Demo: Open source platform for IEEE 802.11p NICs evaluation

Demo: Open source platform for IEEE 802.11p NICs evaluation

Francesco Raviglione
Politecnico di Torino
Torino, Italy
francesco.raviglione@studenti.polito.it

Marco Malinverno
Politecnico di Torino
Torino, Italy
marco.malinverno@polito.it

Claudio Casetti
Politecnico di Torino
Torino, Italy
claudio.casetti@polito.it

***Abstract*—The automotive industry is more and more looking into solutions for connected vehicles, enabling use cases which would be otherwise not realizable. New solutions and protocols are currently being developed both by IEEE and 3GPP; due to its characteristics, IEEE 802.11p will probably be the protocol of choice for all the short range communications between vehicles. As, at the best of our knowledge, very few open source solutions exist, we present an open source platform, based on *PC Engines* embedded boards and Unex DHXA-222 cards, for the testing and characterization of the vehicular amendment of 802.11.**

***Index Terms*—802.11p, Vehicular networks, OpenWrt, Unex DHXA-222, PC Engines APU1D**

I. Introduction

Connected vehicles are becoming more and more a reality in the recent years and several standardization efforts have been put on developing new protocols and solutions, both by IEEE and 3GPP. For DSRC (Dedicated Short-Range Communications), IEEE is considering the introduction of a new amendment, called IEEE 802.11bd [1]. However, before the introduction of this amendment, the protocol of choice for short range communications will likely be legacy IEEE 802.11p, a vehicular amendment based on 802.11a.

This is what drove us to create a Linux based, up-to-date, open source platform for the evaluation of IEEE 802.11p-compatible NICs, providing all the hardware and software elements needed to communicate over 802.11p at a basic level. Being it open source, it is open to any improvement or derivative work and it can be used to experiment new and existing features, sharing the results with the R&D community working on vehicular communications.

II. Platform description

To develop the IEEE 802.11p testing framework, we selected *PC Engines* embedded boards (as shown in Figure 1) which are designed to be a flexible and customizable hardware for networking systems, with the capability of running many Linux distributions and representing an up-to-date solution with good performances when compared to other available embedded boards.

We chose, in particular, the APU1D boards, supporting mini PCIe cards, which are now more and more frequently found on the market with respect to mini PCI ones [2]. Two slots are provided, allowing the installation of any compatible WLAN card and increasing the customizability of this solution.

These boards mount a dual-core AMD G-series T40E x86 CPU with 64 bits support and 2 GB of DDR3-1066 DRAM, including the possibility to install an mSATA SSD as secondary storage.

Due to the availability of an mSATA slot, we selected a SATA III Transcend *MSA370* MLC NAND Flash SSD for each board, with a capacity of 16 GB each, providing enough space even for more memory-demanding applications.

As WLAN cards, we installed a Unex DHXA-222 card on each board, whose main characteristics are summarized below:

- Half-size mini PCI express chips
- Dual band (2.4 and 5 GHz)
- Support for ITS frequencies in the 5.8/5.9 GHz band
- Bluetooth 4.0 support
- MIMO 2x2 operations
- Declared maximum output power: 17 dBm (up to 18 dBm could be selected in software for Wi-Fi, with two attached 5 dBi antennas)

These cards, based on the Atheros *AR9462* chipset, are supported by the *ath9k* Linux driver. Other chipsets supported by this driver have been successfully tested and used in other research works, including the OpenC2X project [3], and for this reason we decided to adopt these cards in our tesbed.

As a software platform, we selected the latest (at the time of this writing) stable version of OpenWrt, i.e., release 18.06.1, with Linux kernel 4.14.63.

The choice of OpenWrt, among all the embedded Linux distributions, was driven by the number of benefits it can provide to a system like the one under study:

- It is specifically designed for networking and embedded devices.
- It supports several architectures, not only the ones based on x86, as the *PC Engines* boards.
- It is currently supported by a wide community, including an official forum and a complete and up-to-date documentation.
- Instead of providing a static framework, it is highly customizable, allowing the user to select only the needed modules and packages. This makes the OpenWrt distribution particularly suitable also for low memory and cheap devices.
- It is up-to-date: the latest version was released few months before the time of writing. Moreover, both the

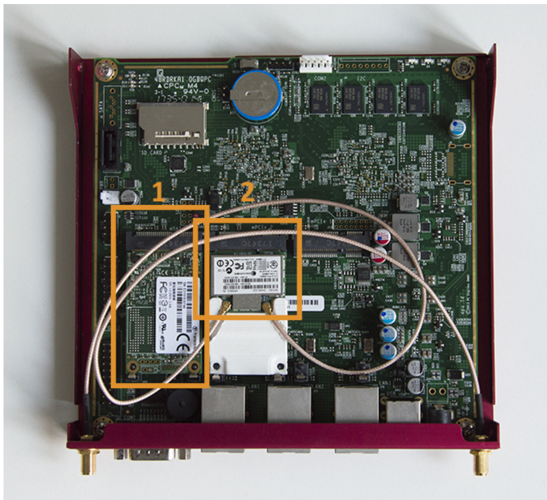

Fig. 1. The picture highlights two additional components of one of the APU1D boards: (1) the Transcend 16GB MLC SSD, mounted on the mSATA slot, and (2) the Unex DHXA-222 WLAN card, mounted on one of the two mini PCIe slot, with two U.FL to RP-SMA pigtail connectors.

Linux kernel and OpenWrt are being constantly updated with bug fixes and new features. This is quite important when working with vehicular network applications, as up-to-date software can introduce new features and patch existing ones, improving performance and possibly already include some important V2X features which already made their way inside the kernel (such as Outside Context of BSS mode, which is now selectable by means of the `iw` utility).

- Last but not least, we were able to identify a number of works, in the literature, based on OpenWrt, such as [4].

Our focus was put on open source solutions, as few of them are available nowadays, at the best of our knowledge. Working with open source solutions also provides two main advantages:

- They have an important role in the vehicular networking research community, as new and experimental features can be coded and then shared, allowing other groups to create their own platform on top of ours and collectively improve existing solutions for vehicular networks.
- The implementation and deployment of new features is much easier, since the source code is available and it can be patched without any major concern about licensing and/or unavailable proprietary functionalities. This also enables researchers to test new features with ease before they possibly reach commercial systems.

Our setup is licensed under the GPLv2 license and available on GitHub [5]. It is described in details in the next sections, focusing on physical, MAC and higher layers.

A. Physical layer

The physical layer is managed in hardware by the Unex WLAN cards. This approach differs from the one taken by SDR solutions implementing physical layer algorithms in software; one such SDR implementation, based on GNU radio, was presented in [6].

In order to enable the usage of the DSRC channels, in the 5.8/5.9 GHz frequency band, the *ath9k* driver needed to be patched in order to let the user select the ITS frequencies and properly support Outside Context of BSS (OCB) mode, together with the 10 MHz-wide channels foreseen by the standard [7].

These patches, including a modification enabling the collection and management of statistics related to the hardware queues, were already provided by the OpenC2X embedded platform, developed by *CCS Labs* in University of Paderborn [3].

As they were developed with kernel 3.18 in mind, they were ported, with minor modifications, to the latest stable OpenWrt version, running on Linux kernel version 4.

In order to prepare the boards for the communication over the 802.11p channels, we also wrote a configuration script, which is run at startup and which sets all the parameters for OCB communications, including an initial value for the physical data rate, for the wireless interface IP address for the transmission power.

B. MAC layer

Since one of the MAC-layer enhancements required by the IEEE 802.11 standard is the EDCA functionality [7], we focused our attention to it.

The Linux *mac80211* wireless subsystem, on which *ath9k* is leveraging, was patched to enable the direct selection of any of the four traffic classes (from AC_BE to AC_VO) by working with a dedicated socket option (`SO_PRIORITY`). This patch was again provided by the OpenC2X platform and it was ported to OpenWrt 18.06.1.

Another important patch we developed is related to the iPerf network measurement tool, which is considered quite a standard when evaluating goodput and packet loss inside networks. Both versions 2 and 3 of this program were lacking the possibility to select a traffic class to send the traffic at, making it impossible to test, with this tool, the EDCA functionalities, which are required in V2X communications [7]. Thus, we introduced a patch enabling the selection of a traffic class thanks to an additional command-line option, taking as argument a string corresponding to the desired priority (BK, BE, VI, VO), which has been integrated inside the modified OpenWrt distribution [5].

C. Higher layers

Concerning higher layers, we validated our platform through several tests, mostly using the UDP protocol over IPv4, which is one of the transport protocols foreseen by the standard and which is well-suited to broadcast communications, which are common in most V2X use cases.

We coded a Linux library to help developers create applications running on top of the operating system, simplifying the usage of raw sockets to send packets over any wireless medium, with any protocol of choice. It is currently supporting

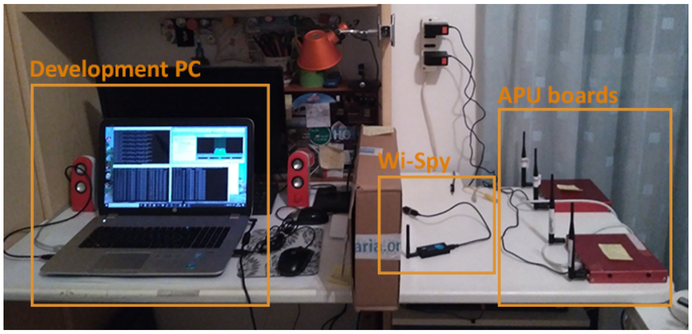

Fig. 2. Concept of the setup which will be presented during the demo, taking as reference one of the workbench layouts we used when validation out setup.

IPv4 and UDP, but it has been built with modularity in mind, in order to facilitate the addition of new protocols, such as WSMP (WAVE Short Message Protocol [8]). WSMP, in particular, could be implemented thanks to raw sockets, without the need of creating an ad-hoc kernel module to be inserted inside the Linux kernel, which may be subject to updates, increasing the overall system complexity.

Based on this library and on the custom protocol LaMP (Latency Measurement Protocol), we also developed, as part of a separate project, an open source flexible Linux latency testing tool (LaTe), which is now in its beta-testing stage, which can be run on the setup described here. This tool is available on GitHub and it has been released with an open source license [9].

III. Demo description

Our platform will be demonstrated following a setup which is similar to the one presented in figure 2.

In particular, two laptops will be available, connected through SSH, with two *PC Engines* APU1D boards, running our setup.

Thanks to the two computers it will be possible to interact with the boards, which will be set to communicate over any of the seven DSRC 10 MHz-wide 802.11p channels. A *Wi-Spy* spectrum analyzer will be available too, showing a proof the spectrum usage, at different power levels and in real-time, to the demo attendees. This demonstration will also showcase a patched version of the Kismet *Spectools* software, providing a graphical interface for the data coming from the spectrum analyzer, introducing few enhanced features, including the possibility to view and select the 5.8/5.9 GHz channels [10].

We plan to show four different demos:

- The first will let the attendees evaluate the latency between the boards, thanks to the LaTe tool [9], logging the collected data to *csv* files, which will be parsed in real-time by MATLAB, displaying a graphical representation of the obtained results to the users.
- The second will involve iPerf, which will be used to test the reachable throughput with a given Access Category (AC) in presence of another interfering traffic flow, generated by another client-server instance of the network measurement tool. The attendees will be able to experience how a higher throughput can be reached when using a higher priority AC, no matter which is the interfering traffic priority.
- The third will run a simple game on both the PCs, making them communicate though the boards, which will transfer game data using the DSRC channels. This will let the attendees experiment how 802.11p communication can be used to provide an online gaming experience to the users.
- The fourth will involve a live camera feed application, through an HD webcam, with all the multimedia data flow passing through the boards, thus effectively connecting the two endpoint by means of an IEEE 802.11p wireless link. This demo will also show how DSRC can be used to transfer multimedia data, which is one of the possible uses cases for vehicular networks.

IV. Conclusions and future work

We developed an open source framework for testing and benchmarking the performance of IEEE 802.11p-compatible cards, allowing user applications to run on top of the operating system and, therefore, to test also more demanding and complex V2X use cases. Through the demo, we plan to show how the selected hardware and software components can work together to provide a reliable IEEE 802.11p basic framework.

Future improvements will include the real deployment on vehicles, in the near future, to perform mobility tests, and the implementation and evaluation of additional features, such as the WSMP protocol, which is still missing.

References

[1] IEEE. Ieee 802 pars & icaids under consideration [online]. http://www.ieee802.org/PARs.shtml.

[2] J. de Jongh, J. van de Sluis, D. Heuven, A. Voronov, and I. Passchier, "Ieee 802.11p [ctu-iig] on pcengines apu1d running voyage," Feb 2016.

[3] S. Laux, G. S. Pannu, S. Schneider, J. Tiemann, F. Klingler, C. Sommer, and F. Dressler, "Demo: Openc2x — an open source experimental and prototyping platform supporting etsi its-g5," in *2016 IEEE Vehicular Networking Conference (VNC)*, Dec 2016, pp. 1–2.

[4] A. Abunei, C. Comşa, and I. Bogdan, "Implementation of a cost-effective v2x hardware and software platform," in *2016 International Conference on Communications (COMM)*, June 2016, pp. 367–370.

[5] Github - francescoraves483/openwrt-v2x [online]. https://github.com/francescoraves483/OpenWrt-V2X.

[6] B. Bloessl, M. Segata, C. Sommer, and F. Dressler, "Performance assessment of ieee 802.11p with an open source sdr-based prototype," *IEEE Transactions on Mobile Computing*, vol. 17, no. 5, pp. 1162–1175, May 2018.

[7] "IEEE standard for information technology–telecommunications [...]," *IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012)*, pp. 1–3534, Dec 2016.

[8] "Ieee standard for wireless access in vehicular environments (wave) – networking services," *IEEE Std 1609.3-2016 (Revision of IEEE Std 1609.3-2010)*, pp. 1–60, April 2016.

[9] Late + lamp home page [online]. https://francescoraves483.github.io/LaMP_LaTe/.

[10] Github - francescoraves483/spectools-dsrc [online]. https://github.com/francescoraves483/spectools-dsrc.