Research Statement

Moein Khazraee (moein@csail.mit.edu)

Network bandwidth is outpacing our ability to process packets in software, which consequently forces network developers to resort to specialized hardware. However, hardware development is inherently intricate and laborious, resulting in lengthy and high-cost developments. Integrating specialized hardware into a networked application requires hardware-software co-design, which is challenging because developers with markedly different specializations have to collaborate and go through revision cycles to make an efficient and high-performance system. My research has focused on building frameworks to lower the *barrier to entry* for use of customized hardware in networking, by finding the practical abstractions to facilitate collaboration across hardware and software boundaries. I am uniquely suited to address this problem because of my expertise in both networked systems and computer architecture. I have built hardware frameworks for network middleboxes, wireless networks, networked applications such as machine learning in datacenters, and more recently I have started to build a framework for Silicon Photonics (SiP) based networks.

My approach is to develop a deep understanding of the requirements and opportunities in networking, alongside benefiting from my background in computer architecture, to make high-performance packet processing frameworks on top of the already available networking platforms with customizable hardware (FPGAs). As these frameworks are targeted for various aspects of networking, they have to provide befitting abstractions. For example, we need hardware acceleration for signal processing in wireless, deep packet inspection in middleboxes, and efficient communication within Machine Learning (ML) clusters in datacenters. I have built hardware frameworks for wired (Sec. 1) and wireless networks (Sec. 2), as well as extended the use of specialized hardware in datacenters (Sec. 3). Shire [1] is the first 200 Gbps framework for wired network middleboxes, which brings a C-like programming model to dynamically configure, invoke, and update the hardware accelerators. SparSDR [5] is the first framework for wireless networks that maintains the universality of Software Defined Radios (SDRs), while allowing wide-band real-time processing on a low-end processor. These frameworks provide order-of-magnitude development and equipment cost reductions, and SparSDR has already been open-sourced and adopted by several research groups.

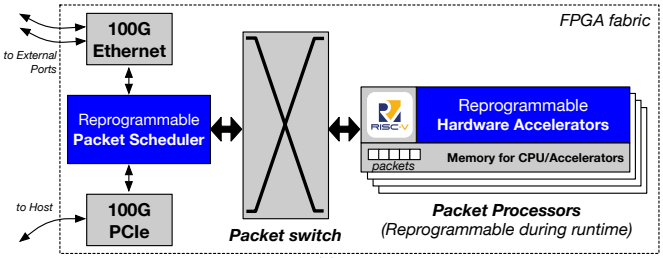
Furthermore, my work as a computer architect was focused on lowering the barrier to use custom hardware accelerators for planet-scale applications (Sec. 3.1). We evaluated the impacts of performance, power and cooling on the Total Cost of Ownership (TCO) of a datacenter made from fully customized chips [4, 8, 10]. We also made the first comprehensive manufacturing and development cost model in academia, which counter-intuitively showed that using the most advanced technology is not necessarily the best option [7, 9]. This work not only gave me remarkable insight into the hardware acceleration capabilities, but also highlighted the need for more efficient networking.

I believe the next breakthrough in networking will come from the use of new technologies such as SiP, as well as co-optimization of the network stack from the physical layer to higher network layers, even all the way to the application layer for dominant applications such as ML. To this end, we have demonstrated the benefits of co-optimizing the physical layer network topology and application-level parallelization strategy for ML training workloads, using the current reconfigurable optical switching technology [2]. We were able to achieve up to three times reduction in training time (Sec. 3.2). I will further discuss SiP related research in future work.

1 Hardware specialization in wired networks

There is ample opportunity to benefit from specialized hardware for networked applications. Currently 100 Gigabits per second (Gbps) links are becoming widespread in enterprises, 400 Gbps links are being deployed in datacenters, and bandwidths in the order of Terabits per second are expected in the near future. Even though customized hardware accelerators can achieve orders of magnitude higher performance than general purpose processors, their development and maintenance is challenging and laborious without a framework, especially for agile and high-performance workloads found in network middleboxes.

The goal of the Shire framework [1] is to handle the common functionalities, such as data distribution and on-the-fly updates, and let the developer focus only on the desired functionality. Shire builds upon the familiar abstraction of network packets, where a packet scheduler load-balances the incoming packets among the packet processors. In these packet processors, there is a RISC-V soft core sitting next to the hardware accelerators, alongside a



(a)

```
// Read ip from header
unsigned int src_ip =
    *((unsigned int*) (desc->data + 14 + 12));
// Pass the value to the IP checker engine
ACC_SRC_IP = src_ip;
// Drop or forward by changing descriptor
if (ACC_FW_MATCH) desc->len   = 0;
  else            desc->port ^= 1;
// Send the packet out
pkt_send(desc);
```

(b)

(a) Architectural overview of the Shire framework. Shire provides a simplified abstraction for FPGA-accelerated high-performance network middleboxes. The packet scheduler distributes the incoming packets among the packet processors, and into their hybrid memory, which is accessed by a RISC-V processor and its attached hardware accelerators. Software on RISC-V controls the data flow. The host can access packet processors memories and swap them during runtime. (b) Example firewall code parsing the header, calling the accelerator, and making decisions.

tailored hybrid memory that is a central piece in achieving the desired performance. Shire acts similar to a software middleware: packets arrive at the memory of a processor and the software controls the data flow among hardware accelerators, and communicates with the host and the rest of the system through the abstraction of descriptors. Shire also provides host-side libraries for the developers to be able to interact with packet processors, access their memories, and swap them during runtime. Shire will be open-sourced after publication for public use.

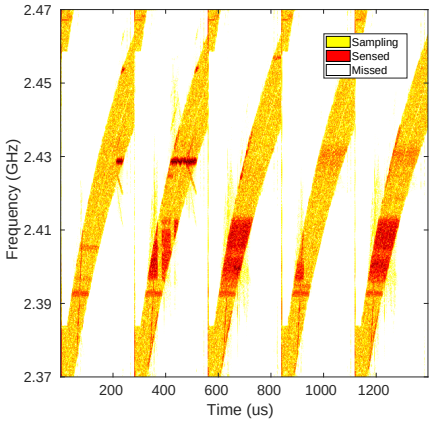
Shire enables software control and software-like debugging, resulting in significantly faster hardware development. For example, **we made the first 200 Gbps intrusion detection system with *runtime* updatability in less than 3 weeks**, by plugging-in a *compile time* updatable string-matcher engine from another work. That work **took about 2 years as they had to develop the surrounding infrastructure for 100 Gbps**. We also made a basic firewall in one week by just implementing a basic IP matcher accelerator. The primary research challenge was to find the practical abstractions that can be implemented efficiently for high-performance systems, yet are practical and well-known. Design considerations for high-performance and multi-die FPGAs, partial reconfiguration constraints required for runtime hardware updates, and use of slow processors made from logic fabric—more than 10 times slower than embedded processors such as ARM—further made implementation of this framework intriguing.

2 Hardware specialization in wireless networks

I am fascinated by the universality offered by SDRs, making them protocol independent. They achieve this universality by decoupling the radio frontend—including analog RF components and digitizers—from the signal processing backend. However, they have limitations which made them a niche specialized hardware, and I took novel approaches to unlock two new capabilities for commodity SDRs: (1) full spectrum monitoring, and (2) wide-band processing on low-end processors.

2.1 Customized hardware to sweep frequency bands

Hardware accelerators can offer new capabilities and innovations for wireless networking. As an example, I designed an accelerator to enable fast sweeping of the wireless frequencies in the order of nanoseconds, instead of milliseconds achievable through software [6, 3]. This enabled us to sweep the full spectrum range of 5 GHz in only 5 milliseconds. We also designed a novel algorithm to detect packet type based on partially captured signals. Through this process we were able to **sense and detect packets across the entire spectrum with a commodity narrow band capturing device (∼$1000), instead of an expensive spectrum sensor (∼$500,000)**.



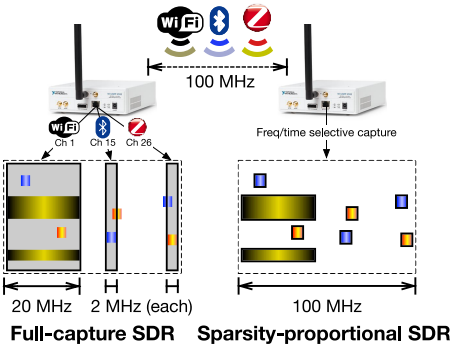
SweepSense is a low-cost radio architecture that senses the frequency spectrum by rapidly sweeping across it. Our novel algorithm classifies the packets from the partially captured data across the full spectrum, at 500 times lower cost than the previously available wide-band radios.

2.2 Efficient framework for SDRs

The main drawback of SDRs is their demand for high network bandwidths and processing power. For instance, using a 100 MHz wide SDR to capture the frequency hopping of a Bluetooth signal, we need to process 3.2 Gbps of continuous data on a high-end server, to extract only 1 Mbps of useful information from infrequent transmissions. My goal in the SparSDR [5] framework was to bring back the concept of packets and only deliver active transmissions to the software. Therefore, I implemented active signal detection in hardware, in addition to channelization. Using

this framework, developers can select the desired bands in a wide-band radio capture, and only need to process the actual transmissions.

The SparSDR framework keeps the universality promise of SDRs, and significantly increases their efficiency. For example, we were able to **capture 4 times wider bands with the same radio frontend device**, and do **real-time Bluetooth decoding with a Raspberry Pi as backend**. We also unlocked **3 times wider bandwidth for the low-cost (∼$200) PlutoSDR.** I believe the improved efficiency of SDRs can lead to wider adoption, as well as leaving more room for the developers to try out their ideas and keep up with the performance demands. This work has already been used for a study on the privacy of COVID-19 contact tracing with Smartphones for a recent paper [1], where they captured raw data for several days without internet coverage. Previously this **required ∼400 TB of data using a wideband SDR, alongside a high-end workstation**. Using SparSDR, they used a **commodity SDR device** alongside a **mini-PC**, which were fitted inside a small box and the total capture became **only 1.2 TB**. Besides, several research institutes such as the Lawrence Livermore National Laboratory, the Secure Mobile Networking Lab in Germany, and SofterHardware enthusiasts for amateur SDRs showed interest in use of SparSDR.



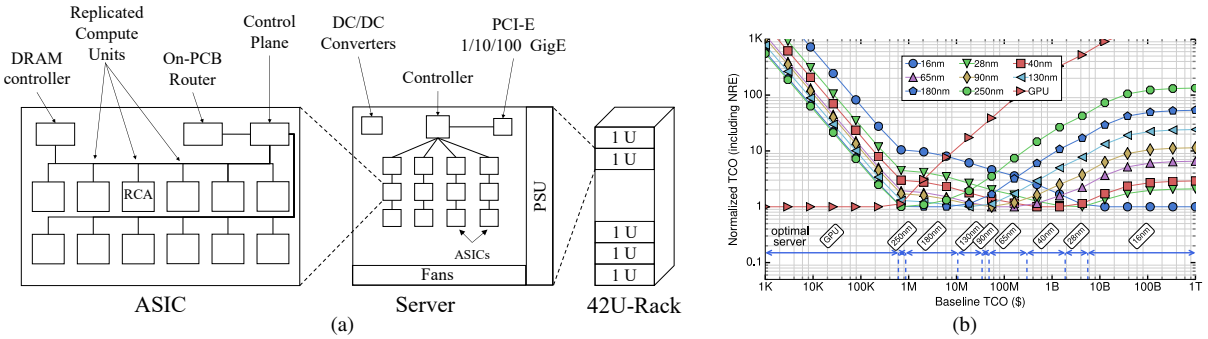
SparSDR is a resource-efficient framework for SDRs whose backhaul bandwidth and compute power requirements are proportional to the active transmissions. SparSDR dramatically reduces processing power requirement, while retaining both flexibility and fidelity. As an example, it enabled 4x wider band captures from a commodity SDR module, and also enabled performing real-time Bluetooth decode on a Raspberry Pi instead of a high-end server.

3 Hardware specialization for datacenters

Use of specialized hardware for highly demanded high-performance compute workloads inside datacenters has become inevitable. I took two approaches to further customize the hardware for datacenters: (1) Designing fully customized servers made from fully customized chips (2) Using reconfigurable optical network switches to co-optimize ML training parallelization strategy and the network topology among the specialized compute servers.

3.1 Use of fully customized chips for scale out applications

As a computer architect, I focused on lowering the barrier to use custom hardware accelerators for planet-scale applications that can be scaled out. I developed a framework that **uses a single hardware accelerator as the abstraction, and scales it out to the scale of a datacenter** [4, 8, 10]. We specialized the ASIC server by employing optimized ASICs, a customized circuit board, custom-designed cooling and specialized power delivery systems, in addition to a tailored external memory and networking. We evaluated applications with different characteristics: Bitcoin, Litecoin, ML inference and training, and video transcoding. We showed that the performance-optimal or energy-optimal designs are not Total Cost of Ownership (TCO) optimal. Furthermore, **we developed the first comprehensive model in academia for estimating customized hardware development costs**, such as development, manufacturing and operational costs [7, 9]. This work provided me with the opportunity to extend my work at Google, where I was involved in doing the due diligence for use of ASIC chips for YouTube transcoding workloads, which eventually became the Google VCU chip that was announced earlier this year.

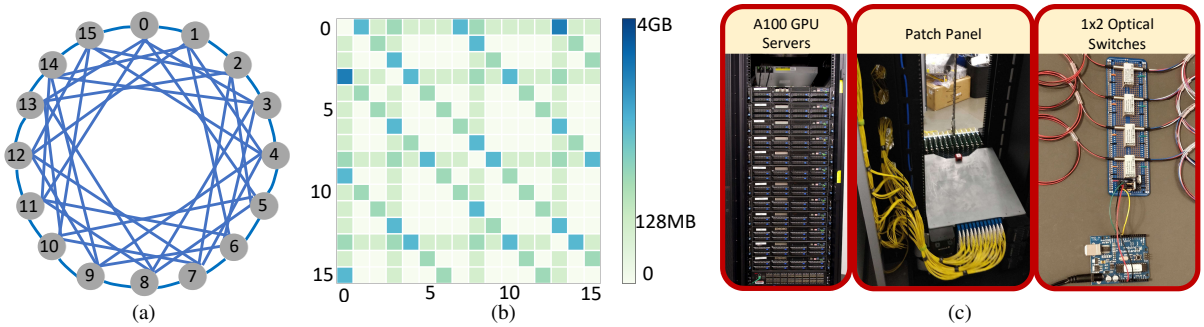


(a) High-level abstract architecture of an ASIC Cloud. Specialized replicated compute accelerators (RCAs) are multiplied up by having multiple copies per application-specific integrated circuit (ASIC), multiple ASICs per server, multiple servers per rack, and multiple racks per datacenter. The power delivery and cooling system are customized based on ASIC needs. If required, external memory and networking are also considered. (b) TCO-optimal ASIC Cloud for Bitcoin. For each total cost value, the best server scenario is set to 1 and values for other scenarios represent the ratio of total money spent. For many applications, cutting edge node technologies become TCO-optimal only for extreme-scale ASIC Clouds.

[1] Givehchian, Hadi, et al. "Evaluating Physical-Layer BLE Location Tracking Attacks on Mobile Devices.", *IEEE Security and Privacy*

3.2 Use of reconfigurable optical network switches for ML

We observed that network traffic is highly unbalanced for several ML training applications, which leads to underutilized links. Since ML parallelization strategy dictates the network traffic, we co-optimized the network physical topology and parallelization strategy to achieve a more balanced network utilization [2]. The main research challenge was figuring out how to explore the solution space, which is an NP-hard problem, as well as implementing the solution efficiently with currently-available network equipment and modifying the software stack accordingly. **We achieved up to 3 times higher performance at the same total cost**.



(a) (b) (c)

(a) Use of Ring-AllReduce permutations to increase direct connectivity among nodes. (b) By co-optimization of parallelization strategy and network topology we made the traffic pattern significantly more uniform, resulting in up to 3x lower training times. (c) Our testbed using a patch panel with automatic robot grip, and 1x2 optical switches in a look-ahead approach to enable fast topology updates between workloads.

4 Future work

The ambition for my future work is to make use of specialized hardware widespread, for both current and future networks. To this end, I will follow two objectives: make use of specialized hardware not to be niche and exclusive to large companies (Sec. 4.1), and lower the barrier to entry for innovations upon newer technologies (Sec. 4.2). Furthermore, as most of my previous work has been interdisciplinary, I will elaborate the directions that I foresee collaborating with my colleagues (Sec. 4.3).

Among future avenues that I envision to pursue my goals, I believe we can achieve the first objective by enabling developers to share their specialized hardware just like sharing software libraries, for example for middleboxes and SDRs which are among the most compute-intensive elements in networking. Regarding the second objective, I plan to make open-sourced hardware frameworks for new technologies, which will in turn enable researchers to innovate without necessarily owning the actual hardware. I am particularly interested in the rising technology of SiP, which I believe is a unique opportunity for the networking community to impact the future of networks. All these directions match **my research methodology to (1) use specialized hardware for a few applications to find the proper abstractions, (2) make a framework to lower the barrier to entry to use the specialized hardware, and (3) collaborate with other researchers to enable unconventional innovations.**

4.1 Make specialized hardware to become similar to software libraries

For the short- and medium-term I am going to make the SparSDR framework more energy efficient, and add more flexibility to the Shire framework. Energy efficiency is becoming of utmost importance in the age of the Internet of Things (IoT) and I believe there is room for an order of magnitude improvement to reinvigorate SDRs for this purpose. We are building a hybrid system where the SparSDR framework adds some metadata, which helps the software running on an energy-efficient integrated processor to only select the signals from the desired protocol. Furthermore, I am going to collaborate with FPGA manufacturers such as Xilinx, to do minor modifications to the hardened IP blocks already available in their Versal FPGAs. This can significantly reduce the area overhead of the Shire framework, and increase the clock speed of the processors sitting next to the accelerators. These two improvements will result in more slack for the developer, in terms of the amount of the code they can run in software, and area to use per accelerator. Based on these two works, I can accomplish my long-term goal of making specialized hardware be more accessible and also swappable during runtime, like a software library. Examples of such libraries would be protocols for low-cost and energy-efficient SDRs, and hardware accelerators ready for partial reconfiguration on the supported FPGAs.

4.2 Enable innovations for specialized hardware using new network technologies

The SiP technology uses MicroRing Resonators to increase the achievable bandwidth by an order of magnitude, while improving energy efficiency and lowering latency. The fundamental concept is to directly attach fibers to the silicon chip, and use several wavelengths per fiber to achieve a higher bandwidth. However, SiP introduces new challenges such as crosstalk among wavelengths on the same fiber, or higher bit error rates than current short-range transceivers, and these characteristics are different for each manufacturer.

I see a new technology and its challenges as a fertile ground for research and collaboration. Recently, we have started a collaboration with Columbia University to study the impact of bandwidth on bit error rates. I want to bring some of the techniques from the wireless domain and tailor them to SiP to get higher aggregate throughput. The implementation challenges, while facing constraints of high-performance compute workloads in the datacenters, as well as the need for cross layer co-optimization, make this problem unique and intriguing. As the first step, I am developing a cycle-accurate hardware simulator for the SiP connections among hardware accelerators. We utilize this hardware framework with our collaborators at UC San Diego to evaluate the potential of SiP for ML training, which is the next step to our current NSF grant that was awarded 1.2 million dollars.

These two projects are the building blocks for my medium-term goal of making a simulator that enables collaboration and co-optimization among three domains of optical interconnects, chip design, and networking. The goal is to explore the physical possibilities available in the optical interconnects, benefit from the novelties in chip design, and co-optimize through network and application layers. In addition to Columbia University, I have also started to collaborate with AyarLabs and HPE to be able to model their SiP and find the proper abstractions. For the long term, I see high bandwidth and low latency SiP links as a potential to go beyond the per chip compute limitations of today. This opens up several potential research directions, such as design of reliable yet performant communication links, network traffic scheduling, and finding the optimal topology among pool of compute nodes.

4.3 Opportunities for collaboration with other faculty

I am looking forward to collaborating with my colleagues from other fields, as well as continuing my current collaborations with faculty members at MIT and UC San Diego. I believe my currently developed frameworks, and the ones that I am set to build in the future, are great opportunities for collaboration. The Shire framework can be a great building block for disaggregated and serverless computing system researchers. My works in wireless domains enable low cost wireless sniffers and spectrum monitors that can be a starting point for geographically spread deployment for threat detection by security researchers. Wireless and communication theory researchers can build on top of the SparSDR framework to achieve a unified gateway for various IoT devices and in 5G networks. Finally, the SiP simulator paves the path for collaboration with researchers in optical and chip design domains, and as the next step we can bring the optical compute and SiP technologies together.

References

[1] **M. Khazraee**, A. Forencich, G. Papen, A. Snoeren, A. Schulman, "Shire: Making FPGA-accelerated Middlebox Development More Pleasant", *arXiv*, 2022

[2] W. Wang, **M. Khazraee**, Z. Zhong., Z. Jia, D. Mudigere, Y. Zhang, A. Kewitsch, M. Ghobadi, "TOPOOPT: Optimizing the Network Topology for Distributed DNN Training", Submitted to *USENIX NSDI*, 2023

[3] Y. Guddeti, R. Subbaraman, **M. Khazraee**, A. Schulman, D. Bharadia, "Towards Low-Cost, Ubiquitous High-Time Resolution Sensing for Terrestrial Spectrum", *ACM GetMobile*, 2020

[4] M. B. Taylor, L. Vega, **M. Khazraee**, I. Magaki, S. Davidson, D. Richmond, "ASIC Clouds: Specializing the Datacenter for Planet-Scale Applications", *Communications of the ACM*, 2020

[5] **M. Khazraee**, Y. Guddeti, S.Crow, A. Snoeren, K. Levchenko, D. Bharadia, A. Schulman, "SparSDR: Sparsity-proportional Wideband SDRs", *ACM Mobisys (International Conference on Mobile Systems)*, 2019

[6] Y. Guddeti, R. Subbaraman, **M. Khazraee**, A. Schulman, D. Bharadia, "SweepSense: Sensing 5 GHz in 5 Milliseconds with Low-cost SDRs", *USENIX NSDI*, 2019

[7] S. Xie, S. Davidson, I. Magaki, **M. Khazraee**, L. Vega, L. Zhang, M. B. Taylor, "Extreme Datacenter Specialization for Planet-Scale Computing: ASIC Clouds", *ACM SIGOPS OSR (Operating Systems Review)*, 2018

[8] **M. Khazraee**, L. V Gutierrez, I. Magaki, M. B. Taylor, "Specializing a planet's computation: ASIC Clouds", *IEEE Micro top picks*, 2017

[9] **M. Khazraee**, L. Zhang, L. V Gutierrez, M. B. Taylor, "Moonwalk: NRE optimization in ASIC Clouds, or, accelerators will use old silicon", *ACM ASPLOS (Architectural Support for Programming Languages and Operating Systems)*, 2017

[10] I. Magaki*, **M. Khazraee***, L. V Gutierrez, M. B. Taylor, "ASIC clouds: specializing the datacenter", *ACM/IEEE ISCA (International Symposium on Computer Architecture)*, 2016 (* equal contribution))