Fieldable Cross-Layer Optimized Embedded Software Defined Radio is Finally Here!

Jithin Jagannath, Anu Jagannath, Justin Henney, Noor Biswas, Tyler Gwin, Zackary Kane, Andrew Drozd

Marconi-Rosenblatt AI/ML Innovation Lab, ANDRO Computational Solutions, LLC, Rome NY

{jjagannath, ajagannath, jhenney, nbiswas, tgwin, zkane, adrozd}@androcs.com

Abstract—The concept of cross-layer optimization has been around for several years now. The primary goal of the cross-layer approach was to liberate the strict boundary between the layers of the traditional OSI protocol stack. This is to enable information flow between layers which then can be leveraged to optimize the network's performance across the layers. This concept has been of keen interest for tactical application as there is an overwhelming requirement to operate in a challenging and dynamic environment. The advent of software defined radios (SDR) accelerated the growth of this domain due to the added flexibility provided by SDRs. Even with the immense interest and progress in this area of research, there has been a gaping abyss between solutions designed in theory and ones deployed in practice. To the best of our knowledge, this is the first time in literature, an embedded SDR has been leveraged to successfully design a cross-layer optimized transceiver that provides high throughput and high reliability in a ruggedized, weatherized, and fieldable form-factor. The design ethos focuses on efficiency and flexibility such that optimization objectives, cross-layer interactions can be reconfigured rapidly. To demonstrate our claims, we provide results from extensive outdoor over-the-air evaluation in various settings with up to 10-node network typologies. The results demonstrate high reliability, throughput, and dynamic routing capability achieving high technology readiness level (TRL) for tactical applications.

I. INTRODUCTION

Wireless communication has undoubtedly become a ubiquitous part of our lives and we are constantly striving to meet the evolving needs. This includes all connected devices in our smart homes, our cellular network, the entire concept of internet-of-things (IoT) networks controlling manufacturing, industry automation, smart grid metering, space communications, underwater networks, tactical networks among others. As we move from 5G (5th Generation) to 6G (6th Generation), the need to optimize the scarce resources is becoming evident and inevitable [1]–[5].

Traditionally, the strictly layered architecture proposed by the open systems interconnection (OSI) reference model has been the prevalent design for a majority if not all modern networking architectures. This is strict in the sense that they are designed to maintain only a limited interface between the neighboring layers [6]. Realizing the deficiencies in this layered architecture, cross-layer optimized approach has been proposed over the past decade to maximize the utilization of scarce resources by "erasing" the strict boundaries between various layers of the protocol stack. In other words, any attempt to violate the OSI reference model is considered a cross-layer design [7]. While there are abundant solutions proposed in literature [6], the majority of it is limited to simulations that may have strong assumptions and/or do not consider all the hardware constraints and rigidness that may be encountered during a real-life deployment. During the next phase of advancement, the advent of software defined radios (SDR) provided the much-needed impetus to this concept providing the flexibility to implement novel cross-layer architectures. This enabled some of these efforts to be extended to hardware-based testbed evaluations. In most cases, these efforts still used one or more dedicated (non-embedded) host computers to implement the solutions which were then connected to SDRs. Even with these advances, to the best of our knowledge, there does not exist a ruggedized fieldable SDR with a comprehensive cross-layer optimization capable software module implemented on an embedded ARM processor. The main reason for this is the various hurdles that are associated with developing the solution from theory to effective hardware deployable software.

Contribution: In this article, we present the first, completely stand-alone, ruggedized, and fieldable cross-layer optimized solution build using a low SWaP (Size, Weight, and Power) SDR. In this case, we implement an energy-aware cross-layer protocol that aims to maximize network lifetime for enabling telemetry collection of tactical test and evaluation ranges. The primary contribution of this work is the realization of theoretical or simulation-level concepts to a fieldable hardware entity. We first discuss the hardware-level modification required to customize the baseline SDR into a fieldable solution. Next, we outline how the protocol stack was designed and implemented on a computationally constrained ARM processor. To demonstrate the feasibility of the implementation and the designed transceiver in terms of throughput and reliability, we performed extensive outdoor experiments with up to 10 nodes in the network.

Impact: In this paper, we have demonstrated the feasibility and effectiveness of designing and developing an embedded SDR-based cross-layer optimized solution. The proposed architecture and design principles can be leveraged to implement and mature several of the novel cross-layer optimized solutions to meet the evolving needs of both tactical and commercial communication systems. Our implementation principle has been to facilitate rapid reconfiguration of network objective with only few lines of code thereby enabling a *truly* software-defined radio that evolves with the growing requirements. We hope and believe the unique cross-layer design methodologies and extensive outdoor evaluations (field trials) will serve as an impetus for maturation of novel cross-layer solutions.

II. RELATED WORK

Cross-layer approaches have been explored as a novel solution to address a wide range of problems in wireless communication due to its perceived benefits in sustaining communications in a dynamic and constrained environment [6], [7]. This includes RF domain for tactical network [8]–[10], commercial network [11]–[14], acoustic underwater networks, and even in the upcoming visible light communication networks [15]–[18]. These solutions are designed for various objectives such as optimizing throughput and/or latency [19]–[22], fairness [8], energy consumption [12], [13], [15], resource management [14], [17], efficient multi-path TCP [11]. In this work, we focus on reviewing the maturity of these state-of-the-art cross-layer solutions to *expose the absence of deployable embedded SDR based solutions.*

As in any novel research, simulations are the obvious first choice to establish feasibility and performance gains over existing approaches. Most works rely on MATLAB [10], [20], NS3 [12], OMNET [9] or similar simulators. Majority of the works that propose cross-layer optimization have been limited to simulation [8]–[12], [19]–[21] due to the challenges, time, and effort it takes to evaluate them on hardware testbeds. In most cases, these simulations are executed under various assumptions and/or are abstracted from the physical layer (PHY) of the protocol stack and are restricted to packet-level simulators [6], [10], [20]. This abstraction implies several intricacies of real-world deployment are overlooked or set aside to be handled in the future which often thwarts the maturation level of cross-layer solutions.

In several cases, the next logical step is to evaluate the feasibility and performance on hardware-based testbed. The advent of SDR has significantly bolstered efforts in this direction. At the same time, there is a distinction in the maturity/utility of solutions that have been implemented using a host PC controlling the SDRs and ones efficiently ported to a standalone embedded SDR. The key difference is in the rapidity of development on computationally capable hardware as opposed to carefully optimized (often C/C++ or VHDL/Verilog) implementation on embedded (resource-constrained) hardware. The host PC based development approach saves time and resources to rule out impractical solutions before significant time is spend in optimizing the implementation for final mature deployment. Therefore, based on the resources, the risk associated with novel cross-layer solutions either approaches can be adopted to mature network optimizing solutions. Several of the solutions discussed earlier have been successfully extended to preliminary hardware testbeds [22], [23]. In most of these cases, SDRs like the USRPs are used in association with the host PC. Beyond SDRs, well-defined commercial wireless protocols like WiFi and LoRa (for PHY) have also been used along with microcomputing platforms to design cross-layer approaches [24]. In this case, a cross-layer approach - distributed energy-efficient routing (SEEK) - was implemented on Raspberry Pi to maximize the network lifetime using LoRa as the PHY. In this form, it was highly restricted in throughput

(due to LoRa) as a trade-off for longer transmission range.

To summarize, the majority of the cross-layer optimized work is limited to simulations and has not been successfully validated on a hardware platform. This is a major hurdle and shortcoming of the current state of research and development. In the recent past, some of this has been mitigated by preliminary hardware-based evaluations but has often provided limited performance in terms of metrics like throughput (due to low sampling rate constrained by computations) and/or has been depended on external computational platforms. Due to these reasons, even with the advances made in the field, there is no known cross-layer optimized embedded SDR solution built using commercial-off-the-shelf (COTS) embedded SDR that, (i) can be deployed as a standalone unit, (ii) makes cross-layer optimized distributed routing decisions, (iii) is ruggedized for outdoor deployment, and (iv) can provide reliable and high throughput (up to 11 Mbps) links over large distances (1 km for up to 5.5 Mbps).

III. SYSTEM DESIGN

In this section, we describe the system design of the cross-layer optimized transceiver and how it has been executed.

A. Embedded Software Defined Radio Platform

One of the key objectives of the work was to ensure a modular and programmable, portable, handheld, battery-powered standalone solution which should operate in harsh conditions for several hours. This implied that the foundation of the design needs to be a low SWaP embedded SDR. The proposed solution was implemented on a Epiq Solutions' Sidekiq Z2 SDR [25] (Fig. 1). It consists of an Analog Devices' AD9364, Xilinx Zynq XC7Z010-2I system on chip (SoC), and the key device specifications are provided in Table. I

TABLE I: Specification of Z2

Specs.	Values
Frequency	70 MHz - 6 GHz
Sample Rate	Up to 61.44 MS/s
Size	30 x 51 x 5 mm
Weight	8 grams
Processor	Dual-core ARM

Fig. 1: Z2 SDR

Several hardware customizations were necessary to accomplish the objective of designing a fieldable transceiver using the COTS SDR. This includes power amplifier, filters for the frequency of interest, power supply system that is capable of supplying power from battery during standalone remote operation but could also operate from DC power source when available. To aid the implementation of various cross-layer routing techniques that use location of nodes (such as geographical routing [24]) an embedded GPS receiver was also included in the final design. The block diagram of the final transceiver and the ruggedized prototype is shown in Fig. 2.

B. Cross-layer Software Architecture for Embedded System

The custom cross-layer protocol stack for the transceivers is implemented on the Zynq SoC of the Sidekiq Z2 plat-
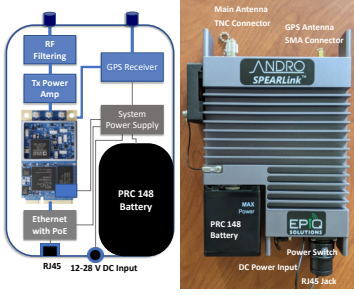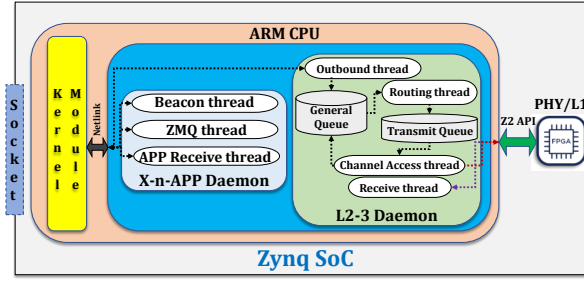
Fig. 2: Transceiver Design
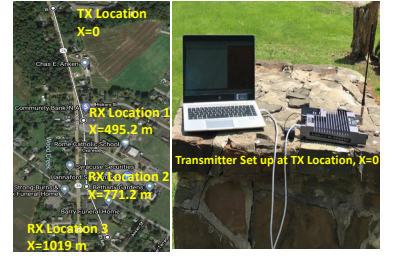

Fig. 3: Software Architecture.

Fig. 4: Range testing setup and map-view of the locations.

form. Specifically, the PHY is an IEEE802.11b implemented purely on the FPGA subsystem while the upper layers are implemented in the C/C++ language on the embedded Linux operating system of the Dual-core ARM Cortex A9 CPU of the SoC. In this section, we will detail the software architecture of the upper layers (above PHY/L1) to establish long-range mesh networking.

Our primary goal in building the software architecture was to maintain *reconfigurability*. The reconfigurability was enforced by adopting a modular design framework with defined functions for each module while being resource-efficient. We define reconfigurability as the ability to modify the protocol characteristics such as routing objective, specifics of the cross-layer information exchanged, etc. Essentially, designing the entire radio stack on the SoC presents a non-trivial challenge owing to the memory, computational, and latency constraints. Since an unorganized framework could add overhead from unnecessary resource utilization consequently increasing the system latency.

The software architecture is broadly categorized into user and kernel space with daemons running in the user space as in Fig. 3. We resort to daemons each of which hosts its own threads to attain a parallelized architecture that does not perform redundant and unnecessary memory accesses. In other words, we can say the user space hosts multi-threaded processes which interface with the kernel space. However, there are certain design challenges with moving key functions to the kernel space. Kernel programming requires the most trusted operations as any bug or corruption may cause severe system crashes. Nevertheless, kernel space enjoys the benefits of low latency memory access operations. Furthermore, we also leverage the transport layer and IP headers supported by the Linux kernel. Figure 3 shows a socket which is a generalized custom socket architecture for interfacing with external devices. The socket architecture is generalized to be able to reprogram and enable interfacing with a wide range of socket protocols such as ZMQ, UDP, etc.

The stack employs two daemons, namely; L2-3 and Cross-layer & Application (X-n-APP) that interact with the kernel module via Netlink sockets. Notice that the entire stack only has two daemons since it's more efficient to use threads over processes as they share the process' resources. On the other hand, having more processes would require more computational and memory resources. The kernel module handles the

transport layer, packet encapsulation, and partial IP header population prior to passing over to the L2-3 daemon. The L2-3 daemon is the cross-layer L1, L2 (MAC), and L3 (Network) module which performs CSMA/CA-based medium access and SEEK routing. Another level of cross-layer interactions occur when the L2-3 daemon acquires L1 information such as the link reliability, data rate, etc., via the Sidekiq Z2's PHY application programming interface (API). Hence, the term cross-layer as it involves interaction between L1, L2, and L3 to perform the optimized decision making. Due to limited space, we do not delve into the details of the cross-layer SEEK routing algorithm that is employed here. We urge interested readers to refer to our previous work [24] for detailed discussion. Therefore, we only define our utility function adopted to perform distributed optimized routing here,

$$\mathcal{U}_{ij} = \eta_{ij} \left(\frac{\max\left[\Delta Q_{ij}, 0\right]}{q_i} \right) \left(\frac{d_{is} - d_{js}}{d_{is}} \right) \left(\frac{E_r^j}{E_0^j} \right), \forall j \in \mathbb{NB}_i$$

where η_{ij} is the number of bits/Joule of transmission energy successfully transmitted from i to j and can be replaced by reliability when using constant power and modulation, E_0^i and E_r^i is initial and residual battery energy respectively, effective distance progressed by i choosing j is represented as $d_{is} - d_{js}$, and ΔQ_{ij} is the differential backlog. To accomplish this objective in a *distributed* manner, each node aims to maximize this value by just collecting information from its local neighbors. This information is exchanged by means of periodic beacon packets. The effectiveness of SEEK has been demonstrated in [24]. The emphasis of this work and contribution is not SEEK but the design and feasibility of maturing similar solutions for tactical applications. *Due to the modular implementation, just by changing a few lines of code that defines the utility function, one can reconfigure the stack to execute a new optimization objective.*

The outbound DATA packets that are handed over to the L2-3 daemon from the kernel module are queued in the *General Queue* awaiting the best route assignment. This is accomplished by the *Outbound* thread which continuously listens for incoming packets from the kernel module. The *SEEK* thread continuously performs route computation and assignment for the outbound packets in the *General Queue*. Following route assignment, the packets will await their transmission opportunity in the *Transmit Queue*. A third thread - *Channel Access* - performs CSMA/CA awaiting a clear-

to-send (CTS) from its intended next-hop/destination to dequeue the segment (set of DATA packets) from the *Transmit Queue*. The segment for which the CTS was received will be forwarded to the FPGA via the L1 API for *over-the-air* transmission. It must be noted that if the intended next-hop for a segment is unresponsive, the segment will be returned back to the *General Queue* for rerouting. Additionally, a *Receive* thread continuously monitors for incoming packets from the FPGA, processes, and responds (such as with CTS or ACK) accordingly based on received packet type. A supplementary auxiliary thread also tracks DATA timeouts.

The X-n-APP daemon handles two tasks; (i) preparing the beacon packet with necessary information to be shared with immediate neighbors for distributed optimization, and (ii) handling messages for any application such as a graphical user interface (GUI) in this case. The beacon packets as well as the GUI requires location information to reflect the most recent coordinates on the GUI. We choose a efficient approach and access the GPS module from only one location - X-n-APP daemon - in the stack to avoid GPS pinging from multiple software locations. This retrieved GPS location information is subsequently populated in the beacon message. The GPS location extraction as well as beacon packet construction is carried out by the *Beacon* thread. The *Beacon* thread periodically sends this beacon messages to the lower layers for outbound transmission where the remaining fields such as residual battery and current buffer backlog are updated. Here, we note that unlike the DATA, the beacon packet is treated as control packet and are not queued in the general or transmit queues of the L2-3 daemon rather it is directly sent to the L1 FPGA for transmission.

The X-n-APP daemon also functions as a generalized application daemon for direct interfacing with an application such as a GUI. Both *ZMQ* and *APP receive* threads are designed to support such applications and hence can be customized based on the application at hand. Since the use case of this particular transceiver is remote deployment in test and evaluation ranges, a GUI application is developed to interface with the transceiver at a central command and control location. We emphasize that the remote monitoring and configuration (parameters like data rates, frequency etc.) capability enabled by the GUI will ease the operator load by alleviating the need to physically travel to the deployed locations. However, we do not elaborate on the GUI design or its implementation aspects as it is beyond the scope of this article rather state that the daemon is capable of handling any such applications.

We further reemphasize that the software-defined stack is designed to be reconfigurable to modify the cross-layer routing as the requirements evolve in the future or depending on the desired network application. It is noteworthy that the utility function discussed above is representative of one such example of the algorithm where the network application desires energy-aware routing. This specific choice was dictated by the customer's requirement. *The broad impact of this work is the reconfigurable nature of the software-defined stack such that it keeps evolving to meet the future requirements rapidly.*

TABLE II: Parameters Of Outdoor evaluation

Parameters	Values
Data rates	1, 2, 5.5, 11 Mbps
Transmit Power	1-3.5 W (based on attenuation)
Center Frequency	430 MHz
Bandwidth	22 MHz
Payload Size	1000 Bytes
Segment Size	32 Packets

IV. OUTDOOR EXPERIMENTS AND RESULTS

In this section, we performed extensive outdoor experiments to evaluate the performance of the implemented software modules in a realistic environment. Since the proposed solution is implemented on SDR, it can be extended to any frequency of interest supported by the target hardware. To accomplish the experiments, we received a temporary experimental license from FCC to utilize the 430 MHz frequency within a geographical area. It is important to point out that the purpose of the experiments is to evaluate the performance of implemented cross-layer optimized algorithms on general purpose processor (GPP) supported by a FPGA-based PHY layer for outdoor deployment. The novelty and utility of the SEEK algorithm itself has been established in previous work [24]. To the best of our knowledge, this is the first time such a comprehensive evaluation has been undertaken using deployment-ready cross-layer optimized embedded SDRs. The default parameters are listed in the Table II unless otherwise specified.

A. Transmission Range Evaluation

The target deployment scenario for the device is remote test and evaluation sites, hence, the design goal for transmission range was to achieve up to 1 km. The setup of the transmitter and the key testing locations from our range testing experiments is shown in Fig. 4. For each reported value in the Table. III, the results were averaged over 10,000 packets transmitted in each run. *We define **reliability** as the percentage of packets received with respect to packets sent.*

TABLE III: Link reliability for varying transmission ranges

Data Rate	Distance	Reliability
2 Mbps	495.2 m	99.9%
	771.2 m	99.77%
	1019 m	98.08%
5.5 Mbps	495.2 m	99.62%
	771.2 m	96.03%
	1019 m	97.16%
11 Mbps	495.2 m	85.28%
	771.2 m	31.56%
	1019 m	13.9%

Table. III shows that the performance was consistent for both 2 Mbps and 5.5 Mbps up to 1019 m but the performance of 11 Mbps degraded sooner (1 Mbps eliminated due to limited space). Due to the FCC license restrictions and urban environment, the next feasible point was at 1942 m from the transmitter at which point no packets were received. The
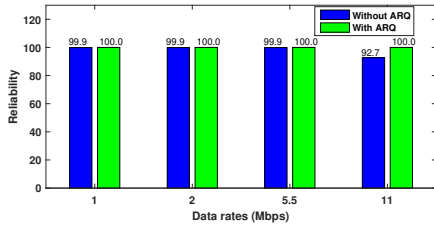
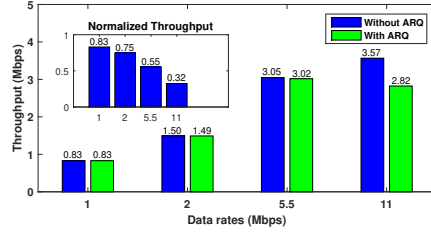Fig. 5: Reliability vs data rates

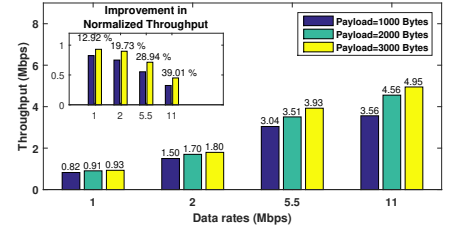Fig. 6: Throughput vs data rates
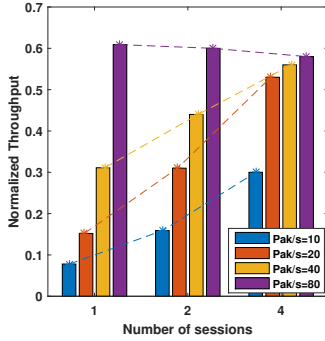

Fig. 7: Throughput for varying payload
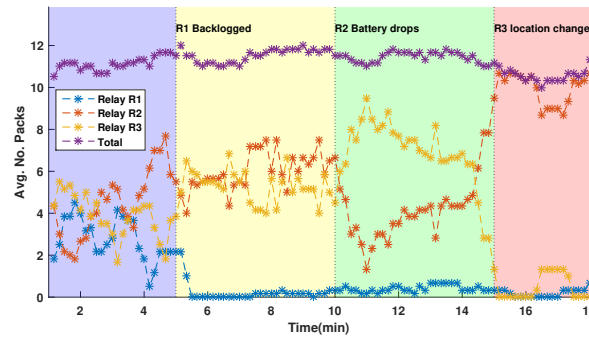

Fig. 8: 10-node Experiment
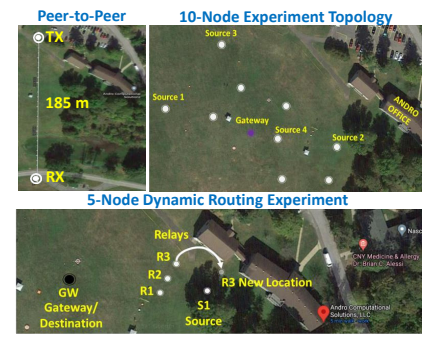

Fig. 9: 5-node Experiment (dynamic routing)

Fig. 10: Outdoor Topologies

performance degradation of 11 Mbps was attributed to sensitivity to lower signal-to-noise-ratio and multi-path propagation effects. Further in-lab cabled analysis revealed there is room for improvement in sensitivity (15 dB) in the future iterations of the FPGA PHY. Overall, the experimentation was successful as we hit our target distance of 1 km even at 5.5 Mbps.

B. Peer-to-Peer Experiments

As described earlier, the decision to implement the PHY on FPGA and the rest of the protocol stack was to achieve the best combination of high data rates and the required flexibility to implementing cross-layer routing protocols. Yet, it is important to study the delays and associated overhead it takes to make the necessary calls between the GPP and FPGA to support packet transmission and how it impacts the data rates. To accomplish this, we perform outdoor peer-to-peer testing for varying data rates and payload sizes to analyze the impact.

First, in Fig. 5, we study the reliability at the different data rates with and without automatic repeat request (ARQ) enabled at the MAC layer. It can be clearly seen that the transceiver has high reliability close to 100% for 1, 2, and 5.5 Mbps even without ARQ enabled. At 11 Mbps the reliability decreases to 92% without ARQ and returns to 100% with ARQ with the trade-off of throughput. Figure 6 depicts how the peer-to-peer throughput varies at different PHY data rates 1, 2, 5.5, 11 Mbps. *The **throughput** in this case only considered the payload of the packets and does not include the headers or the control packets*. The throughput considered here is also referred to as goodput in some literature. The figure also depicts the **normalized throughput** *which is defined as the ratio between throughput and data rate*. The design choice provides solid performance at 1 and 2 Mbps especially considering it depicts the goodput achieved. The impact of the overhead becomes visible as the data rate increases to 5.5 and 11 Mbps. This hypothesis is further substantiated in our experiments that

show an increase in normalized throughput with the increase in payload as shown in Fig. 7. The normalized throughput increased by up to 39% for 11 Mbps when the payload was increased from 1000 Bytes to 3000 Bytes.

C. 10-Node Network Experiments - Network Capacity

In this experiment, we evaluate how the network handles increasing number of sessions and source (packet generation) rate. The 10-node topology is shown in Fig. 10. The experiments were conducted at 1 Mbps of data rate, the source rate was varied from $[10, 20, 40, 80]$ packets/s, and the number of sessions (independent sources generating traffic to the gateway node) were set to 1, 2 and 4. This is motivated by a typical use case for our intended deployment scenario. As expected, Fig. 8 demonstrates that the normalized throughput increases and gets saturated both when the source rate increases and the number of sessions increases. There is no drop in performance even when there are multiple sessions (4) with high source rates (80) and the normalized throughput is maintained at the saturated value (~ 0.6). This shows the efficiency of the network in handling multiple traffics and session rates as expected even when the traffic increases.

D. 5-Node Network Experiment - Dynamic Routing

For timely evaluation, we designed a specific experiment such that we can determine the effectiveness of the implementation of SEEK algorithm to dynamic changes in the network. To accomplish this, we consider a topology shown in Fig. 10. The source *S1* continuously transmits packets destined for the gateway node *GW*. *S1* chooses the appropriate relay among *R1*, *R2*, and *R3* based on SEEK algorithm. We plot the average packets received from each relay node every 10 s interval. The values are averaged using a moving window of 60 s duration to get smoother curves.

In the first part of Fig. 9 (blue shade), in a uniform setting, the traffic is evenly distributed. Next, we increase the backlog of the *R1* to emulate a congestion scenario at a node. As it can be seen (yellow shade), *S1* learns to avoid *R1* and routes the packets through *R2* and *R3*. Next, the residual energy of *R2* is rapidly dropped by removing the DC power supply and the residual energy in the battery was close to 10%. As it can be seen, *S1* recognizes this in the next part (yellow shade) and prefers *R3* so that lifetime of the network can be extended. Finally, we now move the *R3* node behind the source *S1* as shown in Fig. 10 such that it is no more a feasible next hop. In this situation, with *R1* congested, *R3* is no more the possible choice for forward progress, *S1* must return to *R2* with a lower battery since that is the only possible choice to relay the packets to the destination. *It is also interesting to see that even with all these changes, the total packets/s at the GW remains very stable demonstrating the rapid adaptability of the cross-layer optimized nodes.* In this experiment, we have demonstrated the gamut of routing decisions the nodes can make in a distributed manner by just gathering information from its immediate neighbors demonstrating the effectiveness of the cross-layer optimized routing using an intuitively representative experiment.

V. Conclusion and Future work

This article introduces the first known cross-layer optimized transceiver that has been designed, developed, and matured to provide high throughput and reliability by implementing the protocol stack (except PHY) on an embedded ARM processor. We have demonstrated through extensive outdoor experiments the capability, range, and demonstrated dynamic routing under varying network conditions. The modular design allows reconfiguration of the cross-layer protocol stack to rapidly customize components like, information flow, cross-layer interactions, and optimization objective itself by changing a few lines of codes. We hope this successful demonstration of cross-layer optimized embedded transceivers will provide directions for future cross-layer optimized solutions to benefit tactical and commercial applications.

References

[1] M. Katz, M. Matinmikko-Blue, and M. Latva-Aho, "6genesis flagship program: Building the bridges towards 6g-enabled wireless smart society and ecosystem," in *Proc. of IEEE Latin-American Conference on Communications (LATINCOM)*, 2018, pp. 1–9.

[2] Y. Zhao, J. Zhao, W. Zhai, S. Sun, D. Niyato, and K.-Y. Lam, "A survey of 6g wireless communications: Emerging technologies," *ArXiv*, vol. abs/2004.08549, 2020.

[3] A. Jagannath, J. Jagannath, and T. Melodia, "Redefining wireless communication for 6G: Signal processing meets deep learning," *arXiv preprint arXiv:2004.10715*, 2020.

[4] W. Saad, M. Bennis, and M. Chen, "A vision of 6g wireless systems: Applications, trends, technologies, and open research problems," *IEEE Network*, vol. 34, no. 3, pp. 134–142, 2020.

[5] E. Calvanese Strinati, S. Barbarossa, J. L. Gonzalez-Jimenez, D. Ktenas, N. Cassiau, L. Maret, and C. Dehos, "6g: The next frontier: From holographic messaging to artificial intelligence using subterahertz and visible light communication," *IEEE Vehicular Technology Magazine*, vol. 14, no. 3, pp. 42–50, 2019.

[6] B. Fu, Y. Xiao, H. Deng, and H. Zeng, "A survey of cross-layer designs in wireless networks," *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 110–126, 2014.

[7] F. Foukalas, V. Gazis, and N. Alonistioti, "Cross-layer design proposals for wireless mobile networks: A survey and taxonomy," *Commun. Surveys Tuts.*, vol. 10, no. 1, pp. 70–85, Jan 2008.

[8] X. Wang and K. Kar, "Cross-layer rate optimization for proportional fairness in multihop wireless networks with random access," *IEEE Journal on Selected Areas in Comms*, vol. 24, pp. 1548–1559, 2006.

[9] I. Nosheen, S. A. Khan, and F. Khalique, "A mathematical model for cross layer protocol optimizing performance of software-defined radios in tactical networks," *IEEE Access*, vol. 7, pp. 20 520–20 530, 2019.

[10] J. Jagannath, T. Melodia, and A. Drozd, "DRS: Distributed Deadline-Based Joint Routing and Spectrum Allocation for Tactical Ad-hoc Networks," in *Proc. of IEEE Global Communications Conference (GLOBECOM)*, Washington, DC USA, December 2016.

[11] T. Zhu, X. Chen, L. Chen, W. Wang, and G. Wei, "Gclr: Gnn-based cross layer optimization for multipath tcp by routing," *IEEE Access*, vol. 8, pp. 17 060–17 070, 2020.

[12] M. J. Herrmann and G. G. Messier, "Cross-layer lifetime optimization for practical industrial wireless networks: A petroleum refinery case study," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3559–3566, 2018.

[13] J. Jagannath and S. Furman and A. Jagannath and A. Drozd, "Energy Efficient Ad Hoc Networking Devices for Off-the-Grid Public Safety Networks," in *Proc. of IEEE Consumer Communications & Networking Conference (CCNC)*, Las Vegas, NV, USA, January 2019.

[14] S. Barmpounakis, N. Maroulis, M. Papadakis, G. Tsiatsios, D. Soukaras, and N. Alonistioti, "Network slicing - enabled ran management for 5g: Cross layer control based on sdn and sdr," *Computer Networks*, vol. 166, p. 106987, 2020.

[15] Y. Zhou, H. Yang, Y.-H. Hu, and S.-Y. Kung, "Cross-layer network lifetime maximization in underwater wireless sensor networks," *IEEE Systems Journal*, vol. 14, no. 1, pp. 220–231, 2020.

[16] N. Cen, J. Jagannath, S. Moretti, Z. Guan, and T. Melodia, "LANET: Visible-Light Ad Hoc Networks," *Ad Hoc Networks*, vol. 84, 2019.

[17] M. S. Demir and M. Uysal, "A cross-layer design for dynamic resource management of vlc networks," *IEEE Transactions on Communications*, vol. 69, no. 3, pp. 1858–1867, 2021.

[18] J. Jagannath and T. Melodia, "VL-ROUTE: A Cross-Layer Routing Protocol for Visible Light Ad Hoc Network," in *Proc. of IEEE Symp. on a World of Wireless, Mobile, and Multimedia Networks (WoWMoM)*, Washington D.C., USA, June 2019.

[19] L. Ding, K. Gao, T. Melodia, S. Batalama, D. Pados, and J. Matyjas, "All-spectrum Cognitive Networking through Jointly Optimal Distributed Channelization and Routing," *IEEE Transactions on Wireless Communications*, vol. 12, no. 11, pp. 5394–5405, November 2013.

[20] L. Ding, T. Melodia, S. Batalama, J. Matyjas, and M. Medley, "Cross-layer Routing and Dynamic Spectrum Allocation in Cognitive Radio Ad Hoc Networks," *IEEE Transactions on Vehicular Technology*, vol. 59, pp. 1969–1979, May 2010.

[21] C. She, C. Yang, and T. Q. S. Quek, "Cross-layer optimization for ultra-reliable and low-latency radio access networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 1, pp. 127–141, 2018.

[22] J. Jagannath, S. Furman, T. Melodia, and A. Drozd, "Design and experimental evaluation of a cross-layer deadline-based joint routing and spectrum allocation algorithm," *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, pp. 1774–1788, 2019.

[23] G. Sklivanitis, E. Demirors, A. Gannon, S. N. Batalama, D. A. Pados, and S. N. Batalama, "All-spectrum cognitive channelization around narrowband and wideband primary stations," in *Proc. of IEEE Global Communications Conf. (GLOBECOM)*, San Diego, CA, December 2015.

[24] J. Jagannath and S. Furman and A. Jagannath and L. Ling and A. Burger and A. Drozd, "HELPER: Heterogeneous Efficient Low Power Radio for Enabling Ad Hoc Emergency Public Safety Networks," *Ad Hoc Networks (Elsevier)*, vol. 89, pp. 218 – 235, 2019.

[25] Sidekiq-Z2. Epiq Solutions. Accessed July 19, 2021. [Online]. Available: https://epiqsolutions.com/static/Epiq-Solutions-Sidekiq-Z2-8a9337a3ad0d6387032e52c2db623625.pdf