PAMĚŤOVĚ EFEKTIVNÍ VYHLEDÁNÍ NEJDELŠÍHO SHODNÉHO PREFIXU PRO SMĚROVÁNÍ VE 100 GB/S SÍTÍCH

Jiří Matoušek
Výpočetní technika a informatika, 2. ročník, prezenční studium
Školitel: Zdeněk Kotásek

Fakulta informačních technologií, Vysoké učení technické v Brně
Božetěchova 1/2, 612 66 Brno

imatousek@fit.vutbr.cz

Abstrakt. Zpracování síťových dat v současných páteřních sítích již není možné provádět s využitím obecných procesorů, ale ke zpracování je nutné využít specializovaný hardware. V rámci disertační práce s názvem *Využití rekonfigurovatelných obvodů v oblasti počítačových sítí* je zkoumána oblast využití technologie FPGA pro implementaci operace vyhledání nejdelšího shodného prefixu, která představuje hlavní část procesu směrování paketů. Tento příspěvek popisuje nově navrženou paměťově efektivní reprezentaci sady prefixů ze směrovací tabulky, která se v případě sad IPv6 prefixů vyrovná současným nejlepším řešením a pro IPv4 prefixy dosahuje výrazně lepších výsledků. Díky snížení paměťových nároků reprezentace prefixové sady je možné k jejímu uložení využít snadno a rychle přístupnou paměť na čipu FPGA. Reprezentaci prefixové sady tak není nutné ukládat do externí paměti, díky čemž dosahuje výsledné řešení propustnosti přibližně 155 Gb/s.

Klíčová slova. LPM, FPGA, směrování, zřetězené zpracování

1 Úvod

S narůstajícím využitím služeb poskytovaných prostřednictvím Internetu narůstá také množství dat přenášených v počítačových sítích. Přenosy velkých datových objemů jsou v současných sítích uskutečnitelné jen díky nárůstu podporovaných přenosových rychlostí, viz například technologie 100 Gb/s Ethernetu [1]. Nárůst přenosových rychlostí však klade zvýšené požadavky na zpracování síťového provozu, které již není možné v páteřních sítích provádět na obecných procesorech. Zpracování síťového provozu je třeba akcelerovat v hardware, například s využitím technologie FPGA.

V rámci své disertační práce na téma *Využití rekonfigurovatelných obvodů v oblasti počítačových sítí* se zaměřuji na oblast směrování v páteřních sítích. Směrování paketů je jednou ze základních síťových operací a je prováděno na základě dat — prefixů síťových adres určujících jednotlivé podsítě — uložených ve směrovací tabulce. Výpočetně nejnáročnější součástí směrování je nalezení nejdelšího prefixu, který odpovídá cílové adrese směrovaného paketu (anglicky *longest prefix match*, *LPM*).

Směrovací tabulky v současných páteřních směrovačích obsahují přibližně 450 tisíc IPv4 a 13 tisíc IPv6 záznamů [2], které je třeba při podpoře přenosové rychlosti 100 Gb/s prohledat přibližně za 6 ns. Externí paměti disponují dostatečnou kapacitou pro uložení rozáhlých prefixových sad ze směrovacích tabulek, avšak přístup k nim je relativně pomalý a energetický náročný. Při využití technologie FPGA je možné nahradit externí paměti dostupnou distribuovanou pamětí na čipu, která však má omezenou

kapacitu. V rámci tohoto příspěvku je proto představena paměťově efektivní reprezentace sady IPv4/IPv6 prefixů a hardwarová architektura pro práci s touto reprezentací na čipech FPGA. Tato reprezentace prefixové sady umožňuje využití paměti na čipu FPGA, čímž je dosaženo propustnosti přes 100 Gb/s.

Základem představeného řešení je stejně jako u jiných excelentních LPM algoritmů, s nimiž je navržené řešení srovnáváno, datová struktura *trie* nazývaná též prefixový strom. Jde o binární stromovou strukturu, jejíž uzly reprezentují prefixy určené cestou ve stromu od kořene k danému uzlu. Uzly trie, které reprezentují prefixy z dané množiny, se nazývají prefixové uzly, zatímco zbývající uzly trie se nazývají neprefixové uzly.

Implementace operace LPM může využívat přímo datovou strukturu trie, avšak častější je využití tzv. vícebitových trie, které umožňují zpracování více bitů v jediném kroku. Příkladem algoritmů založených na principu vícebitové trie jsou *Tree Bitmap* (*TBM*) [3] a *Shape Shifting Trie* (*SST*) [4]. TBM uzly mohou reprezentovat libovolný podstrom trie o maximální hloubce dané parametrem *SL* (z anglického *stride length*) a jejich pevně daný tvar je výhodný pro reprezentaci hustých prefixových stromů. Naopak v řídkých trie se uplatní SST uzly, jejichž tvar je možné přizpůsobit tvaru reprezentovaného podstromu trie. Adaptivita SST uzlů je omezena pouze parametrem *K*, jenž udává maximální počet trie uzlů, které mohou být reprezentovány jedním SST uzlem. Posledním srovnávaným algoritmem je přístup představený v [5] a označovaný v rámci tohoto příspěvku zkratkou *PPLA* (*Prefix Partitioning Lookup Algorithm*). Tento algoritmus vyniká především svou paměťovou efektivitou a propustností 262 Gb/s.

Příspěvek je strukturován následovně. Kapitola 2 shrnuje provedenou analýzu paměťových nároků současných LPM algoritmů. Nově navržená reprezentace prefixových sad je představena v kapitole 3 a hardwarová architektura pro práci s navrženou reprezentací je popsána v kapitole 4. Příspěvek dále v kapitole 5 prezentuje výsledky provedených experimentů. Další uvažované přístupy k optimalizaci paměťových nároků operace LPM, které budou rozpracovány v rámci disertační práce, jsou nastíněny v kapitole 6. Příspěvek je zakončen kapitolou 7 shrnující výsledky prezentované v rámci příspěvku.

2 Analýza paměťových nároků LPM algoritmů

V rámci provedené analýzy byly pomocí nástroje Netbench [7] změřeny paměťové nároky algoritmů trie, TBM a SST při reprezentaci různých realných sad IPv4 a IPv6 prefixů z páteřních směrovačů[1] a také při reprezentaci IPv6 prefixových sad vygenerovaných pomocí generátoru [6].

Změřená spotřeba paměti jednotlivých algoritmů při reprezentaci uvažovaných prefixových sad je uvedena v tabulce 1. Při prováděných experimentech byly hodnoty parametrů *SL* a *K* nastaveny tak, aby paměťové nároky jednotlivých algoritmů byly co nejnižší. U algoritmu TBM můžeme tudíž pozorovat závislost optimální hodnoty parametru *SL* na hustotě trie — u husté trie (sady IPv4 prefixů) je výhodnější použití větších TBM uzlů než u řídké trie (sady IPv6 prefixů). V tabulce 1 se také odráží výpočetní náročnost algoritmu SST, jehož reprezentaci generovaných sad IPv6 prefixů se vůbec nepodařilo vytvořit.

Výsledky měření spotřeby paměti při reprezentaci IPv4 a IPv6 prefixových sad potvrzují očekávané vlastnosti algoritmů trie, TBM a SST. Nejméně paměťově náročnou reprezentaci prefixové sady využívá algoritmus SST, který je však výpočetně velmi náročný a neexistuje žádná hardwarová architektura, která by jej implementovala. Při optimalizaci paměťové náročnosti, i s ohledem na hardwarovou implementaci, je tudíž třeba vycházet z algoritmu TBM.

V rámci analýzy byla také provedena klasifikace TBM uzlů při reprezentaci různých prefixových sad, která ukázala, že nejčastěji využité jsou vnitřní TBM uzly neobsahující žádný prefix a listové TBM uzly. Vzhledem k jejich četnosti použití (konkrétní hodnoty viz [9]) může mít i malá optimalizace paměťových nároků těchto uzlů velký vliv na celkovou paměťovou náročnost upraveného TBM algoritmu.

[1] prefixové sady byly získány z `http://data.ris.ripe.net/`, `http://bgp.potaroo.net/` a `http://archive.routeviews.org/`

Tabulka 1: Paměťové nároky LPM algoritmů

Prefixová sada	**Prefixů**	**Paměťové nároky [Kb]**		
IPv4		**Trie**	**TBM (*SL=5*)**	**SST (*K=32*)**
rrc00	332 118	47 639.677	9 689.432	6 930.441
IPv4-space	220 779	24 252.430	5 702.065	4 081.008
route-views	442 748	62 650.455	11 942.068	8 774.961
IPv6		**Trie**	**TBM (*SL=3*)**	**SST (*K=32*)**
AS1221	10 518	3 518.297	1 076.926	588.516
AS6447	10 814	3 673.781	1 125.094	617.124
Generované IPv6		**Trie**	**TBM (*SL=4*)**	**SST**
rrc00_ipv6	319 998	307 641.509	87 257.128	N/A
IPv4-space_ipv6	150 157	153 877.340	43 958.728	N/A
route-views_ipv6	439 880	418 663.730	118 889.431	N/A

3 Nová reprezentace prefixových sad

Na základě výsledků provedené analýzy paměťových nároků LPM algoritmů byla navržena nová reprezentace prefixových sad založená na principu vícebitové trie a využívající celkem 13 různých typů uzlů. 9 typů uzlů je nově navržených a zbývající 4 typy jsou variantami TBM uzlu — standardní TBM uzel pro $SL = 3$ (TBM3) a listové TBM uzly pro $SL = 3, 4, 5$ (TBM3-L, TBM4-L, TBM5-L).

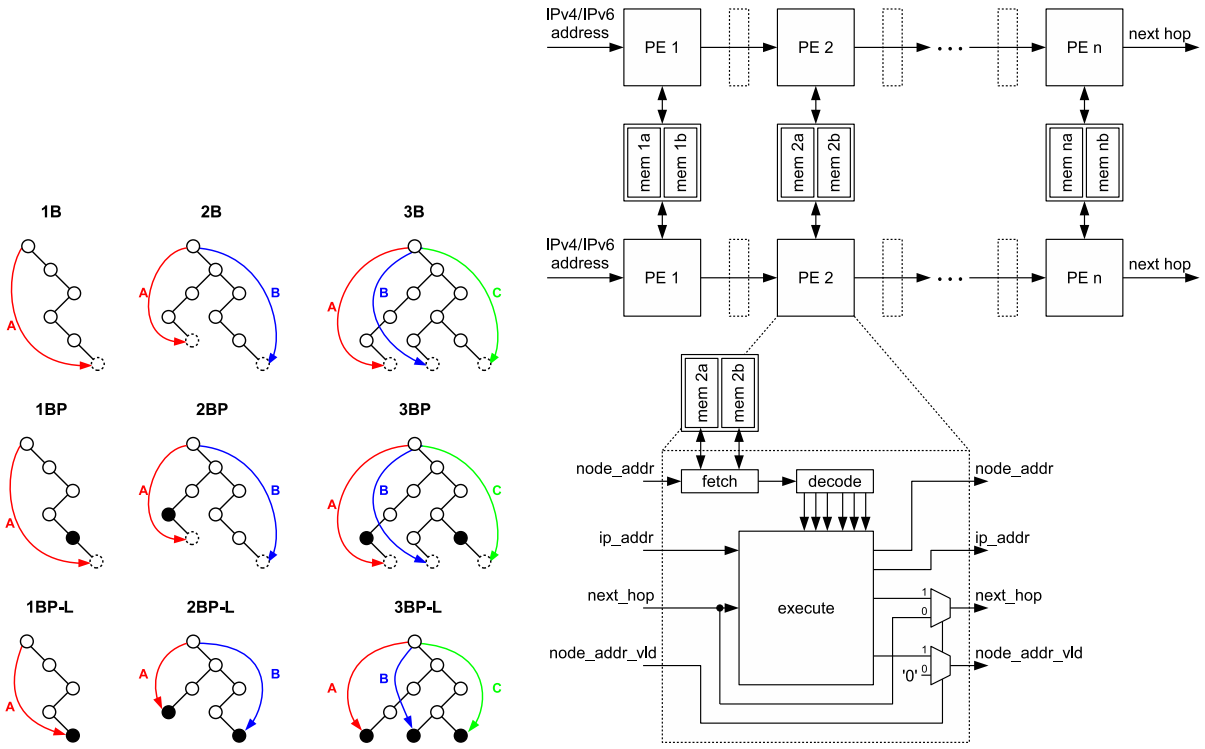
Nově navržené uzly jsou znázorněny na obrázku 1a a jejich označení popisuje situaci, pro kterou je daný uzel optimalizován. Levý sloupec ukazuje skupinu uzlů pro zakódování podstromu trie s jedinou větví (1B), v prostředním sloupci je skupina uzlů určených pro podstromy trie s dvěma větvemi (2B) a vpravo je zobrazena skupina uzlů pro zakódování podstromů trie se třemi větvemi (3B). Každá tato skupina obsahuje tři různé uzly pro různé situace — pro podstromy trie bez prefixových uzlů na horním řádků, pro volitelné zakódování prefixového uzlu v poslední hladině podstromu trie na prostředním řádku (P) a pro koncové podstromy trie (zakončené listovými uzly), u kterých jsou všechny větve povinně zakončené prefixovým uzlem, na spodním řádku (-L).

Mapování navržených typů uzlů na trie reprezentující prefixovou sadu je prováděno od kořene trie směrem k jejím listům. V každé pozici pro namapování uzlu je provedeno ohodnocení mapování všech navržených typů uzlů podle rovnice (1), kde p označuje počet mapováním pokrytých prefixových uzlů, n je počet mapováním pokrytých všech trie uzlů a $size$ udává velikost reprezentace uvažovaného typu uzlu. K namapování je pak zvolen ten typ uzlu, který podle rovnice (1) dosáhnul nejvyššího ohodnocení. Tento postup mapování uzlů na strukturu trie sice nezaručuje dosažení optimálního řešení, ale umožňuje provedení mapování s přijatelnou časovou složitostí.

$$price = \begin{cases} \frac{p}{size} & \text{pokud } \frac{p}{size} > 0 \\ \\ \frac{n}{size} & \text{jinak} \end{cases} \tag{1}$$

4 Hardwarová architektura

Pro práci s uvedenou reprezentací prefixové sady byla navržena hardwarová architektura znázorněná na obrázku 1b. Podpora vysoké propustnosti je zajištěna využitím 2 paralelních zřetězených linek. Pro každý stupeň zřetězené linky je vyhrazen samostatný blok paměti na čipu FPGA, který je však sdílen mezi zřetězenými linkami, což je umožněno díky jeho dvěma nezávislým portům. Paměť dostupná v rámci paměťového bloku je vnitřně organizována do dvou částí, kde každá část obsahuje poloviční počet položek o plné datové šířce. Jedna část pak slouží pro ukládání datových slov na sudých adresách a do druhé část jsou ukládána datová slova umístěná na lichých adresách. Díky tomuto uspořádání je



(a) Nová reprezentace prefixové sady (b) Hardwarová architektura pro práci s navrženou reprezentací

Obrázek 1: Nově navržené řešení LPM operace

možné načíst reprezentaci uzlu z paměti v jediném taktu i za situace, kdy tato informace není zarovnána na okraj datového slova a může být tudíž rozdělena mezi dvě následující datová slova.

Činnost jednotlivých procesních elementů (PE) se podobá zpracování instrukcí v obecném procesoru. Nejprve je provedeno načtení příslušného uzlu z paměti (*fetch*), následně je reprezentace uzlu dekódována (*decode*) a nakonec je provedeno jeho zpracování (*execute*) — vyhledání odpovídajících prefixů a načtení informací o následnících. Jelikož části PE realizující načtení uzlu z paměti a jeho zpracování obsahují složitou kombinační logiku, byly do každého z těchto bloků doplněny 2 sady vnitřních registrů, které umožňují dosažení požadované pracovní frekvence. Každý stupeň zřetězené linky zobrazené na obrázku 1b tak ve skutečnosti reprezentuje 5 stupňů zřetězeného zpracování.

5 Experimentální výsledky

Měření paměťových nároků navržené reprezentace při aplikaci na používané prefixové sady bylo provedeno pomocí nástroje Netbench [7]. Výsledky tohoto měření jsou shrnuty v tabulce 2a, která vyjadřuje paměťovou náročnost navržené reprezentace prefixových sad a algoritmů TBM a SST ve formě počtu bytů paměti potřebných k reprezentaci 1 bytu prefixu (použito i v popisu algoritmu PPLA [5]).

Z uvedených výsledků je patrné, že navržená reprezentace prefixových sad je výrazně paměťově efektivnější než algoritmus TBM a na všech uvažovaných prefixových sadách překonává také algoritmus SST navržený právě s ohledem na nízkou paměťovou náročnost. V porovnání s algoritmem PPLA dosahuje navržená reprezentace výrazně lepších výsledků pro IPv4 prefixové sady (průměrná hodnota u PPLA je 1.0) a mírně horších výsledků pro generované IPv6 prefixové sady (průměrná hodnota u PPLA je 0.9). Paměťová efektivita PPLA na reálných sadách IPv6 prefixů nebyla v [5] uvedena.

Vyhodnocení hardwarové architektury pro práci s navrženou reprezentací prefixových sad je uve-

(a) Efektivita využití paměti (bytů paměti na 1 byte prefixu) navržené reprezentace prefixové sady, TBM a SST

Prefixová sada	**Prefixů**			
IPv4		**Nové LPM**	**TBM (*SL=5*)**	**SST (*K=32*)**
rrc00	332 118	0.610	0.934	0.668
IPv4-space	220 779	0.518	0.826	0.592
route-views	442 748	0.562	0.863	0.634
IPv6		**Nové LPM**	**TBM (*SL=3*)**	**SST (*K=32*)**
AS1221	10 518	0.724	1.638	0.895
AS6447	10 814	0.731	1.665	0.913
Generované IPv6		**Nové LPM**	**TBM (*SL=4*)**	**SST (*K=32*)**
rrc00_ipv6	319 998	1.063	4.363	N/A
IPv4-space_ipv6	150 157	1.109	4.684	N/A
route-views_ipv6	439 880	1.056	4.324	N/A

(b) Využití zdrojů a maximální frekvence navržené architektury po syntéze pro FPGA Xilinx Virtex-6 XC6VSX475T pomocí nástroje Xilinx ISE 14.3

	LUT (% všech)	**Registry (% všech)**	**Frekvence [MHz]**
1 PE	4 038 (1.357 %)	1 827 (0.307 %)	115.407
1 linka (23 PE)	92 874 (31.208 %)	42 021 (7.060 %)	115.407
2 linky (46 PE)	185 748 (62.415 %)	84 042 (14.120 %)	115.407

Tabulka 2: Experimentální výsledky

deno v tabulce 2b, která uvádí spotřebu zdrojů FPGA (LUT a registry) a také maximální možnou pracovní frekvenci. Tyto hodnoty byly získány syntézou navržené architektury pro FPGA Xilinx Virtex-6 XC6VSX475T ve vývojovém prostředí Xilinx ISE 14.3. Uvedené procentuální využití všech dostupných zdrojů se vztahuje ke zmíněnému cílovému FPGA. Kromě spotřeby zdrojů jednoho procesního elementu je také uvedena spotřeba zdrojů na implementaci jedné a dvou zřetězených linek sestávajících z 23, respektive 46 PE. Zapojení 23 PE v rámci jedné zřetězené linky je odvozeno od maximální výšky stromu při reprezentaci používaných prefixových sad způsobem popsaným v rámci kapitoly 3.

Navržená hardwarová architektura je schopná poskytnout v každém hodinovém taktu dva LPM výsledky, což při maximální pracovní frekvenci 115 MHz znamená až 230 milionů LPM výsledků za sekundu. Maximální podporovaná přenosová rychlost je tudíž přibližně 155 Gb/s. Latence vyhledání nejdelšího shodného prefixu je dána dobou zpracování v jednom stupni zřetězené linky (8,66 ns) a jejich počtem ($5 \times 23 = 115$), celkem tedy 995,9 ns. Navrženou architekturu je tak nutné doplnit ještě o paketový buffer s minimální kapacitou 18,97 KB, který však může být implementován i v externí paměti.

6 Cíle disertační práce

V úvodu příspěvku byla nastíněna současná situace v oblasti směrování v páteřních sítích. Kvůli požadavku na vysoké přenosové rychlosti je třeba zpracování síťových dat akcelerovat na speciálních hardwarových architekturách. Důležitým prvkem těchto architektur jsou rychlé paměti s dostatečnou kapacitou.

V rámci disertační práce se zabývám akcelerací operace LPM s využitím technologie FPGA. Současné FPGA čipy disponují rychlou a snadno přístupnou pamětí na čipu, jenž však má omezenou kapacitu. Proto je třeba hledat způsoby paměťově efektivní reprezentace datových struktur používaných při operaci LPM. Příkladem takové reprezentace je přístup představený v tomto příspěvku, jehož základy byly publikovány na konferenci DDECS 2013 [8] a článek na toto téma byl také přijat na konferenci FPL 2013 jako „regular paper“ [9].

Naprostá většina současných přístupů k akceleraci operace LPM je založena na použití zřetězené linky za účelem paralelizace jednotlivých částí výpočtu. V rámci takového řešení jsou jednotlivé části datové struktury rozděleny do oddělených pamětí přiřazených stupňům zřetězené linky. Vzhledem k dynamické povaze směrovacích informací však nejsou paměťové nároky jednotlivých stupňů konstantní. Jednotlivým stupňům zřetězené linky je proto nutné přidělit dostatek paměti na pokrytí nejhoršího případu,

i za cenu toho, že část přidělené paměti není reálně využita. V rámci disertační práce se proto chci také zabývat využitím částečné dynamické rekonfigurace k dynamické alokaci paměti na čipu FPGA.

7 Závěr

Tento příspěvek představuje nově navrženou paměťově efektivní reprezentaci prefixových sad pro implementaci operace vyhledání nejdelšího shodného prefixu. V rámci příspěvku je také představena hardwarová architektura implementující operaci LPM s využitím navržené reprezentace prefixové sady.

Měření paměťových nároků navržené datové struktury při reprezentaci reálných sad IPv4 a IPv6 prefixů z páteřních směrovačů i generovaných sad IPv6 prefixů ukázalo, že navržená reprezentace má výrazně menší paměťové nároky než algoritmus TBM a dosahuje nižší paměťové náročnosti než algoritmus SST. V porovnání s algoritmem PPLA je navržené řešení z pohledu paměťové náročnosti lepší při reprezentaci IPv4 prefixových sad a mírně za PPLA zaostává při reprezentaci generovaných IPv6 prefixových sad. Navržená hardwarová architektura podporuje propustnost přibližně 155 Gb/s.

Výsledky prezentované v tomto příspěvku byly dosaženy v rámci řešení disertační práce na téma *Využití rekonfigurovatelných obvodů v oblasti počítačových sítí*, jejíž další částí by mělo být navržení řešení umožňujícího dynamickou alokaci paměti na čipu jednotlivým stupňům zřetězené linky s využitím částečné dynamické rekonfigurace FPGA.

Poděkování

Tato práce byla podpořena Evropským fondem regionálního rozvoje (ERDF) v rámci projektu Centra excelence IT4Innovations (CZ.1.05/1.1.00/02.0070), výzkumným záměrem MSM 0021630528, a částečně také grantem FIT-S-11-1 – Pokročilé, bezpečné, spolehlivé a adaptivní IT.

Reference

[1] IEEE Computer Society: Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications; Amendment 4: Media Access Control Parameters, Physical Layers, and Management Parameters for 40 Gb/s and 100 Gb/s Operation. IEEE std 802.3ba-2010, June 2010. ISBN 978-0-7381-6322-2.

[2] (2013, Jun.) IPv6 / IPv4 Comparative Statistics. [Online]. Available: `http://bgp.potaroo.net/v6/v6rpt.html`

[3] W. Eatherton, G. Varghese, and Z. Dittia: Tree Bitmap: Hardware/Software IP Lookups with Incremental Updates. ACM SIGCOMM Computer Communications Review, vol. 34, no. 2, pp. 97–122, April 2004, ISSN 0146-4833.

[4] H. Song, J. Turner, and J. Lockwood: Shape Shifting Tries for Faster IP Route Lookup. In Proceedings of the 13th IEEE International Conference on Network Protocols (ICNP'05). IEEE Computer Society, 2005, pp. 358–367, ISBN 0-7695-2437-0.

[5] H. Le and V. K. Prasanna: Scalable Tree-based Architectures for IPv4/v6 Lookup Using Prefix Partitioning. IEEE Transactions on Computers, vol. 61, no. 7, pp. 1026–1039, July 2012, ISSN 0018-9340.

[6] M. Wang, S. Deering, T. Hain, and L. Dunn: Non-random Generator for IPv6 Tables. In Proceedings of the 12th Annual IEEE Symposium on High Performance Interconnects, 2004. IEEE Computer Society, August 2004, pp. 35–40, ISBN 0-7803-8686-8.

[7] V. Pus, J. Tobola, V. Kosar, J. Kastil, and J. Korenek: Netbench: Framework for Evaluation of Packet Processing Algorithms. In Seventh ACM/IEEE Symposium on Architecture for Networking and Communications Systems (ANCS'11). IEEE Computer Society, October 2011, pp. 95–96, ISBN 978-0-7695-4521-9.

[8] J. Matoušek, M. Skačan, and J. Kořenek: Towards Hardware Architecture for Memory Efficient IPv4/IPv6 Lookup in 100 Gbps Networks, In 2013 IEEE 16th International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS), Brno, CZ, IEEE CS, 2013, pp. 108–111, ISBN 978-1-4673-6133-0.

[9] J. Matoušek, M. Skačan, and J. Kořenek: Memory Efficient IP Lookup in 100 Gbps Networks. In 23rd International Conference on Field Programmable Logic and Applications (FPL 2013), Porto, 2013. Accepted.