Power-Efficient and Scalable Virtual Router Architecture on FPGA*

Swapnil Haria**

BITS, Pilani
Rajasthan, India
Email: swapnilster@gmail.com

Thilan Ganegedara, Viktor Prasanna
Ming Hsieh Dept. of Electrical Engineering
University of Southern California
Los Angeles, CA 90089, USA
Email: {ganegeda, prasanna}@usc.edu

***Abstract*—In the recent years, networking infrastructure has advanced in such a way that router hardware management and power efficiency issues have gained considerable attention. Router virtualization alleviates these issues by allowing a single hardware router to serve packets from multiple networks. We propose a power-efficient scalable architecture for implementing router virtualization using the Virtualized Merged (VM) approach on Field-programmable Gate Array (FPGA). Three novel optimizations are incorporated into the basic VM approach to reduce the dynamic power dissipation of the router hardware and deliver higher throughput per unit power consumed. The reduction in power consumption is in part due to the savings in memory required to store the merged lookup table, which makes our optimized VM approach more scalable with respect to the number of virtual routers per FPGA chip. Also, by exploiting the low power features and clock gating techniques, the optimized VM approach achieves significant power savings. To illustrate the improvements achieved, we tested the optimized VM router using 75 routing tables on a single FPGA, utilizing 50% less memory and consuming 20% less power compared with the basic VM approach.**

***Index Terms*—FPGA, virtualization, power-efficient, scalable.**

I. Introduction

In recent years, the Internet has been growing at a rapid pace, and currently consists of over 5 billion nodes [11]. This growth has been supported by the development of high performance networking hardware, made possible by the continuous advancements in semiconductor technology. Network virtualization [1], [2] was proposed to take advantage of these advancements to improve network efficiency. This has been realized via router virtualization, at the hardware level. Router virtualization allows multiple network hardware to be consolidated onto a single shared platform, thereby enabling centralized management of multiple networks. Reduced hardware costs, reduced power consumption, and ease of management are a few additional benefits. These features have become attractive especially in the context of datacenter networking and Software Defined Networking (SDN) [4].

*This work is supported by the United States National Science Foundation under grant No. CCF-1116781.
** Participation sponsored by the Viterbi-India 2012 Program,University of Southern California.

Router virtualization has been studied in several recent works [5]–[7], [10], [12] and two generic approaches for router virtualization were analysed in [8], namely, Virtualized Separate (VS) and Virtualized Merged (VM). The VS approach creates dedicated logical routing engines on the same hardware platform while the VM approach implements a single routing engine that serves multiple virtual networks. The more power-efficient VS approach is unfortunately not as conducive to scaling, with respect to the number of virtual routers, as VM [8]. However, the VM approach suffers from memory inefficiency when multiple virtual networks are mapped onto the same engine. In order to address these issues, in this paper we propose three novel optimizations to the basic VM approach.

Field Programmable Gate Array (FPGA) has become a widely used platform for networking applications due to its reconfigurability and high performance [16]. Using the numerous logic and memory resources available on the state-of-the-art FPGA, high throughput pipelined architectures can be built for networking applications to meet current and future networking demands. In this work, we use a state-of-the-art FPGA device and demonstrate the power performance and scalability achieved via the optimizations we incorporate in the basic VM approach. The post place-and-route results show that compared with the basic VM approach, the proposed architecture achieves $2\times$ scalability and $7\times$ improvement in power efficiency, while sustaining a throughput of 100 Gbps. We summarize our contributions in this work as follows:

- A scalable and power efficient virtualized router architecture on FPGA
- Three novel optimizations to the basic VM approach
  - Reducing the adverse effect of short routing prefixes
  - Compact representation of routing information for better scalability
  - Power reduction via selective enabling of on-chip memory

The paper is arranged as follows. The background information on router virtualization is summarized in Section II. Section III describes the power and memory issues in VM routers. Section IV contains the key optimizations proposed in this paper. Section V explains our proposed architecture.

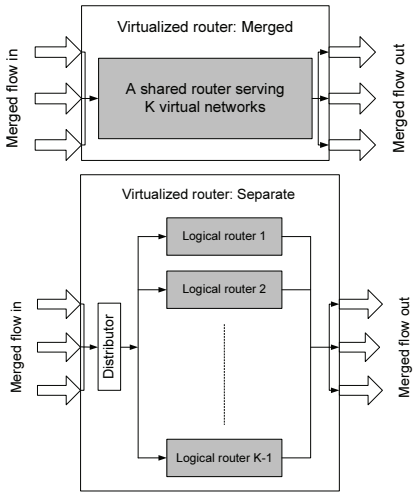

Fig. 1. Merged (top) and separate (bottom) router virtualization approaches

Section VI covers the performance evaluation and Section VII concludes the paper.

II. Background and Related Work

A. Router Virtualization

Router virtualization is an emerging research area that has attracted attention from the industrial and the academic community. The key innovation brought about is that it enables the efficient use of networking hardware by allowing multiple virtual networks to share the same physical substrate. As mentioned in Section I, there are two main approaches to router virtualization and they are as follows:

- Virtualized Separate (VS): Multiple virtual networks share a single hardware platform while maintaining separate lookup engines. The incoming network traffic is distributed across the logical routers, where each logical router is assigned to a virtual network. There is independent control over each of the logical routers [14].
- Virtualized Merged (VM): The multiple virtual routing tables are merged into a single lookup table using a suitable merging technique [5]–[7], [10], [12]. A single logical router is able to route all incoming packets. Lookup is performed using the routing table data corresponding to the specific virtual network that a packet belongs to.

Compared with the conventional non-virtualized approach, where multiple physical devices are required on a per network basis, both the above virtualized approaches have the benefit of being able to share the same hardware platform. As shown in [8], this translates to numerous benefits from scalability and power efficiency standpoints. The two approaches to router virtualization are shown in Figure 1. In this work, we propose techniques to further enhance these features of the VM approach.

B. Trie Based IP Lookup

IP Lookup is the key function in IP routers, which involves retrieving the routing information for all incoming IP packets by matching the destination IP address of each packet against prefixes stored in a routing table [7], [9], [17]. Longest Prefix Matching (LPM) is the algorithm employed to search the routing table and is implemented in all IP routers.

A common data structure used for LPM in FPGA implementations is a trie [6], [9]. The most basic version of a trie will require each node to store two pointer fields and one Next-Hop Information (NHI) field to store prefix information, which is less memory efficient. Various techniques have been proposed to improve the memory efficiency of such tries. Leaf-pushing is one such technique in which all valid prefixes stored at non-leaf nodes are pushed down (or extended) to the leaf level of the trie until all prefixes are represented by a leaf node. This way, the non-leaf nodes store two child pointers while only the leaf nodes store prefix information. We employ leaf-pushing technique in our work and throughout the paper, the term trie refers to a leaf pushed trie.

III. Power and Memory Issues in VM Routers

It has been shown in [8] that the VS approach consumes less power per unit throughput than the VM and the Non-Virtualized (NV) approaches. However, the VS approach necessitates a dedicated pipeline and memory for each virtual router. This acts as a deterrent to the maximum number of feasible virtual routers per chip under the VS approach due to the on-chip resource constraints, limiting its scalability. Hence, for single-chip large-scale router virtualization, the VM approach is preferred.

The VM approach employs a single lookup engine which serves incoming packets belonging to one of several virtual networks. This makes the VM approach better suited for implementation of a scalable virtualized router architecture. However, as shown in [8], the scalability of the VM approach greatly depends on the efficiency of the merging technique. Each leaf-node (i.e. an extended prefix) stores a NHI vector in which the elements are the appropriate NHI for each virtual network. The vector is indexed using the network identifier. Depending on the properties of the prefixes in the routing tables, the occupancy of this vector may vary significantly [8].

A. Power Consumption in VM Approach

Power consumption of a FPGA-based router has two major components: 1) the leakage power which constitutes the static power dissipation and 2) the power consumed by interconnects, logic, and memory elements, which together make up the dynamic power dissipation.

Static power depends on the area of the device as well as the operating temperature, and is majorly unaffected by architecture and circuit level issues. It increases linearly with the number of routers in the NV case. Virtualized router approaches have the distinct advantage that the static power remains constant with increasing number of virtual networks, assuming that all the routers co-exist on the the same chip. This leads to significant savings considering that the average

[1]The occupancy can be defined as the ratio between the non-null elements in the vector to the size of the vector.

static power dissipation of a FPGA-based router is approximately 4 Watts [8].

Dynamic power increases with the number of virtual routers hosted on a chip, as more resources are utilized. In [8], it has been shown that the logic and signal power consumption per stage to be 1.5 mW at 250 MHz. This is small compared to the 6 mW power consumed per 36 Kb block of Block Random Access Memory (BRAM) at 250 MHz, considering that a large number of BRAM blocks are required per stage to store the merged routing table. The power dissipated in the logic elements is low since the logic resources involved in each stage is essentially for a single bit comparison and stage registers of the pipeline.

We achieve a reduction in power consumption of a virtual router using the following two approaches:

1) Dynamic power: By reducing the memory foot print and using clock gating techniques where possible,
2) Per virtual router static power: By increasing the number of virtual routers existing on the same chip.

B. Memory Requirements in VM Approach

The memory required to store the merged routing table in the basic VM approach can be separated into the memory consumed by the pointers to the child nodes at the non-leaf nodes, and the memory needed for storing the NHI corresponding to a particular prefix at the leaf nodes. As mentioned earlier, each non-leaf node contains two child pointers and each leaf node is a vector with every element being a particular NHI indexed according to its corresponding Virtual Network Identifier (VNID). A unique VNID is associated with every virtual network so as to distinguish between the various networks virtualized on the same chip. Under this setup, considering 10 virtual routing tables, the memory required per leaf node is 60 bits and 32 bits per non-leaf node. Thus, the memory required for storing the Next Hop Information (i.e at leaf nodes) is higher than the memory required for storing the pointers (i.e. at non-leaf nodes). More importantly, the leaf node memory requirement also increases at a greater rate with the increase in the number of virtual routers than the memory required for storing non-leaf nodes [8].

With increasing number of virtual networks, the NHI vector grows in size, however the entire vector may not be occupied based on the merging efficiency [8]. This can become a severe bottleneck for scalability when increasing the number of virtual networks. The optimizations proposed in this work reduce both the memory required per leaf node, as well as the number of leaf nodes in the trie representing the merged routing table. We discuss these techniques in detail in the following section.

IV. Power Optimizations for VM Routers

The basic VM router architecture is essentially a pipelined trie search architecture, in which every level of the pipeline corresponds to a trie level. Successive bits of a packet's destination IP are used to traverse the trie, and reach a leaf node. At the leaf node level, the VNID is used as index to access the appropriate NHI from a particular NHI vector.

[2] $200-300$ MHz is the typical operating frequency of IP lookup architecture designs in the literature.

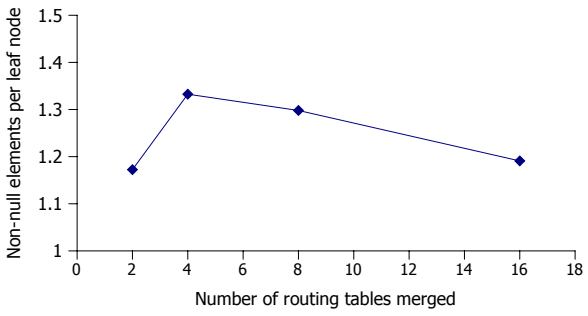

Fig. 2. Average non-null elements per leaf node vector in a merged routing table of 16 virtual routers

We propose an optimized implementation for the basic VM router model, to reduce its power consumption, and in the process, the memory requirements.

A. Optimization I: Memory-Efficient Storage of Leaf Nodes

In the VM approach, the tries corresponding to every leaf node comprises of a vector storing the routing information corresponding to all networks for the prefix represented by the node. The vector is indexed using the Virtual Network Identifier (VNID), to obtain the routing information. The NHI vector may contain null elements when an extended prefix stored in the routing table of one or more routers does not appear in the routing table of another router.

This optimization represents the NHI vector in a compact manner, thereby reducing the memory footprint of the trie. In order to study the occupancy of the NHI vector, we conducted experiments with 16 randomly selected real world IPv4 routing tables obtained from [3] by merging the tries generated from these tables. The routing table sizes varied from 3000 to 7000 prefixes, and contained a total of 65838 prefixes. As shown in Figure 2, the average number of non-null elements per NHI vector stayed within the range $1 - 1.5$, even as the number of virtual routers is increased. This suggests that the vector storage of NHI is an inefficient approach, considering the number of leaf nodes that appear in a trie.

As a remedy to this storage inefficiency, each NHI vector is stored as a complete Binary Search Tree (BST) in our architecture. With this approach, only the non-null elements of the NHI vector need to be stored in the BST. The unique VNID, which corresponds to the NHI vector index in the basic VM approach, is used as the search key. The basic VM router architecture is modified by adding additional pipeline stages for performing BST search at the end of the trie search, as shown in Section V.

The throughput of the virtual router is unaffected by this optimization because of the pipelined architecture. There is a small increase in the latency of the lookup operation, due to the increase in number of pipeline stages necessitated by the BST search. However the increase in latency of the lookup

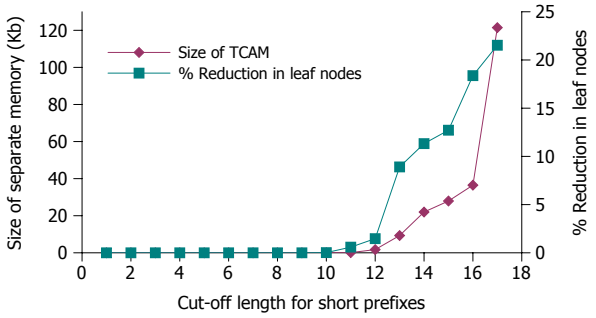

Fig. 3. Effect of Optimization 2 with variation in N_{opt}

operation is minimized by storing as a complete BST, as it provides the best possible ratio between the number of nodes and the height of tree. The height of a complete binary tree with N nodes is at most $O(\log N)$. Storing the NHI vector as a complete BST also ensures that the worst case run time complexity is $O(\log N)$.

B. Optimization II: Separate Storage for Short Prefixes

Short prefixes lead to a greater expansion in the number of non-leaf nodes after leaf-pushing compared to longer prefixes. In the context of the NHI vector, shorter prefixes expand to the leaf nodes of large sub-tries, hence filling the corresponding elements in the NHI vector. This increase in the number of non-null elements per NHI vector adversely affects the first optimization. We mitigate this effect of short prefixes by eliminating them from the merged trie structure and storing them separately in a Ternary Content Addressable Memory (TCAM). We define short prefixes to be prefixes of length smaller than a length N_{opt}. The length, N_{opt}, is determined by the prefix length distribution in the routing tables of the virtual routers, taking into consideration, the trade-off between the size of the TCAM and the additional power consumption introduced by the TCAM. Note that the TCAM match will be dropped if the trie search results in a longer prefix match.

Short prefixes are generally less in number, as per the distribution of the prefix lengths in real world routing tables demonstrated in Figure 3. As a result, the memory required to store the prefixes separately is small. More importantly, the power savings resulting from the reduced memory storage due to the decreased number of trie nodes and NHI vector elements, outweigh the power consumed by TCAM.

Using the real world routing tables described in the previous section, we found that for $N_{opt} = 16$, there was a 18% reduction in the number of leaf nodes, and the size of the TCAM needed was 36.5 Kb. The percentage of short prefixes was only 0.055% of the total number of prefixes.

The above two optimizations lead to a reduction in memory resources used to store the merged trie. This allows for a higher number of virtual routers per chip, due to which the static power being consumed by the device is shared between more routers resulting in lesser average power dissipation. Also, these optimizations improve the scalability of the VM router model, by reducing the memory costs of virtualization.

C. Optimization III: Selective Enabling of Memory

A BRAM block that is enabled and clocked, contributes to the overall power consumption, even if the BRAM block is not being accessed. Using the clock gating technique, the power consumed by the unused BRAM blocks can be eliminated. This leads to power savings of about 5 mW (at an operating frequency of 250 MHz) per disabled BRAM [15]. In this optimization, we explore methods to incorporate clock gating into our lookup architecture.

The search pipeline in this approach consists of three phases: 1) the parallel TCAM search, 2) trie search and finally, 2) the BST search. Even though all three phases may benefit from clock gating techniques, adding such facility comes at the cost of sacrificing the throughput of the overall architecture. For example, in the trie search, in order to include clock gating, each stage memory has to be partitioned into multiple BRAM blocks, which requires each BRAM block to be accessed independently. This complicates the routing of the design and deteriorates the throughput of the design.

The TCAM size in our approach is relatively small (compared to the size of a BRAM block), therefore clock gating will not yield significant power savings. Therefore, we apply clock gating only to the BST search. To enable this, we divide the memory required to store the BST node information into multiple partitions instead of storing them in an individual block. A single partition is able to operate independently and at the preprocessing step, we ensure that nodes from a given BST does not reside in two partitions. This provides us the facility to turn on only one partition for an incoming packet. In the architecture, this is enabled by associating a Unique BRAM ID (UBID) with each leaf node, corresponding to the BRAM partition where the BST pertaining to the leaf node is stored. While the granularity of this partitioning can vary from no partitioning (i.e. total memory is a single partition) to fine grained (i.e. each RAM block is a partition), increasing the number of partitions leads to diminishing returns. In our work, we limit the number of partitions to 10. A FPGA implementation of TCAM is described in [13].

V. Optimized Virtual Router Architecture

The overall architecture is shown in Fig. 4. The short prefixes are stored in the TCAM. Each level of the tree and trie structure is assigned a distinct pipeline stage so that a lookup request can be issued every cycle, thus increasing the throughput. The number of stages in the trie search section is determined by the maximum length of the prefixes (32 for IPv4) and the number of virtual routers merged. The merged routing table is stored as a uni-bit leaf pushed trie. The NHI of various networks of a particular prefix is stored as a BST indexed using its appropriate VNID, at the leaf node representing the prefix.

The lookup procedure is as follows. The destination IP address is extracted from an incoming packet and a VNID is associated with the packet to represent its virtual network before the packet is routed to the lookup engine. There are two

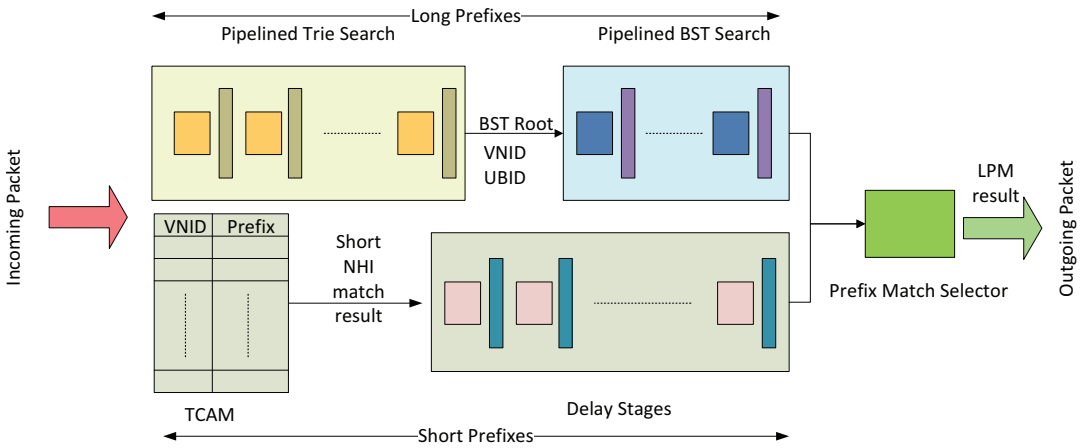

Fig. 4. Modified VM router architecture

parallel lookup operations, for short and long prefix matches respectively.

A TCAM search is performed to identify whether any short prefixes match the incoming packet. The VNID appended to the destination IP address is used as the key for the TCAM search. Any match result from the TCAM stage is passed through dummy stages to match the delay of the long prefix search operation. For the long prefix match lookup, the trie is traversed using the bits of the packet's destination IP address. When a leaf node is reached, the corresponding BST root and the UBID is extracted and the information is carried on to the BST search. The BRAM partition corresponding to the UBID is activated and the BST information is loaded based on the BST root forwarded from the trie search. The VNID is used to search the BST and the match result from BST lookup, if any, is forwarded to the prefix match selector stage. The short prefix match is selected by the prefix selector stage, in the case where no longer prefix match can be found, to be used as the LPM for the incoming packet; else the long prefix match is forwarded as the LPM.

VI. Performance Evaluation

A. Experimental Setup

To illustrate the benefits of the proposed optimizations, we conducted experiments on a state-of-the-art Xilinx Virtex 7 platform (XC7V2000T). Note that none of the proposed optimizations are device or vendor specific.

For the purposes of fair evaluation, we demonstrate the effects of the proposed optimizations over a wide range of parameter values (parameters being percentage of null elements, amount of leaf node reduction of short prefix removal, etc.) as per the results obtained from the experiments conducted on real world routing tables in section IV.

Throughout this paper, for computing the memory requirements, we use 6 bits per NHI field. Note that the length of the child pointers need not be uniform across the pipeline stages. The average pointer width chosen accounts for all the nodes that appear in the merged trie. The baseline implementation

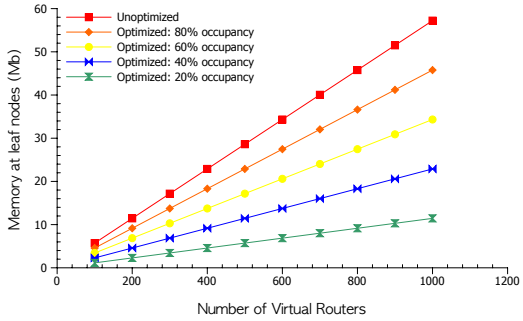

Fig. 5. Variation in memory consumption at leaf nodes with increase in virtual routers

chosen for all the comparisons is the basic VM router implementation on FPGA, as described briefly in Section IV and elaborated in [5].

B. Optimization I

We demonstrate performance improvements due to the first optimization, considering a range of values for the percentage of null elements in the leaf node vectors from 20% to 80%. These values are smaller compared to those observed in Section IV-A, for a fair evalution. The number of virtual routers varies from 10 to 1000 and each individual routing tables has 10000 entries. Even if the percent of null elements is as low as 20%, there is an reduction of about 12 Mb and also 0.35 W (about 8%) of power savings. Significant power and memory savings of 2.55 W (about 50% of total power) and 45.77 Mb are achieved if the percentage of null elements is as high as 80%, which is close to the percentage of null elements observed in real world merged routing tables. Thus we save 52.7% of the total on-chip memory and can increase the number of virtual routers by upto six times the number possible in the unoptimized case.

C. Optimization II

The separate storage of short prefixes, was encouraged by a need to reduce the number of leaf nodes, as most of the

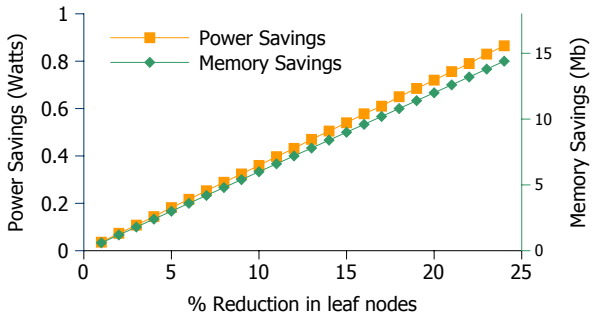

Fig. 6. Power and memory savings for varying amounts of reduction in number of leaf nodes

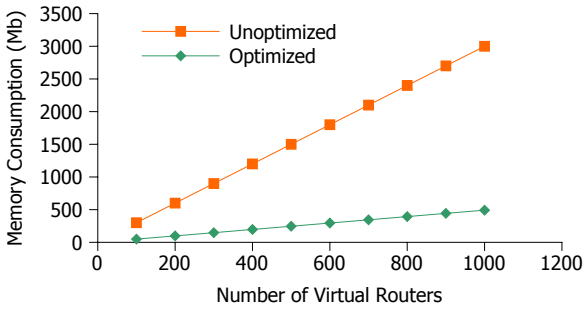

Fig. 7. Memory savings

memory consumed in the VM approach is at the leaf nodes. In Fig. 6, we show the reduction in power and memory requirements, considering varying percentages of reduction in leaf nodes over the range of 1 to 24% in a merged routing table containing 1 M leaf nodes, on applying the second optimization. Considering the 18.45% reduction obtained from the real world routing tables for N_{opt}=16, we get 0.6 W(15% of total) of power savings, and correspondingly 10Mb reduction in the memory required by the merged routing table.

The memory and power savings offered due to the first two optimizations increase with the number of routers, thus leading to a scalable implementation. This is illustrated in Fig 6, using routers having individual rulesets with 500k prefixes. We assume 80% null elements at the nodes, and 18% reduction of leaf nodes, as observed in real-world data as mentioned in Section IV.

D. Optimization III

We determine the power savings offered by selective enabling of the BRAMs by considering the case of 100 virtual routers (to ensure complete usage of available on-chip resources), with 1 M leaf nodes in the merged routing table with 40% null elements, and 6 bit NHI. This results in 60 Mb required to store the entire forwarding information at the leaf nodes and correspondingly 3.6 W of power consumed by on-chip memory. Each BST can store upto a hundred 6 bit NHI values in the worst case, and hence is entirely stored in a single BRAM. Thus, only a single BRAM is accessed per packet by the BST search level of the pipeline, and the power consumed by the BST search decreases almost completely to 2 mW per packet.

E. Power and Memory Analysis of FPGA Architecture

We present here a comparison of the power and memory requirements of the complete architecture with all the three optimizations applied simultaneously, with the baseline implementations of NV, VS and VM(basic) router approaches. To demonstrate the scalability of our model, we considered 75 routers with 100k prefixes in their individual routing tables, which ensured that the on-chip resources were completely used up in our design. Actual IPv4 prefixes were not generated, instead the memory consumption was computed considering the worst case number of trie nodes to be spread evenly across all the trie levels. The increase in the number of non-leaf nodes with the number of merged lookup tables was estimated based on the real-world routing tables described in Section IV. The forwarding information is stored at the leaf nodes as a BST, and we assume a 9% reduction in number of leaf nodes after removing the short prefixes. The computations were done considering 80% null elements per NHI vector. The values assumed above represent conservative estimates considering the observed values discussed in Section IV. The routing tables in all the designs were stored in on-chip memory. Table I shows a comparison of power and memory consumption of the various router approaches.

TABLE I
POWER AND MEMORY COMPARISON

	NV	VS	VM (Basic)	VM (Opt.)
No. of chips required	75	3	2	1
Memory (in Mb)	149.76	149.76	96.09	47.1
Power (in W)	169.18	22.66	33.63	4.59

The number of chips needed for VS and VM router implementations was based on the on-chip memory available on the chosen Virtex 7 FPGA device. The NV router approach was the worst performer in terms of the hardware required (75 FPGA chips) and the power consumed, although the memory requirement was the same as the VS approach. The basic VM router required about 36% lesser memory than the VS model, but at the cost of almost 50% more power. Our optimized VM router approach was able to fit all 75 routers on the same FPGA chip, utilized 50% less memory as compared to the next most memory-efficient approach and more importantly, consumed 20% of the power of the next best power-efficient router approach.

Figure 7 shows a comparison between the memory consumption of our optimized approach and the conventional approach to the VM model for router virtualization. The memory savings increase with the number of routers on the same chip. The possible memory savings obtainable are only limited by the FPGA resource constraints, and the benefits of the developed architecture will only improve with the advancements in FPGA technology.

VII. CONCLUSION AND FUTURE WORK

In this work, we proposed a modified VM router architecture to improve on the power consumption and memory

requirements of the basic router VM approach. The proposed optimizations makes the VM approach an attractive choice for the implementation of scalable and power efficient virtual routers on Field Programmable Gate Array (FPGA) platform. We presented the results of implementing our modified VM router model under a variety of conditions of the merged routing tables. We have illustrated the effect of the proposed optimizations on real world routing tables.

An interesting observation in the paper is that the proposed optimizations lead to the memory and power consumption at the non-leaf nodes now becoming comparable to that at the leaf nodes. Hence, further modifications to the VM router approach should aim at reducing this contribution to the overall power as well, possibly by revisiting the concepts of path-compressed tries.

REFERENCES

[1] J. Carapinha and J. Jiménez. Network virtualization: a view from the bottom. In *Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures*, VISA '09, pages 73–80, New York, NY, USA, 2009. ACM.

[2] N. M. K. Chowdhury and R. Boutaba. A survey of network virtualization. *Comput. Netw.*, 54(5):862–876, Apr. 2010.

[3] CIDR. Cidr report for 25 jun 12. http://www.cidr-report.org/as2.0/.

[4] O. N. Foundation. Software-defined networking: The new norm for networks. https://www.opennetworking.org/images/stories/downloads/white-papers/wp-sdn-newnorm.pdf.

[5] J. Fu and J. Rexford. Efficient ip-address lookup with a shared forwarding table for multiple virtual routers. In *Proceedings of the 2008 ACM CoNEXT Conference*, CoNEXT '08, pages 21:1–21:12, New York, NY, USA, 2008. ACM.

[6] T. Ganegedara, W. Jiang, and V. Prasanna. Multiroot: Towards memory-efficient router virtualization. In *Communications (ICC), 2011 International Conference on*, June 2011.

[7] T. Ganegedara, H. Le, and V. Prasanna. Towards on-the-fly incremental updates for virtualized routers on fpga - unpublished article. In *Field Programmable Logic and Applications (FPL), 2011 International Conference on*, Sep. 2011.

[8] T. Ganegedara and V. Prasanna. Fpga-based router virtualization: A power perspective. In *Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW), 2012 IEEE 26th International*, pages 360 –367, may 2012.

[9] W. Jiang and V. Prasanna. Towards green routers: Depth-bounded multi-pipeline architecture for power-efficient ip lookup. In *Performance, Computing and Communications Conference, 2008. IPCCC 2008. IEEE International*, pages 185 –192, dec. 2008.

[10] H. Le, T. Ganegedara, and V. Prasanna. Memory-efficient and scalable virtual routers using fpga. In *Field Programmable Gate Arrays (FPGA), 2011 International Symposium on*, March 2011.

[11] A. M. Lyons, D. T. Neilson, , and T. R. Salamon. Energy efficient strategies for high density telecom applications. http://imsresearch.com/press-release/Internet_Connected_Devices_About_to_Pass_the_5_Billion_Milestone&from=all_pr.

[12] H. Song, M. Kodialam, F. Hao, and T. Lakshman. Building scalable virtual routers with trie braiding. In *INFOCOM, 2010 Proceedings IEEE*, pages 1 –9, March 2010.

[13] Z. Ullah, M. Jaiswal, Y. Chan, and R. Cheung. Fpga implementation of sram-based ternary content addressable memory. In *Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW), 2012 IEEE 26th International*, pages 383 –389, may 2012.

[14] D. Unnikrishnan, R. Vadlamani, Y. Liao, A. Dwaraki, J. Crenne, L. Gao, and R. Tessier. Scalable network virtualization using fpgas. In *Proceedings of the 18th annual ACM/SIGDA international symposium on Field programmable gate arrays*, FPGA '10, pages 219–228, New York, NY, USA, 2010. ACM.

[15] Xilinx. Reducing switching power with intelligent clock gating. http://www.xilinx.com/support/documentation/whitepapers/wp370IntelligentClockGating.pdf.

[16] Xilinx. Xilinx xcell journal. http://www.xilinx.com/publications/xcellonline/.

[17] K. Zheng, C. Hu, H. Liu, and B. Liu. An ultra high throughput and power efficient tcam-based ip lookup engine. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, volume 3, pages 1984 –1994 vol.3, march 2004.