Implementation of a Cost-Effective V2X Hardware and Software Platform

Adrian Abunei[1], Ciprian-Romeo Comșa[1,2], Ion Bogdan[1]
aabunei, ccomsa, bogdani @ etti.tuiasi.ro
[1] Telecommunications Dept., Technical University "Gheorghe Asachi" of Iasi, Romania
[2] Continental Automotive Romania SRL, Iasi, Romania

Abstract — **Vehicular ad-hoc networks (VANET) aim at exchanging traffic safety and/or general purpose data by vehicles with fixed devices and other vehicles. The IEEE 802.11p standard defines the physical and medium access control layers for VANETs. High costs of commercial devices make practical experiments quite expensive. This work proposes a low cost, customizable device that supports different VANET standards for 5.9 GHz and 700 MHz frequency bands.**

Keywords — **VANET, ITS, 802.11p, OpenWRT**

I. Introduction

Intelligent transportation is under fast developing around the world. Intelligence in transportation systems is empowered by exchange of information between vehicles, i.e., vehicle to vehicle (V2V), and between vehicles and infrastructure (V2I), referred together as V2X. This exchange is enabled by creation of such called Vehicular Ad-hoc Networks (VANET). VANETs can use various wireless networking technologies, such as WLAN, Bluetooth, Visible Light Communication (VLC), ZigBee. In addition, cellular technologies like 3G, LTE, or WiMAX IEEE 802.16 can support VANETs. These communication technologies can support a large number of applications, with different requirements in terms of throughput, frequency of transmission, and latency [1]. In particular, V2V safety applications, such as collision warning, have strict latency requirements that can be satisfied by VANETs. Although different approaches have been also recently proposed [2], the most commonly adopted option for V2X communication is the vehicle-specific version of WLAN, standardized as IEEE 802.11p. However, due to independent development, standardization build up onto 802.11p differs in different countries and continents. As such, in US, the IEEE 1609 WAVE (Wireless Access in Vehicular Environments) protocol stack builds on IEEE 802.11p WLAN operating on seven reserved channels in the 5.9 GHz frequency band. It includes also a standardized message format described by IEEE 1609.3 WSMP (Wave Short Message Protocol). In Europe, ETSI ITS-G5 builds on a variant of the same radio technology with some adaptations operating on up to five reserved channels in the 5.9 GHz frequency band. Authorities of Japan allocated two different bands, 5.8 GHz and 700 MHz, to be operated by a variant of the same technology, standardized as ARIB STD-T109.

To cope with the rapid changes of the wireless propagation channel, in 802.11p, the time domain characteristics were doubled compared to IEEE802.11a, leading to a symbol duration of 8 µs and a guard interval of 1.6 µs. At the same time, the frequency domain characteristics have been halved, e.g., leading to a channel width of 10 MHz.

It was shown in [3] that reliability of communication, important for safety applications, in VANETs on 5.9 GHz can be improved by using a using a supplementary lower frequency, e.g., around 700 MHz. That is, in scenarios where critical messages sent within the 5.9 GHz band do not reach the intended destination due to harsh wireless propagation environment, they can be transmitted within the 700 MHz band, increasing the probability to reach the destination [3]. As compared to 5.9 GHz, the 700 MHz band has a better coverage due to smaller path loss and diffraction loss.

There are several manufacturers of 802.11p radios: NEC, Cisco/Cohda Wireless, Commsignia, Denso, Savari, Kapsch, Siemens, Unex, Autotalks [4], Arada, DGE, Componentality. The last one produced FlexRoad [5], an open source platform based on IEEE 802.11p V2X communications, but not supporting all the specifications of 802.11p protocol. In sum, there are few platforms available in the market and generally their price is high. Some of them are developed in custom configurations, bound to proprietary hardware constraints. Open and configurable solutions are needed for research, evaluation studies and experimental prototypes.

In this paper, a cost-effective V2X platform is proposed. The hardware platform is built by cheap components widely available on the market, is customizable, and allows operation on the 5.9 GHz band and supplementary on the 700 MHz band to support the implementation proposed in [3]. Details about the hardware platform are included into Section II. The software platform, described in Section III, is based on OpenWRT [6], a Linux distribution for embedded systems, on which the 802.11p standard was integrated. Then, Results and Conclusions are the subsequent sections of this paper.

II. Hardware Platform

To allow prototyping and evaluation of the concept proposed in [3], a hardware platform for vehicular communication has the following minimum requirements: low cost, easy customization, ability to work as Road Side Unit (RSU) and On Board Unit (OBU), redundancy radio link, robustness in extreme conditions, running open source software.

In order to meet these requirements, we have adopted, as in [7], the Mikrotik RB433UAH hardware platform [8]. This

This work was funded in part by Continental Automotive Romania SRL Iasi

equipment is specifically designed for broadband Internet access through a WiFi wireless service.

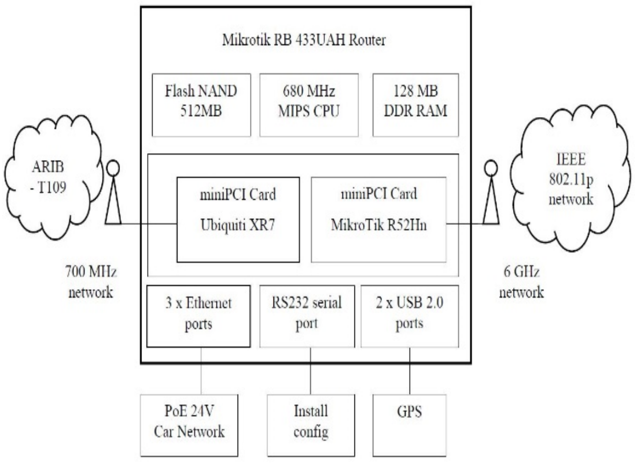

Fig. 1. Hardware components of the V2X platform.

The Mikrotik RouterBoard RB433UAH (that has the block diagram represented in Fig. 1) is equipped with an Atheros AR7161 MIPS 24K processor, 128 Mb RAM, 512 Mb Flash Memory [8]. The power is fed through a 24V PoE adapter. Energy consumption is as low as 10W. The board is equipped with an RS232 serial port, three miniPCI interfaces, three Ethernet interfaces, a micro SD card and two USB 2.0 interfaces. The CAN bus of a vehicle can be connected to the router via the serial port or Ethernet. If needed, other component / wireless interfaces like Bluetooth, WiFi or GPS module could be connected by means of the RouterBoard RB433UAH USB interfaces.

Fig. 2. Mikrotik R52Hn and Ubiquiti XR7 miniPCI cards.

We equipped the unit with two wireless miniPCI cards: Mikrotik R52Hn and Ubiquiti XR7.

Mikrotik R52Hn is an 802.11a/b/g/n wireless adapter with Atheros AR9220 chipset. According to specifications, it supports upper 5 GHz band and it can be used by SW adaption for wireless communications based on 802.11p in the 5.9 GHz band.

Ubiquiti XR7 [9] has an Atheros AR5414 chipset, working on a proprietary 700 MHz band. It can be adapted for vehicular communications based on the Japanese Standard ARIB–T109.

The cost of our experimental set-up without GPS component is about 200 euro. To the best of our knowledge, the lowest market cost for similar equipment is about 800 euro.

Mikrotik RB433UAH platform is running RouterOS, a proprietary router operating system, which was replaced in our application by OpenWRT, a Linux router operating system. This OS is used to implement the 802.11p communication protocol in the 5.9 GHz band for the first card and in the 700 MHz band for the second card, both of them with 10 MHz channel width.

In order to operate in accordance with the 802.11p standard specifications, it is required to change the drivers and software kernel of OpenWRT for the first card and to modify the frequency band for the second card.

According to the manufacturer's technical specifications, Ubiquiti XR7 works in the 748 - 807 MHz frequency band. The transceiver is based on the Atheros AR5414 single-chip processor and includes an Analog Devices voltage-controlled oscillator (VCO) and mixer. The Ubiquiti XR7 card "looks" like a standard 802.11g device in software, but the hardware is configured to provide a frequency offset of 1664 MHz. By using the Linux *ath5k* driver, channels 1-13 from the 2.4 GHz band are converted to the 748-807 MHz band with 5MHz channel bandwidth. However, due to filtering and hardware limitations only the 4 central channels support a maximum transfer capacity up to 24.59 Mbps as they could use the OFDM 64QAM modulation and ¾ coding rate.

The ADF4360-4 is a fully integrated integer-N synthesizer and VCO. Control of on-chip registers is done through a simple 3-wire interface. The design of the ADF chip uses a MCU PIC12F629 to generate a control signal. The output frequency of VCO is mixed with AR5414 modulated signals and the lower sideband of the mixer is selected by means of a bandpass filter. An integrated linear amplifier rises up the output 700 MHz signal power to 28 dBm. ARIB T-109 standard requires a single channel of 10 MHz bandwidth with the central frequency of 760 MHz. The program registries are set through microcontroller such that the ADF 4360-4 generates an output frequency of 1672 MHz (corresponding to channel no. 5 in 2.4 GHz band).

III. Software Setup

Linux was chosen due to its reliability, easy configuration and accessible open source system. It contains a multitude of network tools that can be adapted to user requirements.

OpenWRT is a Linux distribution for embedded devices mainly oriented towards wireless routers, but may be extended to other platforms. OpenWRT software is not intended to be a dedicated distribution to be loaded directly into an embedded device. Instead, a development framework creates a tailored firmware to suit particular needs. This means that one can change any step in the process, simply by changing the templates. There is an automated system for downloading the sources, patching them to work on the designated platform and compiling them correctly for that platform. Almost any standard non-GUI Linux/UNIX program can be compiled for use in OpenWRT, in order to fit in the smaller memory module of the target device. The cross-compilation and packaging techniques are available in the literature [10]. Both the system and the third-party package libraries contain a growing list of standard Linux tools and device drivers. The installation of OpenWRT consists in selecting and compiling the necessary packages. Compilation process is done under host development system. We chose Ubuntu 14.04LTS distribution for cross-compiling to create OpenWRT binary code for RB433UAH. We used the latest stable version of OpenWRT (Chaos Calmer 15.05), compatible with Mikrotik RB433UAH.

Atheros chipsets are found on many manufacturers cards and are used by the open-source community due to their extensibility. They directly call hardware functions and write to hardware registers. Other chipsets from Intel or Broadcom get recognized by the Linux kernel, however, these drivers do not allow the same level of configuration as the Atheros Linux drivers do. Among them, the most common for Atheros 802.11a/b/g PCI/PCI-E chips are *ath5k* and *ath9k*, the former supporting also 802.11n chips.

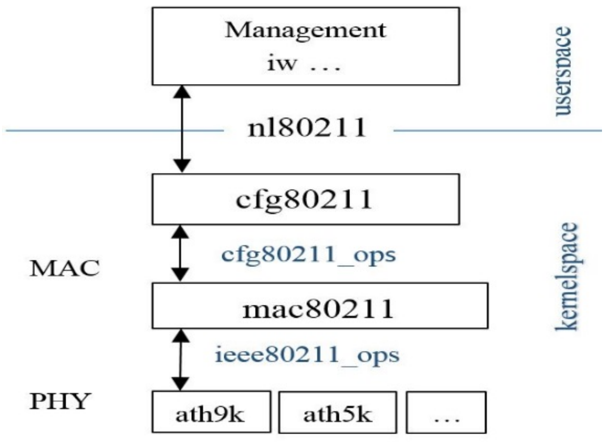

Fig. 3. Linux open-source architecture of the 802.11p stack.

The Linux kernel wireless subsystem, as illustrated in Fig. 3, contains among others two major blocks: *cfg80211* and *mac80211*. They interface the *ath* drivers to the user space. The *cfg80211* is a configuration interface bridging user space and drivers. The *mac80211* is a framework for writing softMAC drivers for wireless devices. The *nl80211* is the netlink configuration interface for user space software. The *iw* is a userspace command-line utility for configuration of wireless devices.

The 802.11p amendment, part of IEEE802.11-2012 standard, specifies extensions for MAC and PHY layer for wireless communications in vehicular environment [11].

The Atheros device driver is adapted on PHY layer to allow the card to operate in the 5.9 GHz band on 10 MHz channels. In addition, the guard interval is doubled as compared to 802.11a. To operate on the channels from 5850 MHz to 5925 MHz, it is necessary to perform modifications on the wireless registration files of OpenWRT.

A new mode OCB (Outside the Context of a BSS) is introduced on MAC Layer, allowing wireless stations (STA) to communicate without being associated to an access point or to each other. OCB mode main properties are: wildcard BSSID (BSSID where all bits are set to 1) is used by each STA; no beacons are sent or received; no association, authentication or encryption are used; and a new flag *dot11OCBActivated* is introduced. When *dot11OCBActivated* is true, a STA is not a member of a BSS (Basic Service Set) and it does not utilize the IEEE 802.11 authentication, association, or data confidentiality services.

We ported to OpenWRT the IEEE 802.11p Linux Kernel implementation which is described in [12]. The OCB mode is implemented in the Linux kernel since version 3.19, but the latest stable version of OpenWRT for Mikrotik is based on kernel 3.18.23. In addition, *iw* wireless configuration utility needs to be recompiled according to the latest version of the kernel. All these changes in the original OpenWRT packages are achieved through *patches* [13]. These files contain specific changes of C++ source software packages. Two of the patches we applied modify the *mac80211* package of OpenWRT that includes also Atheros drivers. The C++ source files belong to *compat-wireless* package of Chaos Calmer OpenWRT.

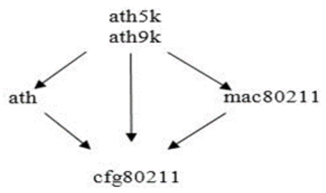

Fig. 4. Diagram dependencies among modules.

Compat-wireless includes the development and stable sources for the kernel wireless drivers such as the *ath* drivers and *mac80211* wireless stack. The *ath* folder, as illustrated in Fig. 4, contains the configuration files for all Atheros-based drivers.

The first patch adds the OCB mode (OCB join and leave handling on *nl80211*, *cfg80211*, *mac80211*), modifies the *ath9k* driver to support OCB mode and enables the operation in 5.9 GHz band for 802.11p and ITS-G5. The second patch modifies the *ath5k* driver files for Ubiquiti XR7 card to support OCB mode and the operation in 2.4 GHz band on 10 MHz channels.

Then, we created another patch file for modifying the *iw* wireless network utility package. It contains new commands for joining or leaving OCB mode in 10 MHz channel. Next, we introduced by patching the kernel of OpenWRT a new module for decoding WSMP messages. For this, we create one C++ source file and a configuration file for compiling.

Before applying the patches prepared above, we need to modify the regulatory domain file to include the 5.9 GHz band.

The selection of the packages is made through command *make menuconfig*, so that both the system and the kernel could support features as IPv6, routing and transport protocols. In addition, it is necessary to choose the type of architecture (ar71xx) and the types of images to be generated.

All these patches described above are applied to the original packages in the cross compiling process on UBUNTU host system. Three files are generated. Their installation process on the Mikrotik platform is described in [14]. We installed OpenWRT on two identical Mikrotik platforms, equipped with the two types of miniPCI cards, Mikrotik R52Hn and Ubiquiti XR7 presented in section II.

IV. RESULTS

The configuration of the two platforms is done through serial terminal program via RS232 port. Table I lists the command lines to set the two platforms in V2X communications.

Next, we set up the two OpenWRT Mikrotik RouterBoards to communicate via IP (layer 3 packets), bridging the interfaces *wlan0* and *eth1* for each platform. We create a virtual interface for *wlan0* to monitor and capture wireless traffic with *ping* and *tcpdump* software programs. The corresponding command lines are as follows:

```
# Add virtual radio monitoring interface on wlan0
iw wlan0 interface add mon0 type monitor flags none
ip link set mon0 up
tcpdump –I mon0 –v
```

Table I

Input command	***Description***
iw reg set RO	Set the proper regdomain
Output	
cfg80211: Calling CRDA for country: RO *cfg80211: Regulatory domain changed to country: RO* *cfg80211: DFS Master region: ETSIcfg80211: (5850000 KHz - 5925000 KHz @ 20000 KHz), (N/A, 2000 mBm), (N/A)*	
Input command	***Description***
ip link set wlan0 down ***iw wlan0 set type ocb*** ***ip link set wlan0 up***	Configure a wireless interface to the OCB Mode
Output	
goto... locla->ocbs++ *### __ieee80211_recalc_idle: active: 1; local->ocbs: 1* *netif carrier ok!* *br-lan: port 2(wlan0) entered forwarding state*	
Input command	***Description***
iw wlan0 wlan0 ocb join 5890 10 MHZ	Connect OCB network to a particular frequency with 10 MHz wide channel
Output	
### __ieee80211_recalc_idle: active: 1; local->ocbs: 1	
Input command	***Description***
iw wlan0 info	Wireless interface status
Output	
Interface wlan0 *addr d4:ca:6d:14:c7:d7* *type outside context of a BSS* *channel 178 (5890 MHz), width: 10 MHZ, center1: 5890 MHz*	

Fig. 5 shows a snapshot of *tcpdump* that indicates the packets transfer over *wlan0* interface. The bitrate is 12Mb/s (adjusted according to received signal strength of – 69dB), on 5890 MHz channel with 10 MHz bandwidth, in agreement with the 802.11p specifications.

According to [1], the active road safety requirements for use cases related to collision warning and vehicle overtaking do not allow a communication latency higher than 100 ms. We checked the latency for the case when a file is transferred between devices and we noticed that the results are much less than the upper bound of 100 ms, as shown within the next lines:

```
PING 192.168.1.1 (192.168.1.1): 56 data bytes:
64 bytes from 192.168.1.1: seq=0 ttl=64 time=0.540 ms
64 bytes from 192.168.1.1: seq=1 ttl=64 time=0.281 ms
64 bytes from 192.168.1.1: seq=2 ttl=64 time=0.379 ms
```

The Frequency band generated by our platform was checked for the 5.890 GHz with 10 MHz bandwidth by means of a spectrum analyzer. As such, Fig. 6 shows the measured spectrum of the generated signal, in accordance with the 802.11p specifications.

```
tcpdump: listening on mon0, link-type IEEE802_11_RADIO (802.11
plus radiotap header), capture size 65535 bytes
192.168.1.154 > fra07s64-in-f24.1e100.net: ICMP echo request, id
23879, seq 240, length 64
18:25:32.067834 452965777us tsft 12.0 Mb/s 5890 MHz 11a/10Mhz
-69dB signal [bit 29] 0us Acknowledgment RA:d4:ca:6d:14:c7:d7
(oui Unknown)
18:25:32.067903 12.0 Mb/s [bit 15] 84us IP (tos 0x80, ttl 53, id
19581, offset 0, flags [none], proto ICMP (1), length 84)
fra07s64-in-f24.1e100.net > 192.168.1.154: ICMP echo reply, id
23879, seq 240, length 64
```

Fig. 5. Capture of the network traffic on data transfer

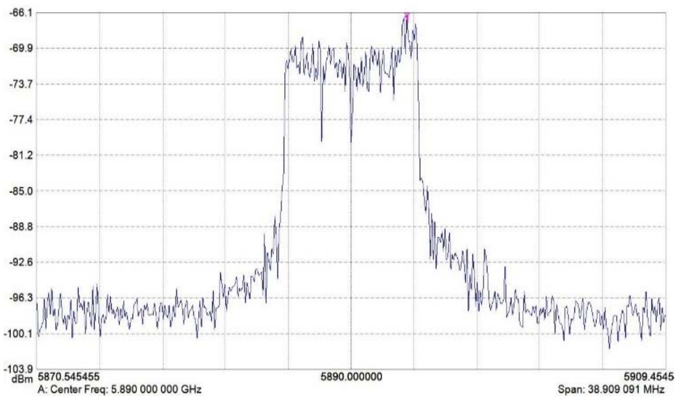

Fig. 6. Spectrum analyzer data of the generated signal.

V. Conclusion and Future Work

This paper describes a V2X low cost hardware and software platform based on the Mikrotik RB433UAH RouterBoard. It operates by default on 5.9 GHz band and it switch on 760 MHz band when communication on 5.9 GHz fails.

We intend to use this platform to demonstrate the solution we proposed in [3]. Also we plan to adapt our platform solution for other development boards, working on Atheros based PCI-E or USB cards in order to elaborate vehicular applications on different interfaces and low cost hardware platforms.

VI. References

[1] G. Karagiannis, O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin*, et al.*, "Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions," *Communications Surveys & Tutorials, IEEE,* vol. 13, pp. 584-616, 2011.

[2] S. Mumtaz, S. Huq, K. Mohammed, M. I. Ashraf, J. Rodriguez, V. Monteiro*, et al.*, "Cognitive vehicular communication for 5G," *Communications Magazine, IEEE,* vol. 53, pp. 109-117, 2015.

[3] A. Abunei, C.-R. Comsa, and I. Bogdan, "RSS improvement in VANETs by auxilliary transmission at 700 MHz," in *Signals, Circuits and Systems (ISSCS), 2015 International Symposium on*, 2015, pp. 1-4.

[4] L. Andia, M. Carlsson, and A. Freund, "A reconfigurable IEEE 802.11 p/ARIB RF transceiver for V2X," 2014.

[5] Componentality, *[Online]. Available: http://componentality.com/en/ flexroad/ [Accessed: Feb. 10, 2016].*

[6] OpenWRT, *[Online]. Available: https://www.openwrt.org [Accessed: Feb, 20, 2016].*

[7] N. Agafonovs, G. Strazdins, and M. Greitans, "Accessible, customizable, high-performance ieee 802.11 p vehicular communication solution," in *Ad Hoc Networking Workshop (Med-Hoc-Net), 2012 The 11th Annual Mediterranean*, 2012, pp. 127-132.

[8] Mikrotik, *[Online]. Available: http://routerboard.com [Accessed: Feb. 20, 2016].*

[9] "Ubiquiti XtremeRange7," *[Online], Available: https://dl.ubnt.com/xr7_datasheet.pdf, [Accessed: Feb, 20, 2016].*

[10] F. Fainelli, "The OpenWrt embedded development framework," in *Proceedings of the Free and Open Source Software Developers European Meeting*, 2008.

[11] I. S. Association, "802.11-2012-IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," 2012.

[12] R. Lisový, M. Sojka, and Z. Hanzálek, "IEEE 802.11 p Linux Kernel Implementation," 2014.

[13] OpenWRT, "Working with patches," *[Online]. Available: https://wiki.openwrt.org/doc/devel/patches [Accessed: Feb, 20, 2016].*

[14] "Installing OpenWRT on a Mikrotik Routerboard RB433," *[Online]. Available : https://wiki.openwrt.org/toh/mikrotik/rb433 [Accessed: Feb. 20, 2016].*