TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky a mezioborových inženýrských studií

Studijní program: B2612 – Elektrotechnika a informatika
Studijní obor: 1802R022 – Informatika a logistika

Realizace distribuce Linuxu pro řídící systém PXA s real-time podporou Xenomai

Realization of Linux distribution for system PXA supporting real-time Xenomai

Bakalářská práce

Autor: **Ondřej Novák**
Vedoucí práce: Ing. Pavel Pirkl
Konzultant: Ing. Jan Kraus

Prohlášení

Byl(a) jsem seznámen(a) s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 o právu autorském, zejména § 60 (školní dílo).

Beru na vědomí, že TUL má právo na uzavření licenční smlouvy o užití mé bakalářské práce a prohlašuji, že **s o u h l a s í m** s případným užitím mé bakalářské práce (prodej, zapůjčení apod.).

Jsem si vědom(a) toho, že užít své bakalářské práce či poskytnout licenci k jejímu využití mohu jen se souhlasem TUL, která má právo ode mne požadovat přiměřený příspěvek na úhradu nákladů, vynaložených univerzitou na vytvoření díla (až do jejich skutečné výše).

Bakalářskou práci jsem vypracoval(a) samostatně s použitím uvedené literatury a na základě konzultací s vedoucím bakalářské práce a konzultantem.

Datum

Podpis

Poděkování

Rád bych poděkoval svým rodičům za podporu a motivaci, kterou mi poskytli nejen při psaní této práce, ale i v osobním životě a během studií. Dále bych chtěl poděkovat vedoucímu práce Ing. Pavlu Pirklovi a zejména konzultantovi Ing. Janu Krausovi za věnovaný čas, odborné vedení a cenné rady týkající se vypracování práce. Závěrem bych chtěl poděkovat své přítelkyni za víru a trpělivost, kterou se mnou měla a její pomoc při opravě tohoto dokumentu.

Anotace

Tato bakalářská práce pojednává o využití Linuxové distribuce pro embedded systémy. Zabývá se hlavně modulem Xboard PXA255 od společnosti Kontron. Účelem práce je zprovoznit a otestovat tento modul. Teoretická část obsahuje poznatky o zařízení, jeho vlastnostech a možnostech využití. Pokračuje výběrem vhodné Linuxové distribuce, kterou by bylo možné pro modul použít. Konec teoretické části pojednává o real-time systémech. Praktická část práce se zabývá zavedením operačního systému na modul. Ukazuje postupy nastavení a propojení systémů. Následuje instalace zvoleného systému, její zavedení na modul a otestování aplikací. Závěrem celé práce je ukázka možností vytvoření vlastního OS pro embedded systémy.

Annotation

This bachelor work is about using a Linux distribution for embedded systems. Especially it occupyes with module Xboard PXA255 made by company Kontron. Purposes of this work are to get to work and to test this module. Teoretical part of the work contains informations about the device and atributes and possibilites of it's features. Next there is choosing of suitable Linux distribution, which could be used for the Kontron module. Last part occupyes with informations about real-time systems. Practical part occupyes with installation OS to the module. This part showes procedures of configuration and connecting the system. Next there is installation choosen system, its installation to module and testing applications. In closing part there is example of possibiletes of creating own embedded OS for embedded system.

Obsah

Slovník použitých výrazů

Bootloader je systémový zavaděč jádra. Program, který přejímá kontrolu počítače od BIOSu (boot) a předá ji jádru (kernelu).

BSP (Board Support Package) je realizace nezbytných podpůrných kódů uzpůsobujících operační systémy pro desky embedded systémů. Součástí balíčku obvykle bývá bootloader, který obsahuje minimální potřebnou podporu zařízení, aby mohl načíst operační systém a ovladače všech zařízení obsažených na desce. Někdy bývá součástí balíčku také root filesystém, toolchain pro kompilování aplikací spustitelných na embedded systémech a konfigurátory zařízení.

ELK (Emedded Linux Konfigurator) je grafické prostředí užívané ELinOSem ke konfiguraci vytvářeného systému.

GCC (GNU Compiler Collection) je sada kompilátorů vytvořených v rámci projektu GNU. Původně se jednalo pouze o překladač programovacího jazyka C (zkratka tehdy znamenala GNU C Compiler), později byly na stejném společném základě vytvořeny překladače jazyků C++, Fortran, Ada a další.[1]

GNU projekt byl založen v roce 1983 aby vytvořil kompletní operační systém UNIXového typu, který bude složen pouze ze svobodného softwaru: GNU systém.

GNU toolchain je souborné pojmenování kolekce nástrojů k programování vytvořených v rámci projektu GNU. Tyto nástroje se používají k vývoji aplikací a operačních systémů. GNU toolchain hraje zásadní roli ve vývoji Linuxového jádra a softwaru pro vestavěné systémy. Části GNU toolchainu jsou také přímo použity v nebo portovány do dalších platforem jako je Solaris, Mac OS X, Microsoft Windows (přes Cygwin a MinGW/MSYS) a Sony PlayStation 3.[2]

1 Dostupné na WWW: http://cs.wikipedia.org/wiki/GCC [cit. 2008-05-05]
2 Dostupné na WWW: http://cs.wikipedia.org/wiki/GNU_toolchain [cit. 2008-05-05]

GPIO (General Purpose Input/Output) jsou piny, které mohou být konfigurovány jako vstupy i výstupy. Hojně se jich využívá na poli embedded systémů.

OS (Operation System) je všeobecně používaná zkratka pro operační systém.

POSIX (Portable Operating System Interface) je přenositelné rozhraní pro operační systémy, standardizované jako IEEE 1003 a ISO/IEC 9945. Vychází ze systémů UNIX. Určuje, jak mají POSIX-konformní systémy vypadat, co mají umět, co se jak dělá apod. POSIX zahrnuje různé aspekty operačních systémů, např. správu procesů, práci se soubory, meziprocesovou komunikaci, základní programy (ed, awk, Korn Shell apod.), síťové záležitosti atd. Jedná se celkem o 15 dokumentů. GNU/Linux je od základu navržen podle POSIX, zajišťuje tedy dobrou přenositelnost z a na jiné systémy splňující tento standard.[3]

Preemptivní plánování je metoda, kterou využívá operační systém pro přidělování svého výpočetního času jednotlivým spuštěným procesům. Systém tak může předejít či zastavit právě naplánovanou úlohu ve prospěch úlohy s vyšší prioritou. Tato metoda byla poprvé vestavěna do linuxového jádra řady 2.6.

RPC (Remote Procedure Call) je protokol pro vzdálené volání procedur vyvinutý firmou Sun Microsystems. Je definován v RFC 1050 (verze 1) a RFC 1831 (verze 2). Každá RPC služba je jednoznačně určena svým číslem. Seznam služeb a jejich čísel se nachází v souboru /etc/rpc. O registraci služeb na straně serveru se stará portmapper (démon portmapd). RPC je využíváno hlavně v NFS.[4]

RTOS (Real Time Operation System) je operační systém schopný zpracovávat úlohy v reálném čase.

3 Dostupné na WWW: http://www.abclinuxu.cz/slovnik/posix [cit. 2008-05-09]
4 Dostupné na WWW: http://www.fi.muni.cz/~kas/p090/referaty/2001-podzim/nfs.2.html [cit. 2008-04-26]

UDP protokol (User Datagram Protocol) je jedním ze sady protokolů internetu. Bývá označován jako nespolehlivý. Na rozdíl od protokolu TCP nezaručuje, zda se přenášený datagram neztratí, zda se nezmění pořadí doručených datagramů nebo zda se některý datagram nedoručí vícekrát. Protokol UDP je vhodný pro použití, které vyžaduje jednoduchost nebo pro aplikace pracující systémem otázka - odpověď (např. DNS, sdílení souborů v LAN). Jeho bezstavovost je užitečná pro servery, které obsluhují mnoho klientů. Dále pro nasazení, kde se počítá se ztrátami datagramů a není vhodné, aby se ztrácel čas novým odesíláním (starých) nedoručených zpráv (např. VoIP, online hry).[5]

UTP (Unshielded Twisted Pair) je označení pro nestíněnou kroucenou dvoulinku. Jedná se o druh metalického vedení v sítích LAN.

5 Dostupné na WWW: http://cs.wikipedia.org/wiki/UDP [cit. 2008-04-24]

1 Úvod

Představte si možnost mít digitální domácnost. Když přijdete z práce, přivítá vás příjemná hudba. Topení bude nastavené na požadovanou teplotu. Světla se budou moci automaticky regulovat dle denní doby, anebo se samy pomalu zhasnou, když půjdete spát. Nebudete se muset bát, že se na plotně připálí oběd. Elektronické spotřebiče se budou moci dle vašeho přání samy zapínat a vypínat. Velká část z toho již není pouze hudbou budoucnosti.

Zajisté většina z nás užívá mnoho elektrických zařízení, která nám usnadňují život. Mnozí by si bez nich ani nedokázali představit život. Co byste řekli tomu, kdyby všechna tato zařízení nahradila jedna šikovná malá věc, která by měla veškeré požadované vlastnosti a ještě další navíc. Právě to mohou zajistit embedded systémy.

Moderní život s sebou přináší mnoho nároků a požadavků na nově vyvíjené technologie. Embedded systémy jsou schopny tyto požadavky plnit. Jejich hardwarová konfigurace se vyznačuje malými rozměry, energetickou nenáročností a možností je přizpůsobit a naprogramovat pro složité průmyslové aplikace i úkony každodenního života.

Embedded systémy jsou kombinací hardwaru i softwaru. Jedná se o autonomní počítačové systémy, jejichž smyslem je řídit ostatní systémy, zařízení či externí procesy. Jsou součástí mnoha běžně dostupných zařízení, se kterými každodenně přicházíme do styku. Lze mezi ně zařadit hračky, hodinky, mobily, dokonce i řídící systémy jaderných elektráren. Klasickým příkladem, který každý určitě zná je PDA zařízení. Technologie embedded systémů je v současné době jedním z nejrychleji se rozvíjejících technických odvětví.

Přes všechny klady, které zde byli zmíněny, mají Embedded systémy i jistá omezení. Jsou proto optimalizovány pro vykonávání specifických úkolů, aby bylo dosaženo lepších výsledků. Jako nadstavba se pro ně užívají různé operační systémy, mezi které patří například i Linux. Ten se využívá kvůli efektivnímu spravování přidělených zdrojů, což umožňuje dosažení vynikajících rychlostí i u slabších počítačů a možnosti postavení systému přesně na míru daným potřebám.

2 Modul X-board PXA255 a vývojová deska

Obr. 1: X-board PXA255 modul[6]

X-board PXA255 je embedded modulem vyrobeným společností Kontron. Jeho srdce je tvořeno procesorem Intel PXA255. Jedná se o malé zařízení, které může nabídnout hodně možností využití. Například procesor Intel X-Scale PXA255, kterým je deska osazena, byl vyvinut jako řídící modul pro PDA (Personal Digital Assistant) zařízení.

Kontron dodává modul společně s vývojovou deskou, displejem a dalšími sounáležitostmi pro jejich zdárné propojení a uvedení do provozu. Dalo by se říci, že je k dostání jakýsi minipočítač, který lze využít v rámci testování různých výpočetních aplikací. Další možností je modul rovnou zařadit jako řídící prvek nějakého procesu, kde je potřeba zobrazovat data s možností nadále s nimi pracovat. X-board nabízí opravdu široké možnosti jeho využití. Díky množství aplikací pro embedded systémy ho lze využít v telekomunikacích, jako již zmíněný prostředek pro testování, provádění výpočtů a jiných operací. Můžeme se s ním také setkat uvnitř kokpitů letadel, kde bývá využíván pro nezbytné činnosti spojené s řízením provozu letadla, dále ve výtazích a dalších prostředcích všedního života.

Systémy založené na X-board PXA modulech bývají využívány zejména tam,

6 Starterkit for X-boardTM <PXA> Document Revision 1.0. s. 6. Dostupné na WWW: http://www.mite.cz/xboardpxastarterkitfiles/XBD3K110.pdf [cit. 2007-12-15]

kde se místo stává nadstandardním požadavkem, kde je nutno využít malý a spolehlivý systém s možností připojení na TFT displej s velmi nízkými energetickými nároky na spotřebu.

2.1 Embedded systém

Embedded systém (vestavěné, zabudované zařízení) je jednoúčelovým systémem, ve kterém je řídicí počítač zcela zabudován do zařízení, které ovládá. Na rozdíl od všeobecně použitelného počítače (například osobního) jsou embedded zařízení navržena pro předem definované činnosti. Vzhledem k tomu, že systém je vyvíjen pro konkrétní účel, mohou tvůrci svůj návrh plně optimalizovat pro určené aplikace a tak i podstatně snížit cenu výrobku. Některé ze systémů pracují v reálném čase. U těchto zařízení zpoždění činnosti nebo akce ovládané řídícím procesorem může vyústit ve fatální následky nebo poruchu činnosti jako například k blokaci motoru.

Počítače do dlaně (PDA), mobilní digitální pomocníci (MDA) a inteligentní mobilní telefony jsou také často označovány jako embedded zařízení vzhledem k vlastnostem hardwaru i přes to, že z hlediska softwaru jsou rozšiřitelné a všeobecně použitelné podobně jako osobní počítače.

U systémů vyráběných ve velkých sériích jako jsou například mp3 přehrávače či mobilní telefony je většinou jedním z primárních cílů ve fázi návrhu nízká cena. Tvůrci těchto zařízení často používají hardware, který je přesně dostačující pro implementaci požadovaných funkcí. Například digitální přijímač DVB-S pro příjem satelitní televize zpracovává každou sekundu obrovské množství dat, zpracování je ovšem provedeno v zákaznickém integrovaném obvodu, který provádí pouze tuto jedinou činnost. Hlavní procesor pouze nastaví a spustí tento proces, případně zobrazují menu či animace, proto stačí jeho menší výpočetní výkon.

Software určený pro embedded systémy je často označován jako firmware a je uložen v ROM nebo FLASH čipech na rozdíl od programů v osobním počítači, uložených na pevném disku. Tento software často počítá s omezenými prostředky zařízení – malá nebo žádná klávesnice, omezený nebo žádný displej, malé množství paměti RAM a podobně.

Embedded systémy jsou vestavěny do zařízení, od nichž se očekává, že budou schopny pracovat léta bez chyb a v některých případech se samy zotaví z poruchy. To umožňuje technika watchdog, která restartuje počítač pokud program po jistou dobu neupozornil, že je v pořádku. Programy bývají většinou vyvíjeny a testovány pečlivěji než programy pro osobní počítače. V návrhu se nepoužívají nespolehlivé mechanické součástky jako disky, přepínače, tlačítka.

2.2 Technické specifikace X-board

Obr. 2: Starterkit pro X-board <PXA>[7]

Starterkit je názvem pro celý balíček, který lze obdržet od společnosti Kontron. Obsahuje veškeré potřebné náležitosti, aby po zapojení a nastavení bylo možné spustit systém a začít s vývojem a testováním. Sestavu tvoří X-board PXA255 modul, vývojová deska. Pro ní je optimalizován a nainstalován operační systém Windows CE,

7 Starterkit for X-boardTM <PXA> Document Revision 1.0. s. 4. Dostupné na WWW: http://www.mite.cz/xboardpxastarterkitfiles/XBD3K110.pdf [cit. 2007-12-15]

který lze najít na přiložené compact flash kartě. Důležitou součástí sestavy je 5,7“ široký TFT display, který po zprovoznění slouží jako displej počítače.

Deska tohoto počítače je osazena procesorem Intel Xscale PXA255 o frekvenci 400MHz, který je založen na platformě procesorů typu ARM. Existuje i slabší verze o taktu 200MHz. Výhodou těchto procesorů je jejich energetická nenáročnost a kompaktní velikost. Pro jeho napájení je potřeba zdroj, který mu bude schopný dodávat 1,5 W. Přestože modul dosahuje velikosti kreditní karty, nabízí velmi širokou paletu vestavěných zařízení. Jedná se o integrovaný grafický čip, 10/100Mb Ethernet pro připojení k internetu, možnost zprovoznění až tří USB portů, dva RS232 porty a integrovanou AC97 zvukovou kartu. Paměťové moduly typu DRAM (64 MB) a Flash (32 MB), kde je uložen nepostradatelný systémový zavaděč, dávají X-boardu plnou způsobilost ke spouštění a provozování různých aplikací, které zde bude možno využít.

Modul je uzpůsoben pro práci s více operačními systémy, závisí pouze na uživateli, který si zvolí. Na výběr je možnost využít již předinstalovaný OS Microsoft WinCE, dále Linuxovou embedded distribuci, popřípadě systémy WxWorks a QNX.

Tab. 1: Technická specifikace X-board PXA255

Technická specifikace X-board PXA255	
CPU	Intel® XScale™ PXA255 400 MHz
pracovní paměť	32 / 64 Mbyte DRAM na desce
USB	3 x USB 1.1 porty OHCI (2 master a 1 client rozhraní)
Flash	16 / 32 Mbyte na desce (s vestavěným bootloaderem)
Audio	AC'97 rozhraní kompatibilní s AC'97 verze 2.0
Grafika	LCD s různými výstupy (DSTN a TFT)
Rozlišení	Rozlišení LCD: 800 x 600 Pixel 16 Bit / pixel
Sériová komunikace	2 x Sériový port TTL (vysokorychlostní COM, jeden s podporou Bluetooth)
Ethernet	10/100 Base-T Realtek RTL8100BL řadič
Standardní výbava	několik GPIO pinů (sdílené s IrDA, SSP a COM3)
	rozhraní MultiMediaCard (MMC)
	JTAG rozhraní pro snažší odstraňování chyb
Doplňkové vybavení	Rošiřující GPIO porty pro multimediální kartu
Pevný disk	1x IDE rozhraní
Rozměry	68 x 49 x 8 mm (2,6 x 1,9 x 0,3“)
Energetická spotřeba	Nízská energetická spotřeba: typ. 1,5 W @ 3.3V (očekávaná)
Teplota	Provoz: od 0°C do 70 °C
	Mimo provoz: od -10°C do +85 °C
Vlhkost	V provozu: od 10% do 90%
	Mino provoz: od 5% do 95% (nekondenzující)
Operační Systém	WinCE, Linux, WxWorks, QNX

2.3 Kontron

Kontron je původem německá společnost, která se řadí k předním světovým firmám ve vývoji a prodeji embedded systémů. Hlavní zaměření společnosti je orientováno na vývoj embedded systémů a aplikací pro sféru komunikačních technologií, automatizace, dopravy, medicíny, armádního využití, leteckého průmyslu, testovacích a měřících odvětví. Kontron zaměstnává více než jeden a půl tisíce lidí. Její pobočky a výrobny můžeme nalézt v mnoha státech po celém světě od Evropy, severní Ameriky až po Asii.

3 Výběr vhodné linuxové distribuce pro modul PXA

Linux se převážně používá jako jádro GNU/Linuxových distribucí. Ty jsou sestavovány jednotlivci, týmy dobrovolníků i komerčními firmami za cílem finančního zisku. Typická distribuce zahrnuje jádro, systémový a aplikační software spolu s prostředky, jak celý systém nainstalovat.

3.1 Požadavky na systém

Klasické linuxové distribuce pro stolní počítače a servery, jako je SUSE, Fedora, Debian a další, jsou svojí stavbou velice rozsáhlé, jelikož jsou vytvářeny dosti univerzálně se zaměřením na uživatele. Obsahující velké množství aplikací, knihoven, dokumentace a jiných věcí, které jsou pro embedded systémy nepotřebné či dokonce nepoužitelné. Vyznačují se značnou náročností na výkon hardwaru, což způsobuje jak zmíněné množství aplikací, tak grafické uživatelské rozhraní, které jim dodává příjemnější uživatelské rozhraní.

Embedded operační systémy jsou dosti odlišné od stolních a serverových verzí Linuxu. Jsou vytvářeny pro zařízení s relativně omezenými zdroji, jako je menší kapacita pamětí RAM a ještě o poznání menšími sekundárními pamětmi. Namísto pevného disku bývají u Embedded systémů využívány paměti typu flash, které jsou energeticky nenáročné, stabilní, ale mají menší kapacitu v porovnání s pevnými disky. Podle toho bývají upravovány operační systémy a aplikace, aby byli co nejmenší. Používá se pouze to nejnutnější, co systém musí umět. Embedded Linux je většinou vytvářen pro jeden specifický hardware s aplikacemi, které jsou požadovány. Umožňuje to vysokou míru optimalizace systému pro dané zařízení, což splňuje první požadavek pro následující vytvoření real-time operačního systému.

Požadavky na OS pro X-board:

- podpora procesoru Intel PXA255, popřípadě platformy ARM X-Scale
- podpora pro baseboard od Kontronu

- Linux kernel řady 2.6
- cena operačního systému (nejlépe zdarma)
- real-time systém, nebo možnost rozšíření systému o real-time nadstavbu

3.2 Vytvoření vlastní distribuce

Možnost vytvoření vlastní distribuce je velikou předností systémů založených na GNU/Linux. Umožňuje vývojáři postavit systém, který bude přesně „šitý“ dle jeho potřeb a přání. Má volnou ruku ve výběru jádra, přidružených balíčků a aplikací, které bude chtít využívat. Je to také jedna z mála možností, jak získat systém pro embedded zařízení úplně zdarma. Vytvoření vlastní distribuce je dosti složitý a časově náročný proces, ovšem dozvídáme se mnoho o vnitřním fungování a strukturách operačního systému.

Typická distribuce k obecnému použití obsahuje linuxové jádro, knihovny a nástroje, příkazové shelly a tisíce dalších balíčků aplikačního softwaru, jako jsou kancelářské balíky a grafické prostředí X Window System přes kompilátory různých jazyků, textové editory a vědecké nástroje. Toto je nutné vzít a předělat. Existuje více možností, jak vytvořit Linuxovou distribuci. Lze začít od úplného počátku, kdy se stáhne samotné jádro, na kterém postupně vystavíme systém. Lze použít některý z programů, které byly vytvořeny právě pro tyto případy.

3.2.1 Vytvoření vlastního systému

Linuxový systém je vždy založen na jádře. Je tedy nutné ho nejdříve vybrat. Pro různé platformy a zařízení existují patche jádra, které obsahují potřebná nastavení a rozšíření o real-time podporu. Pro jádro lze získat i konfigurační soubor .config obsahující nastavení kernelu. Následuje vytvoření root filesystému, který musí vždy obsahovat nezbytné součásti pro spuštění systému.

Minimální požadavky filesystému:

- adresáře: /dev, /proc, /bin, /etc, /lib, /usr, /tmp
- potřebné utility: sh, ls, cp, mv, atd.
- konfigurační soubory: rc, inittab, fstab, atd.
- zařízení: /dev/hd*, /dev/tty*, /dev/fd0
- runtime knihovna pro zajištění funkcí užívaných aplikacemi

3.2.2 Buildroot

Buildroot je nástroj obsahující makefily a patche, které umí vygenerovat souborový filesystém i cross-kompilační toolchain pro cílový systém. Je samozřejmostí, že do samotné kompilace lze přidat i jádro systému, které lze později upravovat. Buildroot je tedy schopen vytvořit plnohodnotný operační systém pro počítače různých platforem. Umožňuje kompilovat systém pro obvyklé procesory x86 i procesory ARM. Majoritní využití tohoto programu spadá do oblasti práce s embedded systémy.

Toolchain je balíčkem obsahujícím veškeré nástroje, které umožňují kompilaci kódů do vytvořeného systému. Skládá se z překladače (nejčastěji gcc), binárních nástrojů jako assembler a linker a samozřejmě nutných standartních knihoven jazyka C.

Na podobném způsobu, jakým pracuje Buildroot, jsou postaveny i další nástroje, které byly taktéž vytvořeny pro práci s embeded systémy. Jedná se například o Crosstool a Makerootfs, který je vytvořen přímo pro embedded systémy založené na ARM procesorech.

3.3 Komerční distribuce

Použití komerční distribuce je jednodušší možností. Jedná se o systémy, které bývají, až na výjimky, vytvářeny pro komerční sféru využití. Znamená to, že na nich pracují odborníci, kteří staví systém na zakázku dle potřeb zákazníka. S dostatkem peněz není problém získat distribuci podle představ. Je nutné se připravit na to, že požadovaná cena bude dosti vysoká. Mnoho předních společností, ne jenom ty

zabývající se embedded systémy, má ve svém programu podporu univerzit a jejich studentů. Vycházejí z jednoduché filozofie. Univerzitě poskytnou nějaký systém či zařízení úplně zdarma, nebo za minimální poplatek. Na oplátku může univerzita produkt využít pro vývoj, z čehož společnost nadále těží. Dále získává do svých řad schopné zaměstnance ze sféry studentů. Důležitým kritériem je i propagace firmy.

3.3.1 Linux Kontron

Kontron dodává k modulu vlastní operační systém. Jelikož je přímo vytvořen společností pro jejich X-board, nabízí nejlepší možnost kompatibility s modulem. Je zabezpečena veškerá komunikace s hardwarovými zařízeními počítače. Neměly by nastat problémy v rámci nefunkčnosti nějakých nainstalovaných součástí. Nejnovější verze, kterou Kontron dodal k použití, je vytvořena na kernelu 2.6.21.5. Systém je velmi stabilní, avšak nabízí pouze omezené možnosti. Systém není přímo real-timeový, ale jádro podporuje preemptivní plánování.

3.3.2 BlueCat embedded Linux

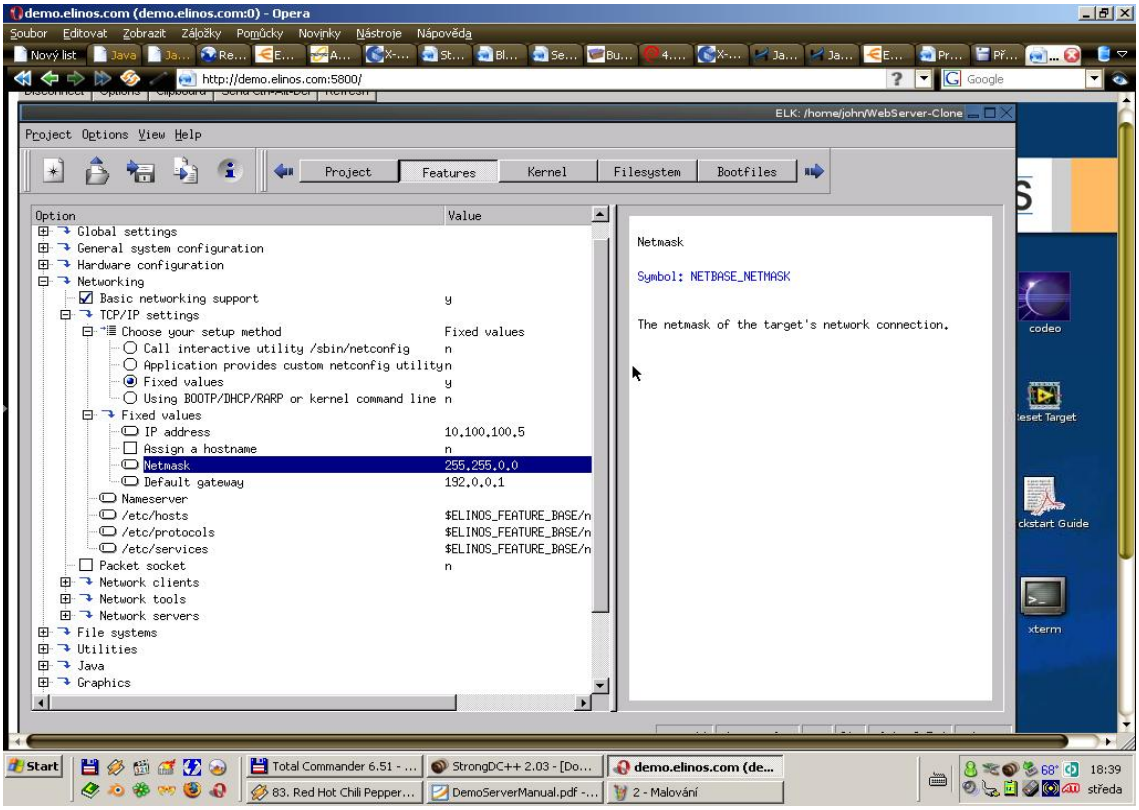
BlueCat Linux je jednou z možných distribucí použitelných pro tuto práci. Systém je přímo navržen pro embedded zařízení. Obsahuje předkompilované balíčky, které umožňují využít vlastnosti procesorů Intel PXA255. Srdcem distribuce je kernel verze 2.6, která pracuje na bázi technologie preemptivního plánování procesů. Systém je hard real-timeový. BlueCat Linux se bohužel nepodařilo získat.

3.3.3 ELinOS

ELinOS je produkt vytvořený německou společností SYSGO Real-Time Solutions GMBH. Nejedná se o operační systém, ale o balíček sloužící k vytváření Linuxových embedded systémů pro průmyslové aplikace. Nová verze ElinOS 4.2 nabízí možnost pracovat s kernely 2.4.31, 2.6.15 a 2.6.22. Veškeré podporované hardwarové

platformy nemají možnost použít nejnovější jádro. Balíček je doplněn o hard real-time rozšíření založené na PikeOS technologi. Podporovaný hardware je doplněn řadou nových BSP vytvořených a certifikovaných ve spolupráci se světovými výrobci hardwaru jako Freescale, AMD, Digital Logic, českého výrobce Teco a.s. a dalších. Důležitá je široká podpora různých platforem, pro tuto práci zejména ARM procesorů.

Vytváření samotných systémů probíhá v uživatelsky příjemném prostředí, které zajišťuje grafický konfigurátor ELK. První možnost ozkoušet si vlastnost systému a konfigurátoru společnost nabídla ihned z počátku. Po zaregistrování byl zpřístupněn internetový server s demoverzí ELinOSu.



Obr. 3: Demoverze ELinOS

Zde je k dispozici jednoduchý tutoriál pro vytvoření zkušební verze systému. Díky demoverzi ELinOSu si lze vyzkoušet práci s konfigurátorem, která vyústí kompilací vytvořeného systému. Vše v příjemném prostředí na vzdáleném serveru.

Zástupci společnosti nám po zkontaktování vyšli vstříc. Přestože se jedná o německou společnost, má v Čechách silné zastoupení a dokonce své sídlo. Ujal se nás

jednatel Jan Rolo, který se s námi setkal a během pár týdnů zprostředkoval smlouvu na bezplatné získání systému pro Technickou univerzitu v Liberci. Naneštěstí samotné převzetí systému se dosti protáhlo. Tuto část mělo na starost německé vedení firmy a díky personálním změnám trvalo přes tři měsíce, než jsme systém obdrželi.

4 Real-Time

Výraz real-time, doslovně přeložen jako reálný (opravdový) čas, je nejčastěji spojován s operačními systémy. Real-time operační systémy jsou takové systémy, které okamžitě reagují na vstupující událost. Jsou využívány například pro navigaci, kde musí počítač reagovat na stálý tok nových informací a to bez jakéhokoliv přerušení. Pojem reálného času je samozřejmě relativní. Při řízení rychlých procesů může jít o dobu odezvy v řádu mikrosekund, která je běžnými softwarovými prostředky nerealizovatelná. Normální operační systémy proto nebývají real-timeové a odezva na událost se u nich může pohybovat v sekundách, dokonce i minutách. Linux je právě systémem, který trpí nedostatkem podpory reálného času. V real-time systémech se klade silný důraz na dobrou správu časování a přerušení. K zajištění správnosti časování je nutné provést změny na úrovni funkcí jádra, např. změnit zacházení s přerušeními nebo s metodami plánování. Takto lze získat real-time platformu s nízkou prodlevou a možností plnit náročné požadavky na předvídatelnost v plném ne-real-time prostředí Linuxu (přístup k TCP/IP, grafické zobrazení a okenní systémy, souborové a databázové systémy atd.).

Real-time má obecně široké zastoupení u embedded systémů. Jejich použití bývá často závislé na rychlosti vykonání aplikací. Provedení příkazů a vyhodnocení výsledků musí proběhnout jako prioritní operace v nejkratším možném čase. Například příliš dlouhá odezva u brzdového systému, nebo u řídících programů letadla by mohla vyvolat katastrofální následky. Dalším příkladem může být ovládání jednoduchého robota řízeného real-time systémem, který zajistí, že uživatelovy příkazy budou provedeny okamžitě.

4.1 Dělení real-time OS

Real-time operační systémy lze rozdělit do dvou základních skupin:

1) Hard real time OS je systém, kde by případné nedodržení časových limitů mělo katastrofální následky. Požadavek na stanovení času průběhu reakce je absolutní.

Určení průběhu zajišťuje deterministické chování systému, kdy pro danou operaci lze předem určit její průběh. U RTOS je kladen důraz na preemptivní plánovač, velký počet nastavitelných priorit vláken a přesné hodiny reálného času.

2) Soft real time OS, kde je možné dovolit drobné odchylky v reakcích.

4.2 Linuxové real-time systémy

Mezi nejpoužívanější real-time systémy patří RTAI, Xenomai, KURT, VxWorks a RTLINUX .

4.2.1 RTLinux

RTLinux je komerční linuxová real-time modifikace označovaná jako hard real-time varianta Linuxu, která umožňuje ovládání robotů, data pořizujících systémů, zařízení citlivých na čas, výrobních dílen a dalších nástrojů. Kromě úpravy jádra přináší sadu vlastních nástrojů a modulů, které dodatečně rozšiřují možnosti jádra. Snaží se co nejvíc přiblížit POSIX 1003.1 Real-Time standard specifikaci. [8]

4.2.2 RTAI/fusion

RTAI/fusion je v podstatě přídavný modul, který využívá služeb HAL (Hardware Abstraction Layer), což je abstraktní vrstva nad hardwarem. Nejedná se o real-time operační systém, jako například VXworks nebo QNX. Je založen na linuxovém jádře a poskytuje možnost udělat jej plně preemptivním. Snaží se o přesunutí běhu procesů z prostoru jádra do uživatelského prostoru. RTAI/fusion je založen na projektu Xenomai.

Xenomai je real-time systém vytvořený pro rozšíření Linuxového jádra, za účelem zajistit hard real-time podporu pro uživatelské aplikace. Vychází z architektury použití RTOS jádra.[9]

8 PELČÁK, V. *Real-time modifikace Linuxu.* Abíčko, březen 2006, s. 23-35.
9 PELČÁK, V. *Real-time modifikace Linuxu.* Abíčko, březen 2006, s. 23-35.

4.2.3 VxWorks 6.x

VxWorks je operační systém pro řízení v reálném čase, který se vyznačuje real-time mikrojádrem. Toto jádro zahrnuje většinu nástrojů pro podporu reálného času. Řadí se k nejrozšířenějším operačním systémům reálného času zejména v oblasti průmyslových aplikací embbended systémů. Mezi základní charakteristiky patří neomezený počet procesů/úloh, preemptivní plánování, rychlá a flexibilní meziprocesová komunikace, dědění priorit, atd. Systém není POSIX kompatibilní.[10]

4.2.4 KURT

KURT Linux byl vyvinut v Information and telecommunication technology center na Kansaské Univerzitě. Je to real-time modifikace linuxového OS, která umožňuje plánování real-time událostí s rozlišením na 10 mikrosekund. Kurt může pracovat ve dvou módech: Focuss mód umožňuje spouštění pouze real-time procesů; a smíšený mód, který má stále nadřazené provedení real-time procesů, ale umožňuje spouštět i ostatní procesy.

10 Dostupné na WWW: http://cs.wikipedia.org/wiki/Real-time_operating_system [cit. 2008-05-09]

5 Realizace propojení

Prvním cílem práce je zavést Linuxový systém na modul PXA. Jakým způsobem toho docílit? Jediná možnost je vytvořit síťové spojení, které umožní stažení jádra a zavedení filesystému pro modul. Notebook Acer Aspire 3022LMi byl použit jako server pro navázání spojení s X-boardem a následné poskytnutí operačního systému.

Acer Aspire 3022LMi:

procesor :	AMD Mobile Sempron 2800+ (1.6 GHz)
paměť:	512 MB DDR RAM
pevný disk :	HDD 20 GB
ethernet :	1GB LAN
operační systém :	Ubuntu 7.10 – Gutsy Gibbon

Propojení cílového a vývojového počítače je zabezpečeno pomocí Ethernetu a rozhraní RS232 (sériové porty). Sériový kabel se používá pro vytvoření komunikace a základní ovládání mezi deskou a PC. Nejdůležitější vlastností je, že na X-boardu zpřístupní bootloader, nebo-li systémový zavaděč. Na desce X-boardu lze najít dva konektory RS232. Pro propojení je využit COM2, který umožňuje zpřístupnění systémového zavaděče. Síťový kabel je použit pro načtení jádra a zpřístupnění filesystému.

Notebook není vybaven RS232 portem potřebným k propojení. Byl použit převodník USB-Serial od společnosti Prolific.

Přes správné propojení počítačů může nastat situace, kdy nebude možné navázat komunikaci. Při pokusech zavádět systém jsem se setkal s neschopností přidělit cílovému počítači adresu pomocí DHCP, přestože vše bylo nastaveno a zapojeno dle předepsaných parametrů. Problém byl v síťové kartě notebooku. Zaváděcí sekvence

bootloaderu se spouštějí neprodleně při startu cílového počítače. Pro vytvoření spojení je tedy nutné modul restartovat. Notebooky v rámci energetické úspory mají defaultně nastaveno vypínání některých zařízení, pokud nejsou zrovna funkční. To se vztahuje i na síťovou kartu. Pokaždé s restartováním X-boardu se zároveň vypínala a zapínala i síťová karta. Nebylo tudíž možné předat parametry definované DHCP serverem a systémy propojit do sítě. Pro vyřešení tohoto problému stačí mezi zařízení přidat switch, nebo jiný aktivní síťový prvek, aby se zamezilo vypínání síťové karty notebooku.

Ačkoliv se řešení problému zdá být dosti jednoduché, trvalo velmi dlouho, než jsme byli schopní se přes tento problém dostat.

6 Nastavení systémů

6.1 Příprava vývojového systému

Požadavky na vývojový systém:

- Software: Linuxová distribuce, která je přizpůsobena pro X-board.
- Vývojový hardware: jakýkoliv počítač založený na platformě x86 s alespoň 10 GB místa na disku.

Pro vývojový počítač bylo nutné vybrat vhodnou linuxovou distribuci, na níž bude probíhat celá práce. Postupně byly vyzkoušeny Linuxové systémy Suse 10.3, Fedora 8, Mandriva Linux One 2008 a nakonec Ubuntu 7.10 Gutsy Gibbon. Vybrána byla poslední jmenovaná distribuce díky dobrému pracovnímu prostředí a široké internetové podpoře.

Ačkoliv všechny tyto operační systémy patří pod Linux, práce s nimi bývá často v různých aspektech odlišná. Základní příkazy jsou většinou podobné, přesto prvotní orientace v prostředí nového systému může být matoucí.

Nezbytné součásti pro vývojový systém:

- nainstalovaný Linux s vytvořeným uživatelským účtem, určeným pro vývoj systému;
- podpora pro BOOTP, DHCPD a NFS;
- balíček „sudo“ zajišťující rozšíření práv pro vývojový účet;
- balíčky jazyka C;
- “ncourses“ balíček;
- X-Windows systém (pro grafickou konfiguraci jádra);
- “qt“ balíček pro možnost grafické konfigurace.

Jedná se o základní části, které musí systém vždy obsahovat. Pro každou distribuci Linuxu budou stejné. Dále bude popisována přímo práce s OS Ubuntu 7.10.

6.2 Nastavení BOOTP/DHCP, TFTP a NFS serverů

Veškeré potřebné součásti systému pro X-board byly již vytvořeny. Nyní je nutno nastavit systém, aby se mohl spojit s cílovým počítačem, poskytnout mu obraz jádra a nakonec zpřístupnit root filesystém.

6.2.1 DHCP server

Pro začlenění prvku do sítě použijeme DHCP server. DHCP vznikl rozšířením BOOTP protokolu a je s ním zpětně kompatibilní. DHCP server je třeba nakonfigurovat na vývojovém počítači a povolit mu přidělovat IP adresu BOOTP klientovi. Nejdříve je nutné nainstalovat příslušný balíček, který danou službu zpřístupní. Tím je *dhcp3-server*. Instalace se provede zadáním příkazu do konzole:

```
$ apt-get install dhcp3-server
```

Server je nutné ještě nakonfigurovat. S instalací vznikl adresář */etc/dhcp3/* ve kterém je soubor *./dhcpd.conf,* který nastavení umožní. Obsah souboru je textového charakteru a určuje, co bude server dělat, na jakém zařízení bude naslouchat, jaké služby použije, jak bude přidělovat IP adresy atd. Soubor po instalaci obsahuje vzorové nastavení, které je dobré si pročíst, pro získání základního přehledu o možnostech serveru. Pro pozdější zlepšení orientace při ladění je dobré původní obsah souboru vymazat.

Pro správnou komunikaci DHCP serveru s X-boardem by měl soubor *./dhcpd.conf* obsahovat následující řádky:

```
allow bootp;
        ddns-update-style none;
        ddns-updates off;
        subnet 192.168.0.0 netmask 255.255.255.0 {
              default-lease-time 1209600;
              max-lease-time 31557600;
              host nov {
                    hardware ethernet 00:00:60:DA:E3:AB;
                   fixed-address 192.168.0.10;
                    server-name "pandora";
                    filename "vmlinuz";
                    option root-path "/tftpboot";
                       }
                 }
```

Co znamenají jednotlivé vstupy?

Každá řádka v souboru má svůj specifický význam a určuje různé možnosti, jak server použít. Je důležité pochopit jednotlivé příkazy pro DHCP, aby bylo možné server správně a efektivně používat. Konfigurování souboru /etc/dhcp3/dhcpd.conf se vzdáleně podobá jednoduchému programování. To umožňuje velké množství příkazů a možností nastavení.

Důležitou částí textu je *host nov { }*. V jejích závorkách je uvozena část, kde je definováno hlavní nastavení pro cílový počítač. Slovíčko *nov* určuje, jak byla pojmenována cílová stanice, jiný význam nemá.

Význam parametrů nastavení:

“lease time“

Kromě IP adresy server eviduje také čas, po který bude IP adresa klientovi poskytnuta. Poté co vyprší, smí server adresu přidělovat jiným klientům.

“hardware Ethernet”

Tímto příkazem je určena MAC adresa připojeného cílového počítače. Adresu lze získat během nastavování bootloaderu, který při svém spuštění vypisuje hardwarové prvky počítače. V tomto případě se adresa získá z výpisu konzole minicomu při zadání příkazu “t“ v nastavení zavaděče. Zvolením tohoto parametru dojde ke spuštění bootovací sekvence a vypíše se MAC adresa příslušného zařízení. Řádek, který hledáme, by měl vypadat následovně:

RTL8139Init:: MAC = 00-00-60-DA-E3-AB

“fixed-address”

Určuje statickou IP adresu, která bude přidělena během bootovací sekvence cílovému počítači. DHCP většinou nepoužívá statické přidělování, nýbrž dynamické, kdy pro skupinu síťových stanic vymezí nějaké pásmo IP adres a ty jim posléze přiděluje. V tomto případě by bylo zbytečné využívat dynamického přidělování adres, jelikož je k dispozici pouze jedna stanice, na které se pracuje.

“server-name”

Určuje jméno (pokud je DNS server dostupný), nebo IP adresu TFTP serveru, ke kterému by se měl cílový počítač připojit. Jedná se o nastavení jména vývojového počítače.

“filename”

Touto položkou lze určit cestu k jádru, která má být zavedena na X-board. Kernel bývá obvykle v podobě souboru zImage, popřípadě vmlinuz, který je umístěn v základním adresáři pro TFTP server (/tftpboot).

“option root-path”

Určuje umístění cílového root adresáře, který lze najít na vývojovém počítači. V tomto adresáři jsou umístěny veškeré potřebné soubory pro nabootování do cílového počítače, který si tento adresář připojí jako jeho root filesystém pomocí služby NFS.

Nyní lze přistoupit k samostatnému puštění serveru. Samotná konfigurace souboru */etc/dhcp3/dhcpd.conf* ke spuštění nestačí. DHCP je služba, která musí běžet jako daemon. Musíme ho spustit manuálně nebo jako součást nějakého spouštěcího skriptu. U serverů je smysluplnější druhá varianta, které lze dosáhnout zadáním následujícího příkazu do konzole:

```
$ /etc/init.d/dhcp3-server start
```

Výpis oznámí, že spuštění serveru proběhlo úspěšně:

```
root@pandora:/etc/dhcp3# /etc/init.d/dhcp3-server start
 * Starting DHCP server dhcpd3                          [ OK ]
root@pandora:/etc/dhcp3#
```

Po prvním nastavení existuje dosti velká šance, že tohoto výpisu se nedočkáme. V případě špatného nastavení souboru */etc/dhcp3/dhcpd.conf* skript vypíše chybové hlášení. Díky zprávě by mělo být možné danou chybu odstranit, jelikož výpis obsahuje číslo řádku, ve kterém je chyba a přímo chybu, která nedovoluje službu spustit.

Spuštění serveru bylo provedeno příkazem */etc/init.d/dhcp3-server start*.

Možné příkazy pro práci se službou:

start	spuštění serveru
stop	vypnutí služby
restart	nutné provést po každé změně konfiguračního souboru */etc/dhcp3/dhcpd.conf*, aby se provedli dané změny

force-reload	v případě, že nelze z nějakých důvodů znovu načíst (restartovat) službu
status	ukáže, v jakém stavu se server nachází

Nyní lze ověřit, jestli je DHCP server správně nastaven a spuštěn. Při spuštění procesu bootování (parametrem "t" v konzoli minicomu) by měla být přijata nastavená IP adresa cílového počítače. Pokud se tak stalo, program vypíše správnou adresu. DHCP server je správně nastaven a je spuštěný.

6.2.1.1 Co je to DHCP

Dynamic Host Configuration Protocol je aplikační protokol z rodiny TCP/IP. Používá se pro automatické přidělování IP adres koncovým stanicím v síti.

Současně s IP adresou posílá server stanicím (klientům) další nastavení potřebná pro používání sítě jako je adresa nejbližšího směrovače (default gateway), masku sítě, adresy DNS serverů. Ve větších sítích se správce někdy rozhodne posílat i adresy doporučených NTP, WINS, SMTP serverů, stáhnutí ARP cache a jiné. Navíc je možné definovat i uživatelské parametry. Části kódu, kterým klient nerozumí, ignoruje.

DHCP protokol přináší několik výhod:

- uživatelé si na počítači v souvislosti s připojením k síti nemusí nic nastavovat
- zaručuje, že se na síti nevyskytnou dvě stejné IP adresy (tzv. konflikt IP adres)
- správce sítě může „přečíslovat" síť nebo změnit vlastnosti sítě s minimálním zásahem do práce uživatelů

DHCP protokol je rozšířením staršího BOOTP protokolu, který přiděloval IP adresy na neomezenou dobu. DHCP je s BOOTP obousměrně kompatibilní. Znamená to, že DHCP klienti dovedou získat nastavení z BOOTP serveru a DHCP server může přidělit IP adresu BOOTP klientovi. Zde je nutná opatrnost, protože BOOTP klient bude již jednou přidělenou IP adresu používat navždy.

6.2.1.2 Co se děje na síti

Klienti žádají server o IP adresu. Ten u každého klienta eviduje půjčenou IP adresu a čas, dokdy ji klient smí používat (doba zapůjčení, angl. lease time). Po vypršení může server adresu přidělovat jiným klientům. Klient komunikuje na UDP portu 68, server naslouchá na UDP portu 67.

Po připojení do sítě vyšle klient broadcastem DHCPDISCOVER paket. Na ten odpoví DHCP server paketem DHCPOFFER s nabídkou IP adresy. Klient si z několika nabídek vybere jednu IP adresu a o tu požádá paketem DHCPREQUEST. Server mu ji vzápětí potvrdí odpovědí DHCPACK. Jakmile klient obdrží DHCPACK, může už IP adresu a zbylá nastavení používat. Klient musí před uplynutím doby zapůjčení z DHCPACK obnovit svou IP adresu. Pokud lhůta uplyne aniž by dostal nové potvrzení, klient musí IP adresu přestat používat.

Protokol definuje roli i tzv. DHCP relay agenta. Používá se v situaci, kdy existují dvě nebo více sítí oddělené směrovačem a jen jedna síť má server. V takovém případě správce na směrovači zapne relay agenta a nastaví jej tak, aby všesměrové (broadcast) DHCP dotazy ze sítí bez DHCP serveru přeposílal do té sítě, která ho má. Agent k přeposílanému dotazu přidá masku sítě, kde klienta zaslechl, aby DHCP server poznal, ze kterého adresního rozsahu má klientovi adresu přiřadit.[11]

6.2.2 TFTP server

Trivial File Transfer Protokol (TFTP) slouží k poskytnutí obrazu jádra klientskému počítači. Přenos probíhá po ethernetu. Pro jeho spuštění je nezbytné nainstalovat příslušný systémový balíček pro Ubuntu. Insatlace se provede po zadání příkazu:

```
$ apt-get install tftpd-hpa
```

Tímto způsobem je použit pro běh tftp serveru klient *tftpd-hpa.* Bylo by možné použít i program *xinetd.* Pro tuto práci jsem však zvolil tftpd-hpa.

11 Dostupné na WWW: http://cs.wikipedia.org/wiki/DHCP [cit. 2008-04-17]

Pro distribuce Debian/Ubuntu je vlastní, že konfiguračním souborem pro TFTP server je soubor */etc/inetd.conf*. V něm je nastaveno, že jako výchozí adresář bude použit */var/lib/tftpboot.* Je dobré tuto cestu změnit na adresář */tftpboot,* do něhož lze vytvořit symbolický odkaz pracovního adresáře */home/developer/projekt/BSP-Linux-XBD3-Package-0_4/tftpboot/Xboard_PXA_Project*, aby bylo možné pracovat s co nejjednodušším trasováním cest při nastavování serveru. Symbolický odkaz se vytvoří následujícím příkazem:

```
$ ln –s /home/developer/projekt/BSP-Linux-XBD3-Package-0_4/ /tftpboot
```

Pro správné nastavení serveru musí soubor */etc/inetd.conf* obsahovat následující nastavení:

```
tftp   dgram   udp   wait   nobody  /usr/sbin/in.tftpd /usr/sbin/in.tftpd -s /tftpboot
```

Je nutné si uvědomit, že *tftpd-hpa* server se spouští samostatným skriptem umístěným v */etc/init.d* adresáři. Po zadání příkazu:

```
$ /etc/init.d/tftp-hpa start
```

se nestane nic a počítač ani nedá vědět, že služba je nefunkční. V případě, že bychom rovnou přešli k samostatnému bootování jádra a jádro by se správně načetlo, došlo by přesto k chybovému zastavení zavádění systému. Terminál by zobrazil zprávu, že nastala chyba při *TFTPREAD()* a bude vyžadovat, aby došlo k ověření nastavení UDP.

Chybová zpráva by vypadala následovně:

```
TFtpReadFile()  !CheckUDP: Not UDP (proto = 0x00000001)
```

Dle základního nastavení by se měl daemon pro TFTP server spustit automaticky v době načítání samotného systému. Základní nastavení serveru jako služby najdeme v souboru */etc/default/tftpd-hpa.* Po jeho otevření lze okamžitě vidět, že spuštění

deamona pro tftpd-hpa je defaultně zakázané. Pro spuštění by obsah tohoto souboru měl vypadat následovně:

```
#Defaults for tftpd-hpa
RUN_DAEMON="yes"
OPTIONS="-l -s /tftpboot"
```

V této chvíli je již TFTP server plně nastaven a nezbývá, než ho spustit:

```
$ /etc/init.d/tftp-hpa start
```

Tentokrát reaguje systém na zadaný příkaz bezproblémovým spuštěním služby. Jestliže vytvoření systému pomocí skriptu *setup.sh* (kapitola 7) proběhlo správně, tak nyní bychom měli být schopni načíst soubory z hostujícího systému do X-bordu.

Pro vyzkoušení správného nastavení TFTP serveru, je nutné mít stále připojený cílový počítač pomocí správně nastaveného terminálu. Aby bylo možné systém načíst, je nutné spustit bootovací sekvence X-boardu. Ty jsou aktivní během spuštění počítače, tudíž je nutné cílový počítač restartovat. Systém se propojí s bootloaderem přes terminál. Postup nastavení:

- Pomocí klávesnice <SPACE> se vyvolá menu systémového zavaděče pro X-board.
- Pomocí tlačítka <t> inicializujeme BOOTP broadcast.
- Měla by být ověřena a přijata IP adresa přidělené DHCP serverem.
- Poté konfigurační soubor pro bootování (boot.cfg) a obraz jádra (vmlinuz) budou přeneseny (TftpReadFile potvrdí přenesení pomocí OK).
- Samozřejmě systému se nepodaří nabootovat, jelikož počítači ještě nebyl poslán jím vyžadovaný filesystém pomocí NFS služby. To se projeví chybovými hlášeními v terminálu.

```
Root-NFS: Server returned error -5 while mounting /tftpboot/pxaroot
VFS: Unable to mount root fs via NFS, trying floppy.
VFS: Cannot open root device "nfs" or 02:00
Please append a correct "root=" boot option
Kernel panic: VFS: Unable to mount root fs on 02:00
```

6.2.2.1 Co je to TFTP?

Trivial File Transfer Protocol je velice jednoduchý protokol sloužící pro přenos souborů. Obsahuje jen základní funkce protokolu FTP. Jeho specifikace byla poprvé uveřejněna v roce 1980.

TFTP je určen pro přenos souborů v případech, kdy je běžný protokol FTP není vhodný pro svou komplikovanost. Typickým případem je bootování bezdiskových počítačů ze sítě, kdy se celý přenosový protokol musí vejít do omezeného množství paměti, která je k dispozici na bezdiskovém stroji.

TFTP funguje nad nespojovaným protokolem UDP, musí tedy obsahovat vlastní řízení spojení. Koncepce je jednoduchá, v jednom spojení lze přenést jen jediný soubor. Při komunikaci na síti se vždy pohybuje jen jediný paket. Po odeslání jednoho paketu program čeká na jeho potvrzení a teprve poté posílá další. Kvůli tomuto zjednodušení poskytuje protokol na linkách s velkou latencí jen malou přenosovou rychlost.[12]

6.2.3 NFS server

Poslední ze zástupu serverů je NFS, který umožňuje poskytnutí root filesystému cílovému počítači. K zprovoznění služby je nutné nainstalovat příslušný balíček. Instalace docílíme příkazem:

```
$ apt-get install nfs-kernel-server
```

Konfiguračním souborem služby je */etc/exports*. Pro odkázání NFS na vytvořený filesystém je nutno soubor */etc/exports* doplnit o následující řádku:

```
$ /tftpboot *(rw,no_root_squash)
```

12 Dostupné na WWW: http://cs.wikipedia.org/wiki/Tftp [cit. 2008-04-24]

Server spustíme příkazem:

```
$ /etc/init.d/nfs-kernel-server start
```

V případě změn v nastavení NFS nebo nahrazení expedovaného systému je vždy nutné NFS server restartovat. Jinak by se dané změny neprojevily a mohlo by dojít k chybovému zastavení načítání systému. Příkaz restartu:

```
$ /etc/init.d/nfs-kernel-server restart
```

6.2.3.1 Co je to NFS?

Network File Systém je internetový protokol pro vzdálený přístup k souborům přes počítačovou síť. Umožňuje zavedení diskového oddílu ze vzdáleného počítače a jeho následné využití jako lokálního disku.

Protokol NFS je navržen jako bezestavový, což znamená, že server si nemusí nic pamatovat. Provádí pouze požadavky, které mu přichází a nemusí vůbec nic vědět o tom, jaký vztah k sobě mají tyto požadavky mají vztah. Vzhledem k tomu, že ve většině případů běží nad UDP, musí být server schopen zvládnout duplicitní požadavky. Každý požadavek musí obsahovat veškeré informace nutné pro jeho vykonání. Například požadavek na zápis do souboru musí obsahovat identifikaci souboru, pozici, na kterou se mají data zapsat, velikost dat a samozřejmě data samotná.

Komunikace s NFS serverem probíhá na dvou úrovních. Nejprve je potřeba pomocí mount protokolu připojit adresářový strom, se kterým se bude pracovat. Tak jako vlastní NFS využívá i tento protokol RPC. Při požadavku na připojení adresáře dojde k autentizaci klienta. Poté je mu předán identifikátor kořenového adresáře, který se použije při komunikaci protokolem. Jakmile je strom připojen, může začít komunikace s protokolem NFS, který již obsahuje příkazy pro práci se soubory.[13]

13 Dostupné na WWW: http://www.fi.muni.cz/~kas/p090/referaty/2001-podzim/nfs.2.html [cit. 2008-04-26]

6.3 Propojení a zavedení systémů na X-board

Aby byly počítače schopny spolu komunikovat, je nutné zabezpečit jejich propojení pomocí sériového a ethernetového kabelu. Existuje také možnost k X-boardu připojit klávesnici, aby se dal ovládat přímo a TFT panel, aby bylo vidět, co děláme. Tyto dvě možnosti nejsou nezbytné, pro práci nebyly využity.

Samotným propojením kabelů se nezajistí zprovoznění komunikace. Je nutné použít nějaký terminál pro komunikaci skrz RS232 porty. Velmi vhodným programem pro tuto práci je Minicom. Nainstalujeme ho zadáním příkazu:

```
$ apt-get install minicom
```

6.3.1 Nastavení Minicomu

Minicom je třeba správně nastavit, aby bylo možné vytvořit spojení s X-boardem. Konfiguraci inicializujeme pomocí příkazu:

```
$ minicom -s
```

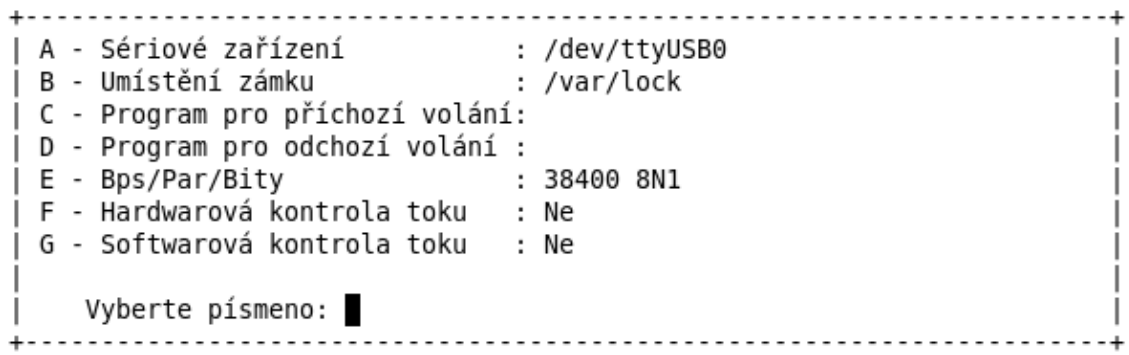
Příkaz otevře textové okno pro nastavení programu.

```
+-------[konfigurace]--------+
| Jména a cesty k souborům   |
| Přenosové protokoly        |
| Nastavení sériového portu  |
| Modem a vytáčení           |
| Obrazovka a klávesnice     |
| Uložit nastavení jako dfl  |
| Uložit nastavení jako..    |
| Konec                      |
| Ukončit Minicom            |
| Konec                      |
| Ukončit Minicom            |
+----------------------------+
```

Obr. 4: Nastavení minicomu

V Ubuntu 7.10 je Minicom plně přeložen do českého jazyka, což usnadní práci lidem, kteří nemluví anglicky. Orientace v konfigurátoru je velmi jednoduchá. Je nutné nastavit pouze parametry pro komunikaci po sériovém portu.

Z nabídky vybereme *Nastavení sériového portu*, čímž se zobrazí další možnosti nastavení:



Obr. 5: Správné nastavení sériového portu v minicomu

Vybíráním jednotlivých písmen z nabídky se zpřístupní možnosti nastavení jednotlivých položek. Důležité je správně nastavit sériové zařízení. Lze si všimnout, že ačkoli se jedná o konfiguraci sériového zařízení, je použit USB port. U dnešních uživatelských notebooků se již nepočítá s nutností použití RS232 portů, proto nejsou zaváděny. Absence je řešena USB-serial redukcemi, které umožňují pracovat stejně, jako kdyby byl použit RS232 port.

Pro nalezení připojeného sériového zařízení lze použít jednoduchý příkaz:

```
$ ls -l /dev/ttyU*
```

Měli bychom dostat výpis všech zařízení, které v daném adresáři odpovídají příkazu, který by měl vypadat následovně:

```
crw-rw---- 1 root dialout 188, 0 2008-04-09 15:18 /dev/ttyUSB0
```

Zajímavý je pouze konec výpisu, který udává zařízení použité k propojení. V Minicomu ho lze definovat pomocí parametru *"A"*. V případě použití sériového portu by vyžadovaným zařízením bylo */dev/ttyS0.*

Postup pro nastavení minicomu:

- Zadá se parametr A a zařízení je nutné nastavit na */dev/ttyUSB0*.
- Zadá se parametr E, který zobrazí tabulku pro nastavení rychlosti, parity a dat. Postupně se vyberou možnosti G a Q, čímž se získá v aktuálním nastavení požadované vlastnosti *38400, 8N1*.
- Následuje návrat do tabulky *Nastavení sériového portu*, kde se pomocí parametru F zakáže hardwarová kontrola toku.
- V tabulce základní konfigurace se vybere položka *Uložit nastavení jako dfl*, pro uložení nastavení.

6.3.2 Spuštění Minicomu a zavedení OS

Vše potřebné je již zadáno. Nastavení minicomu lze opustit a spustit vlastní program. Program se spustí po vyvolání jednoduchého příkazu do konzole:

```
$ minicom
```

Program je nyní spuštěn, zatím však nedošlo k navázání spojení mezi počítači. Aby bylo možno počítače propojit, je potřeba zapnout, popřípadě restartovat, cílový počítač. Ve fázi startu je na X-boardu spuštěn bootloader, který se snaží zavést operační systém pro počítač. Právě ten vše spustí.

V případě správného propojení systémů dostaneme v minicomu následující výpis:

```
Beginning System Initialization ...

        SDCLK[1]   = MemClk
        MemClk     = 99.53 MHz
        Run Mode   = 4 * MemClk
        Turbo Mode = Run Mode
        Mode:    RUN

    Flash = 32 MB, SDRAM = 64 MB

##############################################################
##                    Kontron XScale Bootloader             ##
##                         XBD3R111.024                     ##
##                    (C) Copyright 2004 Kontron            ##
##############################################################

Press [ENTER] to boot from flash
Press [SPACE] to open menu
Boot from flash after 5 seconds.
```

Nabízejí se dvě možnosti. Nechat systém nabootovat z flash paměti umístěné na X-boardu, nebo se dostat do menu nastavení zavaděče. Jelikož v paměti nic není, tak první možnost pro nás nemá význam.

Pomocí klávesy *[SPACE]* se vyvolá následující tabulka nastavení:

```
    ### Boot Loader Configuration ###

    <0> IP address:     192.168.0.230
    <1> Subnet mask:    255.255.255.0
    <2> DHCP:           Enabled
    <5> Download image via CF Card
    <7> Download image via RTL8139 PCI Ethernet
    <t> Download image via RTL8139 BOOTP/TFTP
    <u> Use Debug Ethernet: Disabled
    <p> PCI Menu
    <e> Erase Flash Menu
    <d> Menu Delay:         5
    <a> Menu Auto Select:   "5"
    <s> Save Configuration

    Enter your selection:
```

Toto menu otevírá hned několik možností. Je nutné nastavit masku podsítě, aby byla shodná s nastavením ve vývojovém počítači. Na IP adrese nezáleží, jelikož X-bordu přidělí adresu vytvořený DHCP server při spuštění zavádění systému. Aby vše bylo možné, je důležité nastavit v menu položku *DHCP Enable*. Menu nabízí několik způsobů, jak zavést jádro systému do cílového počítače. Nejdůležitější se ukrývá pod parametrem < *t* >. Umožňuje stažení jádra přes síť.

Spuštění zavádění jádra (< *t* >):

Je-li vše správně nastaveno, X-boardu by měla být přidělena správná IP adresa (pokud je DHCP server spuštěn a správně nastaven). Dále by mělo být možné stáhnout kernel přes TFTP. Ten by se měl nejdříve načíst a následně rozbalit do paměti počítače. Výpis zavádění jádra:

```
Device identification: XSC1BD58283

BOOTP/TFTP download
BootpRead()
BootpRead(): Got Host
ArpSend(192.168.0.1)
ArpSend(): Got Host MAC 00-0A-E4-E0-85-FA
TFtpReadFile(boot.cfg)
TFtpReadFile(): Waiting for data
TFtpReadFile(): Read 92 bytes total
+ValidateAtagFile(0xA00B9100,28416)
MEM 0xA0000000 len 0x04000000
TFtpReadFile(vmlinuz)
TFtpReadFile(): Waiting for data
TFtpReadFile(): Read 1611028 bytes total
TFtpReadFile OK
Uncompressing Linux.....done, booting the kernel.
```

Jestliže vše proběhne úspěšně, mělo by se vypsat množství systémových hlášení, která např. řeknou, že root file systém byl připojen pomocí NFS. Po doběhnutí, by měl být automaticky spuštěn tellnet deamon, který zajistí konečné spojení s X-boardem a možnost jeho ovládání přes síťové rozhraní. Systém požádá o zadání přihlašovacího jména do systému (implicitně je nastaven login root) a poté již máme plný přístup k zavedenému systému na počítači X-board.

Výpis:

```
############################################################
#           Linux startup finished                         #
# Shell access:                                            #
#  - RS232 on COM1 (login as root without password)        #
#  - telnet (login as root without password)               #
#  - local shell (switch to tty2 by pressing ALT+F2)       #
############################################################

Starting pid 792, console /dev/ttyS0: '/sbin/getty'

Xboard_PXA login: root
login[792]: root login  on `ttyS0'
BusyBox v1.2.2 (2008.03.21-00:44+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.
~ #
```

6.4 Možné problémy při načítání

Jak poznat z výpisu Minicomu, kde je problém?

1. Nefunkční, nebo špatně nastavený DHCP server, popřípadě špatné propojení systémů. Výpis:

```
BOOTP/TFTP download

BootpRead()
BootpRead(): Error
BOOTP Failed
ArpSend(192.168.0.70)
TFTP Host ARP Failed
TFtpReadFile(boot.cfg)
TFtpReadFile(): Waiting for data
```

Ukáže-li se toto hlášení, je jisté, že problém nastal již u DHCP serveru, který pravděpodobně nepracuje. Může se jednat o problém v rámci propojení počítačů, který vyústí v neschopnost navázat spojení. Doporučuji zkontrolovat síťové nastavení vývojového počítače, shoduje-li se s nastavením DHCP serveru dle konfiguračního souboru */etc/dhcp3/dhcpd.conf*. Po provedených změnách je nutné server restartovat.

2. Nefunkční, nebo špatně nastavený TFTP.

   V tomto případě by mělo být vypsáno následující hlášení:

```
TFtpReadFile(): Waiting for data
!CheckUDP: Not UDP (proto = 0x00000001)
```

Na rozdíl od DHCP a NFS serveru zde přímo nelze zjistit, zda-li je služba spuštěná. I přes to je dobré si zkontrolovat, zdali se v adresáři */etc/init.d/* nachází spouštěcí skript pro TFTP server *tftp-hpa*. V případě správného nastavení by po zadání příkazu *etc/init.d/tftpd-hpa* mělo být možné dostat výpis možností, jak lze s daným skriptem pracovat. V případě nevypsání hlášení je nutné zkontrolovat soubor */etc/default/tftpd-hpa*, jestli je tam povoleno spuštění deamona pro TFTP a správně nastavena cesta. Pokud je vše v pořádku, mohla by být chyba pouze v samotném nastavení souboru */etc/init.conf*.

3. nefunkční, nebo špatně nastavený NFS server

   Výpis:

```
Root-NFS: Server returned error -13 while mounting /tftpboot
VFS: Unable to mount root fs via NFS, trying floppy.
VFS: Cannot open root device "nfs" or unknown-block(2,0)
Please append a correct "root=" boot option
Kernel panic - not syncing: VFS: Unable to mount root fs on unknown-
block(2,0)
```

Jediné problémy, se kterými jsem se u NFS setkal, byly ve špatném nastavení cesty v souboru */etc/exports*, takže nebylo možné najít filesystém pro přenesení, anebo byla služba pro spuštění vypnutá. Doporučuji zkontrolovat tyto dvě možnosti, popřípadě se obrátit na manuálové stránky NFS.

7 Instalace Linuxové distribuce od Kontronu

Aby bylo možné modul otestovat neprodleně po sestavení a propojení počítačů, měl by být na X-bordu již předinstalovaný systém Microsoft WindowsCE. Toto bohužel nebyl můj případ, musel jsem tedy začínat úplně od začátku.

Po komunikaci s Kontronem jsem získal následující balíčky:

- BSP-Linux-XBD3-JIDA-0_3.tar.gz
- BSP-Linux-XBD3-Kernel-Patch-0_7-kernel_org_2_6_21_5.gz
- BSP-Linux-XBD3-Kernel-Source-0_7-kernel_org_2_6_21_5.tar.bz
- BSP-Linux-XBD3-Package-0_4.tar.gz

Nejdůležitějším z nich je BSP-Linux-XBD3-Package-0_4.tar.gz. Jedná se o instalační balíček celého systému. Obsahuje nástroje pro vytvoření filesystému, jádra a cross-kompilačního toolchainu.

Před nainstalováním systému je nutné upravit vývojový počítač. Prvním krokem je vytvoření uživatelského účtu pro vývoj, pod kterým bude probíhat vytváření systému pro cílový počítač. Nedoporučuje se pracovat jako root, jelikož by mohlo dojít k poškození vývojového systému. V případě správy či nutnosti konfigurace částí systému se tomu vyhnout nedá. Uživatelský účet byl vytvořen již při instalaci operačního systému Ubuntu. Není proto nezbytné vytvářet další účet. Vyskytl se zde ovšem problém se způsobem práce jako superuživatel. Mnoho operací, které budou následně provedeny, vyžaduje, aby uživatel měl i rootovská práva. Nejjednodušším způsobem, jak se stát rootem, je zadat v příkazovém řádku příkaz:

```
$ sudo -s -H
```

Počítač se zeptá na heslo superuživatele. Po jeho zadání lze pracovat jako root. Nyní je nutné přidat uživatelskému účtu práva pro vývoj. Bez nich by nebylo možné spouštět skripty pro vytvoření systému. Například instalační skript implicitně používá příkazu

sudo pro získání přístupových práv. Do příkazového řádku se zadá:

```
$ visudo
```

Příkazem se otevře implicitní konfigurační soubor */etc/sudoers* v editoru Vim.

Sudo je běžným programem pro systémy založené na Unixu. Jeho název vychází ze slov Superuser a Do, což naznačuje jeho funkci. Některým uživatelům dovoluje spouštět programy s rozšířenými přístupovými právy, které jsou vlastní pouze pro superuživatele. Takto lze vyřešit situaci, kdy je vhodné, aby některý uživatel používal programy určené ke správě nebo ovládání systému. Pro rozšíření práv uživatelského účtu je nutné do adresáře */etc/sudoers* přidat následující řádek:

```
developer ALL=(ALL) ALL
```

Tento příkaz udělí uživateli "developer" plná superuživatelská práva.

Při práci v Linuxu se uživateli často stane, že systém odmítne provést nějaký příkaz z důvodu nedostatečného oprávnění pro spuštění. Pokud se tak stane je možné vepsat před příkaz slovíčko *sudo*, které udělí dostatečné oprávnění pro jeho spuštění. Toto lze udělat pouze v případě, kdy uživatel zná heslo pro vývoj.

Pro snadnější práci s editorem doporučuji prostudovat si manuálové stránky Vim. Lze je vyvolat příkazem *man vi.*

Pomocí následujících příkazů se vytvoří adresář, do kterého se zkopíruje a rozbalí instalační balíček systému BSP-Linux-XBD3-Package-0_4.tar.gz:

```
cd /home/developer
mkdir projekt
cp BSP-Linux-XBD3-Package-0_4.tar.gz /home/developer/project
tar -xzf BSP-Linux-XBD3-Package-0_4.tar.gz
cd /home/developer/projekt/BSP-Linux-XBD3-Package-0_4/
```

V adresáři se kromě složek s nástroji pro vytvoření systému nachází soubor bsp.config, instalační skript setup.sh a průvodní manuál. Pomoci souboru bsp.config lze

trochu nahlédnou pod pokličku tomu, co se bude dít po spuštění instalačního skriptu. Dále také umožňuje také změnit název projektu. Pro editaci lze soubor pohodlně otevřít z příkazového řádku pomocí jednoduchého příkazu:

```
$ gedit bsp.config
```

Název projektu je definován položkou *PROJECTNAME*, kde je defaultně přednastaveno *Xboard_PXA_Project*. Pro vytvoření systému zbývá spustit instalační skript, který se skrývá pod označením setup.sh. Pro jeho spuštění je nutné zadat příkaz:

```
$ ./setup.sh
```

Skript se spustí a začne provádět jednotlivé procedury, které vedou k vytvoření systému.

Při spuštění skriptu *setup.sh* nastal problém. Po provedení zaváděcích příkazů proces z neznámých důvodů zkolaboval. Systém pouze vypsal na obrazovku několik nic neříkajících chybových hlášení a to bylo vše. Po důkladných kontrolách nastavení vývojové aplikace a hledání příčiny jsem usoudil, že problém není v instalačním softwaru. Trvalo několik dní, než jsem s pomocí internetu zjistil, že problém vyvolal přímo operačním systémem Ubuntu 7.10.

Psaní skriptů podléhá vlastním pravidlům. Je nutné systém odkázat na to, že vytvořený adresář je právě skript. Vždy to určuje první řádka souboru, kde je napsáno:

```
#!/bin/sh
```

Ta určí, že soubor bude spuštěn pomocí shallu bash. Je zavedenou praxí, že právě */sh* je tím, co odkazuje na bash. Naneštěstí se vývojáři Ubuntu 6.10 rozhodli změnit svoji strategii a právě koncovkou */sh* odkázali jinému shallu, který se jmenuje dash. Ačkoliv jsou bash a dash v mnoha věcech dosti podobné, jejich syntax se částečně odlišuje, například při deklarování proměnných skripty psané v bashi nebudou funkční. Jednou z možností, jak vyřešit problém, je přepsat první řádku každého skriptu z *#!/bin/sh* přímo na *#!/bin/bash*. Nyní již nebudou spory o tom, čím se má skript přeložit. V celé vývojové aplikaci BSP je velmi mnoho skriptů a jejich přepsání by trvalo dlouhé hodiny „mravenčí práce“.

Pro změnu základního systémového shellu lze přikázat balíčkovému manažeru, aby přestal používat dash pro /bin/sh. Je možné dosáhnout toho pomocí příkazu:

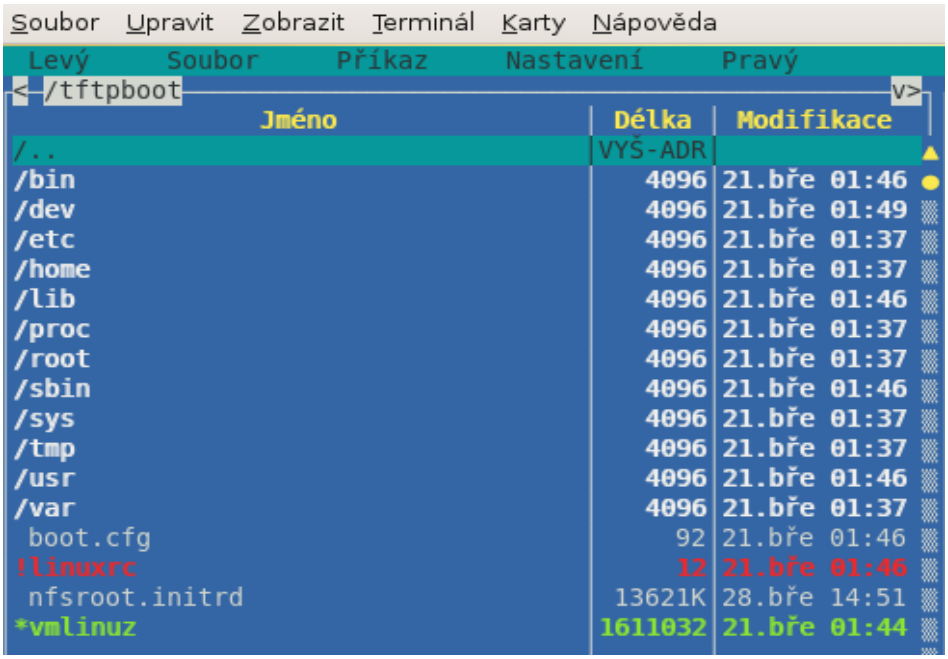
```
$ sudo dpkg-reconfigure dash
```

který zpřístupní nastavení balíčků. Nyní lze rozhodnout zda-li nainstalovat dash jako */bin/sh*. Pro tuto práci je tato možnost nežádoucí, zvolil jsem tedy *NE*.

Nyní lze znovu spustit skript *setup.sh.* Tentokrát je vidět, že probíhá správně. Během provádění se skript několikrát zastavil, v závislosti na tom, zda-li mu chybí či nechybí základní balíčky pro svůj běh. Hlášení o provádění skriptu se zapisují do souboru *./instal.log*, kde také najdeme zápis o chybě, která vedla k zastavení. Z daného zápisu bývá většinou přímo možné zjistit chybějící závislosti a poté již není problém dané balíčky doinstalovat. Systém Ubuntu 7.10 bylo nutno rozšířit o následující balíčky:

gcc	*sudo apt-get install build-essential*
bison	*sudo apt-get install bison*
flex	*sudo apt-get install flex*
patch	*sudo apt-get install patch*
make	*sudo apt-get install make*

V této chvíli by měl systém obsahovat vše potřebné. Skriptu bude trvat delší dobu, než zdárně dospěje ke konci. Záleží pouze na rychlosti použitého počítače. V našem případě trvalo asi dvě hodiny, než skript vytvořil požadovaný systém. V adresáři */home/developer/projekt/BSP-Linux-XBD3-Package-0_4/tftpboot/Xboard_PXA_Project* je nyní vytvořen kompletní systémový strom s obrazem jádra *vmlinuz* a jeho spouštěčem *boot.cfg*.



Obr. 6: Filesystém a jádro vytvořeného systému

Během svého běhu provedl skript následující operace:

- vytvořil prostředí pro cross kompilaci;
- vytvořil jednoduchý demo projekt sestávající z kernelu a busyboxu;
- vytvořil TFTP root adresář, který byl exportován do demo projektu.

Podrobnější popis prováděných operací lze najít v souboru *./instal.log*.

Instalační balíček *BSP-Linux-XBD3-Package-0_4.tar.gz* vytvořil systém, který je postaven na jádře verze 2.6.20.1. Kontron nám poskytl novější verzi jádra. Po rozbalení instalačního balíčku vznikne v adresáři projektu složka *kontron_data*, kde je uloženo jádro stávajícího linuxu a balíček *JIDA* verze 0_2. Tyto balíčky stačí nahradit novějšími verzemi *BSP-Linux-XBD3-JIDA-0_3.tar.gz* a *BSP-Linux-XBD3-Kernel-Source-0_7-kernel_org_2_6_21_5.tar.bz*, které budou využity při vytvoření systému.

8 Testování a kompilování aplikací pro X-board

V této chvíli je již zaveden operační systém na modul a lze tedy začít s testováním a využíváním aplikací.

Operační systém slouží jako prostředí pro spouštění aplikací, neumožňuje však jejich přímou kompilaci. Ta je provedena až pomocí programovacích nástrojů nacházejících se pod toolchainem, který byl vytvořen při instalaci společně se systémem. Nejdůležitějším z nástrojů je překladač gcc.

Při kompilování programů pro X-board pomocí gcc je nutné vždy zadat přesnou cestu vytvořeného překladače. Systém by jinak použil svůj vlastní gcc překladač a přeložil aplikaci pro sebe. Poté by nebylo možné program na modulu spustit.

Cesta k gcc :

```
/projekt/toolchain/gcc-4.0.2-glibc-2.3.6/arm-xscale-linux-gnu/bin/arm-xscale-linux-gnu-gcc
```

8.1 První program HelloWorld

Na vývojovém počítači je vytvořen soubor nazvaný *hello.c*, do kterého se zapíše a uloží následující text:

```
/*hello world v ANSI c*/
#include <stdlib.h>
#include <stdio.h>
int main (void)
{
        puts("Ahoj svete!");
        return EXIT_SUCCESS;
}
```

Přeložení programu se spustí příkazem:

```
$ /home/rawiry/projekt/toolchain/gcc-4.0.2-glibc-2.3.6/arm-xscale-linux-gnu/bin/
arm-xscale-linux-gnu-gcc hello.c -o HelloWorld
```

V původním adresáři souboru hello.c, se kompilací vytvořil nový soubor *HelloWorld.* Ten stačí zkopírovat do sdíleného filesystému pro X-board, například do adresáře */home/user/,* kde máme plná práva pro spuštění. Aplikace je spuštěna příkazem:

```
$ /home/user/HelloWorld
```

V minicomu se vypíše *Ahoj svete!*

8.2 SciMark 2.0

SciMark 2.0 je testovací program (benchmark), který provádí vědecké a numerické výpočty pro měření výpočetního výkonu počítače. Výsledek měření je zapsán v jednotkách Mfops (Millions of floating point operations per second). ANSI C verzi benchmarku lze stáhnout jako zkomprimovaný soubor *scimark2_1c.zip* na adrese *http://math.nist.gov/scimark2/download_c.html*. Je dobré si nejdříve vytvořit adresář, do kterého se soubor stáhneme. Kompilace je provedena následujícím způsobem:

```
unzip  scimark2_1c.zip
cd ./scimark2
/home/user/projekt/toolchain/gcc-4.0.2-glibc-2.3.6/arm-xscale-linux-
gnu/bin/arm-xscale-linux-gnu-gcc hello.c -o scimark2  -lm -O *.c
```

Kompilace proběhne bezproblémově a vytvoří soubor v aktuálním adresáři s názvem scimark2. V příkazu je nutné použít parametr -lm, aby bylo možné program propojit s matematickou knihovnou. Program jinak nelze přeložit, protože mu bude chybět link na math.h. Soubor se zkopírujeme do adresáře */home/user/* do X-boardu a spustí se.

Spuštění benchmarku:

```
$ /home/user/scimark2
```

Aplikace má dva módy spuštění. První lze spustit příkazem *scimark2*, kdy se program spustí s minimálním nastavením pro testové analýzy. Rozšířenou verzí je možnost spuštění jako *scimark2 -large*. Tímto příkazem se provedou stejné výpočetní metody. Testuje se však pro větší počet prvků.

Po spuštění aplikace se začnou provádět výpočetní operace a po dokončení program vypíše následující tabulku:

```
~ # cd /home/user/
/home/user # ./scimark2
**                                                          **
**  SciMark2 Numeric Benchmark, see http://math.nist.gov/scimark    **
**  for details. (Results can be submitted to pozo@nist.gov)        **
**                                                          **
Using       2.00     seconds min time per kenel.
Composite Score:                0.31
FFT                 Mflops:     0.18    (N=1024)
SOR                 Mflops:     0.42    (100 x 100)
MonteCarlo:         Mflops:     0.25
Sparse matmult      Mflops:     0.36    (N=1000, nz=5000)
LU                  Mflops:     0.33    (M=100, N=100)
```

Tab. 2: Porovnání dosažených výsledků benchmarku X-boardu s notebookem ACER Aspire 3022LMi

	SciMark2 – výsledky benchmarku		
	X-board PXA255 [Mflops]	ACER Aspire 3022LMi [Mflops]	nastavení parametrů testu
FFT	0.18	375.91	N=1024
SOR	0.42	373.44	100 x 100
MonteCarlo	0.25	71.97	
Sparse matmult	0.36	366.12	N=1000, nz=5000
LU	0.33	700.17	M=100, N=100
Composite	0.31	377.52	

Tab. 3: Porovnání dosažených výsledků benchmarku X-boardu s notebookem ACER Aspire 3022LMi při rozšířených počátečních podmínkách jednotlivých testovaných analýz

	SciMark2 – výsledky benchmarku		
	X-board PXA255 [Mflops]	ACER Aspire 3022LMi [Mflops]	nastavení parametrů testu
FFT	0.38	172.22	N=1048576
SOR	0.18	27.64	1000 x 1000
MonteCarlo	0.41	361.54	
Sparse matmult	0.25	71.97	N=100000, nz=1000000
LU	0.37	190.33	M=1000, N=1000
Composite	0.70	209.64	

Tabulky 2 a 3 zobrazují dosažené výsledky prováděných testů na X-boardu a vývojovém počítači. Je vidět, že modul PXA je mnohonásobně pomalejší než pracovní stanice. I přesto jsou dosažené výsledky dostačující pro bezproblémové řízení základních úloh.

9 ELinOS a práce s ním

Doručený balíček od společnosti SYSGO obsahuje uživatelský manuál, návod k online registraci systému a balení se dvěma CD. V prvním je samotný software ELinOS určený pouze pro práci s procesory typu ARM/Xscale. Druhé médium obsahuje nástroj Codeo.

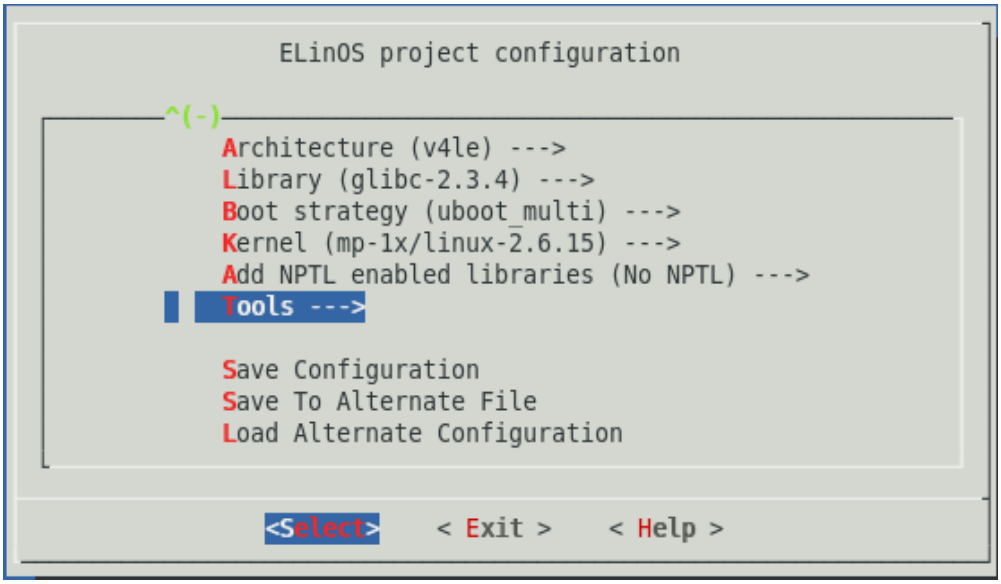
Přiložený manuál je velmi přehledně napsaný návod, jak nainstalovat vývojový software a vytvářet aplikace a řídící systémy pro určenou platformu.

Instalace ELinOSu je velmi jednoduchá. Na médiu najdeme skript INSTALL, který po spuštění vše obslouží sám a systém zavede do adresáře */opt/elinos-4.2*. Pro plné užívání softwaru je nutné dle přiloženého návodu vytvořit online licenční klíč a vložit ho do adresáře */opt/elinos-4.2*.

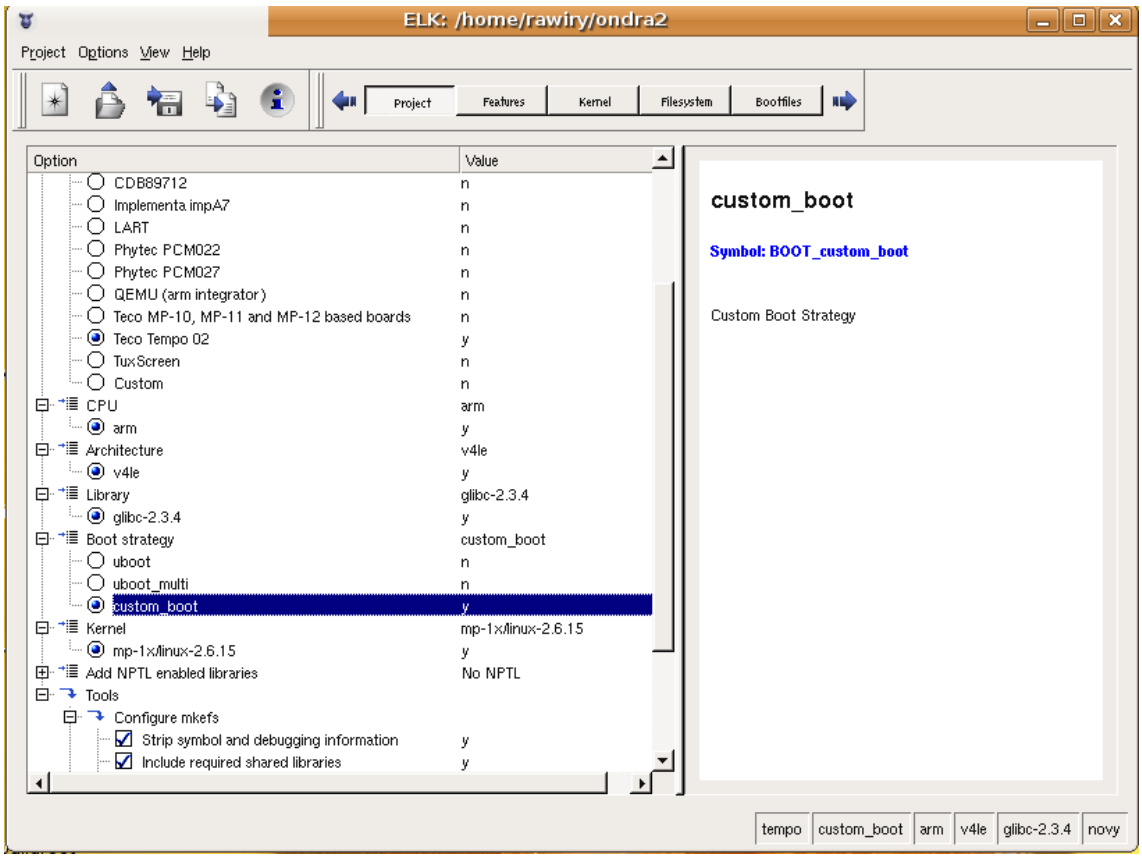
9.1 První kroky vývoje

ELinOS nabízí pro ulehčení vývoje systémů vlastní demoverze. Jsou to předkonfigurované aplikace s různým zaměřením systému, které stačí nastavit dle potřeb a zkompilovat. Je možné zde najít například systém spouštějící aplikaci HelloWorld, systémy s přednastavenými NFS a BOOTP servery, se zavedenou real-time podporou, popřípadě dalším nastavením, které embedded systémy nabízí.

Vývoj systému může probíhat pomocí konzole, kde je k dispozici konfigurační prostředí podobné nastavování jádra. Další možnost vývoje je v grafickém prostředí ELK.



Obr. 7: Základní konfigurátor ELinOSu



Obr. 8: ELK (Embedded Linux Konfigurator)

9.2 Vytvoření systému pro X-board

Důvodem výběru této komerční linuxové distribuce bylo vytvoření embedded systému s real-time podporou, který by bylo možné zavést na cílový počítač společnosti Kontron.

ELinOS od počátku trpěl vážným nedostatkem. Neobsahuje totiž přímou podporu desky X-board od Kontronu, bez které nelze správně systém sestavit. Po komunikaci s centrem podpory společnosti SYSGO mi bylo doporučeno jako výchozí bod použít BSP pro počítač TEMPO 02 společnosti Teco a.s., který podporuje stejný procesor jako X-board. Následující nastavení a kompilace systému proběhli bez větších problémů. Vytvořil se filesystém s jádrem a příslušnými náležitostmi.

Vytvořený systém se nepodařilo úspěšně nabootovat do cílového počítače, jelikož zavaděč X-boardu nedokázal zavést vytvořené jádro. Problém byl pravděpodobně způsoben nekompatibilitou použitých bootloaderů. S otázkou o pomoc jsem se obrátil na vývojové centrum ELinOSu. Tento dotaz však nevedl k vyřešení problému.

10 Vytvoření filesystému pomocí softwaru Buildroot

10.1 Stažení

Pro buildroot jsou vytvářeny každodenní aktualizace, tzv. snapshoty. Databáze starších verzí se nachází na http://buildroot.uclibc.org/downloads/snapshots/. Nejnovější snapshot lze stáhnout z http://buildroot.uclibc.org/downloads/snapshots/buildroot-snapshot.tar.bz2. Po stažení rozbalíme buildroot na disk. Program zatím obsahuje pouze základní adresáře s defaultním přednastavením. Nacházejí se zde makefily a patche, které stáhnou, nakonfigurují a zkompilují software podle později zvoleného nastavení a vytvoří cross-compilančího tool chainu (gcc, binutils and uClibc). Pro každý software existuje právě jeden makefile, který se pozná podle koncovky .mk.

Makefily jsou rozděleny do čtyř skupin:

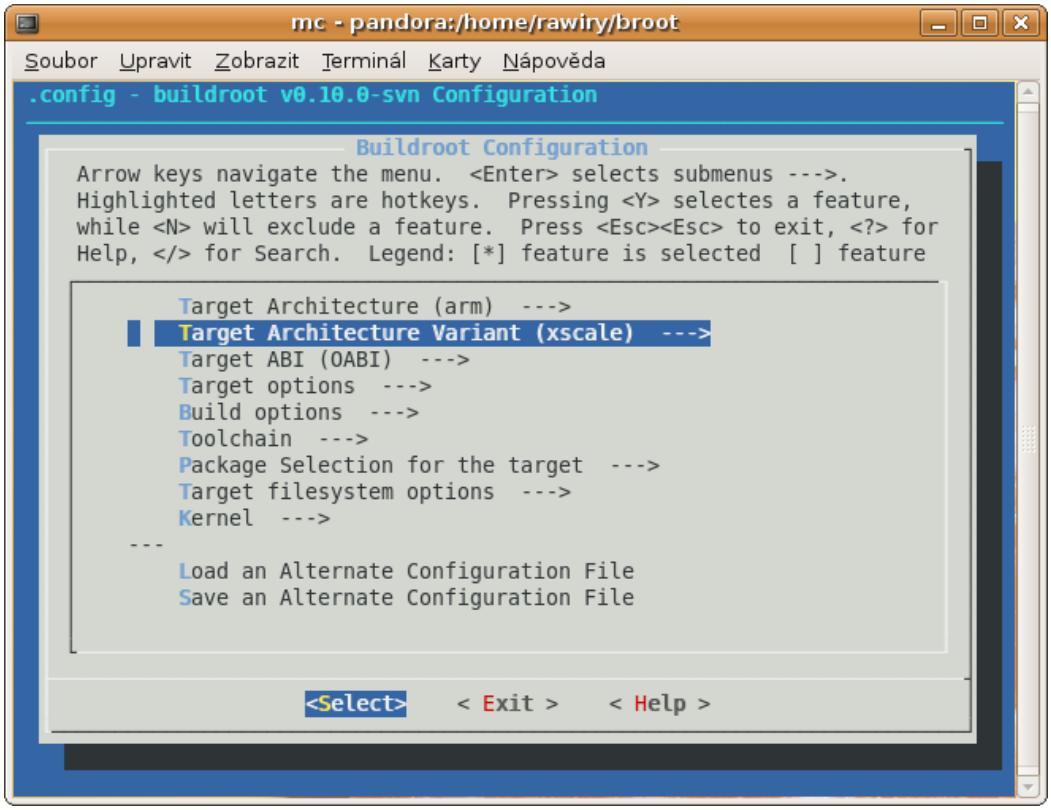
- Project (v adresři project/ directory) obsahuje Makefily a soubory pro vytvoření root filesystému.
- Toolchain (v adressáři toolchain/ directory) obsahuje makefily a soubory pro vytvoření toolchainu: binutils, ccache, gcc, gdb, kernel-headers and uClibc.
- Package (v adresáři package/ directory) obsahuje makefily a soubory pro nástroje definované uživatelem, které se zkompilují do filesystému.
- Target (v adresáři target directory) obsahuje makefily a soubory pro software, který má vytvořit filesystém image : ext2, jffs2, cramfs a squashfs.

Aby mohl buildroot správně pracovat, je nutné do vývojového systému Ubuntu 7.10 doinstalovat sadu nezbytných aplikací:

```
$ sudo apt-get install build-essential libncurses-dev bison flex texinfo zlib1g-dev
gettext openssl-dev autoconf
```

10.2 Nastavení

Prvním krokem je spuštění příkazu *make menuconfig* v root adresáři buildrootu. Nejprve dojde k propojení kompilátoru s gcc a vygenerování souboru .config, ve kterém jsou zapsána nastavení pro vytvoření systému. Poté se zobrazí samotné konfigurační prostředí buildrootu, nápadně podobné konfigurátoru jádra.



Obr. 9: Konfigurační prostředí Buildroot

Sestavení balíčku pro filesystém probíhá jednoduchým způsobem pomocí “enable/disable“ nabídky. Vybrané balíčky se po uložení zapíší do souboru ".config", podle něhož budou poté zkompilovány.

První možnost nastavení je zvolení požadované architektury, pro kterou má být systém vytvořen, a upřesnění jejího typu.

Target Architekture bude nastavené na *arm*

Target Architekture Variant bude *xscale*

Důležitou položkou je **Toolchain**, kde se definují hlavičky jádra. Použitému kernelu verze 2.6.21.5 musí tedy odpovídat i nastavení hlaviček. To je velmi důležité. Při špatném nastavení by pravděpodobně vytvořený filesystém byl nepoužitelný. Dosáhlo by se pouze vytvoření dvou adresářů, z nichž každý by obsahoval vlastní soubor patchů.

Package Selection for the target obsahuje velké množství balíčků a funkcí, které lze pro systém použít. V této části nastavení je možné do systému přidat kompilátory bzip2, shall, balíčky pro práci s grafikou a další užitečné věci. Je nutné dávat si pozor na veškeré závislosti, jelikož v této části vzniká mnoho problémů, které nedovolí správný překlad systému.

Je nezbytné zkontrolovat nastavení pod záložkou Target filesystem, která určí typ vytvořeného filesystému. Pro modul je nutné nastavit EXT2 filesystém.

Položka kernel nabízí možnosti výběru a prvotního nastavení jádra systému. Lze přímo nastavit, aby se použil kernel podle nastavených hlaviček. Rozšířené nastavení umožňuje výběr z více verzí Linuxového jádra. Je zde možnost zadat absolutní nebo relativní cestu k patchy za účelem rozšíření jádra pro zvolené zařízení. S možností volného výběru rozšíření jádra se může stát, že vytvořená kombinace bude zcela nefunkční.

10.3 Kompilace

Kompilace se spustí příkazem *make*. V případě správného nastavení se vytvoří filesystém s jádrem, toolchainem a veškerými náležitostmi. V optimálním případě by se mělo podařit vytvořený systém zavést i do X-boardu, na kterém by měl fungovat. Dosáhnout však správného nastavení není tak jednoduché. Vyžaduje to hodiny i dny nastavování a trpělivosti. Jenom samotný průběh kompilace může trvat několik hodin v závislosti na použitém počítači. Existuje množství chyb, které mohou nastat. Některé je možné odstranit přidáním či odebráním balíčků konfigurace. U jiných často ani nelze určit, co přesně vedlo k přerušení programu. Velkým rádcem se stává internet a zkušenosti jiných uživatelů.

10.4 Výsledek

Výsledkem kompilace by měl být systém, který by bylo možné zavést do cílového počítače.

V rámci práce se podařilo vytvořit pouze filesystém s toolchainem, který byl optimalizován pro Linux kernel 2.6.21.5. Při pokusu o vytvoření jádra program buildroot vždy zhavaroval. Buildroot obsahuje mnoho možností nastavení systému a množství vnitřních závislostí. Pravděpodobný důvod havárií mohl být zapříčiněn neschopností programu propojit filesystém s jádrem, což mohlo způsobit samotné nastavení systému, které nemuselo odpovídat potřebám jádra. Pro jádro není dostačujícím nastavením propojení s příslušnými hlavičkami, důležité je najít pro něj správnou kombinaci s knihovnami gcc, glibc a binutils.

Následně jsem se pokusil zavést vytvořený filesystém do X-boardu s použitím již vytvořeného jádra od Kontronu. Bootloader dokázal filesystém rozpoznat, avšak ke spuštění programu init a následnému zavedení systému nedošlo. Problém byl pravděpodobně v nekompatibilitě knihoven vytvořeného filesystému pomocí buildrootu a jádra od Kontronu.

11 Závěr

Pro tuto bakalářskou práci byly bezplatně získány dvě různé Linuxové distribuce - systém od Kontronu a ELinOS od Sysga.

Prvně jmenovaný operační systém byl optimální volbou co se týče hardwarové podpory. Obsahoval veškeré potřebné ovladače zařízení modulu X-board PXA255. Pro tyto vlastnosti byl primárně využit na oživení a testování zařízení. Proběhlo jeho nainstalování na vývojový počítač a následné přenesení na modul. Systém není real-timeový, ale aktualizace jádra na verzi 2.6.21.5 umožnila využití preemptivního plánování procesů, které je prvním krokem k vytvoření real-time systému. Pro tento systém byly zkompilovány aplikace využité k otestování systému a základních výpočetních možností zařízení.

Důvodů pro výběr ELinOSu existovalo hnedle několik. Systém je plně real-timeový, což řeší nutnost rozšíření jádra o real-time nadstavbu. Byla předpokládána dobrá podpora modulu X-board PXA255. Systém má velmi dobře vytvořené pracovní prostředí a nabízí široké možnosti konfigurace. Po prostudování přiložených dokumentů byl vytvořen systém, který odpovídal parametrům X-boardu a požadavkům zadané bakalářské práce. Naneštěstí při pokusech o zavedení se ukázalo, že bootloader použitý Kontronem pro X-board nebyl kompatibilní se zavaděčem vytvořeným ELinOSem. Systém tedy nebylo možné použít.

Tento systém může být v budoucích projektech a pracích přínosem pro výzkum v oblasti systémů a embedded zařízení.

Rozvoj této práce by mohl spočívat v uzpůsobení ELinOSu pro X-board. Další možností je systém od Kontronu rozšířit a testovat na něm složitější real-time aplikace, popřípadě modul využít pro řízení nějakého procesu.

Seznam použitých zdrojů

BRUYNINCKX, H. *Real-Time and Embedded Guide.* K.U.Leuven, Mechanical Engineering, 2002.

NOVÁK, O. Realizace minidistribuce Linux pro řídící systém mobilního robota. [Ročníkový projekt], Liberec : TUL, 2007.

Distribuce Linux pro řídící systém PXA. Dostupné na WWW: http://www.lynuxworks.com [cit. 2007-10-23]

Distribuce Linux pro řídící systémy PXA. Dostupné na WWW: http://www.sysgo.com [cit. 2007-11-04]

Dokumentace k real-time nadstavbě Xenomai. Dostupné na WWW: http://www.xenomai.org [cit. 2008.02-19]

Dokumentace k řídícímu systému PXA. Dostupné na WWW: http://emea.kontron.com [cit. 2007-11-07]

PELČÁK, V. *Real-time modifikace Linuxu.* Abíčko, březen 2006.

Starterkit for X-boardTM <PXA> Document Revision 1.0. Dostupné na WWW: http://www.mite.cz/xboardpxastarterkitfiles/XBD3K110.pdf [cit. 2007-12-15]

Dostupné na WWW: http://cs.wikipedia.org/wiki/Linux [cit. 2008-05-02]

Dostupné na WWW: http://cs.wikipedia.org/wiki/UDP [cit. 2008-04-24]

Dostupné na WWW: http://www.abclinuxu.cz/slovnik/posix [cit. 2008-05-09]

Dostupné na WWW: http://www.fi.muni.cz/~kas/p090/referaty/2001-podzim/nfs.2.html [cit. 2008-04-26]

Dostupné na WWW: http://cs.wikipedia.org/wiki/GNU_toolchain [cit. 2008-05-05]

Dostupné na WWW: http://cs.wikipedia.org/wiki/GCC [cit. 2008-05-05]

Dostupné na WWW: http://cs.wikipedia.org/wiki/Real-time_operating_system [cit. 2008-05-09]

Dostupné na WWW: http://cs.wikipedia.org/wiki/DHCP [cit. 2008-04-17]

Dostupné na WWW: http://cs.wikipedia.org/wiki/Tftp [cit. 2008-04-24]

Dostupné na WWW: http://www.fi.muni.cz/~kas/p090/referaty/2001-podzim/nfs.2.html [cit. 2008-04-26]

Dostupné na WWW: http://linuxreviews.org/man/tftp/ [cit. 2008-01-08]

Dostupné na WWW: http://kegel.com/crosstool/crosstool-0.42/buildlogs/ [cit. 2008-03-10]

Dostupné na WWW: http://linux.die.net/man/1/minicom [cit. 2008-04-15]

Dostupné na WWW: http://www.abclinuxu.cz/clanky/show/145482?varianta=print&noDiz [cit. 2007-12-15]

Dostupné na WWW: http://cs.wikipedia.org/wiki/DHCP [cit. 2007-12-15]

Dostupné na WWW: http://docwiki.gumstix.org/index.php/Buildroot_on_Ubuntu [cit. 2008-01-06]

Dostupné na WWW: http://www.fi.muni.cz/~kas/p090/referaty/2001-podzim/nfs.2.html [cit. 2008-01-02]

Dostupné na WWW: http://tldp.org/HOWTO/Bootdisk-HOWTO/buildroot.html [cit. 2008-03-25]

Dostupné na WWW: http://cs.wikipedia.org/wiki/Real-time_operating_system [cit. 2008-04-28]

Dostupné na WWW: http://buildroot.uclibc.org/downloads/snapshots/ [cit. 2008-04-13]

Dostupné na WWW: http://buildroot.uclibc.org/downloads/snapshots/buildroot-snapshot.tar.bz2 [cit. 2008-04-13]

Seznam příloh

Příloha 1: BSP pro Teco TEMPO 02

Příloha 2: CD

Příloha 1: BSP pro Teco TEMPO 02

- linuxového jádra 2.6.15
- 64MB RAM Memory
- 32MB/64MB FLASH Memory
- serial ports: FF-UART, BT-UART, HW-UART and ST-UART
- Ethernet 100/10Mb interface (SMC91C111)
- USB gadget controller
- Compact Flash
- Watchdog
- RTC (V30250)
- GPIO interface
- 2 x USB host controller (ISP1161)
- LCD Controller
- A97 Sound codec (UCB1400)
- Touchscreen interface (UCB1400)
- Power voltage monitor (UCB1400)
- Power off button
- Beeper

Příloha 2: CD

CD obsahuje:

- balíčky pro vytvoření operačního systému od Kontronu;
- vytvořený operační systém pro modul X-board PXA255;
- zkompilované aplikace HelloWorld! a SciMark 2.0;
- elektronickou verzi bakalářské práce ve formátu ODT a PDF.