P4-Programmable Raspberry Pi Nodes for Multi-Interface Edge Networking in Networked Airborne Computing

Jagoda Piechocka, Jakub Grzelski, Mariusz Żal
Poznan University of Technology
Institute of Communication and Computer Networks, Poznan Poland
email: (jagoda.piechocka, jakub.grzelski, mariusz.zal)@put.poznan.pl

***Abstract*—Networked Airborne Computing (NAC) systems require flexible networking mechanisms for dynamic packet processing, adaptive routing, and integration of heterogeneous interfaces. This extended abstract assesses the feasibility of low-cost programmable communication nodes for NAC environments using Raspberry Pi platforms and the Programming Protocol-Independent Packet Processors (P4) language. The proposed architecture integrates Ethernet, Wi-Fi, and SPI interfaces, enabling the node to act as both a networking element and an edge gateway. Three P4 compilers, i.e. Behavioral Model version 2 (BMv2), extended Berkeley Packet Filter (eBPF), P4PI are compared together with their control-plane architectures. User-space targets simplify SPI-based telemetry ingestion, while the eBPF backend delivers higher Ethernet/Wi-Fi performance at the cost of greater integration complexity.**

***Keywords*—P4, programmable networking, eBPF, Raspberry Pi, Networked Airborne Computing, UAV communication, edge computing.**

I. Introduction and Motivation

NAC systems rely on cooperating Unmanned Aerial Vehicles (UAVs) forming dynamic, distributed infrastructures for sensing, computing, and communication. Due to node mobility, heterogeneous radio links, and evolving mission requirements, traditional networking stacks with fixed forwarding logic are often insufficient. Programmable data plane technologies, and the P4 language in particular, offer a target-independent way to define parsing, match-action processing, and forwarding behavior directly in the network device. This work investigates the feasibility of realizing such programmability on low-cost Raspberry Pi platforms that combine ARM computing capability with multiple native communication interfaces.

II. Node Architecture

The proposed node runs Linux on a Raspberry Pi and is organized into a *control plane*–responsible for installing forwarding rules and runtime parameters either locally or via external controllers through P4Runtime–and a *data plane* implementing parser, match-action units, traffic manager, and deparser. The node integrates three classes of interfaces: Ethernet for the wired backbone, Wi-Fi for inter-UAV mesh connectivity, and Serial Peripheral Interface (SPI) as a low-latency bridge to onboard sensors, flight controllers, and auxiliary microcontrollers. This multi-interface design allows the node to act imultaneously as a networking element and as an edge gateway for UAV subsystems.

III. P4 Execution Targets on Raspberry Pi

Three representative P4 execution targets for ARM-based Linux platforms are analyzed and compared. BMv2 is the reference user-space software switch–easy to deploy and well suited for prototyping, with limited throughput. The eBPF backend compiles P4 into C/eBPF bytecode loaded into TC/XDP kernel hooks; it offers high packet-processing performance and tight Linux integration through `libbpf`/`bpftool` and eBPF maps. P4PI is a lightweight user-space environment tailored to Raspberry Pi, balancing deployment simplicity with moderate performance for embedded experiments. Control-plane interaction is handled through P4Runtime for BMv2-style targets or directly through eBPF maps for the eBPF backend, with support for runtime updates and, in advanced setups, hot-swapping of data-plane programs.

IV. NAC Communication Scenario

In the considered scenario, each UAV carries a Raspberry Pi node that forms a wireless mesh between airborne platforms, with an optional ground control station acting as a centralized controller. Programmable processing enables adaptive routing, traffic prioritization, application-aware classification, and In-band Network Telemetry (INT). Raw telemetry arriving over SPI can be ingested and directly encapsulated–or embedded as INT metadata–by the P4 pipeline, without heavy involvement of the host Central Processing Unit (CPU). The Wi-Fi and Ethernet interfaces then distribute this data across the mesh or down to the ground station, effectively turning the Raspberry Pi into an application-aware edge gateway.

V. Conclusions

Raspberry Pi platforms combined with the P4 language enable flexible, low-cost programmable communication nodes for experimental NAC testbeds. User-space targets (BMv2, P4PI) simplify integration of non-standard interfaces such as SPI, whereas the eBPF backend delivers higher throughput for Ethernet and Wi-Fi at the cost of greater implementation complexity. Selecting the right execution target therefore depends on the trade-off between wireless link performance and seamless telemetry integration. Future work will focus on experimental evaluation on real UAVs, telemetry-driven adaptive routing, and integration with SDN control frameworks.

Acknowledgment

This research was partially funded by the Polish Ministry of Science and Higher Education (No. 0313/SBAD/1315). This work was partly supported by the NSF/NAWA project IMPRESS-U (NAWA no. BPN/NSF/2023/1/00005, NSF no. 2048266).