VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIOELECTRONICS

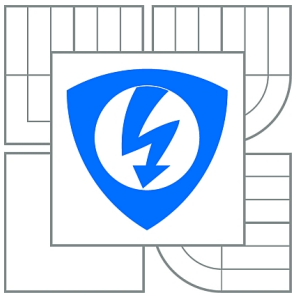
WEB SERVER S MIKROPROCESOREM ARM

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE Bc. JAN TESAŘ
AUTHOR

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF RADIO ELECTRONICS

WEB SERVER S MIKROPROCESOREM ARM

WEB SERVER WITH ARM MICROPROCESSOR

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

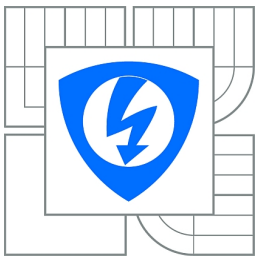
AUTOR PRÁCE — Bc. JAN TESAŘ

AUTHOR

VEDOUCÍ PRÁCE — prof. Ing. ALEŠ PROKEŠ, Ph.D.

SUPERVISOR

BRNO 2013



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav radioelektroniky

Diplomová práce

magisterský navazující studijní obor
Elektronika a sdělovací technika

Student: Bc. Jan Tesař ***ID:*** 115294
Ročník: 2 ***Akademický rok:*** 2012/2013

NÁZEV TÉMATU:

Web server s mikroprocesorem ARM

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s rozhraním Ethernet a s protokolem TCP/IP. Prostudujte hardwarovou koncepci vývojové desky Mini6410 od firmy Friendly ARM a seznamte se s dostupnými operačními systémy pro tuto desku (Android, Ubuntu, Windows CE6, atd), s podporou přístupu k periferiím mikroprocesoru a s metodami programování tohoto modulu. Zvolte aplikačně nejvýhodnější operační systém a vytvořte software pro vzdálený přístup pomocí Ethernetu. Navrhněte a realizujte implementaci HTML stránek do vývojové desky s možností monitorování vestavěných A/D převodníků, pamětí a vybraných sériových rozhraní. Doplňte aplikaci o možnost vzdáleného přístupu k rozhraní videokamery. Ověřte činnost modulu v režimu vzdáleného přístupu. Vytvořte manuál pro konfiguraci desky a vytváření aplikací pro vzdálený přístup.

DOPORUČENÁ LITERATURA:

[1] KÁLLAY, F., PENNIAK, P. Počítačové sítě a jejich aplikace. Praha: GRADA, 2003.

[2] Mini6410 Manual [online]. Guangzhou, China: Friendly ARM, [cit. 15. března 2011]. Dostupné na http://www.friendlyarm.net/downloads.

Termín zadání: 11.2.2013 ***Termín odevzdání:*** 24.5.2013

Vedoucí práce: prof. Ing. Aleš Prokeš, Ph.D.
Konzultanti diplomové práce:

prof. Dr. Ing. Zbyněk Raida
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Diplomová práce řeší návrh a implementaci webového serveru a dohledové webové stránky na vývojovém kitu FriendlyARM Mini6410. Server umožňuje pomocí http vzdáleně spravovat a monitorovat vybrané periferie. Software je instalován v OS GNU/Linux s pomocí balíkového systému PTXdist, jež je v práci popsán. Zařízení by mělo později sloužit ke vzdálené správě hlavice optického atmosférického spoje.

KLÍČOVÁ SLOVA

ARM, Mini6410, FriendlyARM, embedded Linux, embedded webový server, PTXdist, Apache, PHP, HTML, vzdálená správa

ABSTRACT

Diploma thesis is dealing with design and implementation of the web server and management website in the developing kit FriendlyARM Mini6410 with installed OS GNU/Linux. The embedded packaging system PTXdist is described in terms of Kernel configuration and selection of suitable applications. The device should be later used for the remote management of atmospheric optical link.

KEYWORDS

ARM, Mini6410, FriendlyARM, embedded Linux, embedded web server, PTXdist, Apache, PHP, HTML, remote management

TESAŘ, Jan *Web server s mikroprocesorem ARM*: diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav radioelektroniky, 2013. 53 s. Vedoucí práce byl prof. Ing. Aleš Prokeš, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Web server s mikroprocesorem ARM“ 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)

PODĚKOVÁNÍ

Výzkum realizovaný v rámci této diplomové práce byl finančně podpořen projektem
CZ.1.07/2.3.00/20.0007 **Wireless Communication Teams**
operačního programu **Vzdělávání pro konkurenceschopnost**.

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Finanční podpora byla poskytnuta Evropským sociálním fondem
a státním rozpočtem České republiky.

Rád bych poděkoval vedoucímu diplomové práce panu Prof. Ing. Aleši Prokešovi, Ph.D. za odborné vedení, rady a poskytnuté zařízení k práci.

Brno ....

(podpis autora)

OBSAH

SEZNAM OBRÁZKŮ

ÚVOD

V posledních letech se na poli vnořených (angl. embedded) systémů stále více využívá řešení, kdy v dané aplikaci funguje kompletní operační systém (dále OS), tak jak ho známe z PC. Je to dáno technologickým pokrokem, kdy zvětšující se výpočetní rychlost procesorů a jejich zmenšování co do velikosti umožňuje sestavit kompletní počítač na jedné DPS. V drtivé většině je OS nasazen na architektuře Advanced RISC Machines (ARM). Ta je dnes použita téměř ve všech zařízeních spotřební elektroniky typu mobilní telefony, síťové prvky, GPS navigace, v průmyslu, elektronice vozidel a řadě dalších aplikacích. Z hlediska software je možné na ARM nasadit některý z embedded nebo real-time OS tj. Windows CE, Windows 8, Symbian, Linux (Android, Debian, Ubuntu, Gentoo..) a BSD.

Cíl mé práce je vytvořit vhodnou konfiguraci embedded systému, jenž bude později sloužit ke vzdálenému ovládání a monitorování hlavice optického atmosférického spoje. Je tedy potřeba implementovat webový server, na kterém budou pomocí webového prohlížeče dostupné periferie (Universal Asynchronous Receiver/-Transmitter (UART), General Purpose InputOutput (GPIO), Analog-to-Digital Converter (ADC), Inter-IC (I2C), kamera). Realizace spočívá ve vhodném sestavení OS, jeho konfiguraci, následné instalaci potřebných nástrojů a naprogramování software rozhraní webová stránka - periferie. Uživateli (vzdálenému správci) pak stačí k práci se zařízením počítač připojený k internetu. Na přehledné webové stránce může provést operaci kterou právě potřebuje nebo jen zkontrolovat stav AD převodníku či graficky znázornit sledovanou veličinu.

Jako vhodný hardware byla určena vývojová deska Mini6410 firmy FriendlyARM, osazená System on a Chip (SoC) procesorem Samsung S3C6410 (takt až 667 MHz) s architekturou ARM11. Výrobce dodává s deskou také několik pro ni kompilovaných operačních systémů. Jedná se o Linux s prostředím Qtopia, Ubuntu Linux, Android a Windows CE.

Obrovskou výhodou je možnost používání operačního systému, který především zajistí komunikaci s rozhraním Ethernet, aniž by bylo třeba programovat ovladač jako u klasického jednočipového řešení. Pro OS GNU/Linux existuje řada již hotových nástrojů, které použiji v této práci [1]. Jelikož se jedná o open source (otevřené zdrojové kódy), není problém si je zkompilovat pro ARM procesor.

Jistá úskalí přináší správná konfigurace jádra systému tak, abychom získali přístup ke všem požadovaným periferiím. To lze částečně vyřešit pomocí balíčkového systému PTXdist, který se postará o aplikování ovladačů a jejich záplat (patch souborů), kompilaci Kernelu, kompilaci programů a na závěr vytvoření spustitelného obrazu OS. Jiná cesta může být použití linuxové distribuce např. Debian přímo v zařízení, která sice usnadní práci při instalaci nových programů, ale stejně za nás

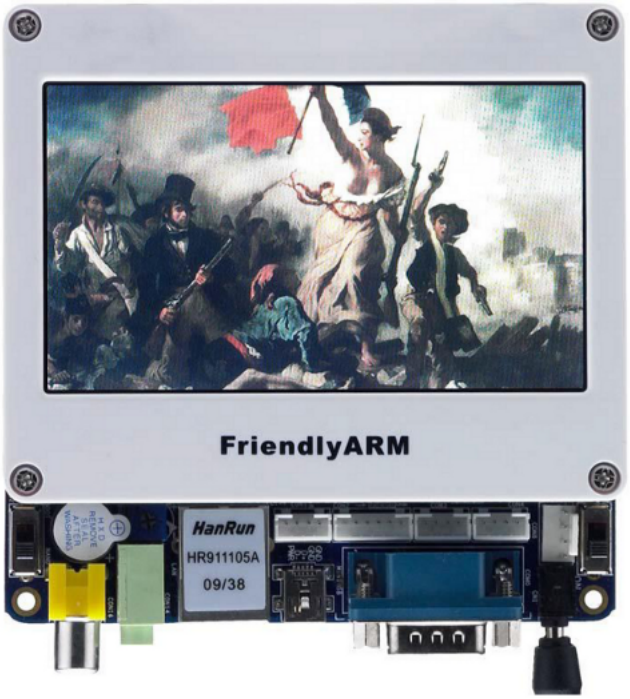
nevyřeší záplatování jádra a jeho správnou konfiguraci.

V práci je popsán jak webový server založený na Common Gateway Interface (CGI), tak i PHP: Hypertext Preprocessor (PHP). Jelikož ale CGI je již starší, nepružná technologie, tak vlastní realizace je postavena na Apache serveru s modulem PHP5. Spolu s databázovým modulem SQLite3 PHP zásadním způsobem ulehčuje programátorovi cestu k realizaci zadání.

1 TEORETICKÝ ROZBOR

1.1 Hardware Mini6410

Celkový design vývojového kitu i se zapnutým displejem je na následujícím obrázku. Jedná se o jednu desku osazenou elektronikou a vyvedenými periferiemi, nad kterou je na distančních sloupcích namontován dotykový displej s úhlopříčkou 4,3". Rozměry kitu jsou 110 x 110 mm. Napájení 5 V ze síťového adaptéru.



Obr. 1.1: Vývojový kit Mini6410 osazený displejem [14]

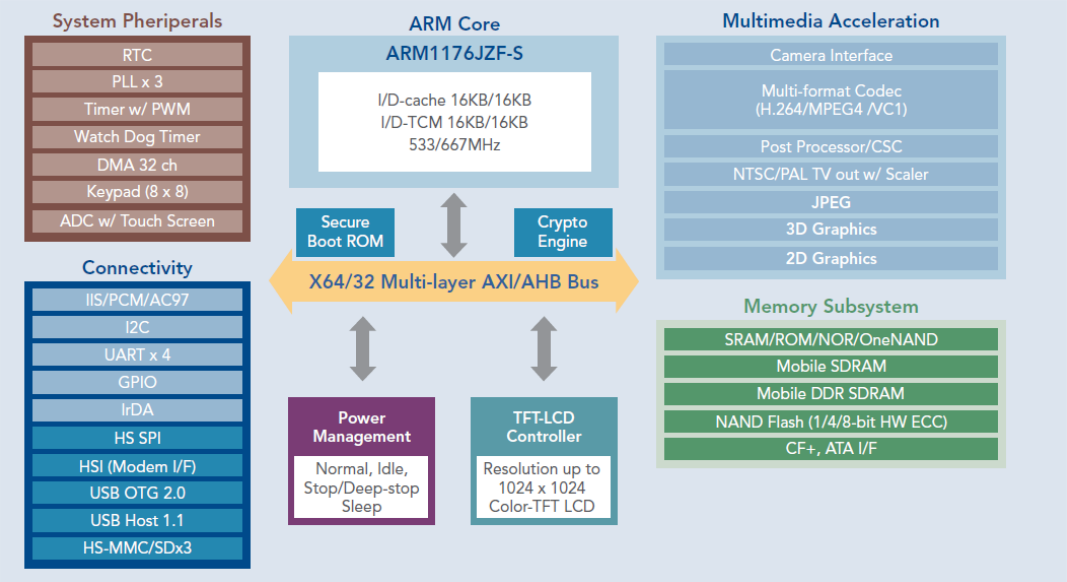
Co se týče spotřeby, výrobce uvádí 0,25 A při provozu bez displeje a 0,5 A s displejem. Z toho vyplývající spotřeba maximálně asi 3 W je velmi příznivé číslo i pro zařízení, které by bylo napájeno třeba jen solárními články.

1.1.1 Procesor Samsung S3C6410

Jádrem Mini6410 je Samsung S3C6410. Je vyroben technologií 65 nm, což jej činí energeticky velmi šetrným - výhoda při autonomním napájení. Taktovací kmitočet 533 až 667 MHz zajišťuje vysoký výkon, hardware 3D akcelerace umožňuje zpracování 3D obrazu - využitelné např. v GPS přijímačích s 3D mapami. Operační paměť tvoří 128 MB DDR RAM připájená na DPS v podobě Surface-Mount Device

(SMD). OS a data lze nahrát do SMD NAND Flash paměti o velikosti 1 GB nebo na připojenou SD kartu.

Procesor se skládá z částí uvedených na obrázku 1.2. Schéma také zobrazuje, které typy multimédií umí zpracovat.



Obr. 1.2: Blokové schema procesoru Samsung S3C6410 [12]

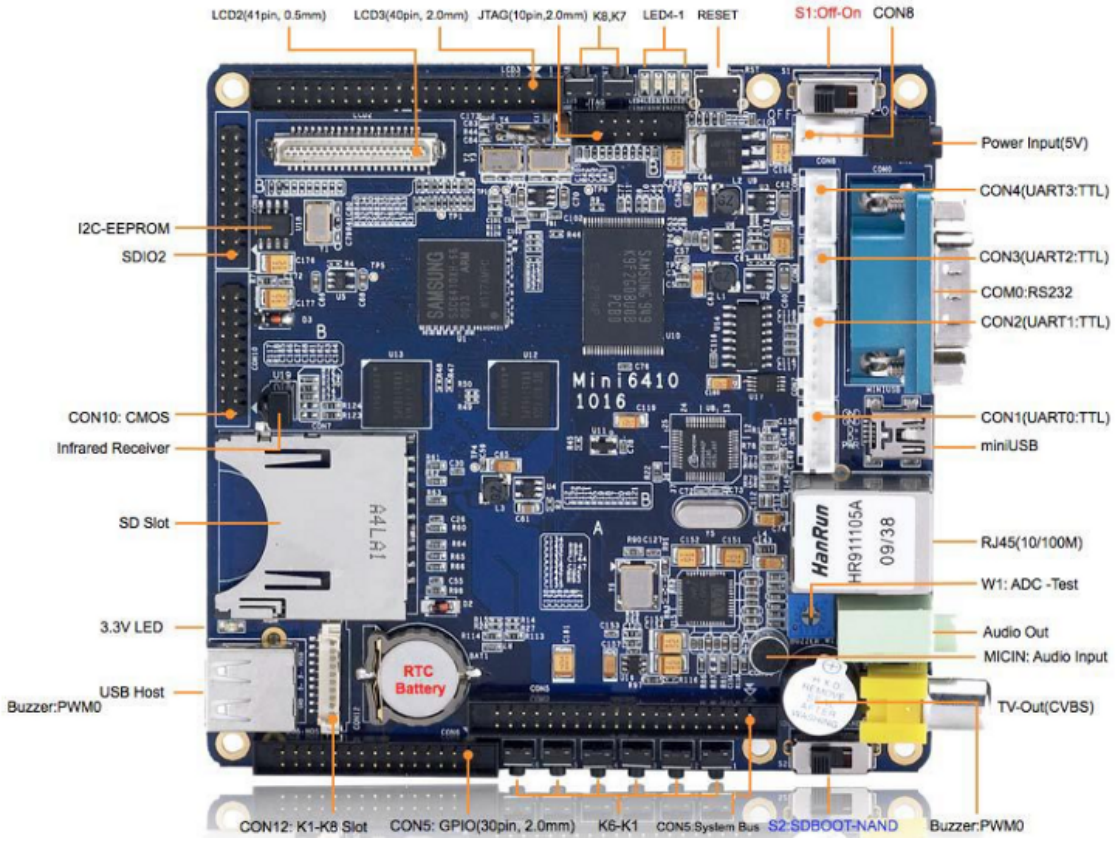
1.1.2 Periferie

Deska disponuje řadou periferií. Jedná se v podstatě o veškerá možná komunikační a datová rozhraní používaná v dnešní době. Samozřejmě je zde možnost také zpracovávat analogový signál na AD převodnících a popřípadě i zvuk na mikrofonním vstupu. Obecné vstupně/výstupní porty se dají rovněž ovládat. O vše se stará nainstalovaný OS obsahující potřebné ovladače. Bez nich je třeba přistoupit k nízkoúrovňovému programování se znalostí významu jednotlivých 32-bitových registrů procesoru.

Výčet jednotlivých rozhraní a prvků:

- 4,3"LCD s 4-vodičovým dotykovým displejem
- 100 MBit Ethernet (síťový čip DM9000, RJ-45 konektor)
- DB9 RS232 5-vodičový seriový port + 4 sériové UART linky
- mini USB slave v. 2.0
- USB host 1.1 port
- 3,5 mm stereo výstup a mikrofonní vstup
- TV výstup (kompozitní video)

- čtečka SD(SDHC) karet
- Ir přijímač
- I2C sběrnice s EEPROM pamětí 256 B
- 8 x 12 bit AD převodník
- 30 pinů GPIO
- 10-pinové JTAG rozhraní
- 20-pinové SDIO rozhraní pro WiFi, (obsahuje I2C a SPI)
- 20-pinové rozhraní pro CMOS kameru
- 40-pinová systémová sběrnice
- 8 pinů přerušení s tlačítky, 4 x LED indikace
- PWM bzučák
- 4 x časovač
- baterie pro obvod reálného času

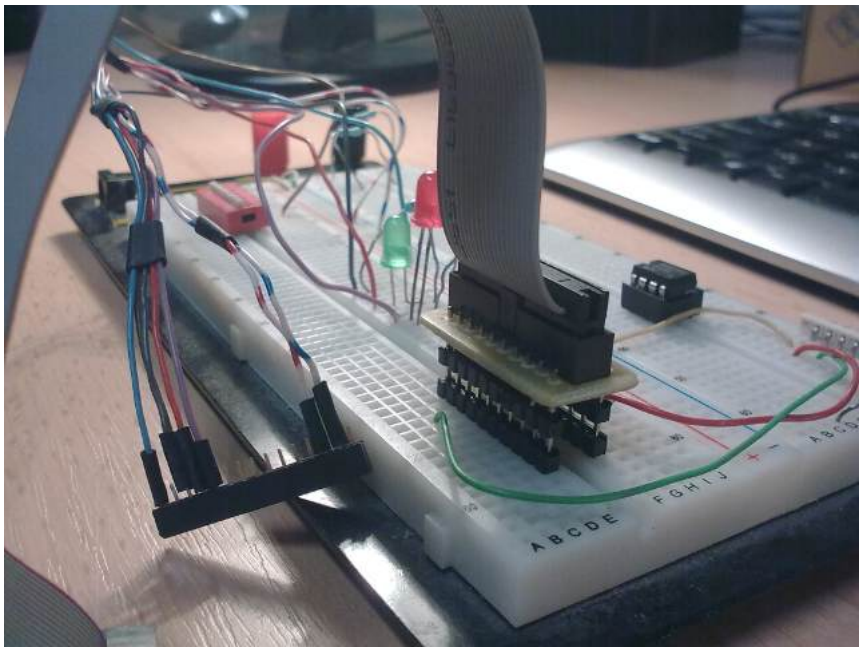


Obr. 1.3: Umístění jednotlivých konektorů a periferií Mini6410 [14]

V práci je cílem zprovoznit vzdálené ovládání především u nejčastěji používaných rozhraní a pamětí. Těmi jsou AD převodníky, UART, GPIO, I2C, EEPROM, čtečka SD karet a CMOS kamera. Vývojový kit disponuje i displejem, který je pro aplikaci

optického spoje také užitečný, v této práci však není jeho programování zahrnuto, jelikož se práce zabývá pouze vzdáleným přístupem.

Menší komplikací je rozteč pinů 2 mm použitá pro všechny konektory plochých kabelů po obvodu desky (Obr.1.3). Lze sice koupit sadu kabelů FriendlyARM, ale ta je pro desku Mini2440, která má trochu odlišné konektory. Na českém trhu prakticky nejde sehnat ploché kabely s konektorovou roztečí 2 mm které by na Mini6410 pasovaly. Proto jsem si vyrobil alespoň jednu dutinkovou lištu s roztečí 2 mm (ta se koupit dá) pro výstupy z CON6 tj. kanály AD převodníku a 16 GPIO pinů. Pro konektor CON9 (SPI, I2C) se dal použít jeden z kabelů pro Mini2440, pro jehož druhý konec jsem vyrobil redukci na 2,54 mm, kterou mohu zasunout do nepájivého pole a testovat signály.



Obr. 1.4: Vyrobená redukce pro 2 mm konektor

1.2 Výběr OS pro Mini6410

FriendlyARM dodává s kitem i software, zkompilovaný přímo pro tuto platformu. Jedná se o několik druhů operačních systémů - Windows CE, Android, Ubuntu Linux a Linux s prostředím Qtopia 2.2.

K jejich instalaci je potřeba mít alespoň 2 GB SD kartu. Na ní se nahraje bootloader. Zde jsou dvě varianty - open source U-boot a nebo Superboot vytvořený firmou FriendlyARM. Já použil Superboot. Jak jej nahrát na kartu je popsáno

v manuálu od výrobce. Veškeré potřebné soubory - obrazy OS, manuály, firmware a ukázky jsou k dispozici on-line a je vhodné si stáhnout aktuální verzi z FTP výrobce [15].

Na SD kartu s bootloaderem lze nahrát všechny dodané obrazy OS zároveň. Správným nastavením konfiguračního souboru `FriendlyARM.ini` se pak zvolí, zda se má daný systém nainstalovat do NAND Flash paměti a nebo bootovat rovnou z karty. Pokusil jsem se spustit postupně všechny čtyři zmiňované OS.

Ubuntu Linux a Android po instalaci nebyly schopny řádně naběhnout. Android sice bootoval zřejmě úspěšně (podle výpisů ze sériové linky), dalo se v něm i vykonávat příkazy přes sériový terminál, nicméně zobrazení na LCD se nezdařilo. Jejich použití jsem tedy pro další práci zavrhl.

Windows CE se po instalaci zdárně nabootoval za několik málo sekund. Problémem však bylo čínské písmo v celém systému. Výrobce pro něj sice dodává ovladače ke všem periferiím, ale raději bych se pro provoz webového serveru přidržel systému GNU/Linux.

Posledním pokusem byla instalace Linuxu s prostředím Qtopia 2.2. Ta proběhla úspěšně a po kalibraci LCD se dá pomocí dotykového displeje či myši připojené k USB se systémem normálně pracovat. Také se lze přepnout do anglického jazyka. Ve výchozím stavu je mimo jiné již spuštěn webový server s ukázkovou stránkou. Je možné se na něj připojit z PC zadáním IP zařízení.

1.2.1 Zkušenost s OS Linux od výrobce FriendlyARM

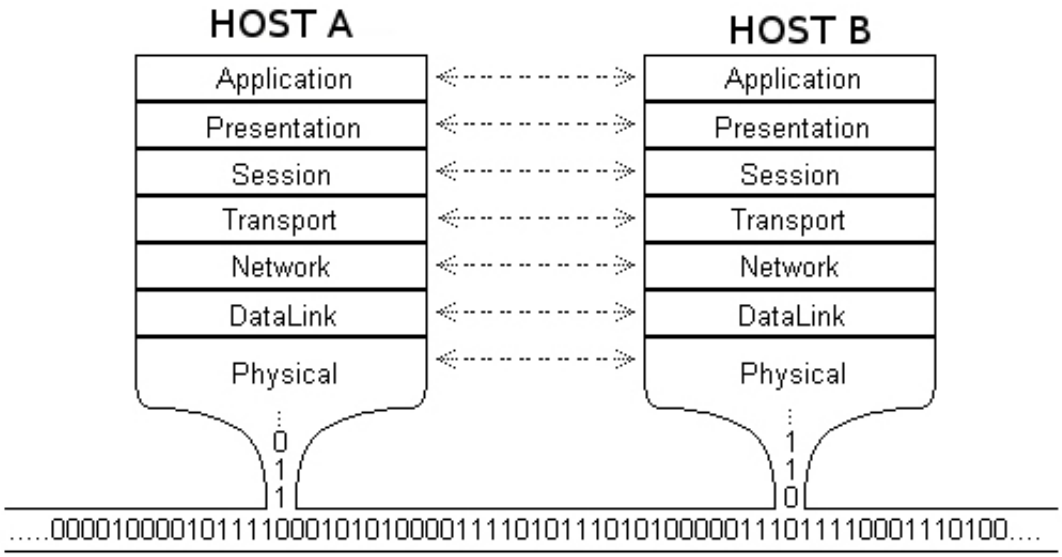
Dodávaný systém se na první pohled zdá být ihned použitelný. Ovšem při detailnějším prozkoumání jsem nejprve narazil na nemožnost ovládání GPIO. Ten jsem odstranil překompilováním jádra Linuxu 2.6.38 dle [4] s vhodně nastaveným konfiguračním souborem pro podporu GPIO. [17]

Jedním z požadavků je možnost čtení z co nejvíce AD převodníků. V dodané verzi jádra jsem však nebyl schopen ani po mnoha různých nastaveních zprovoznit čtení z převodníků. Funkční bylo pouze čtení kanálu ADC0 s připojeným testovacím trimrem.

Nakonec jsem se rozhodl pro zcela jiný přístup k software pro Mini6410. Průzkumem zkušeností ostatních uživatelů této platformy s různými OS jsem dospěl až k balíčkovému systému PTXdist vyvíjeného společností Pengutronix. Bohužel na podporu driverů a záplat od výrobce FriendlyARM se nedá spoléhat. Většinu otázek vznikajících při práci se zařízením si zkušenější uživatelé řeší sami v rámci internetového fóra [16].

1.3 HTTP v TCP/IP modelu

Jestliže chceme zjistit, jakým způsobem se data šíří od klienta k serveru a opačně, je třeba ukázat, co se s nimi děje od jejich vzniku na vysílací straně až po dosažení cílové destinace. Podle referenčního ISO Open Systems Interconnection (OSI) modelu existuje 7 vrstev, kterými data prochází.



Obr. 1.5: Vrstvy OSI modelu

Toto je však jeden z možných pohledů na přenos dat internetem. Zjednodušený a často uváděný model sítě je TCP/IP model. Ten slučuje vrchní tři vrstvy v jednu zvanou aplikační.

Webový server se z hlediska tohoto modelu nachází nad aplikační vrstvou. Jako prostředek své komunikace s klientskými stanicemi používá Hypertext Transfer Protocol (HTTP). Definice http je dána dvěma subjekty - Internet Engineering Task Force (IETF) a World Wide Web Consortium (W3C). Nachází se převážně v doporučení RFC 2616 organizace IETF [9], který definuje verzi HTTP/1.1.

Následně se v transportní vrstvě využije Transmission Control Protocol (TCP) protokol (RFC 1122) pro zajištění spojení serveru s klientem. Tento protokol na rozdíl od User Datagram Protocol (UDP) zajistí spolehlivé doručení zprávy např. od serveru ke klientovi, a to i za cenu několikanásobného znovu-odeslání paketu v případě jeho ztráty či vypršení Time To Live (TTL), jenž určuje maximální možný počet různých podsítí, kterými může paket projít.

Síťová vrstva se stará o správné směrování TCP paketu, nyní již opatřeného IP hlavičkou, v rámci Wide Area Network (WAN) a internetu. Na základě cílové adresy

paketu pak router v každém bodě sítě (cesty) rozhodne, na jaké své rozhraní má paket nasměrovat.

V rámci linkové vrstvy se paket pohybuje pomocí fyzických adres zařízení. V dané podsíti se tedy již nesměruje na třetí (síťové) vrstvě, ale pouze přepíná na druhé (linkové) pomocí přepínačů (switch). Tyto mají ve své přepínací tabulce záznam, jaké zařízení s jakou fyzickou a IP adresou se nachází na konkrétním portu.

Fyzicky se data mohou přenášet elektrickým, elektromagnetickým anebo optickým signálem. Deska Mini6410 je osazena 100MBit Ethernet rozhraním. Jedná se tedy o přenos elektrického signálu dvěma páry kroucené dvojlinky (UTP) s linkovým kódováním typu Manchester. Rámec (frame) v takovémto standardu je ukázán na Obr. 1.6.

Obr. 1.6: Http paket zapouzdřený ve všech TCP/IP vrstvách [13]

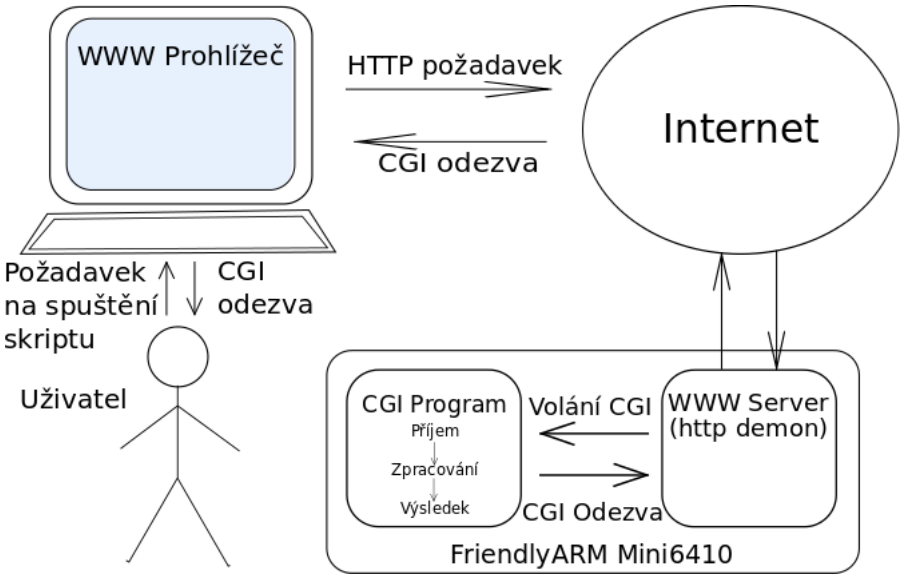
1.4 Protokol CGI

CGI je protokol, jenž slouží k propojeni externích aplikací s webovým serverem. V praxi tedy klient ze svého prohlížeče zavolá namísto html stránky skript a ten je spuštěn jako další aplikace. V rámci skriptu můžeme vykonat v podstatě cokoliv na co má uživatel pod kterým je spuštěn server práva [5].

Jako odlehčený webový server zde poslouží BOA web server, Lighttpd, thttpd nebo mnohé další. Díky otevřenému kódu jsou dostupné i pro tuto embedded aplikaci. Zmiňované servery jsou naprogramovány s důrazem na minimální spotřebu paměti a výpočetní nenáročnost, zároveň však obslouží klienta ve velmi rychlém čase. Předpokladem je, že se na server nevalí stovky dotazů zároveň.

Skripty mohou být psány buď v C/C++ nebo v některém skriptovacím jazyku - Bash, PERL atd. Je-li skript napsán v C, je třeba jej napřed zkompilovat pro danou platformu. Spouští se pak v podstatě jako normální program. Pokud použijeme třeba PERL není potřeba nic kompilovat a vývoj je rychlejší.

Princip CGI je takový že, uživatel vyplní a potvrdí na webové stránce nějaký formulář. Jeho obsah se odešle serveru s příkazem jaký skript se má zavolat. Zavolanému skriptu je předán řetězec ve formě párů name1-value1;name2-value2 atd. Nyní si skript (program) musí rozložit řetězec a správně vyhodnotit, které proměnné



Obr. 1.7: Funkce CGI protokolu

patří jaká hodnota. Výstupem CGI skriptu je html kód, do nějž se vloží libovolná data zpracovaná programem.

Protokol je výhodné použít například chceme-li po systému, aby přečetl stavy AD převodníků. Ty pak zapsal do souboru nebo je poslal zpět uživateli do prohlížeče.

Více o CGI je možno nalézt ve zkratce na elektronickém zdroji [11] nebo přímo v definici [10] na webu IETF.

1.5 Kompilování vlastních programů

Pro kompilování vlastních programů používám nástroje přímo od FriendlyARM dostupné z [15]. Jedná se o Toolchain 4.5.1. Ten je třeba rozbalit a nakopírovat například do `/opt/FriendlyARM/toolschain/4.5.1/` ve vývojářově PC. Předpokládá se, že programátor používá na svém počítači také OS GNU/Linux. Složka `4.5.1` včetně všech vnořených souborů musí mít práva ke spuštění pro přihlášeného uživatele! Dále je potřeba si do souboru `.bashrc` v domovském adresáři přidat na konec řádek:

```
export PATH=$PATH:/opt/FriendlyARM/toolschain/4.5.1/bin
```

Nyní lze kdykoliv zavolat cross-compilator a zkompilovat si svůj vlastní program napsaný v C. Namísto výstupu `priklad` lze psát `priklad.cgi` a vytvořit skript pro web server.

```
arm-linux-gcc -o priklad priklad.c
```

1.6 Apache HTTP Server a jeho moduly

V rámci práce jsem odzkoušel různé menší servery pro CGI skripty. Při skriptování jsem však zjistil, že nutnost kompilovat každou úpravu zdrojového souboru a těžkopádnost programování html kódu v C mi pro práci nevyhovuje. Podařilo se mi tedy zkompilovat pro ARM Apache2 Server a k němu i modul PHP5.

1.6.1 Modul PHP5

Interpret PHP je spuštěn jako modul `mod_php5` serveru Apache2. Konkrétně se jedná o verzi PHP 5.3.3. V kořenové složce serveru `/var/www` stačí vytvořit jakoukoliv cestou (SFTP nebo lokálně) textový soubor s příponou `.php` a server se postará o jeho předání PHP interpretu. Skripty se nemusí kompilovat a proto je vývoj velmi rychlý, s okamžitou odezvou na webové stránce.

Hlavní přínos je pro mě zpracování formulářů hojně použitých na výsledné stránce. Pomocí PHP také řeším skripty pro obnovu dat z databáze po restartu systému a spouštím samostatné procesy PHP interpretu na pozadí pro zpracování úkolů dlouhodobého monitorování apod.

Od verze 5 PHP podporuje objekty. Této vlastnosti lze často využít a usnadnit si práci. Většinu úkolů lze realizovat procedurálně i objektově. Mě se často z hlediska přehlednosti a rychlosti vývoje osvědčuje objektový přístup.

Je nutno podotknout, že podpora pro vývojáře je obrovská. Stačí nahlédnout do manuálových stránek na `www.php.net` a sáhnout po funkci, která často vyřeší daný problém bez nutnosti ji znova programovat.

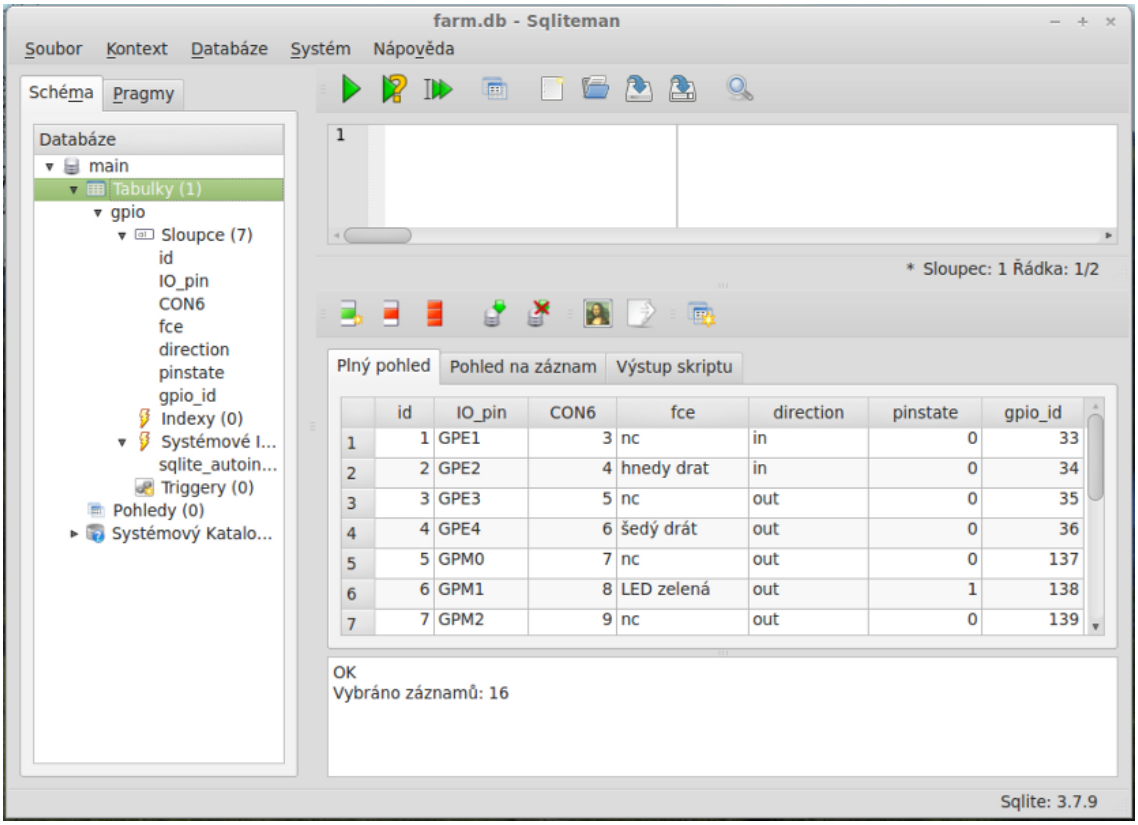
1.6.2 Databáze SQLite3

S potřebou zaznamenávat stav nastavení zařízení či uživatelem definované popisy veličin vyvstala otázka, kam tyto data zapisovat. Zdánlivě pohodlné a okamžité řešení přináší jednoduchý textový soubor, kde si nějakým vlastním systémem můžeme data ukládat a později číst a měnit. Nicméně od tohoto řešení jsem ihned na začátku upustil, jelikož s narůstajícím počtem záznamů a jejich parametrů se nedá něco takového udržovat.

SQLite3 je odlehčená verze SQL databázového systému. Umožňuje vytvářet tabulky záznamů tak jako podobné složitější nástroje typu MySQL. Na rozdíl od MySQL však nepotřebuje žádný databázový server. To je pro embedded aplikaci důležité, jelikož se vždy snažíme šetřit výpočetní výkon a paměť. SQLite3 pouze potřebuje být podporováno v PHP modulu, který SQL dotazy zpracovává. Vlastní databáze se ukládá do jednoho souboru, který může obsahovat více tabulek. Snadné zálohování, přenositelnost a rychlá spustitelnost dělají z SQLite skvělý nástroj pro

vývoj webových aplikací malé a střední velikosti, což je naprostá většina všech webových projektů.

Níže je ukázka vytvořené tabulky v databázi s názvem `farm.db`. Používám program Sqliteman, což je jednoduchý správce SQLite databází napsaný v prostředí Qt. Je dostupný jako open source program s licencí General Public License (GPL)v2 pro Linux i Windows.



Obr. 1.8: Program Sqliteman pro správu SQLite databáze

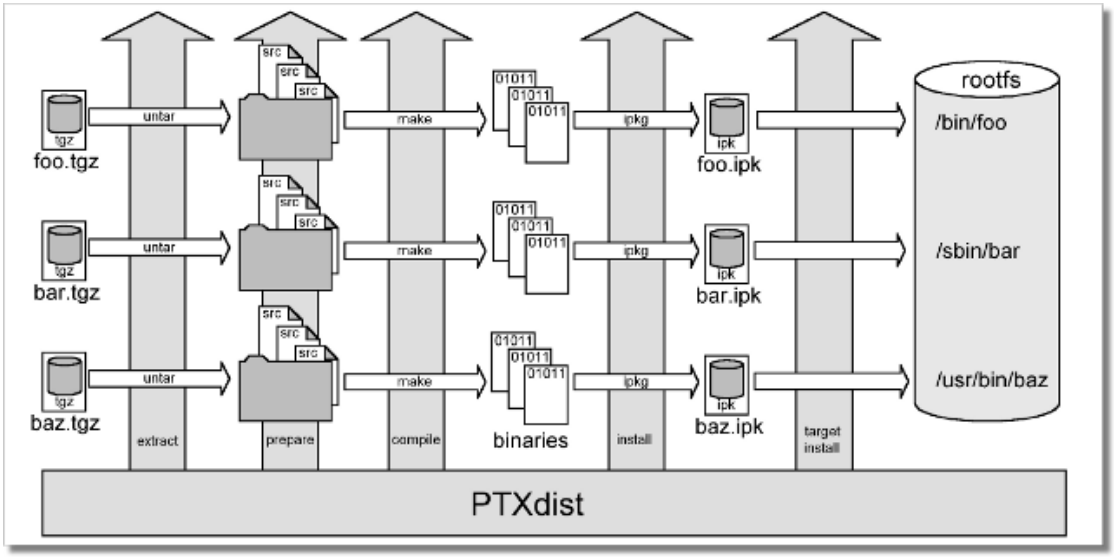
2 KONFIGURACE A OVLÁDÁNÍ MINI6410

V rámci této kapitoly je popsána celková skladba software, jenž je momentálně nasazen na vývojové desce. Pro instalování nových programů a jádra systému se používá PTXdist. Výhodou tohoto systému je, že je neustále udržován a jsou v něm aplikovány záplaty Kernelu nutné pro správný chod veškerého hardware. Dále je popsáno jak přistupovat z prostředí OS Linux k periferiím.

2.1 Balíčkový systém PTXdist

V tomto bodě předesílám, že veškeré programy, které používám pro vývoj, spouštím v operačním systému Linux Mint 13 64 bit. To je distribuce odvozená z Ubuntu 12.10. Následující návody jsou tedy odzkoušeny v tomto OS. Windows pro tuto práci silně nedoporučuji, jelikož se při vývoji převážně využívá linuxových nástrojů a pohodlí jež Linux nabízí.

PTXdist je vyvinut speciálně pro embedded systémy. Není to systém, který by instaloval již zkompilované balíčky z archivu distribuce jako to známe z Debianu či Fedory. `ptxdist` je program, který se spouští na vývojářově PC. Jeho úkolem je stáhnout, zkompilovat a nainstalovat právě a jenom ty open source programy, jež hodláme ve svém zařízení použít. Mimo to také kompiluje Kernel, pro který aplikuje záplaty (patche). Systém si sám rozhodne jaké programy a části jádra stáhnout na základě nastavení, jež provede vývojář.



Obr. 2.1: Princip sestavení OS pomocí PTXdist [8]

Jako cross-compilator je nejvhodnější použít OSELAS.Toolchain, který se nechá stáhnout jako zdrojový kód, který si zkompilujeme pro své PC. Pomocí něj pak PTXdist kompiluje zdrojové soubory pro cílovou platformu (pro nás ARMv6).

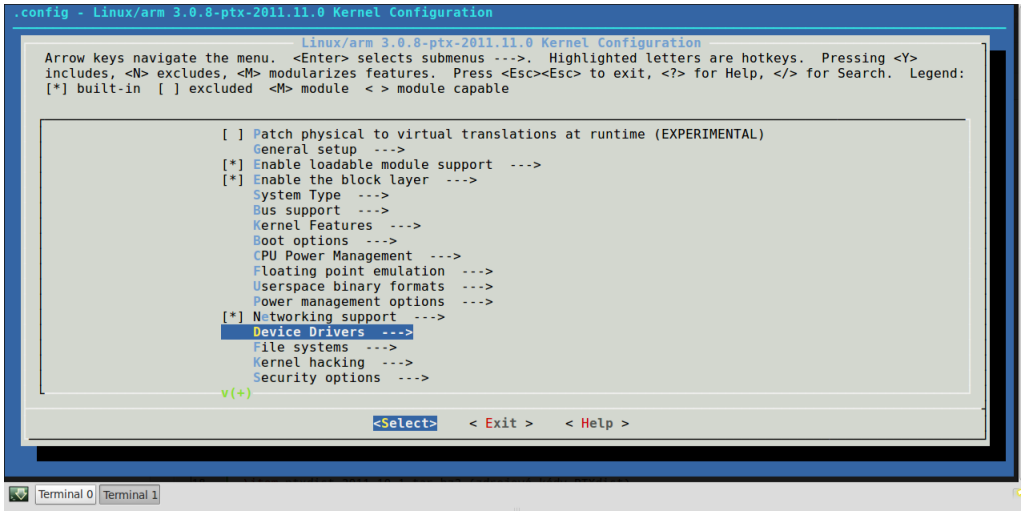
Dále je vhodné si stáhnout tzv. BSP (Board Support Package). Obsahem balíčku je základní konfigurace jádra Linuxu a programů pro Mini6410. Vše je popsáno v návodu výrobce software, jehož je třeba se detailně držet [8].

Výčet souborů, které je třeba stáhnout z `http://www.oselas.com` pro použití PTXdist:

- ptxdist-2011.10.1.tar.bz2 (zdrojové kódy PTXdist)
- OSELAS.BSP-Pengutronix-Mini6410-2011.11.0.tar.gz (podpůrný balíček pro Mini6410)
- OSELAS.Toolchain-2011.03.1.tar.bz2 (cross-compilator od OSELAS)

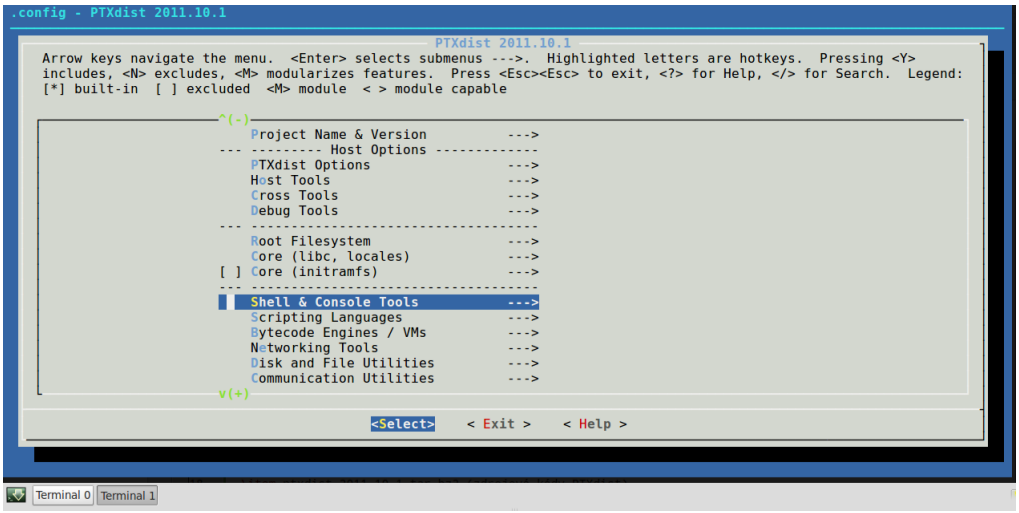
2.1.1 Používání PTXdist

Všechny příkazy provádíme z adresáře projektu, na němž pracujeme tedy `OSELAS.BSP-Pengutronix-Mini6410-2011.11.0`. Většina nastavení jsou již správně předdefinována.
Pro konfiguraci Kernelu je třeba zapnout nástroj `ptxdist kernelconfig`. Pokud si přejeme přidat do jádra další vlastnost či ovladač, je třeba jej zde vyhledat a označit.



Obr. 2.2: Dialogové okno pro nastavení Kernelu

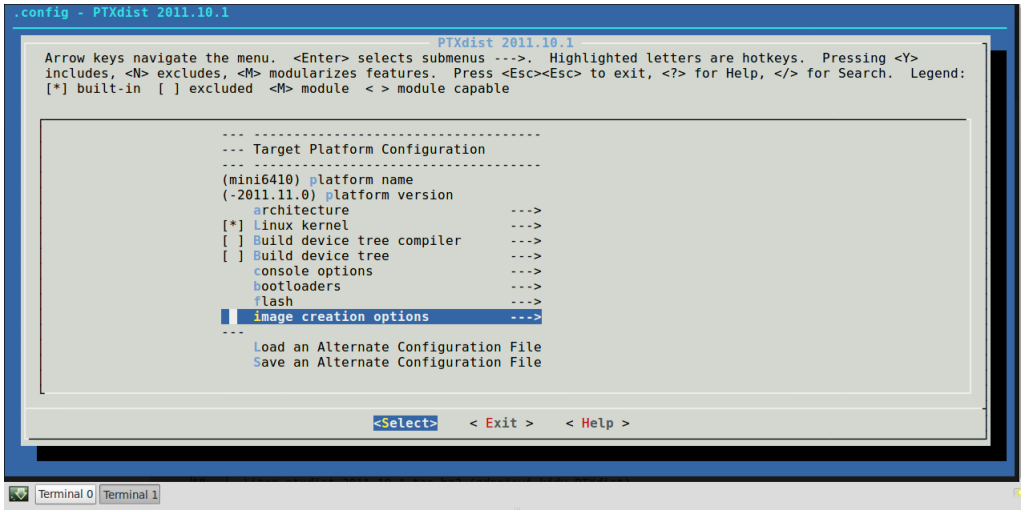
Výběr programů do vytvářeného systému příkazem `ptxdist menuconfig` je naznačen na dalším obrázku.



Obr. 2.3: Dialogové okno pro výběr programů k instalaci

Zde si zvolíme všechny programy jež budou potřeba pro hladký chod a správu serveru - sshd, sftp, dhcpd, ntpd, apache2, php5, sqlite3. Další základní programy jsou již předvoleny, mezi nejdůležitější patří Busybox, což je celá kolekce utilit pro práci v OS GNU/Linux. Díky jmenovaným programům a službám se můžeme k desce přihlásit(ssh), nechat si poslat IP od routeru (dhcp), nahrávat data (sftp), synchronizovat čas z internetu (Network Time Protocol (NTP)) a hlavně je zde důležitý http server Apache. Dle potřeby je možné přidat i další programy z desítek nabízených položek.

Nyní spustíme `ptxdist platformconfig`:



Obr. 2.4: Dialogové okno pro nastavení platformy

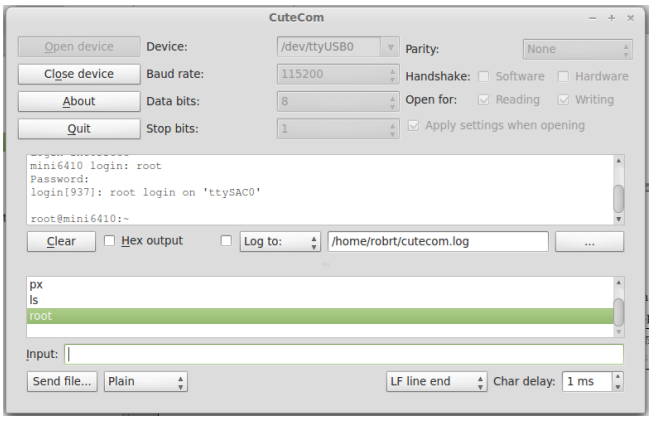
Nastavíme jaký konfigurační soubor se vybere pro kompilaci Kernelu. Dále jaký souborový systém se použije (ext2, jffs2, UBI) a jaký bootloader (zavaděč) se má nainstalovat. Vybereme Kernel 3.0, ext3 filesystem a U-boot zavaděč. Výběr souborového systému ext3 je vhodný pouze pro OS na SD kartě, která se již sama stará o to, aby se nepřepisovaly stále stejná místa paměti a nedošlo k rychlému opotřebení. V případě, že bychom chtěli systém nahrát na Flash paměť, tak je třeba zvolit speciální filesystem jffs2, který je vytvořen pro tento druh média.

Start sestavování systému se provádí příkazem `ptxdist go`. Systém stáhne všechny potřebné zdrojové soubory a provede jejich kompilace dle nastavení výše. Tento proces může trvat řádově desítky minut až několik hodin podle toho kolik programů se musí zkompilovat a hlavně zda kompilujeme i Kernel. Po skončení tohoto procesu se příkazem `ptxdist images` sestaví tři důležité soubory - `u-boot.bin`, `linuximage` a `root.tgz`. Ty se nahrají na SD kartu (nesmí být SDHC! - chyba PTXdist). Postup jak toho docílit je opět v literatuře [8]. Nyní je sestaven OS a stačí pouze nabootovat připravenou kartu. Přepínačem na desce vybereme zavádění z SD karty. Druhou variantu OS v NAND Flash paměti nevyužívám, vzhledem k její menší kapacitě a rychlejšímu opotřebování.

2.2 Nastavení operačního systému

Abychom mohli pohodlně používat Mini6410 je třeba správně spustit a nastavit služby, které jsou v OS.

V první fázi při prvním zapnutí je potřeba se propojit seriovou linkou s rozhraním COM0 na desce. K tomu potřebujeme null modem a popřípadě převodník USB na RS232, pokud nemáme seriové rozhraní. Otevřeme např. program Cutecom, ponecháme výchozí nastavení a otevřeme spojení.



Obr. 2.5: Přihlášení k Mini6410 pomocí seriového rozhraní

Ke konzoli se přihlásíme jako root, heslo zatím není nastaveno. `Inetd`, který se stará o nastavení Ethernetu již běží, jen nemá přidělenu adresu. Dvěma příkazy přidělíme zařízení adresu z rozsahu svého routeru:

```
ifconfig eth0 192.168.1.25 netmask 255.255.255.0
route add default gw 192.168.1.254 eth0
```

Předpokládejme, že PC pro vývoj a deska jsou zapojené v routeru se stejným rozsahem IP adres. Využijeme Ethernetu a komunikaci seriovou konzolí zaměníme za telnet:

```
robrt@robrtnbk:~$ telnet 192.168.1.25
Trying 192.168.1.25...
Connected to 192.168.1.25.
Escape character is '^]'.

root@mini6410:~
```

Získali jsme plnohodnotný přístup k OS. Práce v telnetu je mnohem příjemnější než přes seriové rozhraní. Nyní je třeba provést několik nastavení. Nejprve vytvoříme heslo pro uživatele root příkazem `passwd`, aby bylo možné používat ssh a sftp. Tyto dvě služby se spustí po startu sami.

2.2.1 Nastavení pořadí spouštění služeb

Jakmile se zavede jádro systému začnou se spouštět služby. Jejich konfigurační soubory, spouštěcí soubory a pořadí spouštění jsou definovány v adresáři `/etc`. V `/etc/rc.d` se určuje pořadí spouštění daných služeb. Jednotlivé položky jsou odkazy na skripty v adresáři `/etc/init.d`. Číslo na začátku názvu určuje pořadí spouštění.

```
root@mini6410:/etc/rc.d ls
S00udev
S01rc-once
S12dbus
S12hwclock
S13gpio
S14networking
S15inetd
S16openssh
S20ntpd
S91apache2
S95nfsd
```

```
S97modules
```

Listing 2.1: Výpis /etc/rc.d

Toto je výsledné pořadí, jež jsem vytvořil, aby se mnou požadované služby spouštěly logicky po sobě. Nejprve tedy systémové záležitosti, pak synchronizace času s Real-Time Clock (RTC) (`S12hwclock`), export a obnovení nastavení GPIO ze zálohy (`S13gpio`), nastavení sítě S14-15, spuštění ssh démona pro vzdálený přístup do konzole, synchronizace času s ntp serverem `S20ntpd`, spuštění http serveru `S91apache2`.

2.2.2 Export a obnova nastavení GPIO

Shell skript `/etc/init.d/gpio` se stará po při startu OS o export GPIO pinů konektoru CON6. Ve výchozím stavu je totiž v systému nemůžeme ovládat. Proto jsem vytvořil rutinu, která při každém startu potřebné piny zviditelní. Po exportu je nutné také změnit výchozí přístupová práva. Skupině *ostatní* je na řádku 15 přiděleno právo číst zapisovat i spouštět. Je to z toho důvodu, aby uživatel *www*, pod kterým je spuštěn Apache server, mohl se soubory pracovat.

```
1  #!/bin/sh
2
3  # GPIOs setting for CON6 on FriendlyARM6410 by Jan Tesar
4  #
5  GPIOEXP=/sys/class/gpio/export
6  GPIOUNEXP=/sys/class/gpio/unexport
7
8  case $1 in
9
10   start)
11     echo "exporting selected GPIOs..."
12     echo 33 > $GPIOEXP
13     # zde jsou řádky pro dalších 14 pinů, nezobrazeno kvůli rozsahu
14     echo 200 > $GPIOEXP
15     chmod -R o+rwx /sys/class/gpio/gpio[1-9]*/
16     php5 -f /var/www/config.php
17     chmod -R o+rwx /dev/shm
18     exit 0
19     ;;
20   stop)
21     echo "unexporting selected GPIOs..."
22     echo 33 > $GPIOUNEXP
23     # zde jsou řádky pro dalších 14 pinů, nezobrazeno kvůli rozsahu
24     echo 200 > $GPIOUNEXP
25     exit 0
26     ;;
27   *)
```

```
28      echo "uasge: $0 {start|stop}"
29      exit 1
30      ;;
31  esac
32  exit 0
```

Listing 2.2: Výpis skriptu /etc/init.d/gpio

Řádek 16 spouští PHP skript pro nastavení portů dle zálohované databáze tj. nastavení směru (In/Out) a v případě výstupních pinů i stavu podle poslední známé konfigurace. Data pro tato nastavení jsou načtena z SQLite3 databáze, která je stále aktuální na SD kartě a uchovává nastavení stavů provedené uživatelem na webové stránce i po výpadku napájení.

Řádek 17 opět přidá práva pro skupinu *ostatní*, aby mohla pracovat s `/dev/shm`. To je virtuální paměť, která se fyzicky nachází v RAM a umožňuje rychlý zápis dat bez opotřebení fyzického média. Toho se dále využívá v logování hodnot z ADC, kdy se zde shromažďují rychle měřené záznamy.

```
define("STATE_PATH", "/sys/class/gpio/gpio");

class FARMDB extends SQLite3{ //podtrida tridi SQLite3 ktera hned po
    vzniku otevre farm.db databazi
     function __construct(){
  $this->open('/var/www/farm.db');
     }
}
$db = new FARMDB(); // instance podtridy FARMDB
$sql = "SELECT gpio_id, direction, pinstate FROM gpio";
$result = $db->query($sql);

while($row = $result->fetchArray(SQLITE3_ASSOC)){ // cte jednotlive
    radky tabulky gpio
     if(!isset($row['gpio_id'])) continue;
     $cesta = STATE_PATH."{$row['gpio_id']}/direction"; //vytvori celou
         cestu k nast. smeru
     $fp = fopen($cesta,"r+");
     fwrite($fp, $row['direction'], 4); //nastavi smer

     $cesta = STATE_PATH."$row[gpio_id]"."/value"; //vytvori celou
        cestu k nast. hodnoty
     $fp = fopen($cesta,"r+");
     fwrite($fp, $row['pinstate'], 2); //nastavi stav
}
```

Listing 2.3: Obnova nastavení GPIO skriptem /var/www/config.php

2.2.3 Synchronizace času

V zařízení se nachází obvod reálných hodin. Ten má sloužit jako zdroj časové informace pro OS. Datum a čas je potřeba znát pro fungování webových nástrojů, jež jsem naprogramoval (logování v daném intervalu). Již po dvou letech po zakoupení desky jsem se přesvědčil, že na RTC není spolehnutí, z důvodu rychlého vybití baterie, která jej napájí. Čas potom neukazoval přesně, hodnota se i po správném nastavení za pár hodin lišila o několik desítek minut. Výměna baterie problém řeší, ale jen do doby než se opět vybije a někdo musí opět přijít a vyměnit ji. V případě tohoto zařízení se předpokládá umístění na vzdálená hůře dostupná místa a je lépe problém řešit jinak.

Předpokládejme, že zařízení při nasazení v praxi bude připojeno k internetu. Z toho plyne možnost využít synchronizace času s některým ntp serverem. Po spuštění si zařízení zažádá pomocí udhcpc (odlehčený dhcp klient) o IP adresu, adresu brány a adresu Domain Name Server (DNS) serveru od routeru s DHCP serverem. Nyní může komunikovat se servery v internetu.

Příkazem ve skriptu `/etc/init.d/ntpd` se zapne nejprve deamon pro pravidelnou synchronizaci systémového času se serverem CESNETu. Počká se pro jistotu 10 s než proběhne synchronizace s ntp serverem a následně se aktualizuje RTC na desce z již přesného času systému.

```
    echo "ntpd starting"
    sleep 3
    /usr/sbin/ntpd -p ntp.cesnet.cz
    sleep 10
    /sbin/hwclock -w
```

Listing 2.4: Výpis skriptu /etc/init.d/ntpd

2.2.4 Konfigurace Apache2 serveru

Protože byla v PTXdist zvolena instalace Apache2, byl k němu vytvořen také konfigurační soubor `/etc/apache2/httpd.conf`. Zde je potřeba přidat jednu zásadní věc a tou je načítání modulu PHP5. Jelikož se jedná o dynamicky linkovanou knihovnu, tak není v jádru Apache a musí se přidat ručně. Tím získáme schopnost zpracovat nejen obyčejné html stránky a cgi skripty, ale také php skripty. Na řádek 230 tedy přidáme následující:

```
# PHP Configuration for Apache
#
# Load the apache module
#
LoadModule php5_module /usr/modules/libphp5.so
```

```
AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps

DirectoryIndex index.php
```

Listing 2.5: Konfigurace Apache2 v `/etc/apache2/httpd.conf`

Ponechal jsem nastavení uživatele a skupiny, pod kterými je server spouštěn na *www*. Tzv. DocumentRoot, kde server hledá dokumenty pro klienty je rovněž ponechán výchozí `/var/www`. Hlášení chyb je na řádku 496 zapnuto a nasměrováno do souboru `/var/log/apache2/error_log` umístěného v paměti RAM v systému souborů tmpfs (opět je to vhodné z důvodu opotřebení SD karty). Úroveň hlášení lze nastavit z hodnot debug, info, notice, warn, error, crit, alert, emerg. Já nechávám warn.

2.2.5 Konfigurace PHP5

Nachází se v souboru `/etc/php5/php.ini`. Opět velmi rozsáhlý soubor, kde je třeba změnit několik položek. Nejprve na řádku 307 zmenšit parametr `memory_limit` ze 128 MB na 64 MB. To z důvodu, že máme k dispozici jen cca 120 MB RAM a rozhodně ji nelze nechat zahltit daty od nějakého špatně napsaného skriptu, který by pak mohl vyčerpat všechny prostředky systému a dalším aplikacím[19].

Menší nevýhodou je, že se skripty v php spouštěné na ARM nedají krokovat, jako třeba v jazyce C. Proto je potřeba alespoň zapnout všechna dostupná hlášení, která interpret generuje do webové stránky. Na řádku 354 se tedy zapíše `error_reporting = E_ALL` a na ř. 373 `display_errors = On`. Nejen chyby, ale i obyčejná varování jsou teď zobrazována a podle čísla řádku se dá rychle dohledat problém.

Nastavení dalších vlastností je pak na uvážení programátora, v rámci mých potřeb výše zmíněné stačilo a ostatní hodnoty jsem nechal výchozí.

Dodejme, že vytvořené skripty a obecně dokumenty, které má server zpracovat musí mít nastavena práva alespoň pro čtení pro uživatele *www*, aby je vůbec mohl zpracovat. Adresář `www` a další vnořené pak musí mít vůči uživateli *www* i právo execute tedy spouštění, aby data v nich byla přístupná.

2.3 Přístup k periferiím Mini6410 v OS Linux

Zařízení, která byla rozpoznána jádrem se obecně nalézají v adresáři `/dev/`.

V mé verzi jádra (3.0.8) se záplatami od Pengutronix je pro přístup k hardware využíván filesystem sysfs. Je to část jádra připojená v RAM, tudíž je virtuální nikoliv fyzická. Přípojný bod souborového systému je `/sys/`.

2.3.1 Ovládání GPIO

Rozhraní GPIO nalezneme v `/sys/class/gpio/`.

```
root@mini6410:/sys/class/gpio ls
export        gpiochip121   gpiochip161   gpiochip194   gpiochip38
    gpiochip74    unexport
gpio134       gpiochip0     gpiochip137   gpiochip17    gpiochip26
    gpiochip55    gpiochip9
gpio135       gpiochip104   gpiochip144   gpiochip178   gpiochip32
    gpiochip63    gpiochip91
```

Každé číslo za slovem gpiochip ukazuje na začátek rozsahu konkrétního portu s I/O piny. Je třeba si vyčíst ze schématu jaký pin chceme ovládat, např. některý z konektoru CON6 s čistě GPIO vývody. Řekněme, že chceme ovládat pin GPM0. Musíme najít ve kterém rozsahu se GPM nachází. Zkusmo jsem zjistil, že pin GPM0 je na gpiochip137. Každá položka gpiochip má totiž svůj label, který říká jakému portu rozsah odpovídá [17]. Nyní si vyexportuji pin tak, abych k němu měl přístup a podívám se jakými soubory je pin popsán:

```
root@mini6410:/sys/class/gpio echo 137 > export
root@mini6410:/sys/class/gpio cd gpio137
root@mini6410:/sys/class/gpio/gpio137 ls
active_low  direction  edge  power  subsystem  uevent  value
```

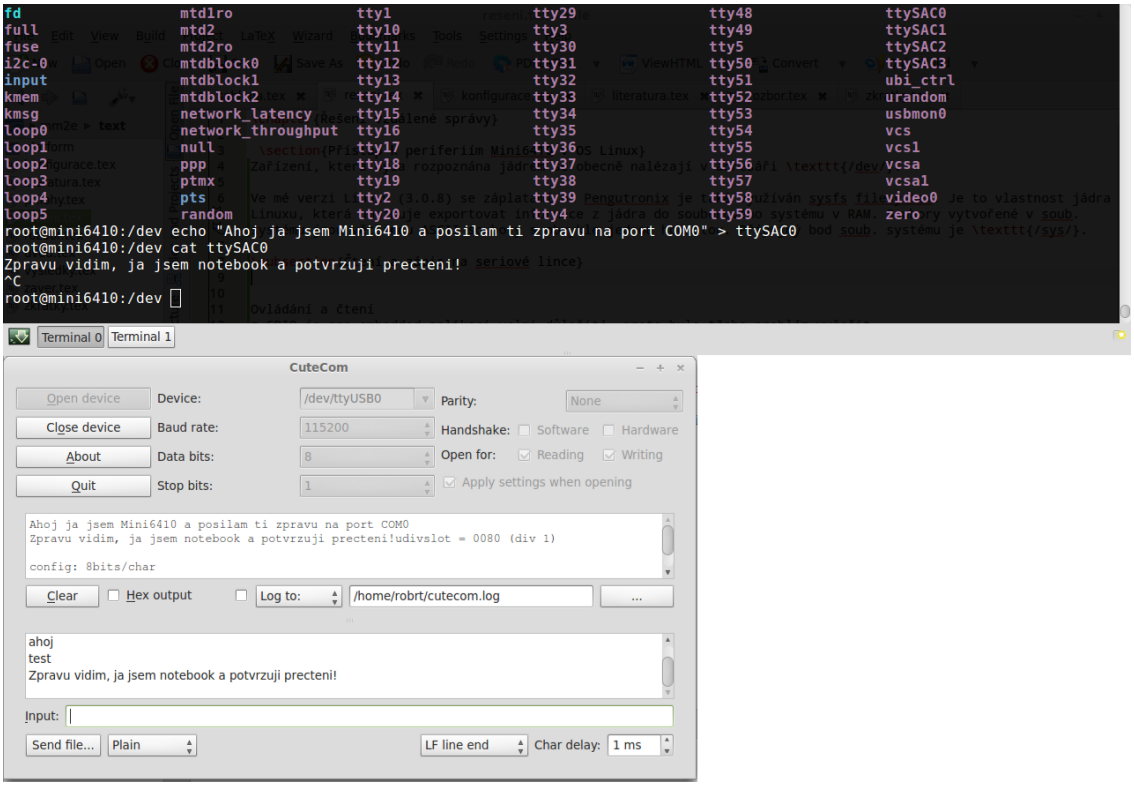
Pomocí příkazu

```
root@mini6410:/sys/class/gpio/gpio137 echo "out" > direction
```

se dá změnit pin na výstupní. Soubor `value` obsahuje údaj o hodnotě na vstupu pokud je pin nastaven jako vstupní. Jinak, pokud je pin výstupní do `value` můžeme zapsat 1 nebo 0 a tím jej nastavit.

2.3.2 Ovládání seriové linky (UART)

Pro seriovou linku v linuxu se standartně používá zařízení v `/dev` s prefixem `tty`. V Mini6410 jsou 4 seriové rozhraní - ttySAC0 až ttySAC3. TtySAC0 je typu RS232 a přes něj komunikujeme null modemem. Následující obrázek ukazuje komunikaci nejprve od Mini6410 směrem k PC a poté odpověď z PC:



Obr. 2.6: Komunikace přes seriovou linku v konzoli na Mini6410 a PC

Aby mohl později ke všem jednotkám UART přistupovat i web server, je potřeba přidat uživatele *www*, pod kterým je Apache spouštěn do skupiny *dialout*. V souboru `/etc/group` na řádek se záznamem dialout doplníme uživatele www takto:

```
dialout:x:106:www
```

2.3.3 Čtení z AD převodníků

Hlavní důvod přechodu na PTXdist je patch Kernelu pro čtení ze všech osmi kanálů AD převodníku. Toto nebylo možné v jiných verzích jádra, jakkoliv jsem se pokoušel je konfigurovat.

Uživatelsky přívětivý způsob jak vyčítat hodnotu na kanálu AD převodníku skýtá opět sysfs. Přesuneme se do adresáře, odkud budeme vyčítat a zobrazíme jeho obsah:

```
root@mini6410:~ cd /sys/class/hwmon/hwmon0/device
root@mini6410:/sys/class/hwmon/hwmon0/device ls
driver      in0_input   in1_input   in2_input   in3_input   in4_input
    in5_input   in6_input   in7_input   modalias    subsystem
hwmon       in0_label   in1_label   in2_label   in3_label   in4_label
    in5_label   in6_label   in7_label   power       uevent
```

Listing 2.6: Výpis vstupů ADC

Nyní čteme to co nás zajímá tj. hodnotu na vstupu Xtého kanálu(inX_input). Pro čtení na kanálu 0 píšeme:

```
root@mini6410:/sys/class/hwmon/hwmon0/device cat in0_input
7732
```

Na další řádek se zobrazí hodnota na vstupu (aktuálně 7732). Převodník je připojen na referenci 3,3V jíž odpovídá číslo 13210.

2.3.4 Ostatní periferie

Samozřejmě deska disponuje řadou dalších rozhraní. Můžeme číst stavy tlačítek, připojit zařízení přes I2C, SPI nebo do USB. Přístup k nim je obdobný výše popsaným postupům. USB host je zde pouze verze 1.1 z čehož plynou omezení pro rychlost a připojitelnost některých zařízení. Navíc pro cokoliv zde připojeného musí existovat zavedený modul v jádře, aby systém zařízení detekoval. V práci USB nevyužiji, kromě testování web kamer.

Pro použití displeje je potřeba zkompilovat knihovny grafického prostředím Qt. Poté lze na vývojářově PC vytvářet aplikace v prostředí QtCreator a cross-compilovat pro ARM. Toho bude jistě před nasazením zařízení do praxe využito, ale je to nad rámec mé práce.

V desce je také pevná flash paměť 1GB. Její připojení poskytuje další prostor pro data. Vzhledem k 2 GB kapacity karty však není další paměť nezbytná.

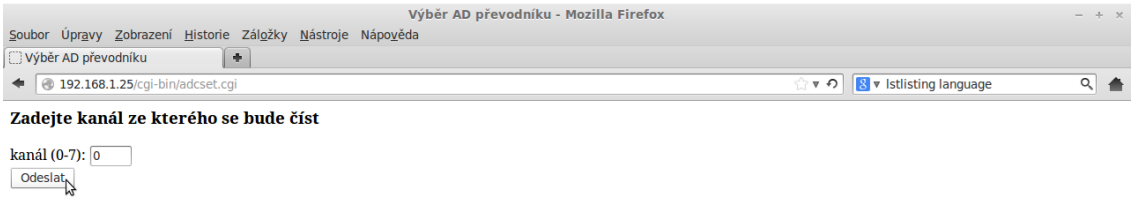
Také je možno připojit dodanou CMOS kameru, jejíž testování je popsáno později.

3 ŘEŠENÍ VZDÁLENÉ SPRÁVY

3.1 Ukázka web serveru s CGI skripty

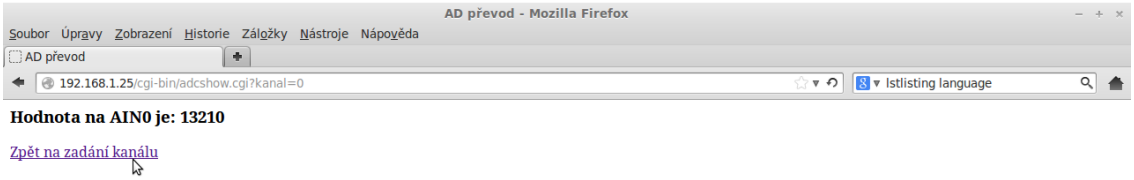
Na základě získaných znalostí o ovládání periferií a funkci CGI[11] jsem vytvořil jednoduchý systém pro čtení hodnoty z vybraného kanálu AD převodníku.

Uživatel zadá ve web. prohlížeči IP adresu Mini6410. Následně je přesměrován na první CGI skript. Zadá se kanál a odešle formulář. Formulářem zavolaný skript



Obr. 3.1: Nastavení kanálu pro čtení ve webové stránce

`adcshow.cgi` rozparsuje hodnotu kanálu. Dle ní pak otevře pro čtení patřičný soubor v adresáři `/sys/class/hwmon/hwmon0/device/` a hodnotu odešle zabalenou v html kódu. Stránce se nastaví meta tagem, aby každé 3 sekundy provedla obnovení. Můžeme tak sledovat, jak se mění vstupní hodnota v diskrétním kroku 3 s.



Obr. 3.2: Čtení hodnoty ve webové stránce

Dále uvedený kód je cgi program `adcshow.cgi` v jazyce C. Kód je ukázkou, jak může spolupracovat jazyk C s html kódem utvářejícím výslednou stránku.

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
int main(void)
{
  FILE *ain;
  char *query;
  char hodnota[10], kanalchar;
  int i,kanal;
  char cesta[41] = "/sys/class/hwmon/hwmon0/device/inX_input\0";// X
      bude nacteno
  query = getenv("QUERY_STRING"); //QUERY_STRING je retezec od
      formulare
  kanalchar=query[6]; //parsovani - na pozici 6 je hodnota kanalu
  printf("%s%c%c\n",     "Content-Type:text/html;charset=utf-8",13,10);
  printf("<HTML><HEAD>\n<meta http-equiv=\"refresh\" content=\"3\">\n<
      TITLE>AD převod</TITLE>\n</HEAD>\n");
  printf("<BODY>\n");
  cesta[33]= kanalchar; //vybere zadany vstup
  ain = fopen(cesta,"r");
  fgets(hodnota,sizeof(hodnota) , ain); //cti z AINX string s hodnotou
  if(ain == NULL) // pokud neni prevodnik dostupny vypis chybu
  {
    printf("%s%c%c\n", "Content-Type:text/html;charset=utf-8",13,10);
    printf("<HTML><HEAD><TITLE>ERROR!</TITLE></HEAD>\n");
    printf("<BODY>\n");
    printf("<H3>ERROR: Nelze číst z převodníku!</H3>\n");
    printf("</BODY></HTML>\n");
    return 0;
  }
  printf("<H3>Hodnota na AIN%c je:  %s</H3>\n",kanalchar,hodnota);
  printf("<a href=\"./adcset.cgi\">Zpět na zadání kanálu</a>\n");
  printf("</BODY>\n</HTML>\n");
  fclose(ain);
  return 0;
}
```

Listing 3.1: Ukázka CGI skriptu

CGI soubory je potřeba na serveru nahrát do složky `/var/www/cgi-bin/`. Obyčejné html stránky se nahrávají o adresář výš. CGI skripty musí mít nastavena práva pro spuštění pro uživatele www pod kterým věží http démon.

K nahrávání souborů doporučuji použít například program Filezilla, kde si předdefinujeme SFTP spojení s Mini6410 a pak již pohodlně kopírujeme z disku na server.

3.2 Realizace web stránky vzdálené správy v PHP

Tato část práce popisuje hlavní část realizace zadání. Výsledek je dosažen kombinací programů a skriptů v jazyce C, PHP, SQL, HTML a Cascading Style Sheets (CSS). K nastudování PHP a SQL jsem použil literaturu [6] a elektronické manuály [19][20]. Pro oživení si znalostí HTML a CSS jsem použil prameny [7][18].

Hlavní dokument `index.php` volá jednotlivé dílčí části, které jsou tvořeny opět jako kusy kódu v PHP aby vše bylo přehledné:

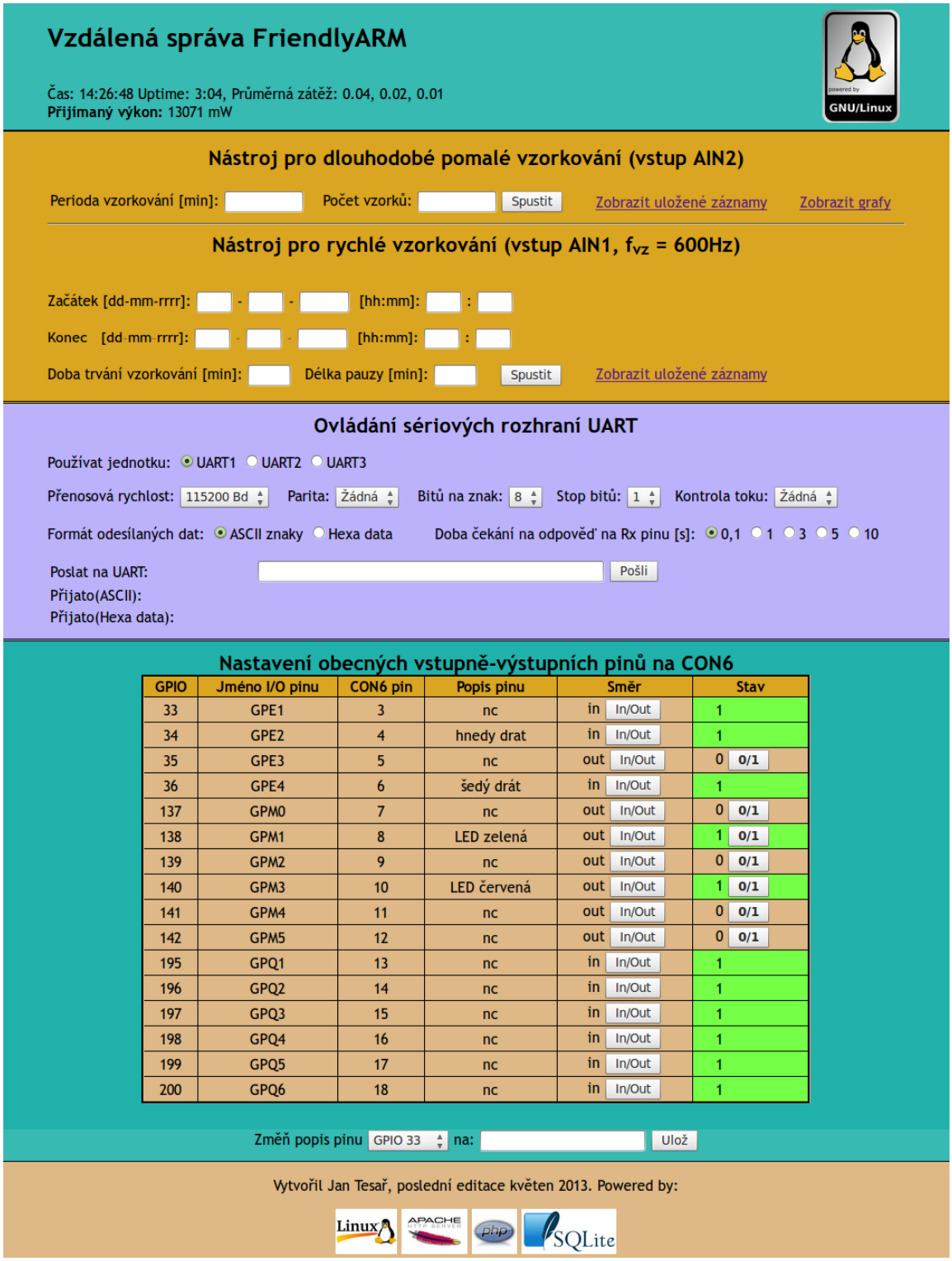
```
<?php include_once("header.php");?>
<?php require ("functions.php"); ?>
<body>
  <div id="all">
    <div id="wrapper">
      <div id="header">
        <h1>Vzdálená správa FriendlyARM</h1>
        <div id="logo">
          <img src="images/powered_by.png"  height="125" alt="Logo" />
        </div>
      </div>

      <iframe  src="system_state.php" id="ifr_monitoring">
        <p>Váš prohlížeč nepodporuje iframes –&gt; zkuste jiný.</p>
      </iframe>
      <div id="longper">
        <?php include("adc.php");?>
      </div>
      <div id="serial">
        <?php include("serial.php");?>
      </div>
      <iframe src="gpio_frame.php" id="ifr_gpio">
        <p>Váš prohlížeč nepodporuje iframes –&gt; zkuste jiný.</p>
      </iframe>
      <div id='gpio_popis'>
        <?php include("description_gpio.php")?>
      </div>

      <div id="footer">
        <?php include("footer.php")?>
      </div>
    </div>
  </div>
</body>
</html>
```

Listing 3.2: Zdrojový kód souboru index.php

Stránka se skládá z pěti hlavních sekcí. Každá z nich slouží k ovládání jiné periferie.

Některé části stránky se sami obnovují v intervalu 10 s z důvodu automatické aktualizace monitorovaného údaje. Toho je dosaženo html vlastností iframe (pozor nejedná se o html rámce), která umožňuje tvořit něco jako stránku ve stránce. Důvodem, proč nenechávám refresh celé stránky je, aby když uživatel vyplňuje nějaký formulář např. pro nastavení UART se mu jeho zadávané hodnoty nepřepsali každých 10 s.

Na následující straně jsem vytiskl celou vytvořenou webovou stránku. Další části kapitoly popisují jednotlivé nástroje obsažené na stránce.



Obr. 3.3: Vytvořená webová stránka vzdálené správy

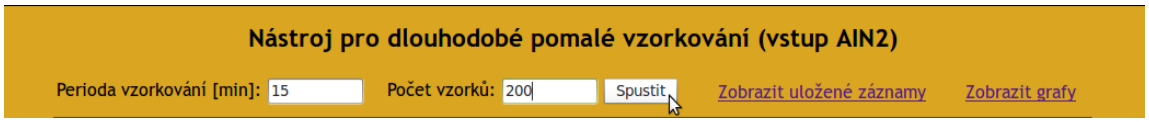
3.2.1 Hlavička stránky - aktuální stav systému

Obr. 3.4: Hlavička stránky s aktuálním stavem systému

V hlavičce je obsažena informace o aktuálním čase, době od posledního restartu a průměrné zátěži procesoru za 1, 5 a 15 minut (1 = 100% vytížení).

Další řádek zobrazuje momentální hodnotu napětí na vstupu ADC kanálu 3. V praxi bude tato hodnota mít význam momentálně dopadajícího optického výkonu. Hodnota na Obr.3.4 je jen ukázková. Hlavička je vložena jako iframe do hlavní stránky a informace je obnovena každých 10 s. Samozřejmě se dá znova načítat třeba každou půl sekundu, kdybychom potřebovali například hýbat s optickou hlavicí a sledovat okamžitou změnu výkonu.

3.2.2 Nástroj pro dlouhodobé pomalé vzorkování



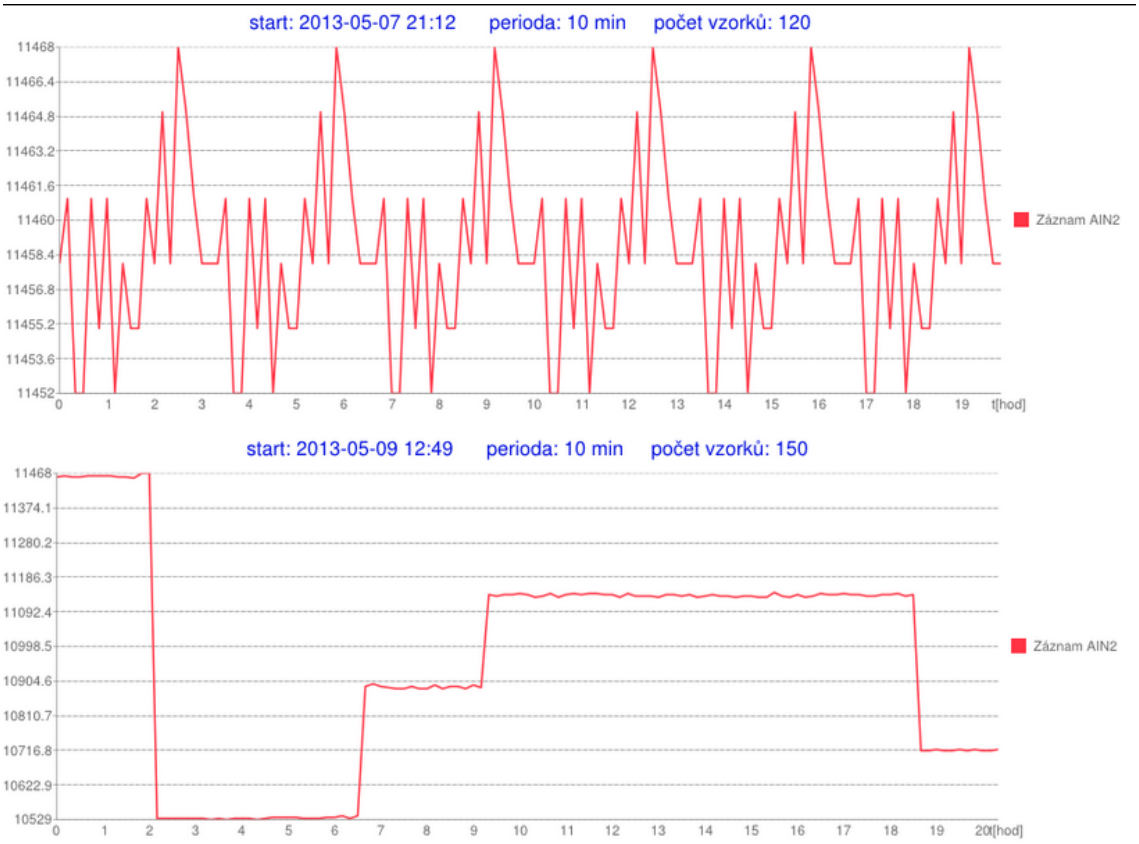
Obr. 3.5: Nástroj pro dlouhodobé pomalé vzorkování

Tento nástroj byl vytvořen pro potřebu dlouhodobého sledování dopadajícího optického výkonu. Vyžaduje zadat periodu s jakou se odebírají vzorky ADC a kolikrát se mají odebrat. Vynásobením těchto čísel dostaneme celkovou dobu měření.

Po odeslání formuláře je spuštěn php skript na pozadí OS. Ten dle zadaných parametrů periodicky čte ADC a hodnotu zapisuje do log souboru na SD kartě. Soubor pojmenuje datem a časem startu a parametry snímání, aby byl později snadno identifikovatelný. Odkaz Zobrazit uložené záznamy otevře složku s logy, které si může uživatel stáhnout k sobě do PC.

Odkaz Zobrazit grafy otevře stránku, na které vykreslí ze záznamů grafy. Tato funkce využívá Google službu gChart, která ze zaslaných parametrů poskládá graf a ten zašle zpět. Aby se s gChart dalo dobře pracovat, používám převzatou open

source knihovnu `gChart.php`, která chápe grafy jako objekty PHP, s nimiž se lépe pracuje [22].



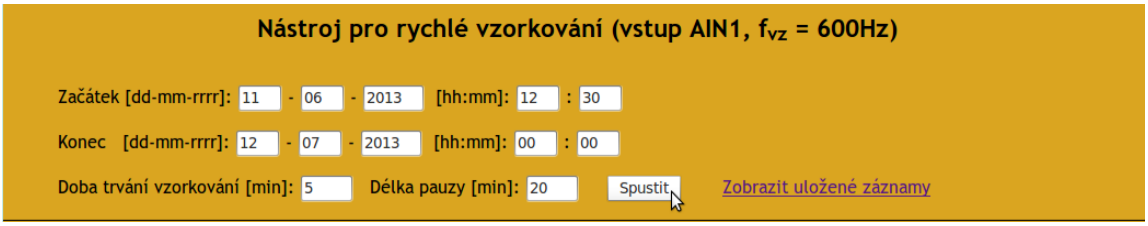
Obr. 3.6: Ukázka vykreslení grafů z dlouhodobých záznamů

3.2.3 Nástroj pro rychlé vzorkování

Abychom mohli zaznamenat velmi rychlé změny stavu atmosféry (tzv. optické turbulence), je potřeba sejmout mnoho vzorků dopadajícího výkonu za sekundu. V ideální případě bychom potřebovali vzorkovat s frekvencí 1 kHz.

Snažil jsem se udělat co nejrychlejší vzorkovací rutinu. Proto jsem program, který toto provádí napsal v C a zkompiloval pro ARM, abych dosáhl maximální výpočetní výkon. Vytvořil jsem vzorkovač s rychlostí 600 Hz. Znamená to tedy, že nedosahuje ideální rychlosti. Není to způsobeno hardwarem. Ten zvládá vyčítat kanál rychlostí 1 MHz. Problém je na straně software. Moje řešení využívá pro všechny periferie přístup přes OS. To znamená, že musí pokaždé otevřít virtuální soubor daného kanálu, ten vyčíst (zde stráví program nejvíce času) a pak zapsat hodnotu do souboru. Ideální přístup by byl pracovat na nižší programovací úrovni a

nastavit si ADC dle potřeb pomocí registrů procesoru a pak přímo tyto registry číst na základě přerušení. Vzhledem k chabé dokumentaci jak toto s procesorem udělat, jsem se do nízkoúrovňového řešení nepustil.



Obr. 3.7: Nástroj pro rychlé vzorkování

Nástroj vyžaduje zadání času spuštění a ukončení. To je výhodné chceme-li spustit vzorkování v době, kdy nemůžeme být přítomni u PC. Když víme, že v danou dobu se budou odehrávat v atmosféře zmiňované turbulence a chceme si je s jistotou zaznamenat, s výhodou si vše dopředu načasujeme. Doba trvání vzorkování říká, jak dlouho se má měřit a zapisovat s frekvencí 600 Hz bez přerušení. Poté přijde pauza, kdy se nic neděje a po ní se opět opakuje měření.

Je třeba podotknout, že se ve výsledku může jednat o objemný soubor dat. Proto je třeba dobře zvážit jak dlouho a jak často se bude snímat. Za 1 minutu snímání se vygeneruje 200 KB dat. Ty jsou navíc sbírány v RAM a teprve po skončení zapisovány na SD kartu, aby se zvýšila rychlost zápisu a omezil počet zapisování na kartu. Záznamy se dají opět nalézt v logovacím adresáři po kliknutí na Zobrazit uložené záznamy.

3.2.4 Komunikační rozhraní jednotek UART

Již řadu let používané, přesto stále populární je rozhraní UART. Vzhledem ke své relativní jednoduchosti jej často vývojáři ve svých aplikacích používají pro komunikaci s vnějším světem. Možnost UART ovládat tedy nechybí ani v mém zařízení.

Obr. 3.8: Komunikační rozhraní jednotek UART

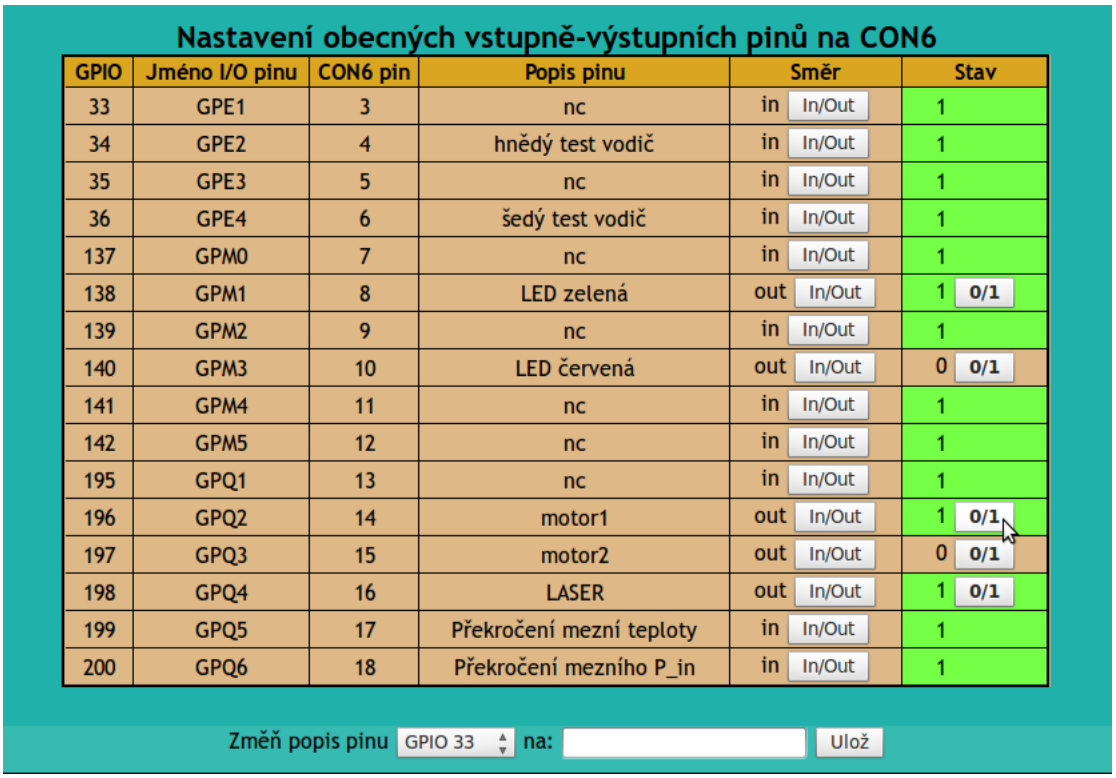
Nejprve si obsluha zvolí, přes kterou ze tří dostupných jednotek chce komunikovat. Čtvrtá jednotka (UART0) není nabízena, jelikož ji používá OS jako systémovou konzoli se standardem RS232 pro případnou konfiguraci systému při nefunkčním rozhraní Ethernet. Na druhém řádku následují standardní nastavení, která není třeba rozebírat. Třetí řádek umožňuje výběr typu kódování. ASCII se používá nejčastěji, avšak můžeme vybrat i zadávání přímých dat v šestnáctkové soustavě. Doba čekání na odpověď určuje, jak dlouho po odeslání zprávy bude UART čekat na odpověď, respektive ji shromažďovat než ji zobrazí. Na řádku Přijato se odpověď objeví, a to jak v ASCII formátu, tak v podobě surových dat hexa formátu.

Jako backend používám open source PHP třídu `php_serial.class.php` [21]. Po odeslání formuláře se hodnoty jednotlivých prvků formuláře uloží do proměnných. Ty jsou poté odeslány jako vstupní argument funkci, jež z nich vytvoří objekt s daným nastavením. Jednou z metod objektu je možnost odeslat data, která se spustí na závěr.

Většinou se nezasílá jen jeden příkaz, ale více za sebou se stejným nastavením. Proto je ošetřeno uchování nastavení i po odeslání dat. Uživatel tak nemusí neustále měnit výchozí hodnoty.

3.2.5 Ovládání GPIO

Mini6410 disponuje desítkami GPIO pinů. Některé ani nejsou vyvedeny od procesoru na konektory. Konektor CON6 obsahuje piny z portů GPE, GPM a GPQ - dohromady 16 GPIO pinů. Ty nám umožňují buďto snímat jakýkoliv dvoustavový výstup na ně připojený a nebo pokud jsou nastaveny jako výstupní tak dvoustavově ovládat připojená zařízení. Pro jejich čtení a ovládání byla vytvořena následující správcovská tabulka opět vložená jako iframe s refresh periodou 10 s:



Obr. 3.9: Tabulka pro nastavení GPIO na CON6

Každý řádek obsahuje číslo GPIO dle datasheetu, jméno podle portu ze kterého pochází, umístění v konektoru CON6, volitelný popis pinu, nastavení směru a současný stav resp. tlačítko pro jeho změnu.

Veškeré údaje v tabulce jsou načítány z SQLite3 databáze. Pouze pokud je daný pin přepnut do režimu In tak je čtena hodnota ze vstupu nikoliv z databáze. Vysoká úroveň na pinu je pro přehlednost značena zeleně. Každých deset sekund se znova načtou všechny stavy. Obsluha tak může pozorovat chování připojených zařízení. Pokud není připojeno nic je na vstupu vysoká úroveň díky internímu pull-up rezistoru.

Kliknutím na tlačítko In/Out může obsluha provézt změnu směru pinu. Jakmile má pin směr out, objeví se i tlačítko pro přepínání vysoké a nízké úrovně. Každé kliknutí znamená jednak příkaz pro systém, aby vykonal požadovanou činnost a zároveň se ihned ukládá stav do databáze. Takže databáze je v každém okamžiku aktuální a po výpadku napájení se vše nastaví do stavu před výpadkem. Viz podkapitola 2.2.2.

Pod tabulkou je ještě řádek pro změnu popisku daného pinu. Pin se vybere z rozbalovací select nabídky a zadá se nový text popisu. Toto jsem zavedl pro přehlednost, aby správce věděl na jakém pinu má jaké zařízení. Vše je opět v databázi.

3.3 Kamera

V rámci zadání jsem se pokoušel zprovoznit dodanou CMOS kameru. Narazil jsem však na zásadní problém. Tím byla nepřítomnost modulu ovladače čipu kamery v PTXdist. Čip má název OV9650, v modulech pro možnou kompilaci jsem našel jen model OV9640. Nicméně je mi známo, z reakcí na fóru FriendlyARM [16], že kamera funguje, ovšem na platformě Mini2440 s jádrem Linux 2.6.29. Vzhledem k tomu, že jsem potřeboval opravy z vyššího jádra 3.0.8, tak jsem se již nemohl vrátit zpět na nižší verzi. Pokoušel jsem se ručně kompilovat modul kamery pro svou verzi FriendlyARM a jádra, ale bez výsledku. Jistě existují cesty jak dosáhnout funkčnosti ať již přidáním zdrojů do PTXdist nebo manuálně kompilací modulu. Bohužel mně se to nepovedlo z důvodů nedostatečných zkušeností a nedostatku času. Myslím, že některé úkony v rámci sestavování OS Linux jsou nad rámec běžné elektrotechnické praxe a potřeboval bych se problematikou déle zabývat a konzultovat s někým zkušeným.

Další moje pokusy byly připojování web kamer různých výrobců k USB. Jedna z nich měla dokonce plnou podporu zavedeného modulu v jádře mého FriendlyARM. Přesto jsem se nedočkal od systému po jejím připojení žádného oznámení o detekci v `dmesg` kruhovém bufferu. Když jsem ji připojil na svůj PC okamžitě jsem viděl ve výpisu `dmesg` zprávy o jejím úspěšném připojení. Všechny připojované USB kamery byly vyrobeny pro USB 2.0. Nejsem si jist, jaký vliv má to, že Mini6410 má pouze USB 1.1 rozhraní, kdežto normální PC má 2.0.

4 ZÁVĚR

Dlouho jsem nevěděl na čem vlastně webový server založit. Když jsem zjistil, že PTXdist nabízí vše, co je potřeba pro fungování PHP, byl jsem rozhodnut pro PHP. Jistě by šlo vše realizovat i s CGI a programy v C nebo Pythonu. Jenže převážně se jedná o programování webové aplikace a PHP je jazyk vytvořený za tímto účelem. Po přečtení knížky [6] jsem byl schopen v něm efektivně pracovat. Vývoj v jiném jazyku by mi trval nepoměrně déle, pokud bych chtěl mít všechny vymoženosti formulářů a jejich zpracování které umí PHP. Paleta funkcí je obrovská a to také usnadňuje práci.

V každém bodě práce jsem jako vývojář mohl jít různými cestami. Již na začátku, při testování systémů jsem mohl zkusit nainstalovat distribuci Debian a pak instalovat aplikace přímo z desky pomocí programu Aptitude. Já však zvolil PTXdist a bylo to dle mého názoru také správné řešení. Především díky vývojářům PTXdist, kteří nabízí záplatované verze jádra.

Obecně podpora pro Mini6410 není tak velká, jako pro starší Mini2440 a už vůbec ne jako pro RaspberryPi. Ta má v současné době celosvětovou komunitu vývojářů a kompilované stovky aplikací dostupné přes Raspbian Aptitude. Nemá však některé periferie, které z principu potřebuje optický spoj nebo jakýkoliv jiný monitorovací systém, například AD převodníky. Dále nemá displej, který se bude do budoucna také využívat.

Pokud má vzniknout skutečný prototyp zařízení pro vzdálené ovládání optické hlavice je na zvážení zda setrvat u modelu Mini6410 nebo přejít na novější a rychlejší Mini210 s USB 2.0 či platformu RaspberryPi s kvalitnější podporou.

V práci se mi podařilo realizovat téměř všechny body zadání až na spuštění modulu kamery z důvodů popsaných v kapitole 3.3. Především webová stránka vzdálené správy je velmi dobře použitelná a odladěná. Jsem si jistý, že vytvořené prostředí je univerzální a dá se na něm postavit obecně jakýkoliv dohledový systém. Během práce jsem se zdokonalil ve svých znalostech OS Linux a naučil se na dostatečné úrovni programovat v PHP.

LITERATURA

[1] PETRLÍK, Lukáš. *Jemný úvod do systému UNIX*. České Budějovice: KOPP, 1997. ISBN 80-85828-28-6.

[2] VYCHODIL, Vilém. *Operační systém Linux: příručka českého uživatele*. 1. vyd. Brno: Computer Press, 2005, 260 s. ISBN 80-722-6333-1.

[3] SIEVER, Ellen. *LINUX v kostce*. Vyd. 1. Praha: Computer Press, 1999, 560 s. ISBN 80-722-6227-0.

[4] KROAH-HARTMAN, Greg. *Linux Kernel: in a Nutshell*. 1st ed. Beijing: O´Reilly, 2006, 183 s. ISBN 05-961-0079-5.

[5] GUNDAVARAM, Shishir. *CGI programování webových stránek a aplikací*. Vyd. 1. Praha: Computer Press, 1998, 453 s. ISBN 80-722-6088-X.

[6] BRÁZA, Jiří. *PHP 5: začínáme programovat*. 1. vyd. Praha: Grada Publishing, 2005, 244 s. ISBN 80-247-1146-X.

[7] SATRAPA, Pavel. *Web design*. Praha: Neokortex, 1997, 414 s. ISBN 80-902-2301-X.

[8] PENGUTRONIX E. K. *Quickstart Manual OSELAS.BSP() FriendlyARM mini6410* [online]. 2011 [cit. 2012-12-10]. Dostupné z: <`http://www.pengutronix.de/oselas/bsp/pengutronix/download/OSELAS.BSP-Pengutronix-Mini6410-Quickstart.pdf`>.

[9] IETF. *RFC2616* [online]. 1999 [cit. 2012-11-18]. Dostupné z: <`http://www.ietf.org/rfc/rfc2616.txt`>.

[10] IETF. *RFC3875* [online]. 2004 [cit. 2012-12-18]. Dostupné z: <`https://tools.ietf.org/html/rfc3875#section-6.2`>.

[11] *CGI Made Really Easy* [online]. Poslední aktualizace 2002 [cit. 2012-12-10]. Dostupné z: <`http://www.jmarshall.com/easy/cgi/`>.

[12] Samsung Corporation. *Datasheet Samsung S3C6410* [online]. Poslední aktualizace 2008 [cit. 2012-10-25]. Dostupné z: <`http://www.samsung.com/global/business/semiconductor/file/media/s3c6410_datasheet_200804-0.pdf`>.

[13] POMERLEAU, Jason. Netmon Inc. *Poster http packet* [online]. 2005 [cit. 2013-05-19]. Dostupné z: <`http://www.barracudadeals.com/freebies/poster/poster_http_packet.pdf`>.

[14] *Manuál k desce FriendlyARM Mini6410* [online]. Dostupné z: <http://www.friendlyarm.net/dl.php?file=mini6410_manual.zip>.

[15] *Server s FriendlyARM soubory* [online]. Dostupné z: <http://www.friendlyarm.net/downloads>.

[16] *Fórum uživatelů FriendlyARM* [online]. Dostupné z: <http://www.friendlyarm.net/forum>.

[17] *Dokumentace ovládání GPIO v Linuxu* [online]. Poslední úprava 2009 [cit. 2012-11-03]. Dostupné z: <http://www.avrfreaks.net/wiki/index.php/Documentation:Linux/GPIO>.

[18] *w3schools web development site* [online]. [cit. 2013-03-29]. Dostupné z: <http://www.w3schools.com>.

[19] *PHP Manual* [online]. [cit. 2013-05-15]. Dostupné z: <http://www.php.net/manual/en/>.

[20] *SQLite manual pages* [online]. [cit. 2013-02-19]. Dostupné z: <http://www.sqlite.org/docs.html>.

[21] SANCHEZ, Rémy *PHP třída php_serial.class.php pro práci s UART* [online]. Poslední úprava 2007 [cit. 2013-04-02]. Dostupné z: <http://www.phpclasses.org/browse/file/17926.html>.

[22] *PHP knihovna gChart.php pro práci s Google Charts* [online]. Poslední úprava 2011 [cit. 2013-05-04]. Dostupné z: <https://github.com/pacbard/gChartPhp>.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

ADC Analog-to-Digital Converter

ARM Advanced RISC Machines

ASCII American Standard Code for Information Interchange

CGI Common Gateway Interface

CSS Cascading Style Sheets

DHCP Dynamic Host Configuration Protocol

DNS Domain Name Server

DPS Deska Plošného Spoje

EEPROM Electrically Erasable Programmable Read-Only Memory

GPIO General Purpose InputOutput

GPL General Public License

HTTP Hypertext Transfer Protocol

I2C Inter-IC

IETF Internet Engineering Task Force

LCD Liquid-Crystal Display

NTP Network Time Protocol

OS Operační Systém

OSI Open Systems Interconnection

PHP PHP: Hypertext Preprocessor

RFC Requests For Comments

RTC Real-Time Clock

SD Secure Digital

SDHC Secure Digital High-Capacity

SFTP SSH File Transfer Protocol

SMD Surface-Mount Device

SoC System on a Chip

SQL Structured Query Language

SSH Secure Shell

TCP Transmission Control Protocol

TTL Time To Live

UART Universal Asynchronous Receiver/Transmitter

UDP User Datagram Protocol

W3C World Wide Web Consortium

WAN Wide Area Network

SEZNAM PŘÍLOH

A MANUÁL K VYTVÁŘENÍ APLIKACÍ PRO VZDÁLENÝ PŘÍSTUP

Příloha si bere za cíl uvést praktické rady jak zacházet se software vývojového kitu. Konfigurace OS se předpokládá taková, jaká je popsána ve vlastní diplomové práci.

Manuál je psán vzhledem k uživateli používajícímu OS Linux pro vývoj na svém PC. Pro uživatele z Windows je potřeba některé kroky a programy pozměnit, popřípadě jsou neproveditelné. Proto silně doporučuji pro tento účel používat Linux, alespoň jako virtuální systém ve Windows.

A.1 Správa desky ze vzdáleného terminálu

A.1.1 Telnet

Pro přihlášení pomocí telnetu proveďte následující:

```
robrt@robrtnbk:~$ telnet "IP adresa přidělená routerem desce"
Trying 192.168.1.25...
Connected to 192.168.1.25.
Escape character is '^]'.

mini6410 login: root
Password: root
root@mini6410:~
```

Listing A.1: Přihlášení pomocí telnet

Toto není preferovaný způsob přihlašování, spojení je nešifrováno. Deska zatím nemá nastaveno bezpečné heslo, před ostrým nasazením jej prosím nastavte!

A.1.2 SSH

Pro přihlášení pomocí SSH proveďte následující:

```
robrt@robrtnbk:~$ ssh root@"IP adresa přidělená routerem desce"
root@192.168.1.25's password: root
root@mini6410:~
```

Listing A.2: Přihlášení pomocí SSH

Toto je preferovaný způsob přihlašování, spojení je šifrováno. Před prvním spojením se systém zeptá zda přijmout RSA klíč, zadáte ano. Deska zatím nemá nastaveno bezpečné heslo, před ostrým nasazením jej prosím nastavte!

A.2 Efektivní kopírování souborů mezi deskou a vzdáleným PC

Již po výchozí instalaci se lze připojit na FTP server, který je vždy spuštěn. Pohodlně toho lze dosáhnout například programem Filezilla. Jméno a heslo jsou stejné jako pro přihlášení do terminálu. V praxi FTP nedoporučuji, není šifrovaný přenos hesla.

A.2.1 SFTP

SFTP je bezpečný a dnes standardní způsob přenosu dat webovými vývojáři. Při prvním přihlášení se musí potvrdit RSA klíč jako u SSH.

Velmi výhodné je mít desku připojenou jako další disk ve své adresářové struktuře. Toho lze docílit pomocí souborového systému sshfs. Doporučuji si vytvořit následující alias v souboru `/home/mujhome/.bash_aliases`:

```
alias farm_sftp='sshfs root@192.168.1.25:/ /media/sftp'
```

Listing A.3: Alias pro namapovaní souborů desky do PC

Příkazem `farm_sftp` se namapuje celý kořenový adresář FriendlyARMu do složky `/media/sftp` na vašem PC. Tady lze otevřít jakýkoliv soubor a po uložení se změny ihned projeví, bez nutnosti cokoliv někam kopírovat jako u FTP.

A.3 Rozšiřování správcovské stránky

Jakmile máme přístup k souborům v SD kartě na desce můžeme začít s úpravami. Složka `/var/www` obsahuje soubor `index.php` a další jím volané skripty.

```
root@mini6410:/var/www ll -R
.:
drwxrwxr-x     5 www      www          4096 May 24 01:29 .
drwxr-xr-x     8 root     root         4096 Jan 18 09:16 ..
-rw-r--r--     1 www      www          4154 May 12 21:22 adc.php
drwxr-xr-x     2 www      www          4096 May 12 15:35 cgi
-rw-r--r--     1 www      www           751 Apr  4 22:55 config.php
-rw-r--r--     1 www      www          1528 Apr 29 11:17
    description_gpio.php
-rw-rw-rw-     1 www      www          6144 May 20 18:52 farm.db
-rw-r--r--     1 www      www           305 Apr 26 13:37 footer.php
-rw-r--r--     1 www      www          3885 May 11 14:54 functions.php
-rw-r--r--     1 www      www         41163 May  7 20:57 gChart.php
-rw-r--r--     1 www      www          3999 Apr 18 11:17 gpio_frame.php
-rw-r--r--     1 www      www           671 May  8 16:33 grafy.php
-rw-r--r--     1 www      www           393 Apr 16 09:16 header.php
```

```
-rw-r--r--    1 www      www           427 Apr 26 12:02
    header_adc_monitor.php
-rw-r--r--    1 www      www           416 Apr 26 13:01 header_gpio.
    php
drwxr-xr-x    2 www      www          4096 Apr 26 13:55 images
-rw-r--r--    1 www      www           961 May 10 08:50 index.php
drwxrwxr-x    3 www      www          4096 May 12 16:52 log
-rw-r--r--    1 www      www         14387 Apr 12 14:18 php_serial.
    class.php
-rw-r--r--    1 www      www            20 Feb 26 22:21 phpinfo.php
-rwxr-xr-x    1 www      www           724 May  7 12:18
    script_adc_longper.php
-rwxr-xr-x    1 www      www          1373 May 12 21:29
    script_adc_shortper.php
-rw-r--r--    1 www      www          6205 Apr 18 11:07 serial.php
-rw-r--r--    1 www      www          1756 May 10 13:31 styles.css
-rw-r--r--    1 www      www           403 May 10 08:49 system_state.
    php

./cgi:
drwxr-xr-x    2 www      www          4096 May 12 15:35 .
drwxrwxr-x    5 www      www          4096 May 24 01:29 ..
-rwxr-xr-x    1 www      www          8441 Feb 23 23:17 fastadc
-rwxr-xr-x    1 www      www           854 Feb 15 10:18 fastadc.sh
-rw-r--r--    1 www      www           803 Apr 30 10:24 slowadc.sh

./images:
drwxr-xr-x    2 www      www          4096 Apr 26 13:55 .
drwxrwxr-x    5 www      www          4096 May 24 01:29 ..
-rw-r--r--    1 root     root         5130 Apr 26 13:26 apache.jpeg
-rw-r--r--    1 root     root         3638 Apr 26 13:55 bckgr.svg
-rw-r--r--    1 root     root         6722 Apr 26 13:27 linux.jpeg
-rw-r--r--    1 root     root         2022 Apr 26 13:25 php.jpeg
-rw-r--r--    1 root     root        47745 Apr 17 23:30 powered_by.png
-rw-r--r--    1 root     root         6436 Apr 26 13:56 pozadi.png
-rw-r--r--    1 root     root         6559 Apr 26 13:25 sqlite.jpeg

./log:
drwxrwxr-x    3 www      www          4096 May 12 16:52 .
drwxrwxr-x    5 www      www          4096 May 24 01:29 ..
-rw-r--r--    1 www      www           744 May  8 16:55 2013-05-07_21
    -12_per10_vz20.txt
-rw-r--r--    1 www      www            42 May  8 16:54 2013-05-08_15
    -23_per30_vz60.txt
-rw-r--r--    1 www      www           763 May 10 09:09 2013-05-09_12
    -49_per10_vz150.txt
```

```
drwxr-xr-x     2 www      www            4096 May 20 17:41 fast_adc

./log/fast_adc:
drwxr-xr-x     2 www      www            4096 May 20 17:41 .
drwxrwxr-x     3 www      www            4096 May 12 16:52 ..
-rw-r--r--     1 www      www          648015 May 12 21:13 2013-05-12_21
    :08_per_1_paus_1.txt
-rw-r--r--     1 www      www         2159989 May 13 10:00 2013-05-13_09
    -40_per_5_paus_10.txt
-rw-r--r--     1 www      www         3240013 May 20 17:41 2013-05-20_17
    -23_per_3_paus_1.txt
```

Listing A.4: Struktura složky `/var/www`

Adresář `log` obsahuje záznamy z AD převodníků, čas od času je možná bude potřeba promazat.

Chceme-li si vytvořit novou funkci v naší správcovské stránce stačí editovat `index.php` ve svém oblíbeném PHP editoru (doporučuji Bluefish, případně PSPad ve Windows). Zorientovat se v tomto souboru je snadné, jeho výpis je v kapitole 3.2. Výsledný vzhled je popsán souborem `styles.css`. V něm si vzhled nové části stránky nadefinujeme podobně jako pro již vytvořené bloky.

V souborech `header*.php` se skrývají hlavičky. Pokud potřebujete pozměnít periodu refresh nějaké části stránky, tak zde to lze nastavit.