ZÁPADOČESKÁ UNIVERZITA V PLZNI

FAKULTA ELEKTROTECHNICKÁ

Katedra elektromechaniky a výkonové elektroniky

BAKALÁŘSKÁ PRÁCE

Testovací prostředí pro embedded Linux

Ondřej Procházka **2016**

ORIGINAL ZADANI

Abstrakt

Tématem této práce jsou embedded zařízení a jejich operační systémy. V první části je provedeno srovnání vybraných hardwarových platforem a přehled uvažovaných operačních systémů. Cílem práce je prověřit různé možnosti načtení operačního systému z lokálního a síťového zdroje. Na základě získaných informací je v druhé části práce navrženo prostředí pro testování různých distribucí na rozdílných zařízeních.

Klíčová slova

Embedded systém, Linux, TFTP, DHCP, NFS, PXE, U-Boot

Abstract

Theme of this thesis is embedded devices and their operating systems. The first part contains comparison of chosen hardware platforms and overview of possible operating systems. Goal of this thesis is to test different possibilities of loading operating systems from local and network sources. The second part is proposition of environment for testing various distributions on different devices, based on gathered information.

Keywords

Embedded system, Linux, TFTP, DHCP, NFS, PXE, U-Boot

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně, s použitím odborné literatury a pramenů uvedených v seznamu, který je součástí této diplomové práce.

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

.....

V Plzni dne 2.6.2016 Ondřej Procházka

Poděkování

Rád bych poděkoval všem, kteří mi byli nápomocni při vytváření této práce. Zvláště pak děkuji vedoucímu práce, panu Ing. Petru Weissarovi, Ph.D, za velmi zajímavé téma, připomínky a pomoc při zpracovávání této práce.

Obsah

1. Úvod

S rozvojem embedded zařízení se zlepšuje i jejich dostupnost pro vývojáře softwaru a hardwarových komponent. Objevují se nové specializované distribuce operačních systémů a firmwarů pro jednotlivá zařízení. To s sebou přináší potřebu testování bez dlouhých prodlev nutných pro přípravu média s operačním systémem.

První částí práce je srovnání hardwaru nejběžnějších embedded systémů a Linuxových distribucí pro ně použitelných. Vzhledem k množství dostupných řešení jsem musel výběr omezit na ty nejdostupnější a nejčastější.

Cílem této práce bylo navrhnout prostředí, které by sloužilo pro testování vlastností embedded systémů různých výrobců s různými operačními systémy. Po prvotním prozkoumání problematiky bylo jednoznačné, že půjde o načítání ze sítě. K tomuto cíli pak směřovala technická část celé práce.

2. Embedded systémy

Pod tímto pojmem jsou zahrnuty různé typy specializovaných zařízení a počítačů. Nejrozsáhlejší oblasti použití jsou v civilní i vojenské avionice, automatizaci, počítačových a mobilních sítích.

Dle oblasti použití je od těchto systémů vyžadována různá míra spolehlivosti, od které se odvíjí nároky na hardware i software. V některých případech nelze takovýto systém opravit ani jednoduše vyměnit - příkladem je technika použitá v satelitech nebo instalacích na nepřístupných místech. Tyto systémy jsou před nasazením rozsáhle testovány za různých hraničních podmínek. Jejich výroba probíhá po kusech nebo v malých sériích s odpovídající cenou.

Opačným případem jsou zařízení projektovaná jako spotřební elektronika, kde jsou jakékoliv opravy nerentabilní. Tato zařízení se vyrábějí ve velkých sériích a spadají sem například zařízení pro automatizaci domácnosti, mobilní zařízení a podobné.

Speciálním případem jsou systémy pro avioniku, zabezpečení dopravě a energetice, kde je na prvním místě bezpečnost. Tato zařízení jsou často navrhována tak že v případě poruchy převezme kontrolu operátor nebo jako zdvojená, kdy převezme řízení záložní jednotka a chyba je signalizována.

2.1. Historie

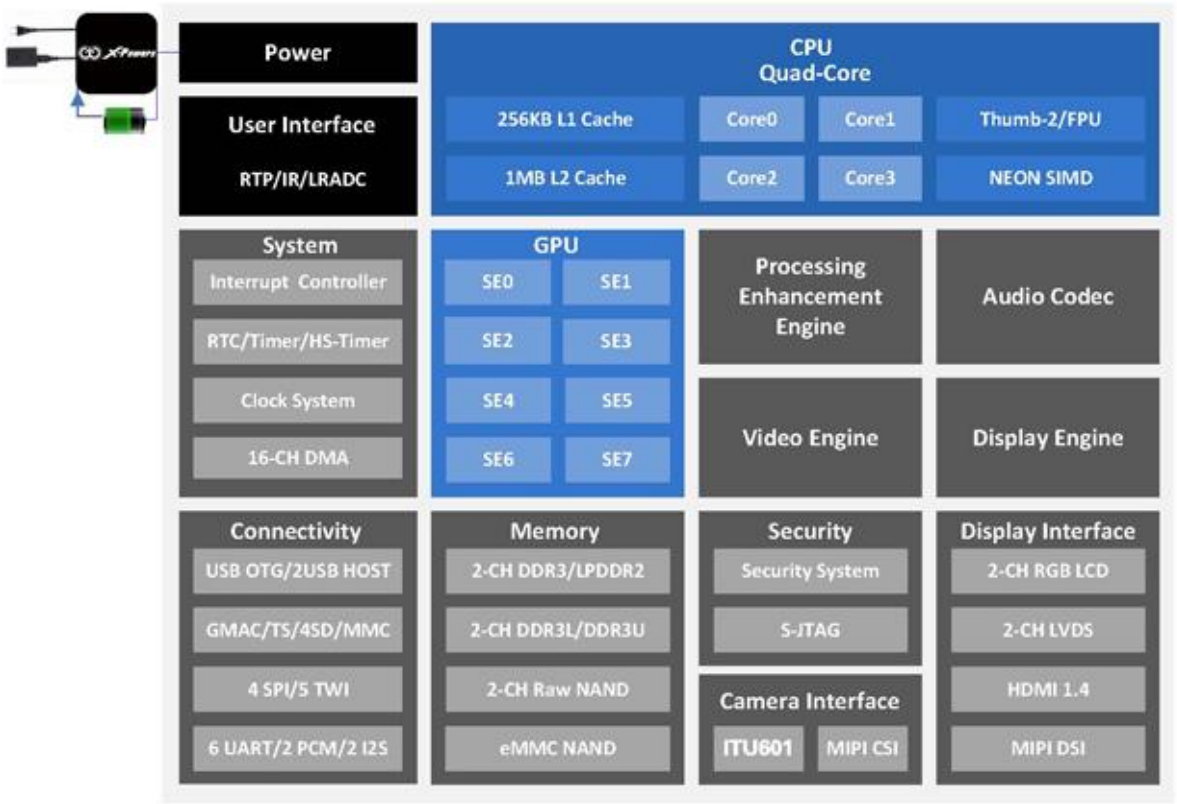
Prvním moderním embedded systémem byl Apollo Guidance Computer, vyvinutý Charlesem Starkem Draperem v laboratoři MIT.

Prvním sériově vyráběným systémem byl naváděcí počítač střel Minuteman I od roku 1961. V roce 1966, při výrobě střel Minuteman II byl nahrazen modernější verzí, která využívala ve velké míře integrované obvody a byla programovatelná. Tento počítač byl schopen provádět i testy zařízení. Sériová výroba těchto počítačů byla jedním z faktorů, který vedl ke snížení ceny integrovaných obvodů a jejich rozšíření.

Vývoj pokračoval s miniaturizací elektroniky až k dnešním zařízením dostupným pro širokou veřejnost. S rozšiřujícím se okruhem využití se také zvětšovalo množství výrobců, architektur procesorů a na nich postavených embedded zařízení. Mezi nejpoužívanější patří procesory ARM, MIPS, PowerPC, x86 a mnoho dalších.

2.2. SOC

System on chip je obvod, který zahrnuje větší množství komponent integrovaných v jednom čipu. Výhodou tohoto řešení jsou kromě nízkých nákladů na výrobu i úspora energie. Na následujícím obrázku je příklad takového řešení - obvod od firmy Allwinnertech s označením A31 použitý pro Banana PI druhé generace [1].



Obr. 1: Allwinner A31

2.3. Embedded zařízení uvažovaná pro testování

2.3.1. Raspberry PI a jeho verze

V současné době existuje 8 typů mikropočítače Raspberry PI, jsou to modely A, A+, B, B+, 2 B, 3 B, speciální nízkonákladové Zero bez ethernetu a Compute Module s onboard FLASH pamětí. S rozšiřující se výbavou stoupá i spotřeba desky. Zatímco model A bylo možné napájet ze standardního USB 2.0, Model B potřebuje zdroj, který je schopen dodat alespoň 1A podle množství připojených periferií [2].

Raspberry Pi 1:

Původní Raspberry s procesorem ARM 1176 a 256 nebo 512MB paměti SDRAM. Hlavním rozdílem modelů B proti A je přítomnost ethernetu a USB hubu. Na desce se nachází 8 nebo 17 GPIO vývodů podle verze, MIPI interface pro kameru.

Obr. 2 Raspberry PI

Raspberry PI 2:

Novinkou tohoto počítače je procesor ARM Cortex A7 stále na 32bitech ale již s 1GB paměti. Cinch pro kompozitní video byl nahrazen sdíleným konektorem jack 3,5mm.

Obr. 3:Raspberry PI 2

Raspberry PI 3 :

Představeno bylo v únoru 2016 a oproti předchozí verzi doznalo několika zásadních změn:

- ARMv8 CPU quad-core 1,2GHz 64bit
- 802.11n WLAN
- Bluetooth 4.1

U tohoto modelu je proti předchozímu přidána bezdrátová síťová karta bohužel s integrovanou anténou. V tomto případě by byl vhodnější konektor typu U.FL. Provedením této úpravy sice ztratím záruku, ale signál o 20 dB lepší hovoří jasně ve prospěch této modifikace.

Obr. 4:Raspberry PI 3

2.3.2. Beaglebone & Beaglebone Black

Nejnovější generace je postavena na procesoru ARM Cortex-A8 s taktem 1GHz (720MHz u starších verzí). Na desce je 512 MB pamětí DDR3 pro verzi Black a 256MB pro původní. Verze black obsahuje 4GB eMMC paměti (FLASH paměť + řadič v BGA pouzdru). Na desce je vyvedeno 65 digitálních pinů a 7 analogových. Beaglebone podporuje 4x UART, 8x PWM, 2x SPI, 2x I2C, 2xCAN Bus, a několik dalších rozhraní [3].

Obr. 5:Beaglebone Black

2.3.3. MikroTik Routerboard

Firma MikroTik dodává více než 10 let síťová zařízení na platformách PowerPC a MIPS nově také ARM a Tile. Kromě jejich systému RouterOS lze u zvláště u starších modelů jako například RB433AH se slotem pro paměťovou kartu použít libovolný kompatibilní systém. U posledních modelů s procesory ARM a 64 jádrovými procesory Tile již nelze bez zásahů do zařízení tuto funkci využít. Výhodou zařízení tohoto výrobce bylo velké množství dostupných portů ethernet a mini-PCI. Pro testování jsem použil RB433AH s procesorem MIPS [4].

Obr. 6: RouterBoard 433AH

2.3.4. Banana PI

Obr. 7:Banana PI - první generace

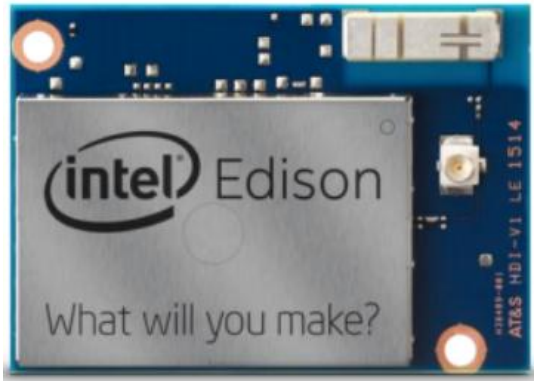
Tento počítač je identických rozměrů a rozložení konektorů jako Raspberry. Jedním z rozdílů patrných na první pohled je přítomnost SATA portu. První verze Banana PI jsou osazeny dvou-jádrovým procesorem ARM Cortex A7 (Allwinner A20), druhá verze obsahuje čtyř-jádrový Allwinner A31S s taktem 1,2GHz. Nejnovější verze Banana PI M3 je vybavena osmi-jádrovým procesorem Allwinner A83T a oproti předchozím verzím má značně rozsáhlejší výbavu. Velikost paměti byla zdvojnásobena na 2GB a přibyla eMMC paměť o velikosti 8GB. Všechny verze obsahují gigabitové síťové rozhraní a od verze 1+ i WiFi standardů b/g/n. Existuje i několik verzí navržených pro specifické účely jako například Banana PI R1 s pěti Gigabitovými rozhraními ethernet vhodný pro malý router s WiFi AP [5].

2.3.5. Intel

Společnost Intel jako jeden z největších dodavatelů procesorů na trhu má ve svém portfoliu i několik procesorů vhodných pro embedded zařízení. Mimo procesorů atom

Obr. 8: Intel Galileo

Intel Galileo jednodeskový počítač založený na platformě x86 který je kompatibilní s počítači Arduino díky čemuž lze použít rozšiřující desky (shieldy) a knihovny pro Arduino. Druhá generace umožňuje navíc instalaci modulu pro napájení pomocí POE po ethernetovém kabelu - norma 802.3af [6].



Obr. 9: Intel Edison

Intel Edison module (verze 2) je miniaturní počítač s rozměrem 25x35x4mm. Je navržen pro aplikaci v oblasti IoT (Internet of Things). Samotný počítač má pouze jeden 70 pinový konektor, díky kterému je možno připojit další periferie. SoC na desce obsahuje dvě jádra Atom Silvermont taktovaná na 500MHz a jedno jádro Quark na 100MHz, vše od společnosti Intel. Integrovaná WiFi a Bluetooth verze 4.0 jsou tak jedinými komunikačními zařízeními přímo na desce. Pro vývoj aplikací s tímto zařízením je potřeba dokoupit desku nazvanou Intel Edison break out board nebo podobnou, které zajišťují přístup k periferiím [7].

2.3.6. Srovnání

Tabulka 1: Srovnání uvažovaných embedded systémů

	RPI A	RPI B+	RPI 2	RPI 3	BBoneBlack	BPI	BPI3	Galileo	Galileo2	Edison v2
CPU	ARM11	ARM11	Cortex-A7	Cortex-A53	Cortex-A8	Cortex-A7	Cortex-A7	X1000	X1000	Atom+Quark
SoC	BCM2835	BCM2835	BCM2836	BCM2837	AM3358	A20	A83T	Quark (x86)	Quark (x86)	Tangier
Instr. sada	32bit	32bit	32bit	64bit	32Bit	32bit	32bit	32bit	32bit	32bit
Počet jader	1	1	4	4	1	2	8	1	1	2+1
Frekvence	700MHz	700MHz	900MHz	1,2GHz	1GHz	1GHz	1,8GHz	400MHz	400MHz	500/100MHz
Paměť	256MB	512MB	1GB	1GB	512MB	512MB	2GB	256MB		1GB
USB	1	4	4	4	1xHost1xClient	2	2	1xHost1xClient	1xHost1xClient	A
FLASH	N	N	N	N	Až 4GB	N	Až 8GB	8MB	8MB	4GB
Ext. úložiště	SD	microSD	microSD	microSD	microSD	SD+SATA	SD+SATA	microSD	microSD	SD***
Ethernet	10/100Mbit	10/100Mbit	10/100Mbit	10/100Mbit	10/100Mbit	1Gbit	1Gbit	10/100	10/100 + POE	N
GPIO *	8/40	17/40	17/40	17/40	65+7x analog.	Až 26	Až 28	14+6x analog.	20	A
WiFi	N	N	N	b/g/n	N	N	b/g/n	N	N	a/b/g/n
Bluetooth	N	N	N	4.1	N	N	4.0	N	N	4.0
GPU	VideoCore IV	VideoCore IV	VideoCore IV	VideoCore IV	PVR SGX530	MALI-400	PVR SGX544	N	N	N
UART	1	1	1	1	4	1	1	2	2	A
IIC	1	1	1	1	2	1	1	1	1	A
SPI	1	1	1	1	2	1	1	1	A	A
CAN	N	N	N	N	2	1	N	N	N	N
PWM	1	1	1	1	8	A	1	6 (8bit)	A (12bit)	A
Časovače	A	A	A	A	4	A	A	A	A	A
Napájení	5V	5V	5V	5V	5V	5V	5V	5V	7-15V	1,8/3,3/5V
Příkon**	1,5W	3W	4W	4W	2W	> 1W	> 4W	>3W	Až 15W	>0,2W
Cena	600 Kč	700 Kč	1000 Kč	1100 Kč	1100 Kč	900 Kč	2000 Kč	2000 Kč	2500 Kč	1400kč

*GPIO - hodnota 7/40 znamená 7 vyhrazených a až 40 celkem. **Příkon - závisí na připojených a zapnutých periferiích. ***SD - přes externí desku (break out board).

Počty periferních zařízení se mohou lišit podle revize, proto je použito následující označení: A - Obsahuje jedno a více zařízení. N - neobsahuje dané zařízení. Číslo - počet daných zařízení.

3. Paměťová média pro embedded systémy

Operační systém pro embedded zařízení je specifický v několika ohledech. Mezi nejčastější omezení patří velikost úložiště, kam jsou vlastní soubory nahrány.

Paměťové úložiště může být lokální nebo síťové. V případě lokálních lze použít celou škálu zařízení od pevných disků přes paměťové karty až po FLASH paměti připájené přímo na desce.

3.1. Lokální úložiště

3.1.1. Pevné disky

Jedním z nejpoužívanějších úložišť pro data jsou pevné disky s magnetickým záznamem jejich výhodou je vysoká spolehlivost, dostatečná rychlost a cena v poměru ke kapacitě i menší než 1Kč/GB. Nevýhodou je horší výkon při práci s velkým množstvím malých souborů, hmotnost a rozměry. Mimo několika málo typů nemají embedded zařízení port vhodný pro připojení pevných disků nebo z nich nejsou schopné načíst operační systém. Dnes je standardem rozhraní SATA 2 nebo 3 s rychlostmi 3 a 6 Gbit/s pro běžné PC a SAS-2 a 3 pro servery s rychlostmi 6 a 12Gbit/s. V roce 2018 bude dostupná další specifikace - SAS4 s rychlostí přes 20GBit/s

Jednou z alternativ pevných disků s magnetickým záznamem jsou SSD. Tyto disky jsou složené z FLASH pamětí a řadiče, který se mimo jiné stará i o funkce prodlužující životnost disku. Příkladem je funkce TRIM, která umožňuje rovnoměrné rozložení zápisů na disk, aby nedocházelo k jeho degradaci nadměrným opotřebením buněk.

3.1.2. Paměťové karty

Jedním z nedostatků paměťových karet je jejich rychlost, která je závislá na kvalitě a třídě karty. Rychlost čtení i zápisu u horších karet může být pod hranicí použitelnosti pro médium, na kterém jsou soubory operačního systému. Kromě přenosové rychlosti je důležitá i přístupová doba a počet I/O operací za vteřinu. Následující tabulky jsou výsledkem orientačního měření pomocí nástrojů dd, hdparm a bonnie++. Pro měření byla použita USB3 čtečka.

Tabulka 2: Orientační rovnání rychlostí paměťových karet

Typ karty	**Velikost [GB]**	**Náhodné čtení [MB/s]**		**Náhodný zápis[MB/s]**	
		512KB	**4KB**	**512KB**	**4KB**
SDHC Class4	4	3,2	0,1	4,8	0,05
SDHC Class10	16/32	21,7	2,9	17,4	0,1
SDXC Cl.10-I	64	72,8	3,8	56,3	1,7

3.1.3. Ramdisk

Ramdisk je vyhrazená část paměti, se kterou lze pracovat podobně jako s pevným diskem. Výhodou ramdisku je rychlost několikanásobně převyšující ostatní média. Zatím nejblíže rychlostí jsou SSD disky s M2 rozhraním, kde teoretická maximální rychlost je 32Gbit/s (4GB/s). Reálná rychlost je zatím cca třetinová. Hlavní nevýhodou ramdisku je to, že data

jsou uložena pouze do restartu systému. Existovalo i několik řešení specializovaných karet do PCI a PCI-express kdy paměť byla zálohována baterií - například Gigabyte i-RAM. S poklesem ceny pamětí a SSD a jejich zvětšující se kapacitou se toto řešení stalo zastaralým.

Tento typ disku lze využít při načtení systému ze sítě, kdy se jádro načte přímo do paměti a při jeho startu je vytvořen ramdisk pro data aplikací.

V následující tabulce je výsledek orientačního testu rychlosti přesunu dat z ramdisku do ramdisku, čtení z ramdisku a zápis do ramdisku. Test byl proveden pomocí příkazu dd pro 4GB dat. Pro srovnání je přidán SSD.

Tabulka 3: Orientační test rychlosti ramdisku

Paměť - velikost	Kopírování [GB/s]	Čtení [GB/s]	Zápis[GB/s]
2CH - 4GB	3,1	6,4	3,2
3CH - 4GB	6,5	8,1	6,7
SSD SATA-3	0,26	0,48	0,41

```
#výroba ramdisku
mount -t tmpfs -o size=4g tmpfs /mnt/ramdisk
#test zápisu 4GB souboru do ramdisku
dd bs=1M count=4000 if=/dev/zero of=/mnt/ramdisk/foo oflag=dsync
```

V případě že potřebujeme určitý souborový systém, například ext4 je postup trochu složitější. Nevýhodou je pokles rychlostí přibližně na polovinu:

```
mount -t tmpfs -o size=4g tmpfs /mnt/ramdisk
dd if=/dev/zero of=/mnt/ramdisk/disk.img bs=1M count=4000
losetup /dev/loop0 /mnt/ramdisk/disk.img
mkfs.ext4 /dev/loop0
mount /dev/loop0 /mnt/disk
```

3.1.4. Životnost Paměťových médií

Dobu životnosti paměťových karet, SSD disků a médií s FLASH pamětí všeobecně ovlivňuje zejména počet přepisů jednotlivých buněk paměti. To je problém pro části paměti, které se často opakovaně přepisují jako například oddíl swap nebo soubory databází.

U pevných disků s magnetickým záznamem je životnost udávána v pomocí parametru MTBF, což je střední doba mezi poruchami. Ta se pohybuje podle typu a určení disku od stovek tisíc po jednotky milionů hodin. Tato hodnota je samozřejmě dále ovlivněna provozními podmínkami a zátěží. Důležitou hodnotou je také parametr BER. Ten udává poměr chyb k celkovému objemu přečtených dat.

3.2. Síťová úložiště

Síťové úložiště dat je jeden ze způsobů jak uchovávat data operačního systému pro embedded zařízení. Jeho výhodou je variabilita použitých zařízení a typů úložišť. Tímto způsobem je možno realizovat i technická řešení, která na samostatném embedded systému nejsou možná.

Jedním z příkladů je záloha dat na RAID poli nebo šifrování obsahu kdy se o samotné šifrování stará výkonný server.

3.2.1. NFS - Network File System

NFS je protokol který umožňuje přístup k souborům přes síť. První dvě verze fungovaly pouze na UDP, verze 3 a 4 podporují i přenos nad protokolem TCP. Připojením disku získáme dle nastavení přístup k datům, ale samotné ukládání a práce se soubory je stále v režii souborového systému na serveru.

Původně vyvinut firmou Sun Microsystems - specifikace RFC1094. Dnes aktuální verze je NFSv4 specifikovaná RFC3530, která implementuje další možnosti zabezpečení například přenos ACL - přístupových práv a další. Na rozdíl od CIFS není tento protokol nativně podporován OS Windows [8].

3.2.2. ISCSI

ISCSI - Internet Small Computer Systems Interface je protokol na IP vrstvě který umožňuje "sdílení" zařízení označovaných v tomto protokolu jako target. Tímto zařízením může být celý disk, jeho oddíl, logický oddíl vytvořený na disku, mechanika a další. Na rozdíl od podobných systémů jako například Fibre channel nevyžaduje vlastní infrastrukturu.

3.3. Chyby přenosu čtení, a zápisu dat

Při každém přenosu dat z úložiště, po síti nebo při ukládání vzniká riziko chyby, které může ovlivnit výsledné chování programu nebo operačního systému. V případě uložení dat na síťovém disku musíme počítat i s možnými chybami při jejich přenosu.

Chybovost přenosu ethernetu by měla být dle specifikace 10^{-12} a nižší [9]. O opravy se zde starají algoritmy využívající kontrolních součtů. Při použití TCP protokolu je zajištěno i dodání packetů na cílové rozhraní, které zpětně potvrzuje příjem packetu. To neplatí při použití přenosu za pomoci UDP kdy je packet s daty vyslán a předpokládá se jeho doručení. Vzhledem k této vlastnosti je potřeba zajistit pro UDP dostatečnou kvalitu přenosové trasy včetně správné konfigurace firewallů nebo QOSu. I přes vyšší dosahované rychlosti při použití UDP protokolu pro přenos souborů z NFS jsem raději zvolil NFS přes TCP.

U pevných disků s magnetickým záznamem udávají výrobci parametry URE - Unrecoverable Read Error a BER - Bit Error Rate. Oba parametry udávají pravděpodobnost výskytu chyby při čtení dat z disku. Hodnota se obvykle pohybuje od 10^{12} výš. To znamená jeden chybný bit na přibližně 100GB dat. U disků určených pro vysokou zátěž - například WD RE uvádí výrobce až 10^{15} což je přibližně jeden bit na 100TB dat [10]. Elektronika disku má navíc vlastní opravné algoritmy, takže možnost této chyby je zanedbatelná.

Vzhledem k výše uvedeným parametrům a faktu že samotné jádro má maximálně několik megabajtů budeme pravděpodobnost chyby při přenosu dat zanedbávat. Jediným opatřením pro zvýšení spolehlivosti bude provoz NFS po TCP

4. OS pro embedded systémy

Operační systémy pro embedded zařízení lze rozdělit na dvě základní skupiny a to jsou univerzální a jednoúčelové. Mezi jednoúčelové řadíme firmwary pro zařízení, které jsou navržené pro jednu specifickou činnost a neumožňují větší změny a úpravy.

Druhou velkou skupinou jsou operační systémy založené na plnohodnotných OS pro osobní počítače nebo servery, ale ořezané o nadbytečné softwarové vybavení kvůli limitované kapacitě úložiště.

4.1. OS založené na linuxových distribucích

Téměř každá dlouhodobě vyvíjená distribuce má port vhodný pro použití v embedded zařízeních. Jejich rozsah a přizpůsobení jednotlivým typům závisí zejména na dostupnosti daného hardware. Na následujících řádcích je uvedeno několik příkladů ke každé distribuci. Značné množství těchto distribucí je založeno na OS Debian nebo z něj vycházejícího Ubuntu. Další distribuce, které mají port pro embedded zařízení na různé úrovni přizpůsobení jednotlivým zařízením jsou například OpenSuse, ArchLinux, CentOS, Kali, Fedora a další.

4.1.1. Debian a Ubuntu

celým názvem Debian GNU/Linux je jednou z nejstarších distribucí. Je vyvíjen dobrovolníky jako univerzální operační systém. Díky jeho konzervativnosti je oblíbený především jako serverový systém. Balíky se však dostávají do repozitáře stabilní verze se značným zpožděním proti ostatním distribucím. To ale zajišťuje velmi dobrou stabilitu a spolehlivost celého systému. Tato distribuce přestala od verze 6.0 obsahovat software s uzavřeným zdrojovým kódem v hlavní části repozitáře a pro tyto balíky je nutno přidávat zdroj. Dnes je pro nejpoužívanější desky k dispozici upravená verze Debianu jako image.

Raspbian - oficiálně podporovaný neoficiální port Debianu pro Raspberry PI. Systém je připraven ve dvou verzích kdy standardní image zabírá přes 4GB a obsahuje kompletní OS připravený k práci a verzi Lite která se svými 1,3GB obsahuje pouze to nejnutnější. Dalšími úpravami a zeštíhlováním lze plně funkční systém stlačit pod 500MB [11].

Bananian Linux - Stejně jako v předchozím případě port Debianu. Hotová image o velikosti 1,8GB je k dispozici pro všechny verze první generace Banana PI. Druhá a třetí generace má vlastní image.

Armbian - je dostupný ve třech verzích. Dvě odvozené z Debianu - pro server i desktop a jedna založená na Ubuntu. Seznam podporovaných zařízení zahrnuje několik desítek typů [12].

Ubuntu je systém odvozený od Debianu a je snahou přiblížit toto prostředí běžným uživatelům. Výhodou je rozsáhlá podpora komunity uživatelů v různých jazycích.

Ubuntu MATE - distribuce odvozená z Ubuntu přímo pro počítače Raspberry.

Ubuntu CORE - speciální verze Ubuntu určená pro embedded zařízení.

4.2. Android

Pro některé desky je k dispozici i OS Android známý převážně z mobilních telefonů. Z hlediska kompatibility systému a aplikací je to jistě přínos, ale podpora práce s hardwarem není

4.3. Speciální OS pro embedded zařízení

Zároveň s výše uvedenými distribucemi existuje značné množství specializovaných distribucí vyvíjených výrobci zařízení nebo komunitou uživatelů. Následující příklady jsou pouze výběrem z množství dostupných systémů.

4.3.1. Busybox

Spolu s linuxovým jádrem je BusyBox používaný jako základ firmware pro nejrůznější zařízení. Obsahuje nejzákladnější sadu programů a příkazů se značně oříznutou funkčností odvozených od plnohodnotných verzí ale s identickou syntaxí [13].

4.3.2. OpenWRT

Tato distribuce je dlouhodobě vyvíjená pro aplikace na síťových prvcích a jako náhrada továrních firmware pro bezdrátová zařízení. Obsahuje podporu pro více než 1200 zařízení různých výrobců. Díky minimální velikosti a rozsáhlým možnostem konfigurace je použitelné pro velké množství aplikací například routery, bezdrátové přístupové body a mnoho dalších.

4.4. Distribuce vyvíjené jako multimediální centra.

OpenELEC je distribuce vyvinutá speciálně pro multimediální centra, jako vlastní nadstavba systému zajišťující multimediální funkce je použit KODI media center. Tato distribuce funguje na většině počítačů s procesorem ARM a x86. Podporuje rozsáhlé množství zařízení a modulů pro příjem satelitních pozemních i streamovaných médií.

Existuje ještě mnoho dalších distribucí v různé kvalitě a fázi vývoje. Známé jsou například XBian, GeexBox a další.

4.5. Windows

Společnost Microsoft vytváří embedded verze svých operačních systémů Windows už od verze XP, která byly nasazena například v pokladnách, terminálech nebo třeba bankomatech.

Oficiálně je Windows IoT prvním systémem určeným pro počítače Raspberry, MinnowBoard a DragonBoard, ale existuje několik zajímavých projektů, které ukazují způsob jak provozovat tento systém na hardware jiných výrobců [14].

5. Návrh řešení

Pro danou situaci se jeví jako nejvhodnější uložení dat na centrálním serveru a jejich distribuce pomocí služeb TFTP a NFS jednotlivým klientům. K tomu bude potřeba server, na kterém poběží mimo výše zmíněných ještě DHCP pro automatickou konfiguraci síťových parametrů. Technické řešení nevyžaduje aby DHCP běželo na stejném stroji jako NFS a TFTP, ale rozhodně to zjednoduší práci. Bootloader bude uložen na SD kartě nebo v integrované FLASH paměti pokud ji zařízení podporuje.

5.1. Zabezpečení

Centrální server bude sloužit mimo jiné jako router se dvěmi síťovými rozhraními (fyzické, VLAN atp.). V ideálním případě celou síť, na které bude probíhat testování, oddělíme od internetu dobře nastaveným firewallem a vytvoříme DMZ, kde nám nebudou komunikaci mezi zařízeními narušovat firewally nebo proxy servery a zároveň získáme bezpečné prostředí pro práci.

6. Server - konfigurace

Základní požadavky na server umožňující provoz systémů se spouštěných ze sítě jsou následující: diskový prostor, rychlá síťová karta (1Gbps) a dostatek paměti.

6.1. Hardware

Jako testovací sestava byl použit server na platformě Intel C602 - socket 1366 s procesorem Xeon a 32GB RAM. Pro úložiště dat jsem zvolil 4x1TB disk v RAID-5. Tato sestava poskytuje dostatek výkonu i pro kompilaci software pro méně výkonné platformy.

6.2. Software

Operačním systémem serveru je Debian s označením Stretch. To je zatím verze testing kde je v balících několik užitečných nástrojů navíc proti stabilní verzi. Systém je samozřejmě 64 bitový. Jádro systému je upravené pomocí několik patchů a kompilované jako monolitické (pro běh systému nepotřebuje externí moduly).

6.3. Práce s obrazy souborových systémů

Pro načtení souborových systémů uložených jako image v prostředí Linuxových distribucí se dobře hodí nástroj kpartx dostupný v repozitářích ve stejnojmenném balíku. Vzhledem k jednoduchému ovládání pomocí parametrů je ideální i pro skripty. Následující kód je ukázkou základních operací:

Vypíše, který oddíl disku bude kam připojen:

```
kpartx -l /data/sys-imgs/2016-05-10-raspbian-jessie-lite.img
loop0p1 : 0 129024 /dev/loop0 8192
loop0p2 : 0 2570240 /dev/loop0 137216
```

Připojí oddíly do /dev/loopXpY; parametr -v znamená podrobný výstup:

```
kpartx -av /data/sys-imgs/2016-05-10-raspbian-jessie-lite.img
add map loop0p1 (250:1): 0 129024 linear /dev/loop0 8192
add map loop0p2 (250:2): 0 2570240 linear /dev/loop0 137216
```

Odpojí oddíly z /dev/loopXpY

```
kpartx -dv /data/sys-imgs/2016-05-10-raspbian-jessie-lite.img
del devmap : loop0p2
del devmap : loop0p1
loop deleted : /dev/loop0
```

6.4. Software - Crosscompilery

Protože embedded zařízení neoplývají nadbytkem výkonu, je vhodné provádět kompilaci zdrojových kódů na stroji s určitou výkonovou rezervou. K tomu potřebujeme několik nástrojů. Balík takovýchto nástrojů se nazývá toolchain a obsahuje čtyři základní skupiny programů. Prvním jsou binutils, jejichž součástí jsou assembler a linker. Druhým je vlastní compiler, třetím balík knihoven pro jazyk C a posledním volitelným je debuger.

Pro architekturu ARM je toolchain přímo v repozitářích jako balík gcc-arm-linux-gnueabi. Samozřejmě lze sestavit vlastní sadu nástrojů, v případě že některý ze stávajících nevyhovuje.

6.5. DHCP server

DHCP server (dynamic host configuration protokol) je síťová služba která umožňuje nastavení parametrů klientských zařízení [15]. Základní funkcí je přidělení IP adresy, které může být statické - IP adresa je svázána s MAC adresou, nebo dynamické kdy je zařízení přidělena volná adresa z poolu daného konfigurací serveru. Dnešní servery fungují i v čistě dynamickém režimu v kombinaci obou způsobů. Jedna IP adresa z poolu je přidělována jednomu klientu, dokud jsou pro nové klienty volné IP, pak jsou přidělovány i ty které byly dříve přiděleny specifickému hostu. Pro linuxové systémy je dostupných několik DHCP serverů. Zvolen byl na základě zkušeností isc-dhcp-server.

6.5.1. Konfigurace

Před začátkem vlastní konfigurace je potřeba serveru říct na kterých rozhraních má poslouchat. Toto se provede pro eth0 následujícím příkazem:

```
echo INTERFACES=\"eth0\" > /etc/default/isc-dhcp-server
```

vlastní konfigurace je uložena v souboru /etc/dhcp/dhcpd.conf. V první části je nastaveno několik společných parametrů pro všechny klienty. První dva parametry jsou nastavení jména domény a DNS serverů následované omezením délky "pronájmu" IP adresy a nakonec je direktiva authoritative, která říká, že server je hlavní, pokud je v sítí více DHCP serverů.

```
option domain-name "muhe.cz";
option domain-name-servers 192.168.0.1, 8.8.8.8;
default-lease-time 7200;
max-lease-time 7200;
authoritative;
```

V další části je definováno nastavení pro rozhraní. Direktiva shared-network říká serveru, že na síťovém rozhraní bude několik rozsahů. Pak následuje nastavení rozsahu. Prvním parametrem je IP adresa sítě, druhým maska, která udává rozsah. CIDR zápis zatím není podporován. Pro jednotlivé rozsahy nastavíme potřebné parametry, které DHCP server přiděluje klientům. Jejich kompletní seznam je dostupný v manuálových stránkách nebo na webu.

```
shared-network eth0 {
      subnet 192.168.0.0 netmask 255.255.255.0 {
            option broadcast-address 192.168.0.255;   #broadcast
            option routers 192.168.0.1;               #gateway
            option subnet-mask 255.255.255.0;         #subnet
            pool {
                  default-lease-time 900;
                  max-lease-time 900;
                  range 192.168.0.10 192.168.0.100;
            }
      include "/etc/dhcp/clients-0.conf";
      }

      subnet 192.168.1.0 netmask 255.255.255.0 {
            option routers 192.168.0.1;               #gateway
            option subnet-mask 255.255.255.0;         #subnet
      include "/etc/dhcp/clients-1.conf";
      }
}
```

6.5.2. Konfigurace DHCP klientů

V souborech clients-0.conf a clients-1.conf jsou uloženy konfigurace DHCP serveru pro jednotlivá zařízení v příslušném subnetu. Technicky je možné změnit pro jednoho každého klienta veškeré hodnoty nastavené globálně pro celý server. V mnoha případech i chybná konfigurace projde přes parser bez chyby a způsobí v nejhorším případě nefunkčnost serveru. Následující kód je minimální konfigurace přidělování pevné IP adresy klientovi s danou MAC adresou:

```
      host rpi3-2 {
      hardware ethernet       00:11:22:33:44:55;
      fixed-address           192.168.0.32;
}
```

Pro načtení souborů z TFTP je potřeba do konfigurace DHCP pro každého klienta doplnit několik informací. Host-name je název hostitele a lze použít ve scriptech na klientském zařízení. Proměnná next-server udává IP adresu TFTP serveru, který poskytuje binární soubor určený na dalším řádku proměnnou filename.

```
      option host-name        "rpi3-2";
      next-server             192.168.0.1;
      filename                "/data/srv/rpi3-2/uImage";
```

V konfiguraci pro každého klienta lze použít i option root-path. Ten definuje síťový adresář poskytovaný NFS serverem, který má být použit jako kořenový adřesář pro systém.

```
      option root-path        "/data/NFS/rpi3-2";
```

U-Boot podporuje i PXE boot a to znamená rozsáhlé možnosti modifikací procesu bootování sytému od menu přes výběr image až po instalaci vlastního systému. Pro použití PXE je potřeba změnit hodnotu proměnné filename následujícím způsobem.

```
      filename "pxelinux.0";
```

V případě že server bude sloužit i pro jiné platformy je vhodné do definice subnetu v konfiguraci DHCP serveru doplnit třídu zařízení, pro které bude hodnoty filename měněna scriptem:

```
class "pxeclients" {
    match if substring (option vendor-class-identifier, 0, 9) = "PXEClient";
    next-server 192.168.0.1;
                if option arch = 00:06 {
                        filename "pxelinux/bootia32.efi";
                } else if option arch = 00:07 {
                        filename "pxelinux/bootx64.efi";
                } else {
                         filename "pxelinux/pxelinux.0";
                }
          }
```

6.6. TFTP & PXE

6.6.1. TFTP

Trivial File Transfer Protocol je způsob který přenosu souborů [16]. Obsahuje pouze základní funkce FTP protokolu a je specifikován RFC1350 a několika aktualizacemi. Často se využívá pro nahrávání firmware do vnitřní paměti embedded zařízení a zvláště síťových prvků. V nejjednodušším nastavení nevyžaduje autentizaci.

Konfigurace se provádí v souboru /etc/default/tftpd-hpa a jeho obsah je přibližně následující:

```
TFTP_USERNAME="tftp"
TFTP_DIRECTORY="/data/srv"
TFTP_ADDRESS="0.0.0.0:69"
TFTP_OPTIONS="--secure"
```

Adresářová struktura serveru je členěna podle zařízení.

```
/data/
└── srv/
    ├── bgb1-1
    ├── bpi1-1
    ├── lnx1-1
    ├── rpi1-1
    ├── rpi2-1
    ├── rpi3-1
    └── rpi3-2
```

6.6.2. PXE

PXE (Preboot eXecution Enviroment) je technologie umožňující start počítačů ze sítě [17].

PXE klient se pokusí ze serveru načíst postupně konfigurační soubor odpovídající jeho ID, MAC adrese, IP adrese v hexadecimálním formátu a na konec soubor default. Při nalezení prvního shodného už dál nepokračuje. Například takto:

```
      /data/srv/pxelinux.cfg/ca8bc98-d9f6-4cd9-6ffd-74a6a4583d
      /data/srv/pxelinux.cfg/00-11-22-33-44-55
      /data/srv/pxelinux.cfg/C0A8000A
      /data/srv/pxelinux.cfg/C0A8000
      /data/srv/pxelinux.cfg/C0A800
      /data/srv/pxelinux.cfg/C0A80
      /data/srv/pxelinux.cfg/C0A8
      /data/srv/pxelinux.cfg/C0A
      /data/srv/pxelinux.cfg/C0
      /data/srv/pxelinux.cfg/C
      /data/srv/pxelinux.cfg/default
```

Načtení jádra je provedeno na základě následujícího kousku kódu:

```
default rpilinux
label rpilinux
        menu label ^rpilinux
        menu default
        kernel bpi1-1/boot/uImage
prompt 1
```

Sekcí označených label může být několik a tímto způsobem lze vyrobit menu například pro boot různých systémů, instalaci systému na stanice nebo jejich automatickou obnovu.

6.7. NFS server

Pro Debian je NFS server dostupný v balíku NFS-kernel-server. Adresářová struktura serveru odpovídá struktuře složky TFTP serveru. Konfigurace je velmi jednoduchá.

6.7.1. Konfigurace NFS serveru

Konfgurace sdílených adresářů je uložena v souboru /etc/exports který má přesně danou strukturu a netoleruje skryté znaky - mezery tabulátory atp. Následující příklad konfigurace ukazuje sdílení adresáře /data/srv/rpi2-1 pro celý segment 192.168.0.0/24 s právy rw:

```
/data/srv/rpi2-1 192.168.0.0/255.255.255.0(rw,sync,no_root_squash) \
```

Vybrané proměnné použitelné v konfiguraci (výtah z manuálových stránek) [8]:

```
IP Adresa:
      ip-adresa
      ip-adresa/maska např.: 192.168.0.0/24
      wildcard - zástupný znak: ? pro jeden znak * pro více znaků
secure  -  port ze kterého byl požadavek vyslán musí být <1024
rw | ro -  zapisovatelný | nezapisovatelný
async   -  server může odpovědět před uložením změn na dsk
sync    -  server odpovídá až po zápisu na disk.
no_wdelay - vynutí okamžitý zápis dat na disk
root_squash - mapuje uživatele root na anonymous
no_root_squash - root má přístup jako root
all_squash - všichni uživatelé degradováni na anonymous
```

6.7.2. NFS & ramdisk

Cílem bylo ověřit nápad jednoho z mých kolegů, jestli se změní rychlost přenosu dat a provádění příkazů, když budou sobory systému uložené v ramdisku na serveru připojené po NFS. Vzhledem k použití 100Mbit rozhraní na většině testovacích zařízení se změna rychlosti neprojevila a změna přístupové doby nebyla průkazná. Jediný přínos byl znatelný v momentě, kdy se disky pro úsporu energie vypnuly. Tento nedostatek lze jednoduše vyřešit nastavením řadiče nebo použitím disku SSD.

Sdílení části ramdisku jsem využil pouze k ověření rychlostí přenosu NFS abych eliminoval zdržení při čtení dat z paměťové karty. I v tomto případě je rychlost limitována nejčastěji rychlostí rozhraní.

7. Boot embedded systému

Většina malých embedded zařízení neobsahuje BIOS nebo podobný software. Z tohoto důvodu potřebujeme bootloader, což je velmi malý program který se načte do paměti zařízení a provede základní operace potřebné pro spuštění systému (načtení jádra do paměti a podobně) a následně systém spustí. V některých případech je jeho funkce sloučena s updatem nebo obnovením systému. Následující stránky popisují postup pro desky Raspberry PI a přesto že postup je téměř univerzální je vždy potřeba ověřit požadavky pro každý typ zařízení. Pro boot ze sítě jsem zkoušel podrobněji tři bootloadery U-Boot, Barebox a projekt založený na Grub2. Z těchto tří jsem nakonec vybral U-Boot, zvláště kvůli rozsáhlé dokumentaci a podpoře širokého množství zařízení.

7.1.1. Boot operačního systému z lokálního úložiště

Následující popis platí pro desky Raspberry PI první generace. U ostatních embedded zařízení bude proces trochu odlišný.

1. zapnutí napájení
2. ARM je držen ve stavu resetu
3. GPU načte bootloader prvního stupně z ROM v SoC, který přečte z karty soubor bootcode.bin a přesune ho do paměti L2 cache a následně spustí.
4. bootloader druhého stupně zapne SDRAM a načte firmware start.elf
5. start.elf přečte soubory cmdline.txt, config.txt
   ----- zde vkládáme vlastní bootloader místo souboru kernel.img -----
6. start.elf spustí soubor kernel.img nebo jiný, který je určen v souboru config.txt
7. načtení systému

Díky nahrazení původního jádra naším bootloaderem, dostáváme mezistupeň v celém procesu, kde můžeme ovlivnit, odkud bude načten systém a mnoho dalších parametrů.

7.1.2. Boot operačního systému ze sítě - U-Boot

U-Boot (v originále Das U-Boot) je opensource projekt bootloaderu používaný převážně pro embedded systémy, i když je možno jej použít i pro PC založených na x86 platformě. Vývoj tohoto bootloaderu začal před rokem 2000 a byl zaměřen na platformu PowerPC. Koncem roku 2002 došlo ke změně názvu na U-Boot a přidání podpory architektury x86. V dalších letech byla postupně přidávána podpora procesorů MIPS32 MIPS64. Dnes obsahuje balík zdrojového kódu konfiguraci pro více jak 1100 zařízení. V tomto případě se nejedná o zcela rozdílné desky ale například i o různé verze a revize jednoho zařízení [18].

U-Boot má několik vývojových větví. Hlavní oficiální je nazývaná Mainline a jde o stabilní verzi. Při zpracovávání této práce nebyl v oficiální větvi dostupný soubor s konfigurací a device model pro Raspberry PI 3 a několik dalších drobností, proto jsem použil zdrojový kód od S. Warrena.

Jedním z hlavních požadavků při výběru bootloaderu byla podpora co největšího množství souborových systémů. U-Boot podporuje kromě EXT2,3,4 i FAT32, VFAT a NFS.

7.1.3. Příprava

Pokud deska nemá vlastní úložiště, například FLASH paměť pro firmware, načítá potřebné soubory z externí paměti. U Raspberry PI je to SD karta. Tuto kartu je nutno rozdělit a naformátovat. Pro první oddíl stačí vyhradit kolem 100MB a naformátovat na FAT32.

Zdrojové kódy stáhneme z git repozitáře následujícím příkazem:

```
git clone  git://github.com/swarren/u-boot.git
```

update zdrojových kódů na aktuální verzi pak:

```
git pull
```

7.1.4. Kompilace a instalace

Před vlastní kompilací je potřeba provést konfiguraci. Ve složce zdrojových kódů u-boot/configs jsou předpřipravené konfigurační soubory pro velké množství desek. Před kompilací je potřeba odstranit jakékoliv staré zkompilované soubory.

pro zjednodušení postupu exportujeme proměnné - složku pro zkompilované soubory, architekturu a crosscompiler:

```
export KBUILD_OUTPUT=/dir/…
export ARCH=arm
export CROSS_COMPILE=arm-linux-gnueabihf-
```

Konfigurace se provede příkazem kdy {TARGET} reprezentuje název desky:

```
make {TARGET}_config
```

Pokud nemáme dostupnou výchozí konfiguraci je nutno ji vytvořit za použití některého z konfiguračních nástrojů například:

```
make menuconfig
```

Stejný příkaz použijeme, i pokud chceme přidat podporu nějakého zařízení. Pak je na řadě samotná kompilace:

```
make
```

Po úspěšném dokončení najdeme v adresáři definovaném exportovanou proměnou soubory u-boot.bin a u-boot které nakopírujeme na připravenou SD kartu. V případě Raspberry PI a jeho klonů musí být formát oddílu FAT32. Po nakopírování souborů můžeme zálohovat původní soubor kernel.img a kernel7.img (stačí přejmenovat) a upravíme soubor config.txt kam doplníme následující:

```
kernel=u-boot.bin
```

karta je tím připravena k použití.

7.1.5. Nastavení

Nastavení bootloaderu se provádí několika způsoby a je závislé na použitém zařízení. Prvním je nastavení přes sériovou konzoli. Následující příklad je pro RPI:

Jako první zkusíme boot z paměťové karty:

```
setenv fdtfile bcm2835-rpi-b.dtb
mmc dev 0
fatload mmc 0:1 ${kernel_addr_r} zImage
fatload mmc 0:1 ${fdt_addr_r} ${fdtfile}
setenv bootargs earlyprintk console=tty0 console=ttyAMA0
      root=/dev/mmcblk0p2 rootfstype=ext4 rootwait noinitrd
bootz ${kernel_addr_r} - ${fdt_addr_r}
```

V případě, že se podaří načíst systém, můžeme pokračovat k dalšímu kroku.

Kompletní seznam proměnných pro prostředí je dostupný na webu projektu U-Boot [19].

Vývojová větev spravovaná S.Warrenem načítá před automatickým bootem soubor uEnv.txt. v případě že máme otestovanou konfiguraci, lze ji uložit přímo do tohoto souboru. Počítá se se začleněním této možnosti nastavení do hlavní vývojové větve v blízké budoucnosti.

7.1.6. TFTP

Pro načtení z TFTP bez použití PXE musíme do příslušné složky přidat soubor obsahující jádro a dtb (device tree blob) soubor příslušného zařízení.

```
setenv fdtfile bcm2835-rpi-b.dtb

usb start
setenv bootargs earlyprintk console=ttyAMA0 console=tty1
      root=/dev/mmcblk0p2 rootwait
dhcp ${kernel_addr_r} zImage
tftp ${fdt_addr_r} ${fdtfile}
bootz ${kernel_addr_r} - ${fdt_addr_r}
```

7.1.7. NFS

Díky tomu že U-Boot podporuje NFS protokol je načtení potřebných souborů otázkou několika řádků. Důležité je dodržet správné adresy pro script.bin pokud je potřebný a pro jádro systému.

```
setenv nfsroot ${serverip}:/data/nfs/rpi2-1
setenv bootcmd nfs 0x43000000 ${nfsroot}/boot/script.bin; nfs 0x48000000
      ${nfsroot}/boot/uImage; bootm 0x48000000
setenv bootargs console=ttyS0,115200 root=/dev/nfs nfsroot=${nfsroot}
      ip=dhcp
saveenv
```

V případě nasazení tohoto způsobu bootování bude pravděpodobně nutné zkombinovat jak stahování souborů z TFTP a NFS.

8. Testování IO

Pro základní otestování funkčnosti periferií jsem kromě sériové konzole potřebné pro nastavení systému používal i dotykový LCD display na deskách Raspberry, znakový display připojitelný přes IIC sběrnici a několik dalších přípravků. U všech testovaných zařízení dané periferie fungovaly a většina jich fungovala i po testování.

9. Chyby a známé problémy

Při testování jsem narazil na několik drobných problémů, které budou vyžadovat další zkoumání.

- Některé distribuce používají starší nebo jiný postup načtení jádra. Je potřeba upravovat image na kartě. Jedním z možných řešení bude z TFTP stáhnout soubor se specifickými příkazy pro dané distribuce.

- Nefunkční USB porty nebo jen některá zařízení v prostředí U-Boot. U některých modelů procesorů lze tento problém vyřešit pomocí patche. Rozšíření podpory USB je slíbeno v další verzi plánované na červenec 2016.

- Image, která je připojená pomocí nástroje kpartx je dostupná lokálně a přes TFTP, ale ne přes NFS. Problémem bude nastavení zabezpečení NFS.

10. Závěr

Všechny body zadání této práce se podařilo splnit. V porovnání embedded systémů jsem vybral ty nejdostupnější. Vzhledem k rozdílům mezi jednotlivými verzemi hardware jsou zde zastoupeny systémy různých výpočetních výkonů, rozměrů i vybavení.

Z uvažovaných operačních systémů vychází nejlépe ty odvozené od Debianu, ale v tomto případě nemusím být úplně objektivní, protože každý uživatel má svou oblíbenou distribuci, která mu určitým způsobem vyhovuje.

Serverová část je kompletní a připravená k nasazení. Stávající server, na kterém byly prováděny testy a pokusy, je značně naddimenzovaný. Pro reálné nasazení bude pravděpodobně stačit i virtuální server. Na bootloaderu U-Boot je potřeba ještě vyřešit načítání distribucí, které vyžadují dodatečné soubory nebo moduly.

V případě testování periferií jsem ověřil základní funkčnost pomocí hotového příslušenství pro dané embedded systémy.

11. Citovaná literatura

[1] **Allwinner Technology.** *Allwinnertech.com.* [Online] http://www.**allwinnertech**.com/en/clq/processora/A31.html.

[2] **Raspberry PI Foundation.** *Raspberrypi.org.* [Online] https://www.raspberrypi.org/products/.

[3] **Beagleboard.org.** BeagleBoard.org. [Online] http://beagleboard.org/bone.

[4] **MikroTik.** *Routerboard.com.* [Online] http://routerboard.com/products/group/11.

[5] **Shenzhen LeMaker Technology Co.,Ltd.** Bananapi.com. [Online] http://www.bananapi.com/product/.

[6] **Intel Corporation.** Intel® Galileo Gen 2 Development Board. *Intel Embedded Design Center.* [Online] http://www.intel.com/content/www/us/en/embedded/products/galileo/galileo-overview.html.

[7] **Intel Corporation.**. Intel Edison Compute Module. [Online] http://www.intel.com/buy/us/en/product/emergingtechnologies/intel-edison-compute-module-iot-463633.

[8] **Haynes & Noveck.** RFC7530 - NFS. Internet Engineering Task Force (IETF). *http://tools.ietf.org.* [Online] https://tools.ietf.org/html/rfc7530. ISSN: 2070-1721.

[9] **IEEE Standards Association.** *ieee.org.* [Online] http://standards.ieee.org/getieee802/download/802.3-2012.zip.

[10] **Wester Digital.** WD Re Datasheet. *wdc.com.* [Online] http://www.wdc.com/wdproducts/library/SpecSheet/CZE/2879-800044.pdf.

[11] **Raspbian.** *Raspbian.org.* [Online] https://www.raspbian.org/.

[12] **Armbian**. [Online] http://www.armbian.com/.

[13] **BusyBox**. [Online] https://www.busybox.net/FAQ.html#whatis.

[14] **Microsoft Corporation.** Windows IoT. *Windows Dev Center.* [Online] https://developer.microsoft.com/en-us/windows/iot.

[15] **R. Droms.** RFC2131 - DHCP. *Internet Engineering Task Force (IETF).* [Online] http://tools.ietf.org/html/rfc2131.

[16] **K. Sollins.** RFC1350 - TFTP. *Internet Engineering Task Force (IETF).* [Online] https://tools.ietf.org/html/rfc1350.

[17] **Intel Corporation.** PXE Specs. *Intel.com.* [Online] 1999. http://download.intel.com/design/archives/wfm/downloads/pxespec.pdf.

[18] **Denx Software Engineering.** *Denx.de.* [Online] http://www.denx.de/wiki/publish/U-Bootdoc/U-Bootdoc.pdf.

[19] **Denx Software Engineering.** *Denx.de.* [Online] http://www.denx.de/wiki/view/DULG/UBootEnvVariables.

12. Seznam Git repozitářů

- U-Boot
  - git://git.denx.de/u-boot.git
  - git://github.com/swarren/u-boot.git
- Barebox
  - git://git.pengutronix.de/git/barebox.git
- Raspberry PI
  - git://github.com/raspberrypi/tools.git
  - git://github.com/raspberrypi/firmware.git
- Banana PI
  - git://github.com/LeMaker/lemaker-bsp.git