROM.write(eepromPortStatus + portNumber, true);
  digitalWrite(portPower[portNumber], HIGH);
}

void setPortStatusDown(byte& portNumber)
{
  EEPROM.write(eepromPortStatus + portNumber, false);
```

```
  digitalWrite(portPower[portNumber], LOW);
}

int getPortCurrent(byte& portNumber)
{
  int m;
  if (DEBUG) {
    Serial << F("portRead = ");
    for (int i = 0; i < 8; i++) {
      Serial.print(bitRead(portRead[portNumber],7-i));
    }
    Serial << F("\n");
  }
  digitalWrite(portReadAddr[0], bitRead(portRead[portNumber],7));
  digitalWrite(portReadAddr[1], bitRead(portRead[portNumber],6));
  digitalWrite(portReadAddr[2], bitRead(portRead[portNumber],5));
  digitalWrite(portReadChipSelect[0], bitRead(portRead[portNumber],4));
  digitalWrite(portReadChipSelect[1], bitRead(portRead[portNumber],3));
  digitalWrite(portReadChipSelect[2], bitRead(portRead[portNumber],2));
  m = analogRead(portMeasurement);
  return m;
}

void setIPAddress(uint8_t* ipAddr) {
  for (int i = 0; i < 4; i++) {
    EEPROM.write(eepromIP + i, ipAddr[i]);
    ip[i] = ipAddr[i];
  }
}

/*
byte[4] getIPAddress() {
  byte[4] IPAdd;
  for (int i = 0; i < 4; i++) {
    IPAdd[i] = EEPROM.read(eepromIP + i);
    ip[i] = IPAdd[i];
  }
  return IPAddr;
}
*/
boolean indexHandler(TinyWebServer& web_server) {
  web_server.send_error_code(200);
  web_server.send_content_type("text/html");
  web_server.end_headers();
  web_server << F("<!DOCTYPE html><html lang=\"cz\"><head><meta charset=\"utf-8\"><
      meta http-equiv=\"X-UA-Compatible\" content=\"IE=edge\"><meta name=\"viewport
      \" content=\"width=device-width, initial-scale=1\"><meta name=\"description\"
      content=\"Managing interface of passive PoE injector.\"><meta name=\"author
      \" content=\"Pavel Frkal\"><title>Management rozhrani PoE injektoru</title>")
      ;
  web_server << F("<link href=\"/bot_min.css\" rel=\"stylesheet\"><link href=\"/
      boot-the.css\" rel=\"stylesheet\"><link href=\"/ie10-wo.css\" rel=\"
      stylesheet\"><link href=\"/theme.css\" rel=\"stylesheet\"><script src=\"/ie-
      em-wa.js\">");
  web_server << F("</script></head><body role=\"document\"><nav class=\"navbar
      navbar-inverse navbar-fixed-top\"><div class=\"container\"><div id=\"navbar\"
      class=\"navbar-collapse collapse\"><ul class=\"nav navbar-nav\"><li class=\"
      active\"><a href=\"#status\">Status</a></li><li class=\"dropdown\"><a href
      =\"#\" class=\"dropdown-toggle\" data-toggle=\"dropdown\" role=\"button\"
```

```
      aria-haspopup=\"true\" aria-expanded=\"false\">Ovl d n port <span class
      =\"caret\"></span></a><ul class=\"dropdown-menu\">");
  for (int i = 1; i <= sizeof(portPower); i++)
    { web_server << F("<li><a href=\"#port/");
    if (i < 10) { web_server << 0; }
    web_server << i;
    web_server << F("\">Port ");
    web_server << i;
    web_server << F("</a></li>");
  }
  web_server << F("</ul></li><li class=\"dropdown\"><a href=\"#\" class=\"dropdown-
      toggle\" data-toggle=\"dropdown\" role=\"button\" aria-haspopup=\"true\" aria
      -expanded=\"false\">Nastaven <span class=\"caret\"></span></a><ul class=\"
      dropdown-menu\"><li>1</li></ul></li><li class=\"active\"><a href=\"#help\">
      N pov da </a></li>");
  web_server << F("</ul></div></div></nav><div class=\"container theme-showcase\"
      role=\"main\"><div class=\"jumbotron\" id=\"hlavni\"><p>Vitejte v nastaveni
      manazovaneho PoE injektoru!</p><p>Pro nastaveni/stav vyberte z nabidky.</p></
      div></div> <script type=\"text/javascript\" src=\"/jquery.js\"></script><
      script src=\"/boot.js\">");
  web_server << F("</script><script src=\"/docs-min.js\"></script><script src=\"
      ie10-wa.js\"></script><script type=\"text/javascript\">var includeDiv = $(\"#
      hlavni\");$(window).on('hashchange', function() {var href = location.hash.
      slice(1);includeDiv.load('./'+href);});</script> </body></html>\n");
  return true;
}

boolean configHandler(TinyWebServer& web_server) {

}

boolean has_ip_address = false;
TinyWebServer web = TinyWebServer(handlers, NULL);

const char* ip_to_str(const uint8_t* ipAddr)
{
  static char buf[16];
  sprintf(buf, "%d.%d.%d.%d\0", ipAddr[0], ipAddr[1], ipAddr[2], ipAddr[3]);
  return buf;
}

void setup() {
  Serial.begin(9600);
  analogReference(INTERNAL1V1);
  for (int i = 0; i < sizeof(portPower); i++) {
    pinMode(portPower[i], OUTPUT);
  }
  for (int i = 0; i < sizeof(portReadAddr); i++) {
    pinMode(portReadAddr[i], OUTPUT);
  }
  for (int i = 0; i < sizeof(portReadChipSelect); i++) {
    pinMode(portReadChipSelect[i], OUTPUT);
  }
  if (DEBUG) {
    Serial << F("Free RAM: ") << FreeRam() << "\n";
  }
  pinMode(10, OUTPUT); // set the SS pin as an output (necessary!)
  digitalWrite(10, HIGH); // but turn off the W5100 chip!
  byte ipTest[] = { 192, 168, 0, 2 };
```

```
  setIPAddress(ipTest);
  // initialize the SD card
  if (!card.init(SPI_FULL_SPEED, 4)) {
    Serial << F("card␣failed\n");
    has_filesystem = false;
  }
  // initialize a FAT volume
  if (!volume.init(&card)) {
    Serial << F("vol.init␣failed!\n");
    has_filesystem = false;
  }
  if (!root.openRoot(&volume)) {
    Serial << F("openRoot␣failed");
    has_filesystem = false;
  }

  if (DEBUG) {
    Serial << F("Setting␣up␣the␣Ethernet␣card...\n");
  }
  Ethernet.begin(mac, ip);

  // Start the web server.
  if (DEBUG) {
    Serial << F("Web␣server␣starting...\n");
  }
  web.begin();
  if (DEBUG) {
    Serial << F("Free␣RAM:␣") << FreeRam() << "\n";
  }

}

void loop() {
  web.process();
}
```