Obsah

Kapitola 1

Úvod

Tato práce si klade za cíl zhodnotit aktuálně běžně dostupné způsoby synchronizace mezi systémy připojenými do elektronické sítě. Elektronickou sítí v tomto případě chápeme paketové sítě, zejména Ethernet, a systémem je myšlen jakýkoliv real-time operační systém. V této oblasti je nutné zajistit často vysoce přesnou synchronizaci času pro zajištění dostupnosti služeb i v náročnějších průmyslových podmínkách a nadále speciálních podmínkách jaké nacházíme ve vesmírných programech, mobilních zařízeních všeho druhu (od uživatelských až po řídící systémy v Automotive) a měřících prostředích v laboratořích.

K výše uvedeným účelům bylo již v minulosti vytvořeno několik způsobů synchronizace, avšak tyto byly aplikačně specifické a nedaly se tedy použít na větší množství zařízení a to především z důvodu vzájemné nekompatibility. Prvním pokusem o využití těchto principů a navržení univerzálně použitelného protokolu bylo vytvoření standardu NTP 1.0. Tato verze vycházela z implementace využité pro synchronizaci směrovačů využívajících směrovací protokol HELLO [11], který pracoval s přesností řádově stovky milisekund. Již s touto ranou verzí bylo možné synchronizovat veškerá zařízení připojená do sítě. Záhy se NTP rychle rozšiřovalo a stalo se výhradním synchronizačním protokolem internetu a tento trend platí dodnes.

Protože však tento protokol nebyl schopen synchronizace s vysokou přesností, byl vytvořen standard PTP, zaměřený na dosažení co nejvyšší přesnosti při zachování podpory pro mnoho technologií na druhé síťové vrstvě. PTP narozdíl od NTP definuje i nepovinnou možnost upravovat časové značky i na úrovni druhá síťová vrstvy, což zajišťuje skutečně vysokou přesnost. V laboratorních podmínkách je PTP s podporou na druhé síťová vrstvě schopné dosáhnout přesnosti jednotek mikrosekund.

Tyto protokoly však dává smysl využívat pouze v případě, že máme k dispozici zdroj absolutně přesného času. Takovým zdrojem můžou být různé druhy atomových hodin, rubidiových hodin, cesiových oscilátorů atd. Z těchto zdrojů je poté pomocí pravidelných pulzů přenesena informace o jejich délce do synchronizačního zařízení, na kterém již běží některý ze synchronizačních protokolů. Takovýto vysoce přesný zdroj času však nelze použít v průmyslu a to zejména z důvodu křehkosti a celková náchylnosti takovéhoto řešení k poškození vlivem neoptimálních vnějších podmínek. Jimi je např. teplota, tlak, světlo, velikost takového zařízení, citlivost na chvění atd. Z těchto důvodů se používá další metody synchronizace, a sice za využití GPS.

GPS synchronizace má obdobnou charakteristiku jako výše uvedené přímé zdroje časových pulzů, avšak je dostupná po celé zeměkouli. Reálné nasazení tohoto řešení zahrnuje využití antény, stíněného, co nejkratšího vedení signálu do GPS přijímače, který zpracuje přijatou GPS informaci a zašle synchronizavané stanici nejčastěji po sériovém portu informaci, že

v následujícím časovém okně (majícím velikost např. 200 milisekund) se objeví na PPS (Pulse Per Second) vodiči připojeném ke GPIO pinu na procesoru v synchronizované stanici vzestupná nebo sestupná hrana vytyčující čas např. 25.6.2012 13:30:57 UTC. Tímto způsobem obdrží synchronizovaná stance velice přesný čas a to nezávisle na veškerám okolí. Tento přístup je rozšířen v časově kritických nasazeních, která často ani nebývají připojená k internetu a jedná se tedy pouze o menší LAN sítě.

Protože neustále roste potřeba přesnější synchronizace času nejen díky nárůstu náročnosti bezpečnostních protokolů využívaných v dnešních jak drátových, tak čím dál více bezdrátových sítích, není udržitelná stávající situace s dostupností vysoké přesnosti času pouze za využití GPS jako jediné univerzálně použitelné metody. Mobilní zařízení nemívají kvalitní GPS přijímače a tedy často i v běžných podmínkách selhávají a nejsou tedy dostatečně spolehlivá pro dlouhodobou synchronizaci času. LAN prvky jsou zase často umístěny ve vnitřních prostorách budov apod., kde není možné GPS signál zachytit a většinou též není možné ke každé z takovýchto stanic přivést kabelem samostatný signál z antény umístěné vně budovy.

Tato práce se zaměřuje na porovnání výkonu protokolů PTP a NTP. A to ve srovnatelném prostředí za srovnatelných podmínek, čímž je především myšlena implementační stránka, kdy oba protokoly byly nasazeny na třetí a čtvrté síťové vrstvě, jejich konfigurace co nejvíce připodobněna a až poté byla provedena měření.

Očekávaným výstupem je zhodnocení požadavků těchto protokolů na real-time vlastnosti hostitelského systému, požadavky na síťové zatížení a určení vhodnosti těchto protokolů pro real-time systémy se zaměřením na SCADA (viz. 1.1) programy pod operačním systémem QNX [1].

1.1 SCADA

`SCADA` (Supervisory Control And Data Acquisition) systémy jsou řídící a monitorovací systémy využívané především v průmyslových oblastech, výrobě, automotive apod. Tyto systémy tvoří jeden úzce propojený celek složený z heterogenních prvků, kterými jsou různá čidla a elektro-mechanická zařízení pro řízení strojů, výroby apod. Jelikož tento celek obsahuje mnoho časově závislých procesů, je nutné, aby každý prvek byl řízen stejným časem, a proto tyto prvky zpravidla spadají do tříd `firm` a `hard` v real-time klasifikaci. Je zřejmé, že takto závislé prvky je bezpodmínečně nutné synchronizovat, a to s vysokou přesností. Běžně užívaná metoda synchronizace pomocí GPS se díky rostoucím nákladům za samostatné GPS moduly stává nevýhodnou. Zejména z důvodu, že obdobných prvků vyžadujících přesnou synchronizaci rychle přibývá.

Na stanicích využitých pro měření synchronizačních protokolů byl nainstalován SCADA systém `MCS 02` od firmy Disam RT sloužící jako referenční kus pro pozorování síťové komunikace. `MCS 02` je distribuovaný řídící a monitorovací systém s modulární architekturou skládající se z serveru (měřící stanice), klientů (PLC - koncentrátorů dat) a dispečerské stanice. Předpokládá se, že tento systém by měl být dlouhodobě provozuschopný za využití synchronizace pomocí protokolů PTP a pravděpodobně i NTP.

[1] QNX operating system. *QNX Software Systems Limited* [online]. Dostupné z http://www.qnx.com/.

Kapitola 2

Precision Time Protocol

Protože již v devadesátých letech se začaly ve velkém rozširovat systémy vyžadující real-time zpracování dat a zárověň propojení mezi sebou na velikou vzdálenost, stávající protokol NTP nebyl dostačující a proto byly zahájeny práce na protokolu PTP, jež měl adresovat nedostatky objevující se v NTP. V roce 2002 byl vydán standard PTP 1, který byl v roce 2008 v některých aspektech silně revidován. Nadále je uvažována pouze verze standardu z roku 2008, tedy IEEE 1588-2008.

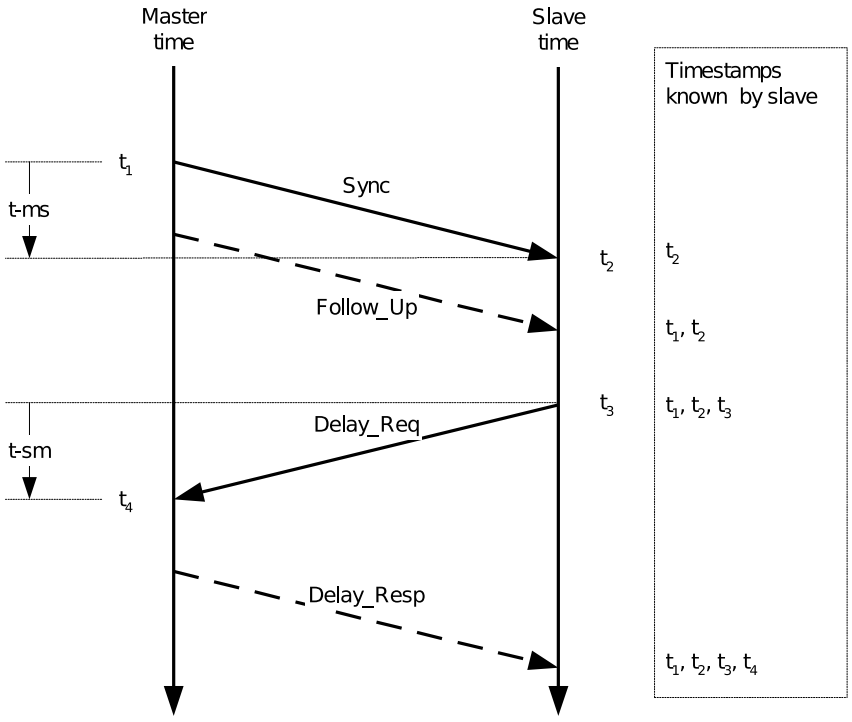
PTP je protokol zajišťující synchronizaci hodin mezi uzly síťové topologie. Tento protokol je navržen pro místa, kde je vyžadována vysoce přesná synchronizace času (odchylky řádově několik desítek nanosekund v laboratorním prostředí) mezi síťovými uzly, avšak z nějakého důvodu není možné použít řešení založené na GPS (viz. kap. 4). Praktické využití tedy nalézá v průmyslových oblastech a to zejména v řídících a monitorovacích systémech SCADA (viz. 1.1).

Principem funkčnosti je vztah `master-slave` mezi dvěma uzly, přičemž `master` periodicky zasílá zprávy, obsahující informace pro korekci času, stanicím `slave`, a to po krátkých intervalech (nejmenší povolený interval je $0.1s$, avšak běžně se používá $1s$). Komunikace probíhá přes multicast s tím, že je možné nasadit čistě unicastovou komunikaci. Ve zprávách je zasílán čas ve formátu `PTP Epoch`, který je přímo odvozený od TAI (International Atomic Time) a čas UTC se dopočítává podle offsetu přítomného v PTP zprávách (viz. 2.2).

Multicastová komunikace probíhá tak, že `master` periodicky zasílá `Announce` zprávy na adresy `224.0.0.107` (`peer delay` zprávy) a `224.0.1.129` (všechny ostatní zprávy) v případě IPv4 a na `FF02::6B` (`peer delay` zprávy) a `FF0x::181` (všechny ostatní zprávy) v případě IPv6 (pro adresování v ostatních podporovaných protokolech viz. [8]). Ukázka jednoduché multicastové komunikace s popisem jednotlivých zaznamenaných časových razítek je znázorněna na obrázku 2.1.

Při unicastové komunikaci `master` rozesílá periodicky `Announce` zprávy všem `slave` uzlům zvlášť. V tomto případě PTP nezajišťuje `objevování` stanic v síti, a je tedy nutné `master` uzlu sdělit které `slave` uzly má obsluhovat. K tomu lze kromě ručního nastavení využít i možností `hraničních hodin`, kterým by byla ručně nastaven seznam `master` uzlů a od nich by byly periodicky vyžadovány `Announce` zprávy.

PTP využívá tzv. domény, které vytvářejí logické celky a tyto se navzájem neovlivňují. Je tedy možné v každé doméně mít jiný čas a zařízení může v každé doméně zastupovat jinou roli. V jedné doméně se však může nacházet pouze jeden uzel `grandmaster clock`, který slouží jako referenční hodiny pro danou doménu, a je tedy vrcholem celé hierarchie PTP uzlů. Tento uzel je volen pomocí `BMC` (Best Master Clock) algoritmu z připojených master uzlů.



Obrázek 2.1: PTP komunikace využívající multicast

BMC algoritmus beží na všech `běžných` a `hraničních` hodinách nepřetržitě a kontinuálně přizpůsobuje stavy fyzických linek změnám v síti a hodinách. Algoritmus je spouštěn při každé významější změně, tedy po přijetí každého nového `DataSet`, `Announce` zprávy atd. Výpočet je prováděn pouze lokálně na synchronizované stanici nad každou fyzickou linkou zvlášť a to ve dvou následujících krocích:

1. porovnání `data sets` každých dvou hodin a vybrání nejlepších z nich

   Cílem je nalézt hodiny, které odvozují svůj čas od lepšího `grandmaster` uzlu, nikoliv hodiny, které jsou lepší (toto je nutná podmínka pro stabilitu algoritmu). Pro porovnání se využívá identita (na rovnost) , priority1, třída hodin, přesnost, koeficient `offsetScaledLogVariance` (viz. [8], kap. 7.6.3.2), priority2 a identita (na nerovnost) v tomto pořadí. Algoritmus je zachycen v [8], obr. 27 a 28 . Tento algoritmus též odhalí chyby, že PTP zpráva byla odeslána a přijata na stejném portu, nebo že zpráva byla zdvojena, případně, že jsou porovnávané zprávy obě od `grandmaster` uzlu.

2. výpočet „doporučeného stavu“ pro každou fyzickou linku

   Při porovnávání je využito charakteristik hodin (třída, kvalita topologie a fyzický port). K porovnání kvality topologie je znovu využit algoritmus pro porovnání `data sets`. Algoritmus je zachycen v [8], obr. 26 .

Ačkoliv jsou podporovány různé síťové topologie, protokol nezajišťuje zasílání zpráv s vlastní eliminací smyček a spoléhá v tomto ohledu na nosné protokoly. Smyčka by mohla způsobit potíže např. při výpočtu zpoždění na uzlu `boundary clock`, protože je podporováno maximálně 255 těchto uzlů. K provozu PTP však není zapotřebí spolehlivé linky (k přenosu

časových informací se využívají nespolehlivé protokoly), ačkoliv je doporučeno předejít zdvojování paketů, jejich vysoké ztrátovosti a změně pořadí.

PTP dále definuje chování pro zařízení, skrze která mají zprávy PTP pouze procházet. Pokud možno, mají tato zařízení upravovat pole `correctionField` ve zprávách PTP, a to tak, že přičtou zpoždění vzniklé přenosem přes vstupní médium a zpoždění vzniklé zpracováním paketu uvnitř tohoto zařízení. Přesnost tohoto postupu závisí na rozdílu délky periody hodin tohoto zařízení a `master` uzlu. Lze tedy tyto průchozí uzly (tzv. transparentní hodiny) synchronizovat s `master` uzlem pasivním pozorováním PTP zpráv (`Sync` a `FollowUp`) spolu s aplikováním potřebných korekcí a poté buď změnit frekvenci oscilátoru (tzv. analogové řešení) a nebo neměnit frekvenci ale spočítat koeficient, kterým se vynásobí změřené zpoždění před přičtením k `correctionField` (tzv. digitální řešení).

Zařízení v PTP doméně jsou rozdělena na následující typy:

I. `ordinary clock` (běžné hodiny)

   Cílové zařízení pro synchronizaci času - může to být `master` nebo `slave`.

II. `boundary clock` (hraniční hodiny)

   Zařízení oddělující jednotlivé segmenty sítě a podporující korekci času v PTP zprávách. Mívá více fyzických portů pro připojení více uzlů.

III. `transparent clock` (transparentní hodiny)

   (a) `end-to-end` (E2E)
   Zařízení podporující `end-to-end delay` měřící mechanismus mezi `master` a `slave` uzly. Toto zařízení tedy neumí měřit zpoždění samotné fyzické linky, nýbrž pouze zpoždění způsobené zpracováním v tomto zařízení.

   (b) `peer-to-peer` (P2P)
   Zařízení podporující i korekci zpoždění fyzické linky. V případě přítomnosti těchto transparentních hodin se pro měření zpoždění mezi `master` a `slave` využívá `peer-to-peer delay` mechanismus.

IV. `management nodes` (řidící uzly)

   Zařízení pro konfiguraci a monitorování hodin (není však podmínkou pro provozování PTP).

Pro měření zpoždění se využívá dvou mechanismů. Mechanismus `request-response delay` měří průměrnou dobu, jakou trvá zaslání zprávy od `master` uzlu ke `slave` uzlu. Zde narážíme na problém s různou délkou zpoždění pro příchozí a odchozí směr komunikace a je tedy nutné navíc provést asymetrické korekce (viz. [8], kap. 11.3). Druhým mechanismem je `peer delay`, který měří zpoždění způsobené propagací skrze fyzickou linku (pro každou zvlášť). Princip tohoto měření je popsaný v [8], kap. 11.4 .

Základní vzorec (viz. 2.1) pro výpočet zpoždění nebere v úvahu asymetrické korekce a využívá mechanismus `request-response delay`. Časy t_n odpovídají časům na obrázku 2.1 a δ je offset mezi časem `master` a `slave`. Předpokládá se tedy, že hodiny na `master` i `slave` se mezi časy t_1 až t_4 nezrychlí ani nezpomalí (tedy, že offset je konstantní).

$$\begin{aligned} t_2 - t_1 - \delta &= t_4 - t_3 + \delta \\ \delta &= (t_2 - t_1 - t_4 + t_3)/2 \end{aligned} \tag{2.1}$$

PTP definuje následující typy zasílaných zpráv:

- Zprávy `Sync`, `Delay_Req`, `Delay_Resp` a `Follow_Up` jsou využívány k přenosu časových informací využívaných k synchronizaci běžných a hraničních hodin podle měřeného zpoždění mezi dotazem a odpovědí.
- Zprávy `Pdelay_Req`, `Pdelay_Resp` a `Pdelay_Resp_Follow_Up` slouží k měření meření zpoždění linky mezi dvěmi hodinami implementujícími mechanismus `peer delay`. Toto zpoždění je využito ke korekci časové informace ze zpráv `Sync` a `Follow_Up` v topologiích složených z peer-to-peer transparentních hodin. Běžné a hraniční hodiny implementující `peer delay` mechanismus se tedy synchronizují podle změřených zpoždění linky a informací obsažených ve zprávách `Sync` a `Folow_Up`.
- Zpráva `Announce` ustavuje synchronizační hierarchii.
- Zprávy `Management` slouží k vyžádání a přenosu `PTP data set` udržovaných jednotlivými hodinami. Pomocí nich je možné síť PTP nastavovat, a to zejména při inicializaci a výpadcích. Tyto zprávy si mezi sebou posílají řídící uzly a hodiny.
- Zprávy `Signalling` slouží k výměně veškerých ostatních informací mezi jednotlivými hodinami. Například jsou využívány pro dohodnutí jak často si budou `master` a `slave` zasílat unicast zprávy.

Každá zpráva obsahuje hlavičku, jejíž struktura je pro všechny zprávy totožná. Tato je znázorněna na obrázku 2.2. Je zřejmé, že struktura je nezávislá na technologii přenosu této informace, ačkoliv je PTP nejčastěji využíván ve spojení s IPv4/6 a Ethernetem. Všechny zprávy lze dodatečně rozšířit pomocí `type length value` (TLV) formátu (viz. [8], kap. 14).

Bits								**Octets**	**Offset**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**		
transportSpecific				messageType				1	0
reserved				versionPTP				1	1
messageLength								2	2
domainNumber								1	4
reserved								1	5
flagField								2	6
correctionField								8	8
reserved								4	16
sourcePortIdentity								10	20
sequenceId								2	30
controlField								1	32
logMessageInterval								1	33

Obrázek 2.2: Společná struktura hlavičky všech PTP zpráv

Mezi podporované transportní protokoly patří Ethernet (IEEE 802.3), UDP IPv4, UDP IPv6, DeviceNet, ControlNet a PROFINET. Avšak je možné využít speciálních hodnot v polích určujících typ i pro další nosné protokoly, které nejsou zahrnuty ve specifikaci.

V běžném nasazení se však nejčastější využívá varianty PTP bez podpory na druhé síťové vrstvě, protože v prostředích kde je nasazen je často nutné propojovat několik rozličných linkových technologií a adaptéry za tímto účelem konstruované nemají pro PTP podporu.

2.1 Portace PTPd na QNX

Pro potřeby měření a vyhodnocení byla využita implementace protokolu PTPd verze 2 [10], která byla pro tyto účely nejprve portována do prostředí QNX. PTPd implementuje pouze

podmnožinu IEEE 1588-2008, a sice pouze části potřebné pro splnění požadavků v IEEE 802.1AS (Timing and Synchronization in Bridged Local Area Networks). Tyto části zahrnují zejména podporu pro multicastovou, unicastovou a hybridní komunikaci skrze UDP IPv4/6 a podporu pro `end-to-end` a `peer-to-peer` měřící mechanismy. Hybridní mód využívá multicastová oznámení pouze k objevování nových `slave` uzlů, ale veškerá další komunikace mezi `master` a jednotlivými `slave` uzly probíhá pouze unicastově.

Tato implementace pracuje pouze na třetí a čtvrté síťová vrstvě [20] a z transportních protokolů prozatím podporuje pouze Ethernet. Implementace je však plně připravena pro doplnění podpory pro označování PTP paketů časovými značkami na druhé síťová vrstvě, čímž by se docílilo markantního přiblížení k výše zmíněné přesnosti. PTPd je dostupné na platformách Linux, uClinux, FreeBSD, NetBSD a po portaci již i na QNX.

Při portaci bylo naraženo na problém, že funkce `clock_gettime()` vrací čas v násobcích délky periody tiku mikrojádra (viz. 5) a tedy je naprosto nevhodná pro použití pro vysoce přesnou synchronizaci narozdíl od ostatních operačních systémů podporovaných PTPd. Bylo zjištěno, že žádnou jinou funkci (jedná se především o volání jádra) QNX nenabízí, a že všechny dostupné funkce pracující s časem nemají vyšší přesnost nežli je délka periody tiku mikrojádra.

Z tohoto důvodu byl navržen princip uložení stavu volně běžícího šedesátibitového čítače v obsluze přerušení každého tiku mikrojádra. Při dotazu PTPd na aktuální čas je znovu přečtena aktuální hodnota tohoto čítače a z rozdílu uložené poslední hodnoty a této nové hodnoty spočítán poměr, kde se v právě probíhající periodě tiku mikrojádra nacházím. Toto nebylo dostačující vzhledem k velice proměnnému charakteru `CLOCK_REALTIME`, který byl využíván jako pevný offset, ke kterému se přičítal poměrný rozdíl z hodnot čítače. Implementováno bylo tedy řešení, kdy se uloží aktuální systémový čas pouze při spuštění démona PTPd a nadále se veškeré výpočty času řídí pouze hodnotami z čítače.

Toto řešení by mohlo být nestabilní v případech, kdy integrovaný oscilátor v CPU, na němž je čítač přímo závislý, bude vykazovat příliš vysoký `jitter`. A proto aby se předešlo náhodným extrémním situacím (způsobeným např. prudší změnou teploty prostředí), kdy by se v důsledku příliš měnil rozdíl po sobě zachycených hodnot čítače, a tedy by jaderné volání `ClockAdjust()` zajišťující pozvolnou úpravu systémových hodin tyto neustále zrychlovalo a zpomalovalo a vlastně jimi nepravidelně kmitalo na nepřípustně vysoké frekvenci, byl přidán výpočet váženého průměru zachycené hodnoty čítače z posledního spočítaného a aktuálně spočítaného rozdílu v poměru vah 2:3. Při registraci obsluhy přerušení byl nastaven příznak `_NTO_INTR_FLAGS_END` (viz. 2), protože se nejedná o časově kritickou aplikaci.

Očekává se, že výše uvedené řešení implementace klíčové funkcionality PTPd pod QNX bude stabilní a plně dostačující i pro budoucí verze PTPd s podporou linkové vrstvy.

Kapitola 3

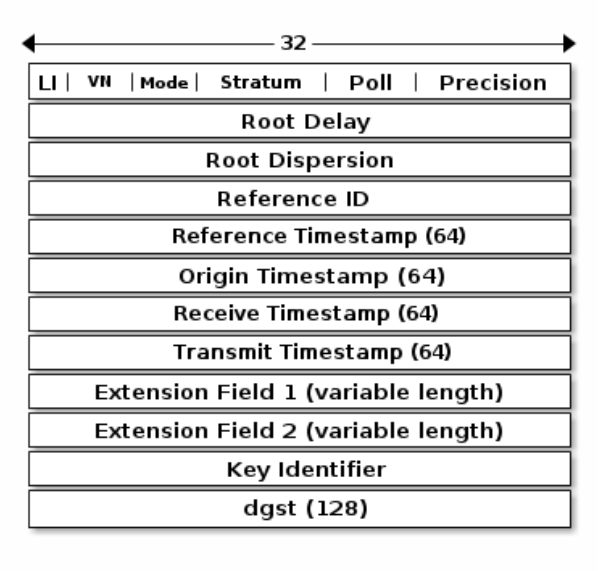
Network Time Protocol

NTP je druhý standardizovaný protokol v historii paketových sítí, hned po protokolu `Internet Clock Service`, jehož je přímým následníkem a náhradou. První verze NTP pod tímto názvem byla specifikována v RFC 958, které popisovalo především obsah paketů a základní výpočet zpoždění (jak je demonstrováno na 2.1). Hlavním důvodem pro jeho existenci v té době byl směrovací protokol `HELLO`, který vyžadoval synchronizaci. Přesnost těchto raných verzí se pohybovala okolo několika set milisekund, což bylo též způsobeno nekompenzováním frekvenčních odchylek a chyb.

NTP verze 1 bylo vydáno roku 1988 v RFC 1059. Tato verze již podporovala symetrický mód i mód klient-server pro asociaci mezi NTP servery. A tedy se mohla začít tvořit hierarchie úrovní serverů poskytujících čas. Tyto úrovně jsou nazývány `stratum`, kde nižší číslo značí přesnější a spolehlivější server. Maximální úrovní je 16, která se považuje za nedostupnost nebo za nedůvěryhodnost daného NTP serveru. NTP verze 2 přinesla podporu symetrické autentizace. NTP verze 3 zahrnula mnoho zásadních změn inspirovaných protokolem DTSS, který byl představen firmou Digital Equipment Corporation. Mezi těmito změnami byla např. inspirace v garantování maximálního a minimálního chybového offsetu, díky čemuž DTSS mohlo být využíváno na kritických místech, kde bylo zapotřebí formální verifikace. NTP však neimplementovalo garanci v podobě úpravy chyb lokálních hodin, ale opravy frekvenčních chyb viz. str. 25 [14] zavedením `maxerr` (maximální chyba, nastavitelné uživatelem) a `esterror` (odhadovaná chyba, nastavovaná přímo démonem). Další změny NTP měly spíše evoluční charakter (automatická konfigurace, zvýšení spolehlivosti, autentizace za využití Publick-Key Cryptography, podpora přímo v Linuxovém jádře apod.). Aktuální verze NTP je 4 z roku 2010.

Princip funkčnosti NTP spočívá v imitaci frekvenčního oscilátoru (PID regulátoru bez derivační složky). Jedná se tedy o jednoduchý fázový závěs (PLL), jemuž se nastavují vhodné koeficienty v závislosti na přijatých zprávách (viz. 3.1) buď z paketové sítě a nebo z lokálního ovladače zdroje přesného času (např. GPS modul poskytující signál PPS). Tento regulátor je přímo „svázaný“ s lokálními hodinami a tedy lokální hodiny kmitají přesně podle jeho frekvence. Toto chování může být v některých podmínkách nežádoucí, ale při vhodném nastavení koeficientů regulátoru utlumit. V případě, že není dostupný žádný zdroj přesného času nebo je přijatá zpráva považována za nedůvěryhodnou (např. obsahuje příliš veliký offset nebo je nastavený příliš dlouhý interval zasílání a přijímání zpráv), přepne NTP do módu FLL (Frequency Locked Loop), kdy se frekvenční rozkmit více ustálí na aktuální frekvenci. Při nevhodně zvolených parametrech regulátoru může dojít k nežádoucímu rozkmitání a tedy znehodnocení systémového času.

Po spuštění démona NTP se kontaktují NTP servery (ať již lokálně fixně nastavené a



Obrázek 3.1: Struktura NTP paketu (zprávy)

nebo objevené skrze multicast 224.0.1.1 v případě IPv4, ff02::101 a ff08::101 v případě IPv6 nebo broadcast) a provede se první výměna zpráv (viz. 3.1) pro inicializaci oscilátoru a tento se spustí. Pokud je v hostitelském operačním systému podpora pro tzv. `Kernel Discipline`, což je označení pro přímou podporu NTP v jádře operačního systému (proměnné využívané v PLL jsou uloženy ve struktuře `timex`), je NTP schopno synchronizace s přesností na 200 mikrosekund. V opačném případě se očekává nezanedbatelně horší výsledek. Simulace regulátoru je počítána od verze 4 v šedesátičtyřbitové plovoucí řádce na rozdíl od verze 3, která je neustále k nalezení nasazená v produkčním prostředí a počítá ve třicetidvoubitové plovoucí řádce. Takto jsou kontinuálně periodicky dotazovány všechny dostupné NTP servery a je vybírán nejlepší z nich podle různých metrik a podmínek [16], přičemž nejprve je podle pozorování chyb a odchylek určeno, zdali je zdroj důvěryhodný či naopak.

Základním předpokladem funkcionality NTP je, že síť je naprosto symetrická [15], str. 4. Tedy že příchozí a odchozí směr komunikace od a k NTP serveru je nezávislý na směrování paketů, různých šířkách přenosového pásma, různých nastavení Quality of Service po cestě apod. Toto je jeden z ohledů, ve kterém se liší NTP a PTP a způsobuje nižší přesnost. Dalším rozdílem je, že NTP je výhradně L4 technologie využívající UDP datagramy na portu 123, a tedy nelze NTP provozovat na jiných sítích než IP, pokud není nasazeno tunelování protokolů. NTP je též schopné se vyrovnat s nespolehlivými linkami využitím nastavitelných koeficientů jako mezí pro důvěryhodnost přijaté zprávy a případným přepnutím z PLL na FLL. NTP podporuje čas pouze ve formátu UTC, a tedy je předmětem přestupné sekundy, pro kterou je v NTP vyhrazeno několik příkazů a struktur zajišťujících pozvolné upravování času v měsící před okamžikem přičtení přestupné sekundy.

Dostupné implementace NTP nejčastěji podporují režim klienta i serveru. Oficiální implementace od prof. Davida L. Millse, PhD, předního návrháře tvůrce všech verzí NTP, obsahuje podporu pro velké množství zdrojů přesného času, které jsou z hlediska NTP speciálními lokálními servery. Dříve bylo nejrozšířenějším zdrojem přesného času pevninské televizní nebo radiové vysílání (DCF77, MSF a WWV), avšak s rychlým rozšiřováním GPS modulů (@gpsmodul) byly tyto téměř vytlačeny. Nejpřesnější podporované zdroje jsou

atomové, rubidiové a cesiové hodiny, které lze taktéž za pomocí referenčního démona použít. Takto synchronizované NTP servery mají stratum 1 a zpravidla nejsou veřejně dostupné k synchronizování, nýbrž pouze NTP serverům se stratum 2, které již veřejně přístupné jsou. Tato referenční implementace nabízí mnohé další možnosti jako např. jednoduché skládání hodin z více zdrojů (nezávisle na lokalitě), základní metody pro řešení výpadku NTP serverů, podporu pro redundanci apod.

Pro účely měření byla využita implementace NTPv4 dostupná v distribuci QNX. Tato verze podporuje Kernel Discipline, avšak informace o tom jak je tato funkcionalita implementována nejsou dostupné. Předpokládá se, že jaderná podpora v mikrojádře nebude příliš vysoká, ale přesto zajistí stabilnější chování než démon v čiste uživatelském prostoru jako v případě portovaného PTPd.

NTP tedy nachází využití v naprosté většině zařízení připojených do paketové sítě. Na internetu je NTP zdaleka nejrozšířenější způsob synchronizace času a to nezávisle na operačním systému. Velká část SCADA systémů je na něm závislá a jedná se tedy o nepostradatelnou položku ve výbavě real-time systémů.

Kapitola 4

Modul GPS

GPS (Global Positioning System) je vojenský družicový polohovací systém americké armády, který se rozšířil do civilního sektoru díky jeho jednoduchosti, univerzálnosti a především tlaku na americkou vládu. Do roku 2000 bylo GPS selektivně dostupné pouze jistým zákazníkům, ale od 1.5.2000 je dostupný všem - za tímto účelem byl navržen nový civilní signál označovaný L2C, jež se začal využívat roku 2005 po vypuštění prvního satelitu s jeho podporou.

GPS systém byl navržen na poskytování dvou informaci - místa a času kdekoliv na planetě. Přesnost pozičního systému se pohybuje okolo 10 metrů, avšak lze ji zvýšit využitím informací z více GPS satelitů a různých aproximačních metod až na jednotky centimetrů. Čas je poskytován ve formátu `UTC` [1] s přesností $10^{-7}s$ pro autorizované subjekty a $10^{-6}s$ pro civilní subjekty [12]. Jednotlivé GPS satelity nesou rubidiové a cesiové hodiny a mezi sebou navzájem se synchronizují s přesností na 20 nanosekund za pomocí broadcastových korekcí času z pozemních stanic. Čas poskytovaný GPS je na satelitech synchronizován s UTC s přesností 100 nanosekund. V [12] je však uvedeno, že náhodný drift je přítomen a nelze ho nijak předpovědět a tedy ani nijak modelovat.

Příjemci GPS signálu obdrží časově posunutý signál, a proto je nutné přijmout signál alespoň ze dvou různých satelitů, aby mohly být provedeny korekce času, v tomto případě offsetu.

GPS tedy poskytuje nejpřesnější běžně dostupnou metodu synchronizace času. Ve `SCADA` systémech se nejčastěji používají samostatné GPS moduly pro každou stanici. Tím je zajištěna velice přesná synchronizace času ve všech stanicích, a to nezávisle na sobě. Tento způsob je však finančně velice nákladný a v mnohých prostředích je komplikované přivést ke každé stanici zvlášť samostatný anténní modul, který je základní podmínkou funkčnosti GPS přijímače. Stále rychleji rostoucí penetrace ethernetové kabeláže v průmyslových oblastech a budovách umožňuje využít některý z výše popisovaných synchronizačních protokolů a GPS modul využít pouze pro stanici s referenčním časem (např. v případě `PTP` by to byla `grandmaster` stanice).

Podporu pro externí zdroj GPS referenčního času lze nalézt především v rozšířenějších real-time systémech jako je QNX, a jedná se tedy z hlediska softwarové dostupnosti a realizovatelnosti o bezproblémové řešení. Toto řešení je též vhodné v prostředích, kde musí být zajištěna absolutní samostatnost jednotlivých synchronizovaných stanic. Jde např. o uzly, jejichž výpadek nesmí ovlivnit zbytek síťové topologie a v žádném případě ohrozit časování v síti. NTP i PTP sice podporují práci v režimech s více `master` hodinami, ale

[1] Coordinated Universal Time, čas odvozený od atomových hodin (narozdíl od GMT, který je odvozený od rotace Země)

v praxi se v takovýchto nasazeních nepoužívají - NTP z důvodu nízké přesnosti a PTP z důvodu malé podpory v zařízeních zapříčiněné přílišnou novostí standardu.

GPS modul dále nebude diskutován, protože pro měření přesnosti NTP a PTP nebyl zapotřebí.

Kapitola 5

QNX

`QNX` je operační systém vyvíjený firmou `QNX Software Systems Limited` od roku 1980 se zaměřením na real-time včetně `hard` real-time. Tento operační systém nabízí kompletní podporu pro běžné uživatelské činnosti - tedy včetne grafického rozhraní, avšak cílí na segmenty, kde je vyžadována vysoká spolehlivost a tvrdá odezva. Mezi nejčastějšími nasazeními QNX nalezneme řídící systémy `automotive`, jaderné a mnohé jiné elektrárny, obranné systémy, medicínské systémy, mobilní zařízení (tablet, smartphone) a v neposlední řadě též `SCADA` systémy.

Nejen vzhledem ke stáří tohoto operačního systému, ale i vzhledem k množství celosvětového nasazení se jedná o velice stabilní operační systém s dlouhodobě udržitelným vývojovým modelem. Stabilita a spolehlivost však nejsou jediné přednosti tohoto systému. Velice důležitým aspektem pro práci s tímto systémem je fakt, že QNX je POSIX [1] kompatibilní a tedy není nutné využívat jeho specifik, pokud to nevyžaduje charakter aplikace.

Protože zaměření tohoto systému jsou především vestavěnné systémy, má QNX velice dobře navžený způsob práce s nízkoúrovňovým programováním. Toho je zajištěno existencí malé skupiny funkcí, které zajišťují atomické čtení nebo zápis předané hodnoty na jisté adrese, jednoduché čtení a zápis registrů veškerých periferií, lehké mapování a uzamykání paměti a především zasílání zpráv mezi procesy, jež je vysoce efektivní a spolehlivé.

Jádro QNX je velice malý mikrokernel, nazývající se `QNX Neutrino` s real-time plánovačem. Mikrokernel je úmyslně postaven bez kontrol aplikací z uživatelkého prostoru, aby se nikde nemohla objevit nepredikovatelná latence. Toto přináší problémy při debugování programů, protože např. nezamčení stránky paměti a zavolání kteréhokoliv jaderného volání nad takovou stránkou způsobí okamžitý a neopravitelný pád systému v případě, že stránka byla dočasně odložena. Meziprocesová komunikace spočívá v zasílání zpráv, kde nad tímto mechanismem je vystavěno i POSIX rozhraní pro semafory a další obdobné mechanismy pro řešení výlučného přístupu ke zdroji.

Časování je v QNX řešeno diskretizací reálného času - všechny události, které se naskytnou v průběhu jedné časové periody jádra, tzv. `tick`, jsou zapsány do fronty v pořadí jejich výskytu a tato fronta požadavků je zpracována až při dalším jaderném tiku. Přesnost časově kritických operací tedy nikdy nebude lepší, nežli délka periody jednoho jaderného tiku. Tento tik nelze nastavit na hodnotu nižší než 10 mikrosekund nezávisle na hostujícím systému. Pokud je naplánována úloha na systémový čas mezi těmito tiky, je spuštěna již v nejbližším nižším násobku tiku.

[1] Portable Operating System Interface, standard vydávaný IEEE zaručující aplikaci využívající toto rozhraní absolutní přenositelnost mezi POSIX systémy

Pořadí provádění nashromážděných požadavků ve frontě lze ovlivnit několika způsoby.

1. změnou systémové `priority` procesu (přenastavením hodnoty `nice`)

   Toto vyžaduje specifikace POSIX. Na rozdíl od většiny ostatních POSIX-kompatibilních systémů přenastavení hodnoty nice zcela pevně ovlivní plánovač mikrokernelu. Tedy platí, že proces s vyšší prioritou (tedy nižší hodnotou nice) se ve frontě provede `vždy` dříve než proces s nižší prioritou. Toto např. v případě Linuxového jádra neplatí, protože plánovač se snaží být co nejvíce férový a ne real-time (nepředpokládáme nyní real-time variantu Linuxového jádra, kde se používá upravený plánovač, který však stejně negarantuje takovouto přednost, nýbrž pouze rapidně zvyšuje pravděpodobnost dání přednosti).

2. použitím příznaku `_NTO_INTR_FLAGS_END` při registraci zpracovávající rutiny pro některá přerušení

   Tento příznak způsobí, že v případě výskytu daného přerušení se rutina spustí až jako poslední rutina vyžadující práci s tímto přerušením. Nastavení tohoto příznaku je běžná praxe. QNX Neutrino mapuje jaderný tik na jedno z dostupných přerušení. V takovém případě tento příznak ovlivní velké množství ostatních procesů se stejnou priroritou a ne pouze procesy mající zaregistrovanou rutinu pro dané přerušení.

Z výše uvedeného tedy plyne, že pokud proces bude spuštěn s nejvyšší prioritou mezi všemi procesy uživatelského prostoru a jako jediný proces mezi procesy s případnou stejnou prioritou nebude mít nastavený příznak `_NTO_INTR_FLAGS_END`, provede se taková akce ihned po provedení všech jaderných akcí.

Kapitola 6

Měření a vyhodnocení

Synchronizační protokoly jsou navrženy často obdobným způsobem a lze tedy předpokládat, že jejich výkon bude podobný. Tento předpoklad se však v některých prostředích může stát chybným jakmile není dodržena specifikace synchronizačního protokolu a nebo není taková situace v protokolu zachycena a tedy chování není definované. V následujících měřěních bude zpracováno jak se tyto protokoly chovají v přítomnosti obyčejného prostředí a zároveň v extrémně náročném prostředí co se zatížení operačního systému a sítě týče.

Dostupné testy těchto protokolů neadresují přímé srovnání ve stejném prostředí za stejných podmínek. Jednotlivé oddělené testy prováděné např. Leeem Cosartem [7] ukazují, že PTP je vysoce přesný protokol, ale jeho výkon závisí na podpoře zařízení, kterými PTP pakety procházejí (měření byla prováděno na zařízeních s hardwarovou podporou časových otisků, ale pakety procházely internetem přes mnoho dalších zařízení bez této podpory). Zpoždění paketů PTP se pohybovala od 29 do 471 milisekund na trase Texas-Kalifornie. Měření NTP [18], [19] ukazují, že lze s tímto protokolem dosáhnout přesnosti přibližně jedné milisekundy při měření rozdílu mezi dvěma stanicemi nezávisle na sobě synchronizovanými ze stejných NTP serverů. Toto meření však nezohledňuje problematiku absolutního zpoždění synchronizovaného času serveru NTP a klienta NTP. Právě tímto měřením se zabývá tato práce, protože právě absolutní čas je směrodatný pro práci vzdálených real-time systémů zahrnujících i SCADA systémy např. pro řízení poruch v elektrárnách, kde je závislost na rychlé reakci přímo podmínkou.

Pro účely měření byly vybrány dva různé průmyslové systémy založené na platformě x86 a x86-64, které vybavením odpovídají počítačům využívaných SCADA systémy. Na tyto počítače byl nainstalován real-time operační systém QNX, který je taktéž častým základem SCADA systémů a byly připraveny nástroje pro komunikaci s tímto systémem za zachování vysoké přesnosti časování. Na jednom z těchto průmyslových PC byl též nainstalován referenční systém SCADA od firmy Disam RT, aby bylo možné sledovat vliv časování na jeho funkcionalitu.

Hlavním cílem následujícího měření je stanovit dosažitelnou přesnost pro získávání a přiřazování časových údajů k zaznamenávaným datům za použití prokolů PTP, NTP a změřit chování těchto protokolů za použití jejich QNX implementací v případě běžných, ale i extrémních podmínek. Tyto hraniční podmínky budou použity pro zhodnocení vhodnosti daných protokolů pro použití v časově kritických nasazeních a v nezvyklých prostředích.

6.1 Očekávávaný výstup

Lze očekávat, že použité implmentace budou vykazovat horší parametry, nežli je uváděno v citovaných dílech. Zvláštní důraz bude kladen na vyhodnocení efektivity přístupů využitých při portaci PTPd, které by měly být téměř shodně přesné jako jaderná podpora v případě Kernel Discipline u NTPd.

Výstupem by mělo být přímé porovnání času z obou stanic a výpočet jejich rozdílu, který by se měl pohybovat v řádech stovek mikrosekund až jednotek milisekund v případě vysokého zatížení sítě, hostitelského PC apod.

Předpokládají se výsledky, že NTP bude méně přesné, ale dlouhodobě stabilnější oproti PTP, které by mělo být schopné okamžitě vhodně reagovat na prudké změny času, avšak přitom udržet kontinualitu času v mezích, které neohrozí funkcionalitu nainstalovaného SCADA systému ani jiných časově kritických aplikací. NTP by se též mělo lépe přizpůsobovat zátěži sítě, protože pro výpočet zpoždění využívá již první zprávy, kterou obdrží od NTP serveru na rozdíl od PTP, které announce zprávy k výpočtu nevyužívá ale vyžádá si znovu zprávu přímo za účelem změření zpoždění. I přes uvedené předpoklady by mělo být portované PTPd schopné reálného nasazení v ostrém provozu, což je též cílem této snahy.

Výsledky měření při kterých byla zatěžována síťová vrstva QNX by měly vyjít u PTP přesně naopak nežli u NTP, protože oba mechanismy využívají výpočet zpoždění zasláním zpráv s časovými razítky (viz. ilustrace 2.1), avšak v případě PTP je tato výměna iniciována `master` uzlem, kdežto v případě NTP `slave` uzlem.

6.2 Vymezení pojmů, popis prostředí

Pro potřeby měření byly zavedeny pojmy označující zdroj absolutního času, ke kterému se všechna měření vztahují - `master` a příjemce časových zpráv - `slave`, který se má synchronizovat se zdrojem absolutního času.

Tyto pojmy budou nadále využívány pro oba testované protokoly. V případě PTP je pojem `master` shodný s pojmem `grandmaster` a pojem `slave` shodný s `slave`. V případě NTP je pojem `master` chápán jako NTP server se stratum 1 a pojem `slave` jako NTP klient s nastaveným serverem, ke kterému se má synchronizovat, představujícím `master`.

Další měřící vybavení se sestávalo z následujících položek, které budou referencovány pod pojmy, pod kterými jsou právě zde uvedeny.

6.2.1 Generátor provozu (Spirent)

Aby bylo možné modelovat reálný síťový provoz a zátěž, bylo nutné zajistit vhodnou infrastrukturu. Ukázalo se, že je nevhodné využít reálnou síť - internet, protože by bylo velice náročné zajistit bezpečnost po dobu měření spolu s přesným dodržením měřících podmínek jako např. nepřetržitý síťový provoz mající určitou šířku pásma a jistou charakteristiku. Z těchto důvodů bylo využito zařízení od firmy Spirent Communications, které je schopné generovat provoz podle popisu, jaký obdrží z řídící stanice.

Všechna měření, která vyžadovala síťový provoz byla provedena se simulací tohoto provozu pomocí Spirent modelu `SPT-2000A`. Toto zařízení bylo vybaveno čtyřmi nezávislými ethernetovými porty o propustnosti 1Gbit/s. Konfigurace tohoto zařízení se provádí skrze uživatelské rozhraní, kde lze naprosto podrobně nastavit všechny detaily o provozu a lze jím simulovat i reálný provoz včetně chyb, různých nestandardních paketů apod. Pro účely měření však postačovalo v některých případech pouze nastavení správného zdroje a cíle

síťového provozu a zejména jeho datové šířky. V případě zatěžování QNX stanic síťovým provozem bylo nutné navíc zajistit, aby pakety přijmuté síťovou kartou byly dále zpracovány jadernou implementací síťové podpory.

6.2.2 FITkit

Hlavní položkou pro měření se stalo zařízení FITkit, které poskytuje mnoho periferií pro vývoj vestavěných systémů. Základní jednotkou je MCU (Micro Controller Unit) MSP430 od firmy Texas Instruments. Toto MCU řídí celý FITkit. Nejzajímavější součástí tohoto kitu je však hradlové pole FPGA, které je připojené na většinu periferií a lze ho tedy využít pro nejrůznější ovládání a velice rychlé výpočty. Mezi ostatním vybavením nalezneme USB převodník emulující dva RS232 kanály, jeden plnohodnotný RS232 TTL převodník úrovní a samostatné oscilátory pro MCU.

Pro účely měření byl využit dostupný mikrokontrolér a FPGA sloužilo pouze pro propojení vhodných periferií za účelem přenosu startovacího signálu měření. V MCU byly využity dvě jednotky záchytu hrany, které mají nominální zpoždění od vystavení hrany po spuštění přerušení 20 nanosekund plus 6 cyklů, přičemž frekvence MCU byla 7372800 Hz (tuto bylo možné zdvojnásobit pro získání dvojnásobné přesnosti, ale ukázalo se, že to nebylo zapotřebí). Celkové zpoždění detekce tedy bylo 0.83 mikrosekundy. Toto zpoždění je však pro měření naprosto irelevantní, protože se jedná o pevný offset přítomný v každém z měření. Nakonec bylo využito USB převodníku pro komunikaci s řídící stanicí, která výsledky měření z FITkitu vyhodnocovala.

6.2.3 Podmínky měření a prostředí

Měření probíhala za pokojové teploty v laboratoři na průmyslových PC připojených do oddělené LAN sítě tvořené jedním nebo dvěma switchi Cisco 3560. Na stanicích byly měřící aplikace spouštěny pod uživatelem root, protože bylo nutné využívat nízkoúrovňový přístup k sériovému portu. Zprvu byly aplikace spouštěny s hodnotou `nice` nastavenou operačním systémem, který ji přideloval vždy o jedna menší nežli rodičovský proces a jelikož byly měřící aplikace spouštěny z emulátoru terminálu, který měl nízkou prioritu, byly i tyto rané výsledky měření značně znehodnoceny jak je uvedeno dále. Poté byla nastavena hodnota `nice` měřící aplikaci na nejvyšší možnou pomocí `nice -n-28` a pro démona synchronizačního protokolu na `nice -n-25`. Takto provedená měření vykazovala diametrálně odlišné výsledky, které odpovídaly očekáváním.

Součástí měřící aplikace `edge_generator` bylo i nastavení délky periody tiku mikrojádra (viz. 5), aby bylo dosaženo co nejpřesnějšího vyvolání rutiny generující měřený signál. Experimentálně bylo ověřeno, že přílišné snížení této hodnoty (až téměř k její spodní hranici) má za následek nepredikovatelnou a častou nestabilitu systému. Protože však bylo nutné zjistit přibližnou hodnotu nejnižší použitelné délky na obou testovacích počítačích, byly spouštěny zátěžové testy ve formě kompilace a postupně upravována tato hodnota až se ustálila přibližně na 40000 nanosekundách, kdy byly systémy stabilní i po dlouhodobém zatížení. Tato hodnota byla tedy využita pro všechna prováděná měření.

Měřící aplikace `edge_generator` sloužila ke generování hran na vodiči DTR na zvoleném RS232 portu. Očekávaným problémem bylo nezanedbatelné zpoždění způsobené hierarchií sběrnic v každém z testovacích počítačů. Ačkoliv obě stanice obsahovaly chipset od firmy Intel, každá měla jiný a s tím i jiný procesor. Protože nebyly k dispozici materiály přesně popisující stavbu sběrnic v těchto chipsetech, předpokládalo se, že zpoždění způsobené propagací skrze tyto sběrnice, ačkoliv nedeterministické (vzhledem k umístění sériového

portu až na nejpomalejší úrovni), je srovnatelé a tedy nebude ovlivňovat zásadním způsobem naměřené výsledky. Před každým měřením bylo sériové rozhraní nastaveno na obou stanicích na totožné hodnoty rychlosti, vypnuto veškeré kešování a veškeré automatizované řízení provozu.

Stanice s CPU Core 2 Duo vykazovala ve všech testech zatížení mnohem vyšší výkon nežli stanice s CPU Atom, což se projevilo nutností nezatěžovat stanici s CPU Atom síťovým provozem, a nebylo tedy možné provést dlouhodobé měření se zatížením obou stanic zároveň. V případě zatěžování pouze `master` nebo `slave` musel být vždy zatěžovaný pouze počítač s CPU Core 2 Duo a tomu přizpůsobeno i spuštění buď `master` nebo `slave` komplementárně na stanici s CPU Atom.

Konfigurace synchronizačních protokolů byla co nejvíce unifikována, aby bylo možné tyto přímo porovnat. V obou případech bylo pevně zvoleno síťové rozhraní, přes které má protokol komunikovat (stanice měly dostupných rozhraní více a tyto by mohly, ačkoliv nepřipojené, ovlivňovat rozhodování např. BMC algoritmu - viz. 2) a dále byla zapnutá maximální úroveň výpisu zpráv na standardní výstup namísto do systémového logu, aby bylo možné pozorovat chování protokolu v reálném čase. PTPd nevyžadovalo žádnou další konfiguraci kromě volby `master` nebo `slave`. NTPd však vyžadovalo konfigurační soubor pro `master` i `slave` a tyto byly nastaveny následovně. Aby bylo možné využít již spočítané hodnoty `drift` při nových spuštěních NTPd a zajistit tedy rychlou adaptaci byla specifikována cesta k souboru driftfile, dále nastaveno, aby NTPd v případě extrémních odchylek tyto vzalo v potaz (volba `tinker panic 0`) a v poslední řadě zajištění co nejkratšího intervalu zasílání NTP paketů (volby `tinker minpoll 4` a `tinker maxpoll 4`, tedy 16 sekund - menší se nedoporučuje, protože na to není interní alogritmus NTP připraven), aby byla dosažena co nejvyšší přesnost spolu s možností co nejpřímější srovnatelnosti s PTPd, které zasílá zprávy každou vteřinu v základním nastavení. Specifické nastavení NTPd `master` a `slave` je popsáno v 6.3.

6.3 Princip měření

Aplikovaný princip měření zpoždění protokolů spočívá v pravidelném, periodickém vystavování hrany iniciovaném z `master` stanice a zároveň ze `slave` stanice ve stejný reálný čas. Délka periody pro generování hrany je přibližně korelována s periodou zasílání synchronizačních zpráv mezi `master` a `slave`. Tato korelace je nutná, aby nebyla měřena přesnost samotných RTC hodin `slave` stanice namísto zpoždění způsobeného propagací zpráv a jejich zpracováním zvoleným synchronizačním protokolem.

Na FITkitu [6] byl implementován program zachycující hrany nezávisle z obou stanic pomocí jednotek záchytu hrany a přerušení. K tomuto účelu byl po resetu FITkitu spuštěn volně běžící šestnáctibitový čítač, nezávislý na aktuálně prováděných výpočtech. Protože tento čítač nemohl dostačovat pro potřeby měření hrany, která byla vystavována QNX stanicema po jedné nebo více vteřinách, bylo nutné simulovat třicetidvoubitový čítač pomocí obsluhy přerušení přetečení tohoto čítače. Rutina obsluhy přerušení detekce hrany obsahovala vždy pouze zápis aktuální hodnoty třicetidvoubitového čítače a posunutí ukazatele na další položku v poli. Tímto způsobem mohlo dojít k posunu nekorespondujících hran v případě výpadku hrany nebo zpoždění většího, nežli délka periody vystavování hrady. Avšak po mnoha měřeních se ukázalo, že tato obava byla mylná, protože k této situaci ani jednou nedošlo. Po pěti stech zachycených hodnotách bylo měření ukončeno a výsledky přeneseny do sběrače (collector) skrze USB.

Tento způsob měření absolutního stavu čítače byl zvolen před způsoby výpočtu rozdílu mezi detekovanými hranami apod. proto, že hodiny ve stanicích `master` i `slave` mohou

vykazovat náhodný `drift` v průběhu měření. Přesnost takto získaných hodnot čítače je závislá na driftu oscilátoru v měřícím nástroji, tedy FITkitu. Osazený oscilátor na FITkitu je běžně dostupný AQ14.745T, vykazující v pokojových teplotách vysokou stabilitu, která se projevuje náhodným zpožděním v řádech desetin nanosekund za vteřinu a tedy má naprosto zanedbatelný vliv na provedená měření. Zpoždění způsobená výpočty při inkrementaci a zapisování hodnoty do pole jsou v řádech desítek nanosekund (instrukce MSP430 mají délku pouze několik cyklů). Tato zpoždění jsou pevná a mohou hrát roli pouze při příchodu přerušení v průběhu obsluhy stávajícího přerušení, k čemuž podle naměřených hodnot nedocházelo (nebyl nalezen ani jeden výskyt). Toto bylo zapříčiněno především malou pravděpodobností způsobenou velkým rozdílem zpoždění mezi vystavovanými hranami stanicemi `master` a `slave`.

```
/* pseudo kód měřícího algoritmu */
master_routine:
    array_master[master_pointer] = get_current_value(COUNTER32)
    master_pointer++
slave_routine:
    array_slave[slave_pointer] = get_current_value(COUNTER32)
    slave_pointer++
start_counter(COUNTER32)
set_sensitive_on_edge(PIN_MASTER)
set_sensitive_on_edge(PIN_SLAVE)
loop while master_pointer < MAX && slave_pointer < MAX
send_results_to_collector(array_master)
send_results_to_collector(array_slave)
```

Zdrojem hran byly stanice `master` a `slave`, které využívaly sériový port k vystavení hrany pomocí signálu DTR, který byl poté napěťovým děličem snížen na hodnoty akceptovatelné jednotkami záchytu hrany na MCU MSP430. Signál DTR přepíná mezi záporným napětím $-9V$ a kladným napětím $+9V$, avšak jednotka záchytu hrany je schopná detekovat pouze kladná napětí. Ukázalo se, že tento nesoulad nezpůsobuje potíže ani výjimečné situace a je dostatečně stabilní. Generátor cíleně nenastavoval příznak `_NTO_INTR_FLAGS_END` (viz. 2), aby se naplánovaná úloha provedla co nejdříve po dokončení jaderných operací QNX.

Generátor hran je spuštěn signálem RD (Receive Data) z RS232, kdy v tomto okamžiku zapíše aktuální systémový čas jako čas kdy má být vystavena hrana a začne kontrolovat v každém tiku mikrojádra, zdali aktuální čas již překročil čas, kdy má být vystavena hrana na DTR a v případě, že ano, tak ji vystaví, ihned vrátí zpět a zvýší hodnotu času, kdy má být vystavena hrana o délku periody, čímž efektivně naplánuje další vystavení hrany. Tímto způsobem je zajištěno monotonické chování a protože signál spuštění je připojen paralelně k `master` i `slave` stanicím, obě zapíší výchozí systémový čas ve stejný reálný čas. K časování tedy nebyly využity standardní prostředky nabízené operačním systémem QNX, protože tyto nedosahovaly potřebné přesnosti a vlastností.

Pro krátkodobá měření byla zvolena perioda vystavování hrany $1s$ a pro dlouhodobá měření $130s$ (právě tato hodnota se ukázala být dobrým kompromisem mezi délkou periody a množstvím uložených dat v MCU s nedostatkem paměti).

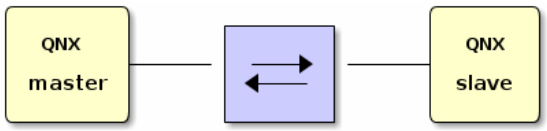
```
/* pseudo kód programu generujícího hrany */
set_clock_period(CLK_PERIOD)
set_serial_port_properties()
```

```
wait_for_RD_signal()
t = round_to_next_nearest_multiple_of(EDGE_PERIOD, get_current_system_time())
loop forever:
    wait_for_next_kernel_tick()
    if t <= get_current_system_time():
        t += EDGE_PERIOD
        set_DTR()
        unset_DTR()
```

Konfigurace démona NTPd v případě `master` stanice obsahovala především nastavení ovladače pro přesný zdroj času, kterým byl oscilátor dostupný v CPU. Tento ovladač byl označen jako stratum 0 pro lokální instanci NTPd, která podle něho nastavovala čas. Je zřejmé, že tento princip má zajímavé důsledky, jelikož se jedná o FIR filtr, tedy ovlivňování sama sebou. V měřeních je tento problém viditelný. Konfigurace `slave` obsahovala pouze adresu `master` stanice, která z pohledu `slave` byla stratum 1 serverem.

Měření byla provedena v různých síťových topologiích a to nejprve bez síťového zatížení, se zatížením LAN sítě a nakonec se zatížením samotných `master` a `slave` stanic. Pro zvýraznění trendu a extrémů jsou body v grafech spojeny. Šipky u přerušované čáry v grafech topologií vyznačují směr toku paketů síťové zátěže. Synchronizační protokoly byly spuštěny vždy alespoň patnáct minut před započetím měření, pokud není uvedeno jinak. U některých měření se objevily nežádoucí chyby např. ve třech prvních vzorcích, a tyto byly ručně odfiltrovány, jelikož zabraňovaly zásadním způsobem v zobrazení ostatních detailů měření. Veškerá zde neuvedená měření lze nalézt v příloze `A` včetně nefiltrovaných a neupravovaných variant. Všechna měření jsou převedena na reálný čas vzhledem ke krystalu ve FITkitu, který nebyl kalibrovaný, a tedy pokud se délka periody vystavování hrany naměřené z master stanice liší pro různé tyto délky přibližně N-násobně, je zjevné, že se jedná o nepřesný přepočet frekvence oscilátoru přítomném ve FITkitu na reálný čas.

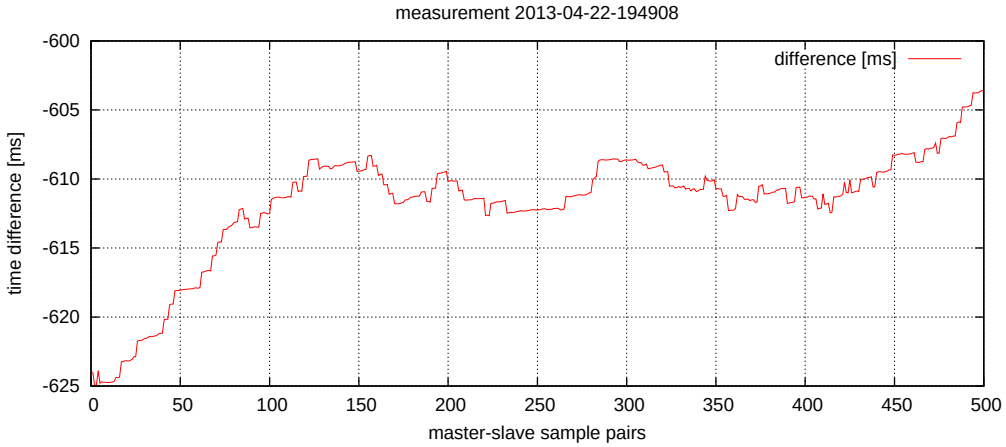
6.4 Jednoduché zapojení s jedním switchem



Obrázek 6.1: Schéma zapojení

1. Měření bez synchronizace s periodou vystavování hrany $1s$ a nezvýšenou hodnotou `nice`

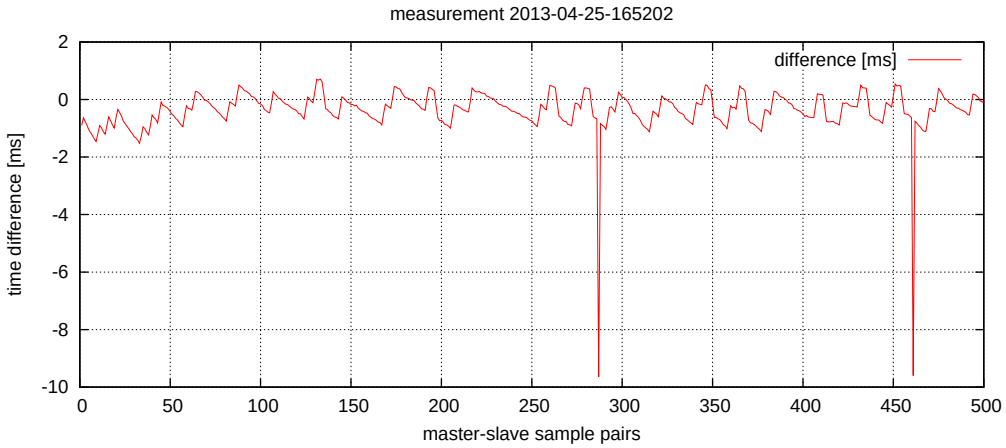
   Následující graf demonstruje vysokou nepřesnost měření způsobenou ponecháním hodnoty `nice` na hodnotě automaticky přiřazené operačním systémem po spuštění procesu. Všechna ostatní dále demonstrovaná měření již mají nastavenou vysokou prioritu a vykazují mnohem lepší výsledky. Za povšimnutí stojí průměrná hodnota rozdílu času obou stanic $-611.8ms$ a medián $-611.1ms$ (absolutní čas se lišil o několik hodin, avšak tímto měřením mělo být zjištěno chování systému při různých hodnotách `nice`).



Obrázek 6.2: Rozdíl naměřených časů mezi stanicemi

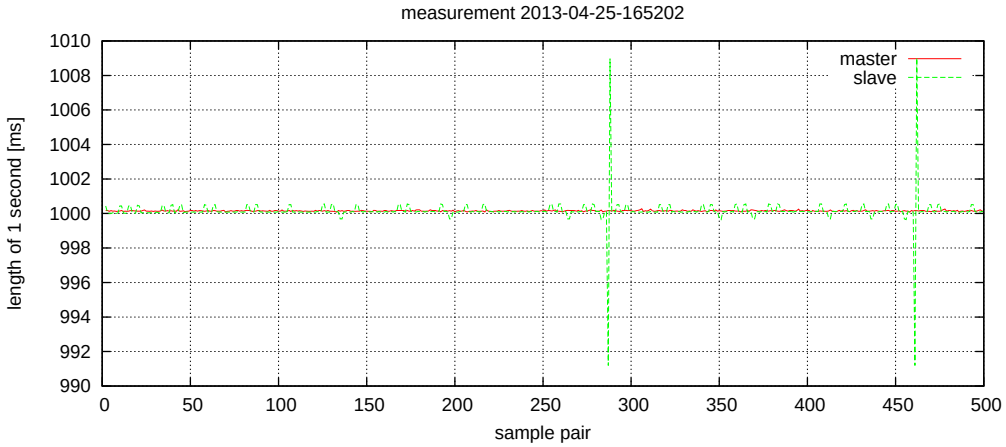
2. Synchronizace PTP, bez zatížení sítě

   Zde je vidět vysoký účinek synchronizace PTP spolu s dostupným servo mechanismem určeným k vyhlazování změn času (tento není předmětem standardu PTP, nýbrž specifikem implementace). Průměr je $-0.367ms$ a medián $-0.318ms$, tedy je vidět tendence `slave` stanice zrychlovat čas oproti `master`.

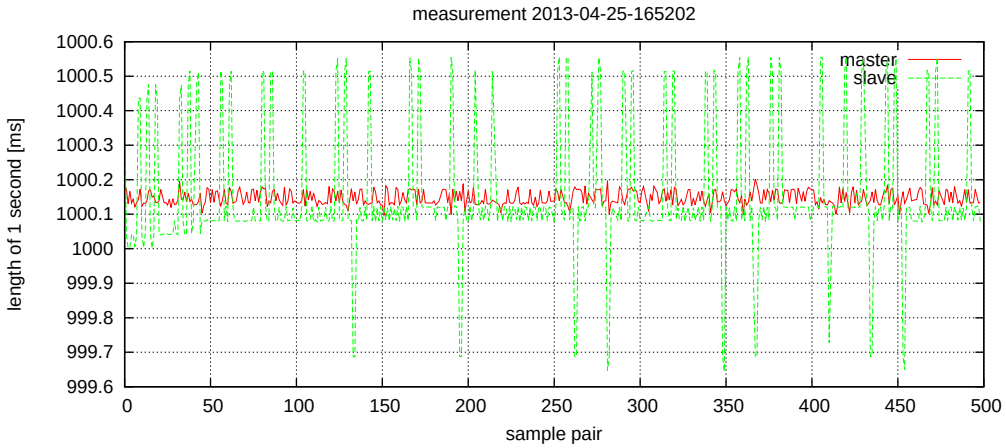


Obrázek 6.3: Rozdíl naměřených časů mezi stanicemi

Následující graf zobrazuje délku period mezi jednotlivými vystavenými hranami. Tedy čím větší výmyk se objeví, tím větší by měla následovat opačná odezva (vyrovnání té jedné „chyby“), pokud má být dodržena stabilita času na stanici. Naměřené výchylky pozorovatelné z předchozího (6.3) a následujícího grafu (6.4) nejsou zjevně vysvětlitelné. Hypotézy zahrnují především výchylku způsobenou zpožděním v hierarchii sběrnic, či implementací ovladače sériového portu v operačním systému QNX. Aby tyto nadměrné výchylky nezpůsobily nečitelnost grafu, je uvedena i filtrovaná varianta grafu, kde všechny hodnoty vzdálenější od průměru více nežli 2σ byly ignorovány. Průměrná hodnota délky této periody činila pro `master` $1000.149ms$ a pro `slave` 1000.151.



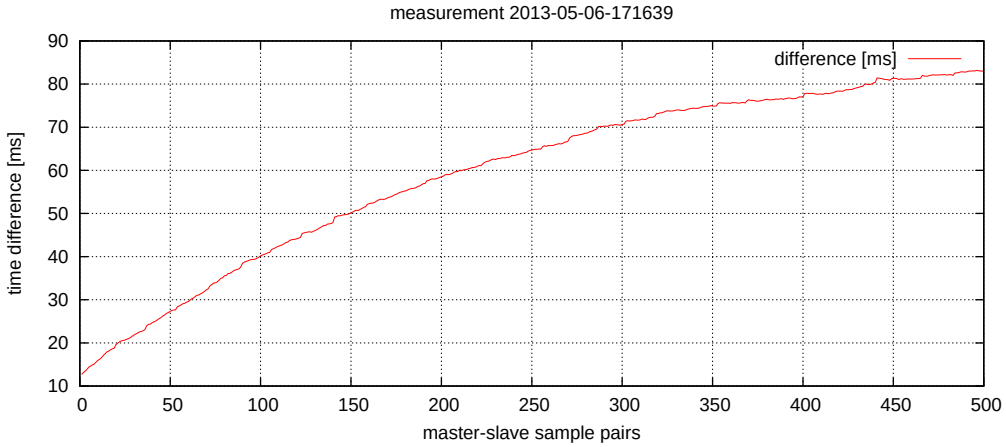
Obrázek 6.4: Délka periody mezi jednotlivými vzorky



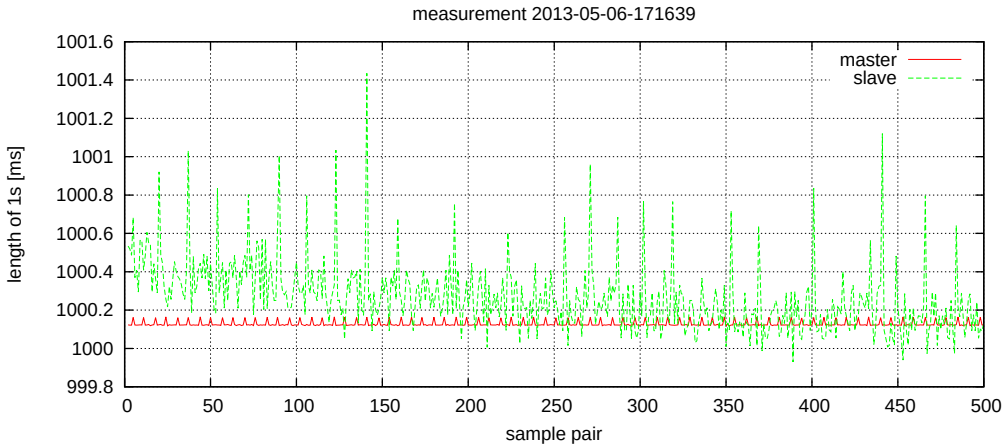
Obrázek 6.5: Délka periody mezi jednotlivými vzorky - filtrováno

3. Synchronizace NTP, bez zatížení sítě

   Z následujících grafů měření, které bylo započato 25 minut po spuštění NTPd lze vidět jak se ustaluje oscilace PI regulátoru v NTP protokolu. Průměrná hodnota rozdílu činila 59.475ms, medián 64.755ms a průměrné trvání doby mezi vystavením hran 1000.128ms na `master` a 1000.269ms na `slave`.

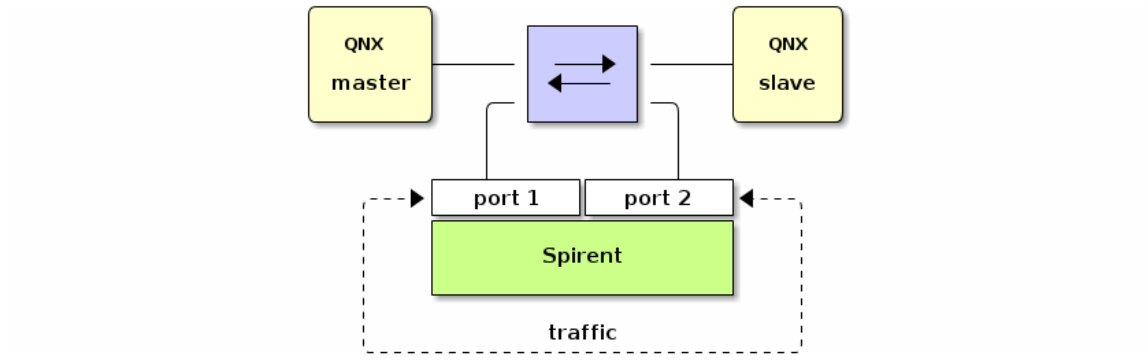


Obrázek 6.6: Rozdíl naměřených časů mezi stanicemi



Obrázek 6.7: Délka periody mezi jednotlivými vzorky

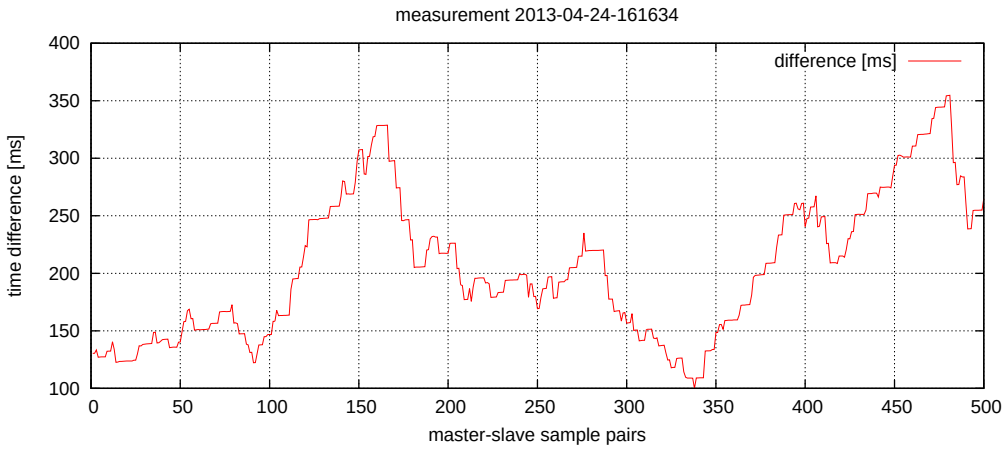
6.5 Zapojení: 1x switch a generátor provozu oběma směry



Obrázek 6.8: Schéma zapojení

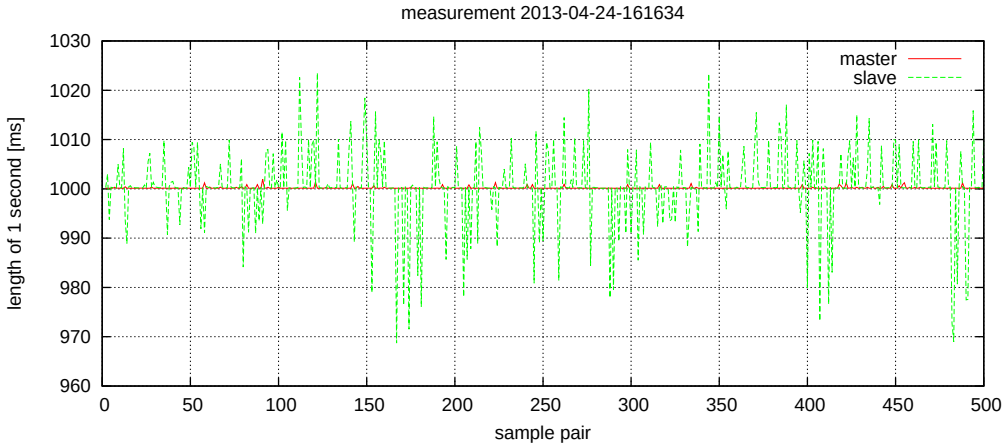
Páteř switche byla zatěžována ICMP Echo Reply pakety, přičemž tento tok zabíral plnou šířku pásma. Směr toku je vyznačen na obrázku 6.8. Bylo provedeno i několik dalších měření se změnou směru toku (tedy ne oběma směry, nýbrž pouze jedním a pokaždé jiným), avšak tato měření se neukazovala zajímavá, a proto nebudou dále diskutována. Lze z nich odvodit, že páteř použitého switche byla dostatečně rychlá a směr toku paketů neovlivňoval synchronizační protokol. Tato zde neuvedená měření lze nalézt v příloze A. Měření se synchronizačním protokolem NTP nebylo z časových důvodů provedeno.

1. Synchronizace PTP



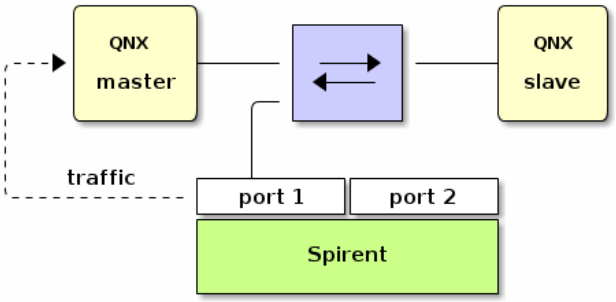
Obrázek 6.9: Rozdíl naměřených časů mezi stanicemi

Na následujícím grafu stojí za povšimnutí problém, kdy se za sebou několikrát objevovaly sekvence délek periody neustále nižší a nižší a poté sekvence výšších a vyšších hodnot. Znamená to tedy, že v těchto chvílích byly systémové hodiny po několik sekund nepřetržitě zrychlovány (perioda uběhla příliš rychle) a nebo zpomalovány.



Obrázek 6.10: Délka periody mezi jednotlivými vzorky

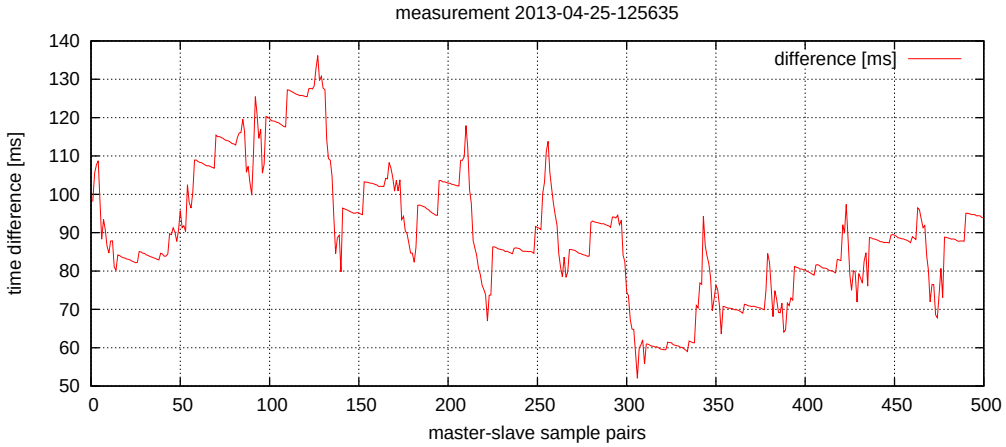
6.6 Zapojení: 1x switch a generátor síťového zatížení na `master`



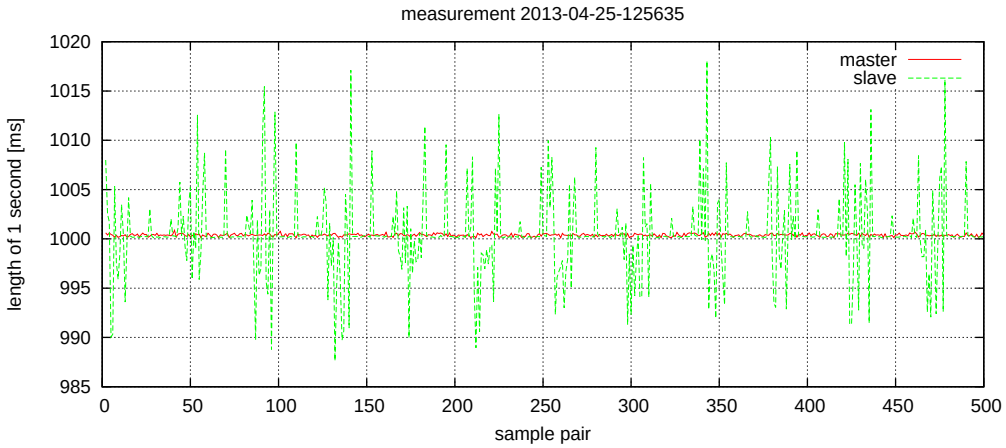
Obrázek 6.11: Schéma zapojení

Stanice byla zatěžována ICMP Echo Reply pakety, přičemž tento tok zabíral plnou šířku pásma. Toto zatížení způsobilo 100% vytížení CPU a tedy plnilo zároveň i roli testu stability systému a použitých démonů pro synchronizaci v takto náročných podmínkách. Měření byla prováděna vícekrát a zde je uvedena pouze nejreprezentativnější varianta. Ostatní lze nalézt v příloze A.

1. Synchronizace PTP



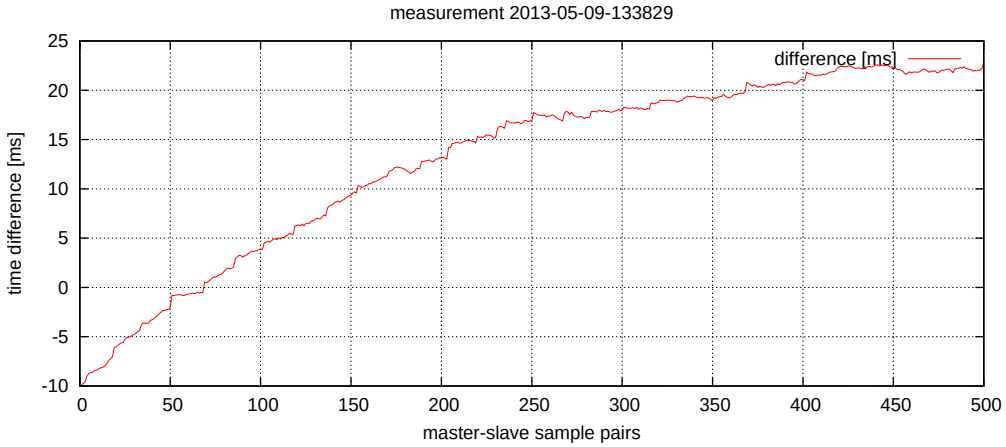
Obrázek 6.12: Rozdíl naměřených časů mezi stanicemi



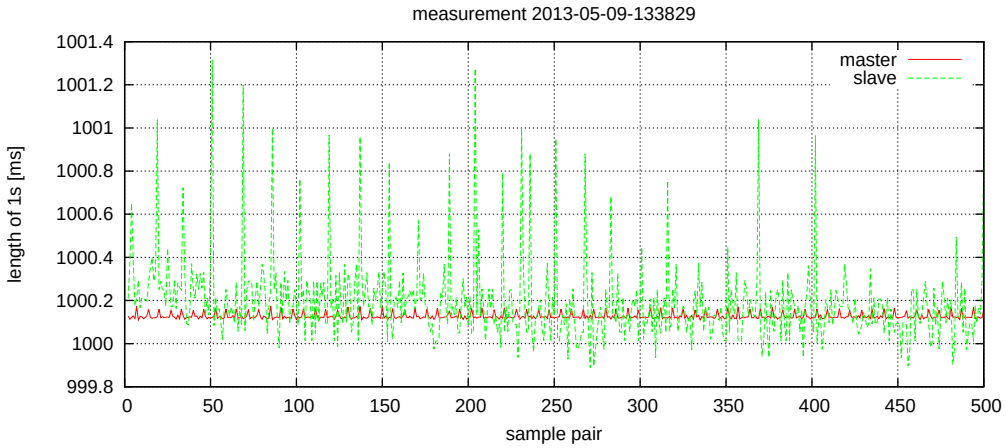
Obrázek 6.13: Délka periody mezi jednotlivými vzorky

2. Synchronizace NTP

   Následující měření bylo započato po sedmi minutách běhu NTPd, a proto lze pozorovat rychlost konvergence.

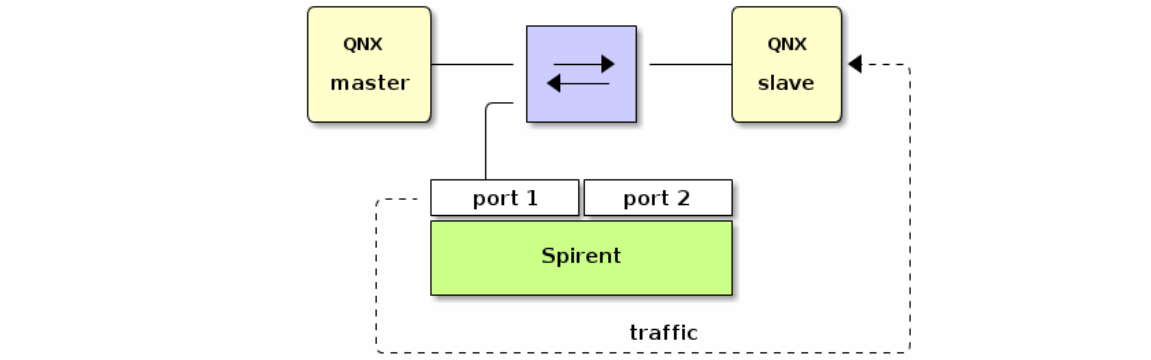


Obrázek 6.14: Rozdíl naměřených časů mezi stanicemi



Obrázek 6.15: Délka periody mezi jednotlivými vzorky

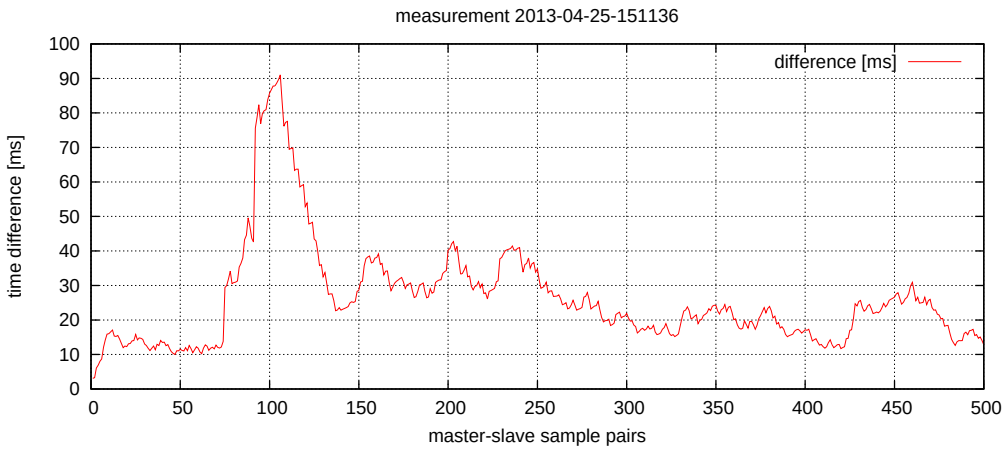
6.7 Zapojení: 1x switch a generátor síťového zatížení na `slave`



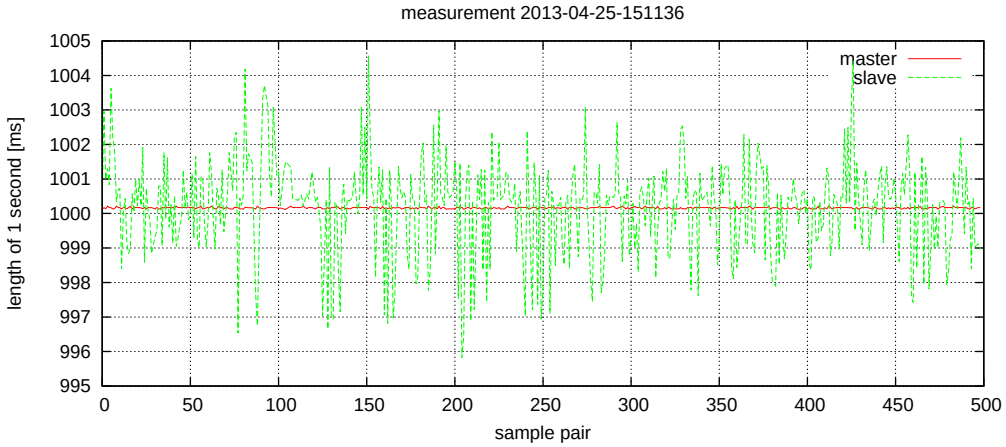
Obrázek 6.16: Schéma zapojení

1. Synchronizace PTP

   V tomto měření se podařilo zachytit PTP při „uvěření“ zprávám, které neobsahovaly kvalitní časový výpočet a proto se objevilo krátkodobě takto vysoké zpoždění, které trvalo přibližně minutu, než bylo opět navráceno do stabilnějšího stavu. Zajímavé též je, že zatěžování `slave` nemá na synchronizaci zdaleka takový vliv jako zatěžování `master` a je tedy nutné brát tento problém v potaz při nasazování tohoto protokolu. Lze z toho vyvozovat, že zjišťování zpoždění je tedy asymetrické.

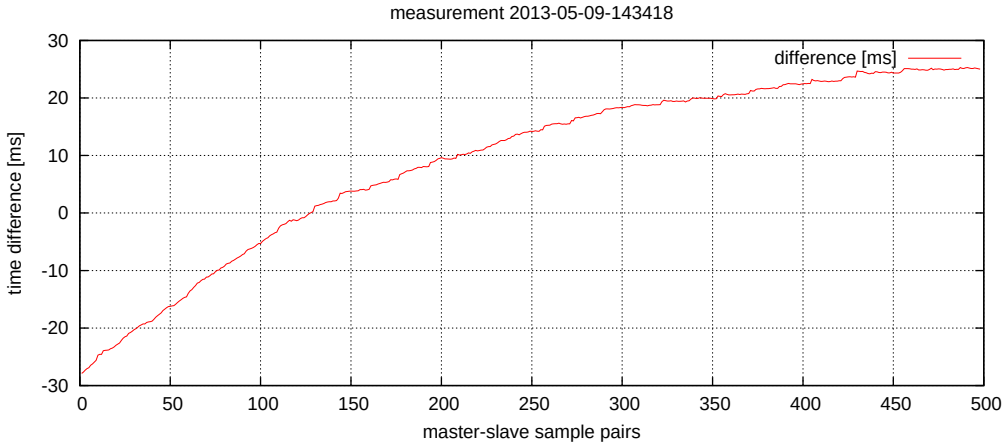


Obrázek 6.17: Rozdíl naměřených časů mezi stanicemi



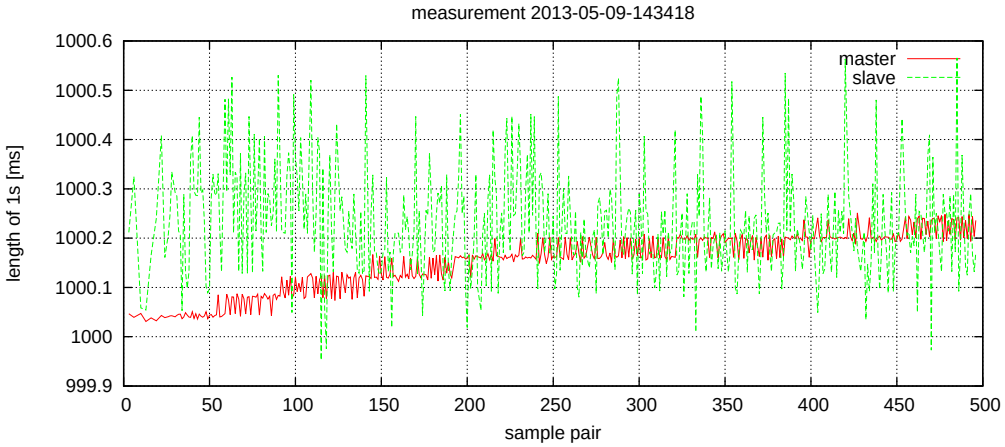
Obrázek 6.18: Délka periody mezi jednotlivými vzorky - filtrováno 2σ

2. Synchronizace NTP



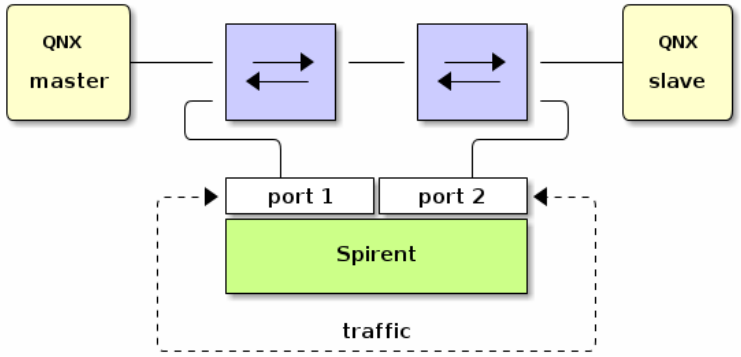
Obrázek 6.19: Rozdíl naměřených časů mezi stanicemi

Zde je na `master` vidět jak působí NTP samo na sebe (FIR filtr), tedy lokální systémové hodiny, ačkoliv byla zatěžována stanice `slave`.



Obrázek 6.20: Délka periody mezi jednotlivými vzorky - filtrováno 1σ

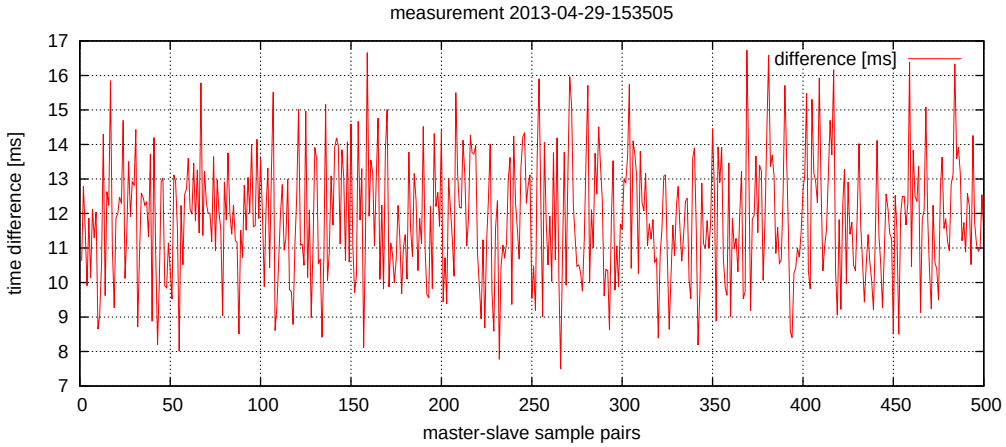
6.8 Zapojení: 2x switch a generátor provozu oběma směry



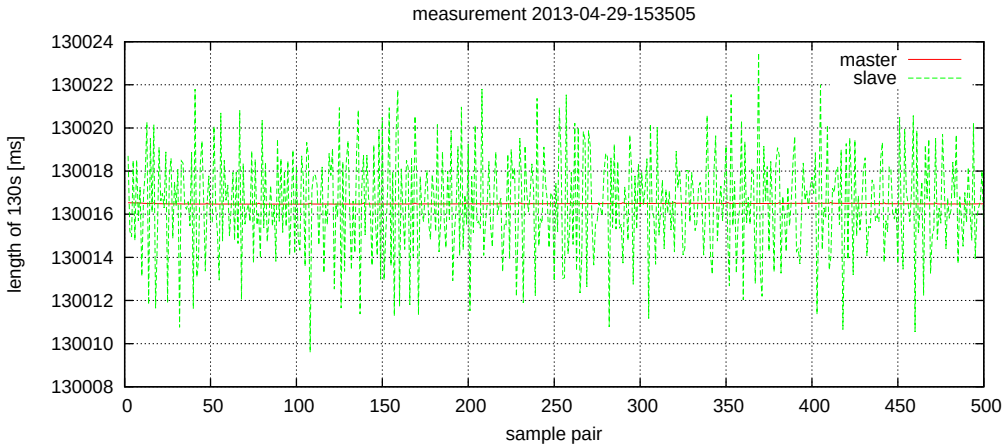
Obrázek 6.21: Schéma zapojení

Tato měření mají demonstrovat plně zatíženou switchovanou síť a to bez zatížení synchronizovaných počítačů. Metriky pro výpočet kvality obdržené zprávy synchronizačního protokolu jsou závislé na počtu `hop`, a tedy toto měření nelze použít pro názornou ukázku chování synchronizačních protokolů ve WAN sítích. Měření byla prováděna po dobu osmnácti hodin s generováním hrany každých sto třicet sekund.

1. Synchronizace PTP



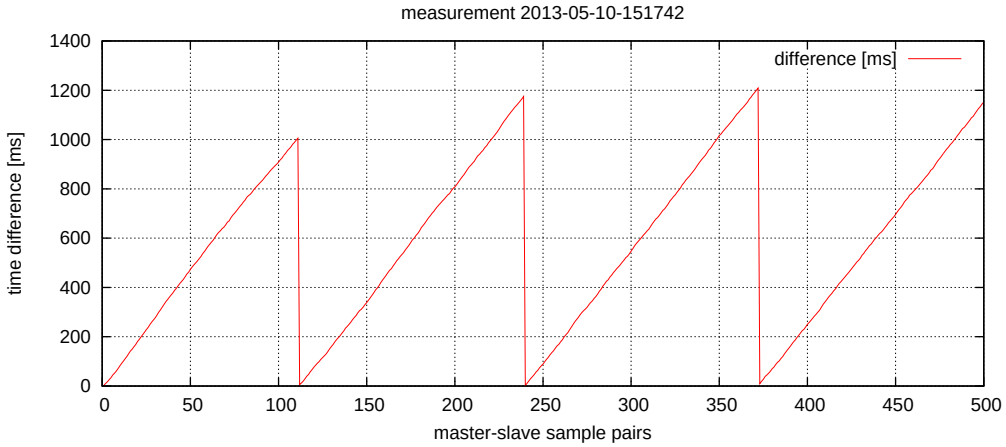
Obrázek 6.22: Rozdíl naměřených časů mezi stanicemi



Obrázek 6.23: Délka periody mezi jednotlivými vzorky

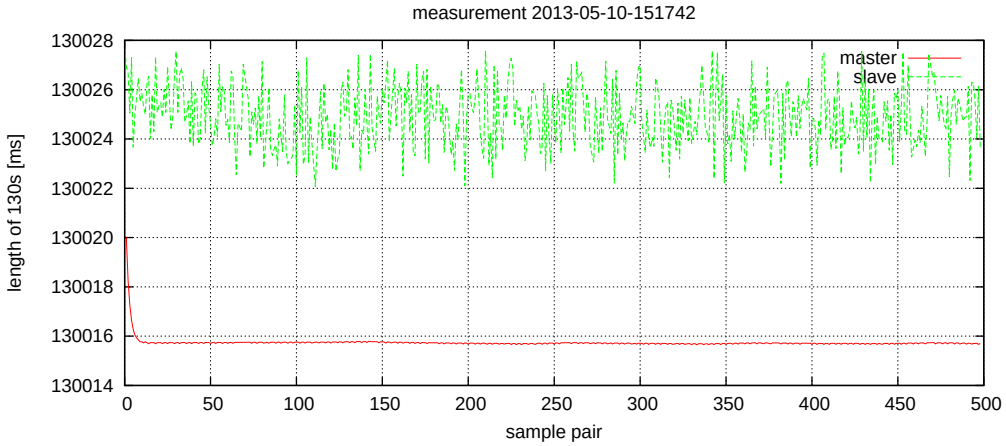
2. Synchronizace NTP

   Při tomto dlouhodobém měření se podařilo zachytit definovaná chování NTP v případě, kdy NTP mnohé obdržené zprávy považuje za špatné (podle různých metrik, mezi které patří např. spočítaný `jitter`) a trvá dlouhou dobu, nežli jim uvěří. Protože NTP bylo nastaveno tak, že velký skok může provést kdykoliv, kdy uzná za vhodné, jsou vidět v grafu místa, kdy NTP příchozí zprávě uvěřilo, přepočítalo veškeré interní koeficienty chyb apod. a začalo znovu odmítat zprávy s většími korekcemi až do jisté doby (po více než čtyřech hodinách), kdy znovu uvěřilo nové informaci a takto neustále dokola. Zajímavé je, že tvrdá korekce probíhala přibližně až po dosažení více než jedné sekundy zpoždění, což může být v některých náročných prostředích skutečný problém.



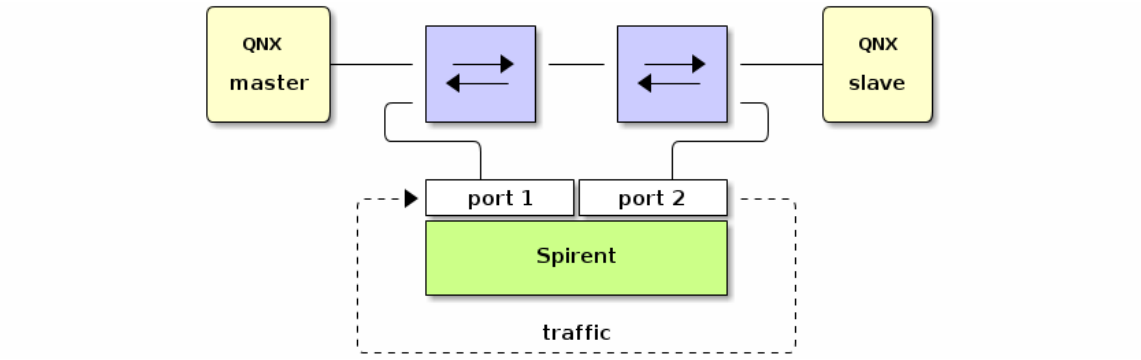
Obrázek 6.24: Rozdíl naměřených časů mezi stanicemi

Na následujícím grafu je vidět chyba vzniklá přepočtem nominální hodnoty frekvence oscilátoru ve FITkitu na reálný čas, kdy naměřené odstupy mezi vystavenými hranami stanicí `master` zdaleka neodpovídají přesnosti tiku (viz. 5) mikrojádra.



Obrázek 6.25: Délka periody mezi jednotlivými vzorky - filtrováno 2σ

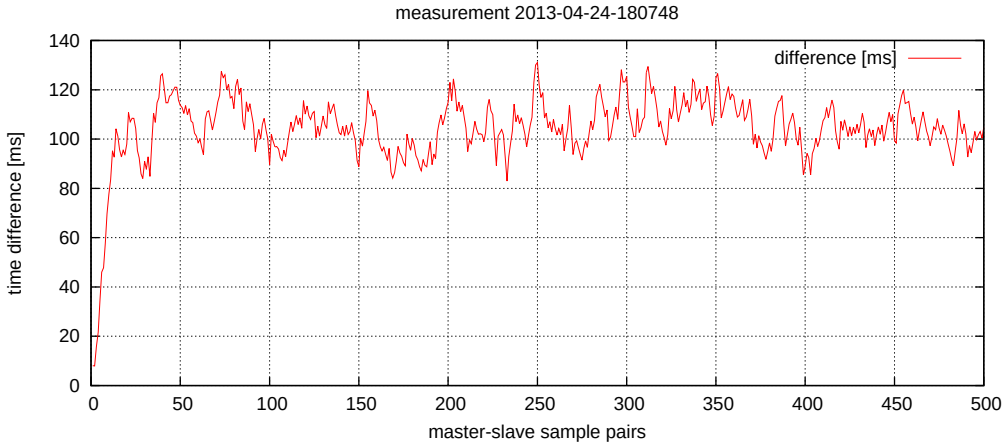
6.9 Zapojení: 2x switch a generátor provozu směrem k `master`



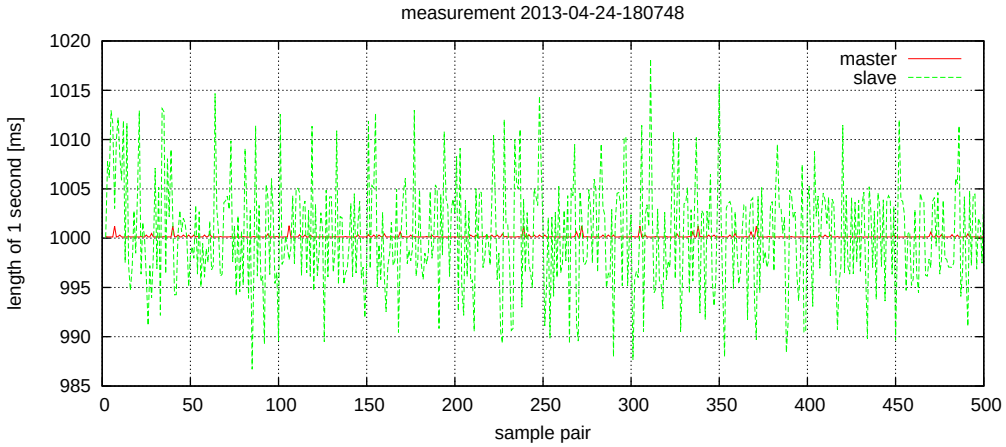
Obrázek 6.26: Schéma zapojení

1. Synchronizace PTP

   Následující měření není přímo srovnatelné s NTP, které bylo měřeno dlouhodobě. Z časových důvodů nebylo obdobné měření provedeno pro PTP, a je zde tedy uvedeno pouze informativní krátkodobé měření po dobu osm minut.

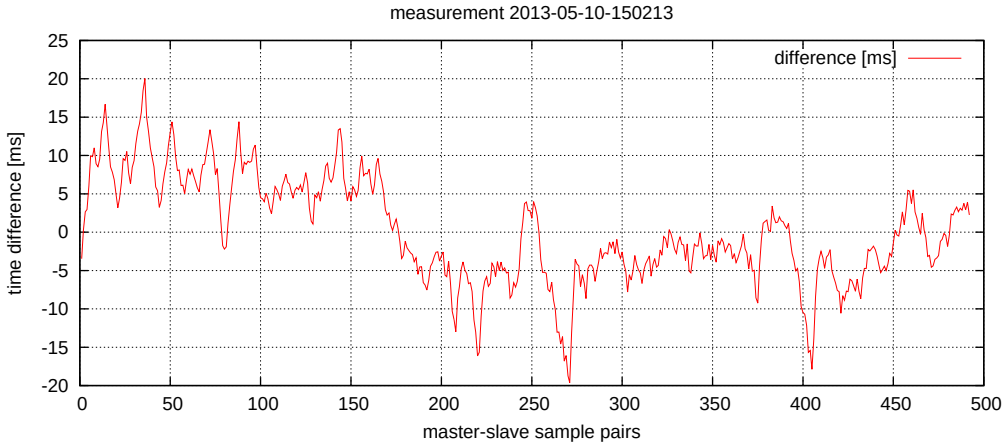


Obrázek 6.27: Rozdíl naměřených časů mezi stanicemi

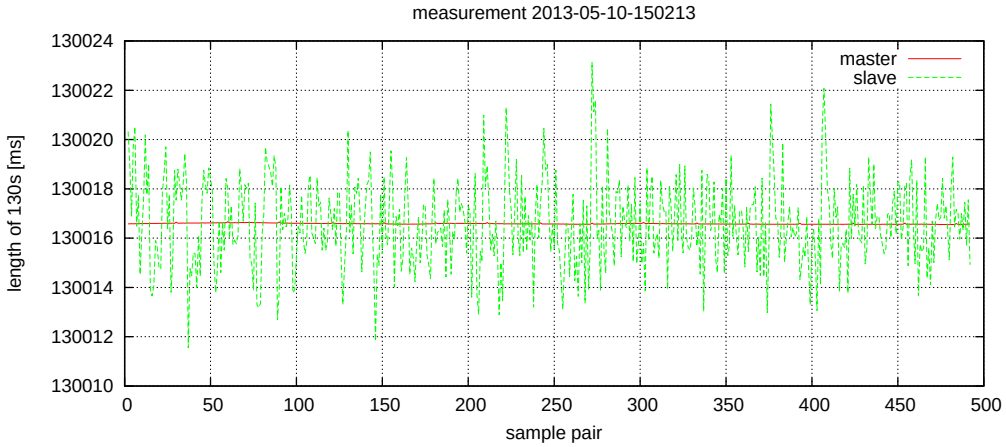


Obrázek 6.28: Délka periody mezi jednotlivými vzorky

2. Synchronizace NTP

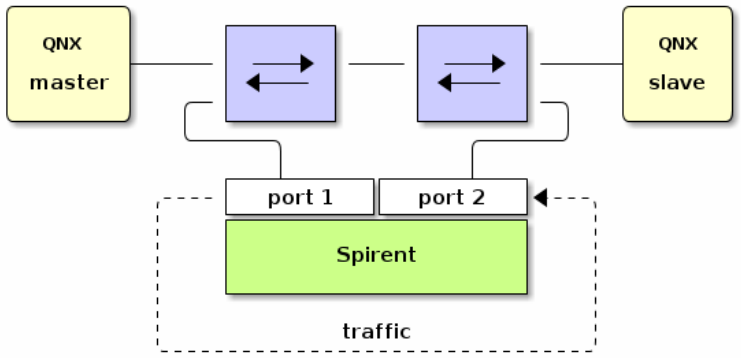


Obrázek 6.29: Rozdíl naměřených časů mezi stanicemi



Obrázek 6.30: Délka periody mezi jednotlivými vzorky

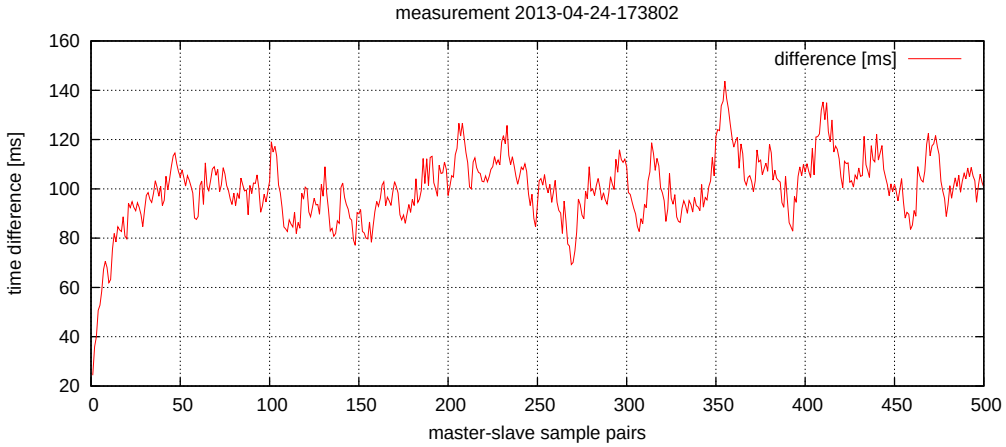
6.10 Zapojení: 2x switch a generátor provozu směrem k `slave`



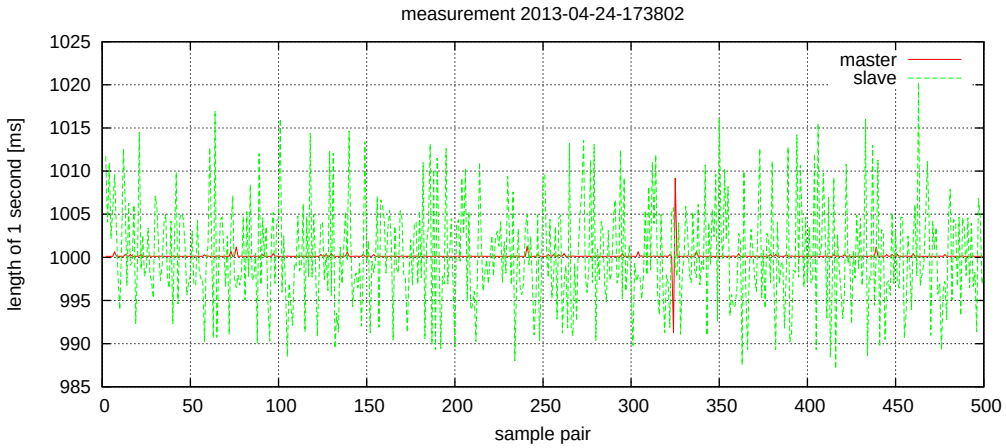
Obrázek 6.31: Schéma zapojení

1. Synchronizace PTP

   Následující měření není přímo srovnatelné s NTP, které bylo měřeno dlouhodobě. Z časových důvodů nebylo obdobné měření provedeno pro PTP, a je zde tedy uvedeno pouze informativní krátkodobé měření po dobu osm minut.

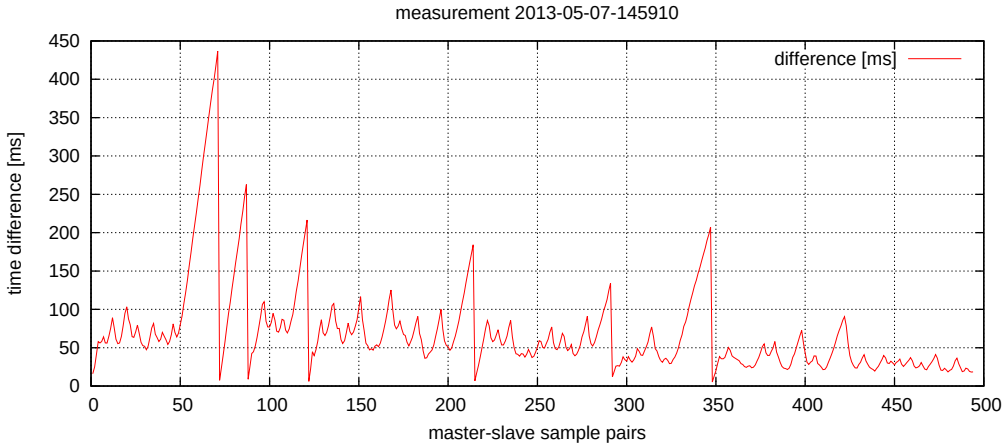


Obrázek 6.32: Rozdíl naměřených časů mezi stanicemi

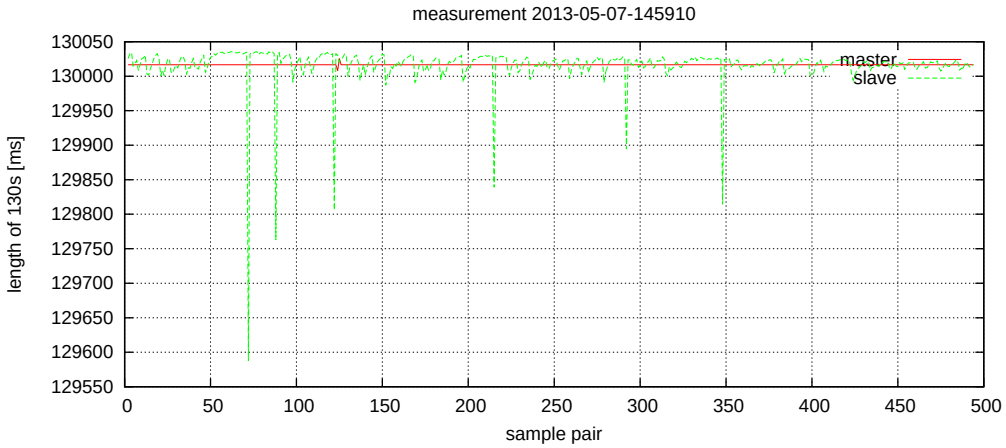


Obrázek 6.33: Délka periody mezi jednotlivými vzorky

2. Synchronizace NTP

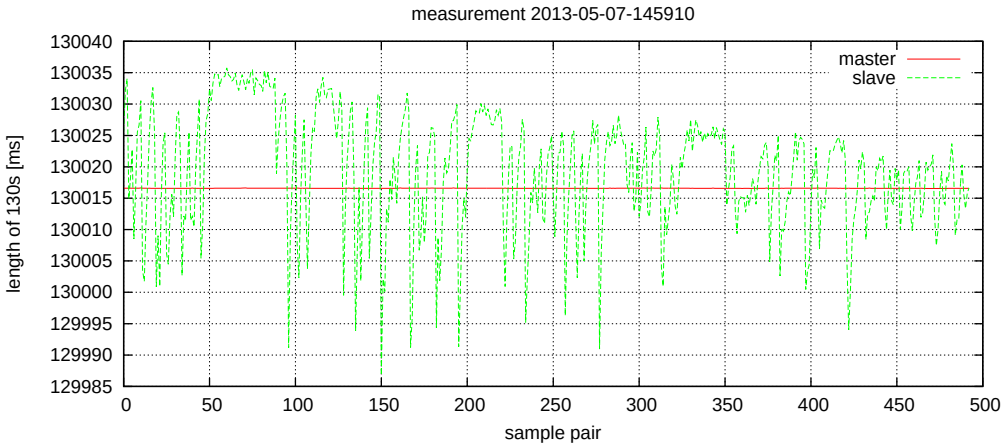


Obrázek 6.34: Rozdíl naměřených časů mezi stanicemi



Obrázek 6.35: Délka periody mezi jednotlivými vzorky

Protože se při měření objevilo mnoho velikých odchylek (největší má více než $400ms$), je zde uvedena i filtrovaná varianta grafu.



Obrázek 6.36: Délka periody mezi jednotlivými vzorky - filtrováno 2σ

Kapitola 7

Závěr

Byla provedena analýza synchronizačních protokolů PTP a NTP, které byly mezi sebou porovnávány za účelem nalézt jejich slabá místa zejména v souvislosti s real-time operačním systémem QNX. Měřící prostředí poskytovalo vysokou přesnost měření (řádově desítky nanosekund), ačkoliv se ukázalo, že tyto protokoly vykazují mnohem menší přesnost v měřících podmínkách. Měřící nástroje využívaly dostupných možností časování s cílem dosáhnout co nejvyšší přesnosti a demonstrovat jejich využití.

Dále bylo zjištěno, že implementace protokolu PTP ve formě PTPd má při nulovém zatížení systému a sítě tendenci zrychlovat a být tedy „napřed“ před zdrojem přesného času - `grandmaster` - o odchylku $-0.318ms$. Při vysokém, téměř maximálním zatížení systému a sítě se však adekvátně zpožďuje (naměřená odchylka $100.180ms$). Naopak NTP se v nezatíženém prostředí hodně odchyluje ($59.475ms$ a více - pozorováno až $85ms$) a při silném zatížení se většinou v dlouhodobém průměru tato hodnota příliš nemění ($69.801ms$) a tedy tyto hodnoty lze brát jako pevný offset, který by po odečtení (tedy jisté úpravě algoritmů v NTPd implementaci pod QNX) vykazoval lepší výsledky.

Provedená měření byla ovlivněna skutečností, že použité switche po jisté době běhu (empiricky zjištěno, že přibližně po deseti minutách) vysoké zátěže přenosové kapacity linek začaly replikovat rámce na všechny porty namísto pouze těch, do kterých byly připojeny zatěžované prvky nebo prvky způsobující tuto zátež. Síťové karty na testovaných počítačích tyto rámce okamžitě zahazovaly a jádro tedy neobdrželo žádnou informaci o změnách (toto lze pozorovat na indikátorech zatížení přitomných ve Photon UI). Linka však byla zatížená, což se mohlo projevit jistými zpožděními. Hypotézy, které by vysvětlily toto chování switchů nebyly nalezeny.

Mezi ostatní zkreslení měření lze též zařadit problémy se stabilitou počítače s CPU Atom, který nebyl schopen stabilně dlouhodobě provozovat vysoce náročné programy na výkon. QNX v takovém případě po náhodné době „zamrzalo“ a často nepomohl ani restart (bylo nutné nechat počítač vychladnout). Např. při maximálním zatížení linky indikátor „zatížení síťové karty“ ukazoval 100% zatížení, kdežto indikátor na druhém testovacím počítači s procesorem C2D za stejných podmínek ukazoval 0% zatížení. Je možné, že se toto chování dělo z důvodu nepříliš dobrého pasivního chlazení a nebo též v souvislosti s nízce nastavenou hodnotou délky periody tiku mikrojádra.

Taktéž problematika délky periody mezi zasíláním zpráv jednotlivými synchronizačními protokoly se jistě projevila na výsledcích, a bylo by tedy vhodné pro další přímá srovnání nastavit PTPd, aby zasílal zprávy v delších intervalech, shodných s NTPd - tedy např. $16s$.

Je též zřejmé, že tato měření nepříliš dobře aproximují běžná prostředí, kde se zátěž síťové linky nepohybuje nad 99.9%. Bylo by tedy vhodné pokračovat v měřeních s méně

zatíženými linkami a to na sítích s více směrovači v cestě, aby bylo možné otestovat plnou účinnost algoritmů pro rozpoznávání kvality a důvěryhodnosti obdržených zpráv. Tyto algoritmy berou v potaz právě počet `hopů` a např. v NTP hrají velikou roli.

Dalším neménně zajímavým pokračováním by bylo nasazení `boundary clock` (k těmto není potřeba hardwarová podpora) na směrovačích či jiných hraničních uzlech a sledovat vliv tohoto zařízení na výkon PTP. V tomto ohledu by tedy bylo možné přímé porovnání NTP, jehož stratum servery s číslem vyšším nežli 0 vykazují právě takové chování jako PTP `boundary clock`. Spolu s tím by bylo možné též měřit časové rozdíly v podobě rozdílů časových razítek zachycené zařízením Spirent, které podporuje protokol PTP i NTP a v souvislosti s podporou skriptování a logování paketů by bylo vhodné porovnat takto získané výsledky s měřením provedeným aktuálně použitou měřící sestavou. Jednalo by se však o nepřímé srovnání, protože časová razítka se ukládají až při odesílání rámce, kdežto testované implementace synchronizačních protokolů pracují až na úrovni transportní vrstvy.

Pro další měření prováděné postavenou měřící sestavou by též bylo vhodné provést kalibraci FITkitu (např. za pomocí poměrně dobře dostupných rubidiových hodin nebo alespoň GPS modulu) a eliminovat nebo alespoň popsat tak nepřesnosti přítomného oscilátoru.

Měření explicitně neuvedená v této práci jsou k dispozici v příloze A. Každé měření je umístěno ve vlastním adresáři pojmenovaném podle času, kdy bylo meření provedeno. Tento adresář obsahuje soubory nejméně soubory `results.raw` (naměřené výsledky, někdy s vynechanými hodnotami z počátku měření pro jejich nerelevantnost), `topology.info` (obsahující popis měřící topologie a veškerých podmínek souvisejících s měřením) a `stats` (obsahující spočítané statistiky).

Přes výše uvedené nedostatky měření lze naměřené hodnoty považovat za reálné a popisující pravděpodobné chování při nasazení v ostrém provozu. Použitá implementace PTP se jeví jako vhodný protokol pro vysoce přesnou synchronizaci na méně zatížených LAN sítích, kdežto dostupná implementace NTP vykazovala nepříliš dobré výsledky a nelze ji tedy do časově kritických sítí doporučit. SCADA systém nainstalovaný na testovacím počítači byl bezproblémově plně funkční i přes zpoždění způsobená vysokou zátěží jak sítě, tak systému. Lze tedy předpokládat, že i přes nepříliš vysokou přesnost NTPd lze použít kteroukoliv z těchto dostupných implementací synchronizačních protokolů.

Výsledky této práce budou sloužit jako podklad pro další pokračování v evaluaci kvality dostupných možností synchronizace, a to se zaměřením na restrukturalizaci podpory pro synchronizační protokoly v jádrech rozšířených operačních systému, zejména Linuxu. Další vývoj v této oblasti bude pokračovat ve spolupráci s ČVUT v Praze.

Literatura

[1] F. Zezulka and O. Hyncica. *Synchronizace v distribuovaných řídících systémech: Precision Time Protocol podle IEEE 1588*, AUTOMA, vol. 2, pp. 17-19, 2010.

[2] J. C. Eidson and K. Lee. Sharing a Common Sense of Time.*IEEE Instrumentation and Meassurement Magazine*, no. 3, 2003.

[3] V. Smotlacha. *Time Issues in One-way Delay Measurement*. Czech Technical University in Prague, 2005.

[4] Q. M. Chaudhari. A Simple and Robust Clock Synchronization Scheme. *IEEE Transactions on Communications*, vol. 60, no. 2, pp. 328-332, Feb. 2012.

[5] V. Igure, S. Laugher and R. Williams. Security issues in SCADA networks. *Computers & Security*, vol. 25, no. 7, pp. 498-506, Oct. 2006.

[6] FITkit. *Platforma FITkit* [online]. 2012-06-23, [cit. 2013-01-18]. Dostupné z http://merlin.fit.vutbr.cz/FITkit/.

[7] Cosart L. *Studying network timing with precision packet delay measurements*, R&D, Symmetricom, Inc., 2009.

[8] Precise Networked Clock Synchronization Working Group. *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, IEEE 1588-2008. 27.3.2008.

[9] David L. Mills, Jim Martin, Jack Burbank and William Kasch. *Network Time Protocol Version 4: Protocol and Algorithms Specification*. June 2010, ISSN: 2070-1721.

[10] PTPd. *Precision Time Protocol daemon* [online]. 2012, [cit. 2013-01-08]. Dostupné z http://ptpd.sourceforge.net/.

[11] David L. Mills. *DCN Local-Network Protocols* [online]. RFC 891, Dec. 1983. Dostupné z http://tools.ietf.org/html/rfc891.

[12] Engineering and Design - NAVSTAR Global Positioning System Surveying. 1.7.2003. EM 1110-1-1003

[13] NTP. *NTP: The Network Time Protocol* [online]. 25.9.2012 [cit. 11.5.2013]. Dostupné z http://www.ntp.org/.

[14] G. D. Troxel. *Time Surveying: Clock Synchronization over Packet Networks*. MIT, May 1994.

[15] David L. Mills, Ajit Thyagarjan and Brian C. Huffman. *Internet Timekeeping Around the Globe*. Dec. 1997.

[16] NTP. *Clock Select Algorithm* [online]. 13.4.2012 [cit. 12.5.2013]. Dostupné z http://www.eecis.udel.edu/~mills/ntp/html/select.html.

[17] Choosing between PTP and NTP [online]. 28.6.2012 [cit. 12.5.2013]. Dostupné z http://www.fsmlabs.com/blog/choosing-between-ptp-and-ntp.

[18] V. Smotlacha. *One-way delay measurement using NTP synchronization* [online]. 2003 [cit. 12.5.2013]. Dostupné z http://www.ces.net/project/qosip/publications/2003/

owd-meas/

[19] Darryl Veitch, Satish Babu and Attila Pasztor. *Robust Synchronization of Software Clocks Across the Internet*. 19.9.2004.

[20] Kendall Correll, Nick Barendt and Michael Branicky. *Design Considerations for Software Only Implementations of the IEEE 1588 Precision Time Protocol*. VXI Technology, Inc. and Case Western Reserve University, Ohio. 2005.

Příloha A

Obsah CD

fitkitSetup/	Adresář s pomocnými soubory pro měřící sestavu.
pics-setup/	Adresář s fotografiemi celého měřícího prostředí
rfc/	Adresář s RFC, kterými byly inspirovány některé součásti měření.
semestralProject/	Adresář s dokumenty pro obhajobu semestrálního projektu.
src/	Adresář se zdrojovými kódy, dokumentací a všemi výsledky měření.
stuff/	Adresář s nezařazeným obsahem jako různé technické dokumentace apod.
README.md	Soubor obsahující návod na aplikaci patche pro PTPd.

Pozn.: Veškeré tyto materiály jsou dostupné v aktuálním znění též online na adrese https://github.com/dumblob/qnx_net_time_sync .