Prototyping Visible Light Communication for the Internet of Things Using OpenVLC

Borja Genoves Guzman, *Member, IEEE,* Muhammad Sarmad Mir, Dayrene Frometa Fonseca, Ander Galisteo, Qing Wang, *Senior Member, IEEE,* and Domenico Giustiniano, *Senior Member, IEEE*

***Abstract*—Visible Light Communication (VLC) has emerged in the last few years as a promising technology not only for high-speed communication but also for serving a new generation of Internet of Things (IoT) devices that may leverage the pervasive lighting infrastructures. Integrating VLC in lighting environments for IoT requires the design of networked and intelligent luminaries and new IoT devices, encompassing the development of innovative technologies and new algorithms. A common experimental platform is necessary to lower the entrance barriers of VLC and speed up the research development. In this paper, we provide guidelines for prototyping VLC for IoT applications, assisted by the open-source platform OpenVLC. We also introduce the new development on OpenVLC, which guarantees support for more powerful LEDs and much longer distance (extending the communication distance from 6 m to 19 m), dimming adaption, among other features. Its low-cost, open-source, and open-hardware designs allow researchers in the community to swiftly adapt it to suit their research purposes.**

***Index Terms*—Internet of Things (IoT), Open Hardware, Open Source, OpenVLC, Prototyping, Visible Light Communication (VLC).**

I. Introduction

With 5G communications leveraging Radio-Frequency (RF) bands currently rolled-out worldwide, the research effort on 6G technologies has already started. The research community is looking for technologies based on new frequency bands that could enable new verticals and attract new stakeholders. Visible Light Communication (VLC) is one of the technologies anticipated to be involved in 6G. As shown in Fig. 1, VLC has attracted a tremendous attention from both academia and industry [1], specially since the TED talk demonstration in 2011 titled “Wireless data from every light bulb”.

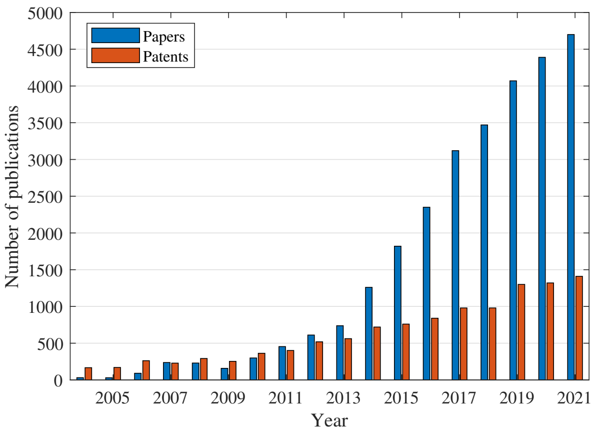


Fig. 1: A consistent increase in the number of published VLC papers and patents (source: Google Scholar).

Compared to RF-based communications, VLC has several advantages: i) large frequency bands, ii) no interference with RF, iii) higher spatial multiplexing and security due to light propagation control, iv) energy efficiency for dual communication and illumination functionality. VLC is being standardized in both IEEE (e.g., 802.15.7, 802.15.13, 802.11bb) and ITU-T (G-9991).

Typically, the front-end devices used in a VLC system are a Light-Emitting Diode (LED) as transmitter of the electrical signal, and a Photodiode (PD) as receiver, which converts the optical received power into an electrical one. LED based solid-state lighting is massively replacing traditional white lighting emitters, already creating an infrastructure for VLC. Retrofitting the LED technology for communication can unleash the full potential of ubiquitous Internet of Things (IoT), going beyond 5G networks. IoT applications typically require data rate from a few kb/s up to hundred kb/s [2]. LED technology is entering the IoT market, enabling light bulbs as Access Points (APs) for communication, or manufacturing low-energy devices with embedded sensory functionalities.

This magazine paper targets the wireless communication research community, communication engineers, and practitioners who traditionally worked in RF technology and need an open-source and low-cost VLC platform for shortening the learning period and exploring the opportunities of experimental VLC. There is a gap between the current available VLC platforms and the demand for entering into the experimental VLC research mainly targeting IoT applications. To fill this gap, *in this paper we present how to prototype a VLC testbed for IoT developments.* We mainly refer to the developed

This work has been partially funded by European Union’s Horizon 2020 Marie Sklodowska Curie grant ENLIGHT’EM (814215), in part by the project RISC-6G, reference TSI-063000-2021-59, granted by the Ministry of Economic Affairs and Digital Transformation and the European Union-NextGenerationEU through the UNICO-5G R&D Program of the Spanish Recovery, Transformation and Resilience Plan, and in part by the grant FJC2019-039541-I (MCIN/AEI/10.13039/501100011033).

Borja Genoves Guzman, Dayrene Frometa and Domenico Giustiniano are with IMDEA Networks Institute. Muhammad Sarmad Mir and Dayrene Frometa are also with the University Carlos III of Madrid, Leganés (Madrid), 28918, Spain (e-mail: {borja.genoves, dayrene.frometa, domenico.giustiniano}@imdea.org, sarmadmir2003@gmail.com)

Ander Galisteo is with Dojo Five, Minnesota, USA. He contributed while being a PhD student at IMDEA Networks (e-mail: ander.galisteo@imdea.org)

Qing Wang is with the Department of Software Technology of Delft University of Technology, 2628 XE, The Netherlands (e-mail: qing.wang@tudelft.nl)

TABLE I: A summary of state-of-the-art VLC platforms, implementations and here presented OpenVLC 1.4.

Name	Optical TX antenna	Optical RX antenna	Processor	Data rate	Distance	Open source	TCP/IP support	Commercial product
OpenVLC 1.4	XHP35A	SFH206K	BBB	1 Mb/s (400 kb/s)	19 m	Yes	Yes	No
Shine [3]	HLMP-CM1A-450DD	SFH203P	Arduino	1 kb/s	1 m	Yes	No	No
EnLighting [4]	SoC bulb	SoC PDs	Arduino	400 b/s	5 m	No	Yes	No
SynLight [5]	LUXEON 3014	SD3421	MSP430F2618	60 kb/s	70 cm	No	No	No
SmartVLC [6]	Philips 4.7W LED	SFH206K	BBB	100 kb/s	3.6 m	Yes	No	No
DenseVLC [7]	CREE XT-E	S5971	BBB	33.9 kb/s	4 m	Yes	Yes	No
Purple VLC [8]	5mm LEDs	SFH213	BBB	100 kb/s	6 m	Yes	No	No
modBulb [9]	unknown	SLD70BG	AGLN250	20 Mb/s	unknown	Yes	No	No
DarkLight [10]	Cree CXA 2520	SD5421	Xilinx Artix-7	1.6 kb/s	1.3 m	No	No	No
LiFi-XC	unknown	unknown	unknown	43 Mb/s	unknown	No	Yes	Yes (PureLiFi)
Trulifi 6002	unknown	unknown	unknown	150 Mb/s	2.8 m	No	Yes	Yes (Signify)
Hyperion	186 Lumen power LED	unknown	LimeSDR	unknown	20 m	No	No	Yes (Hyperion Tech)
LiFiMax	unknown	unknown	unknown	100 Mb/s	unknown	No	Yes	Yes (Oledcomm)
LinkRay	Panasonic displays and LED signboards	Mobile phone camera (app required)	unknown	Few kb/s	unknown	No	Yes	Yes (Panasonic)
MetaLiFi-1P	Metachip (MiniLED)	S13954-01CT	unknown	30-230 Mb/s	3 to 25 m	No	Yes	Yes (HCCL Tech)

OpenVLC platform (www.openvlc.org), a low-cost, full-stack, and open-source (both software and hardware) platform for VLC research and prototyping. OpenVLC perfectly fulfills all the requirements for providing VLC-based IoT services, demonstrating to be not only a good starting point for experimental VLC research lowering the entrance barriers, but also a proper platform for performing cutting-edge research. After more than eight years of developments, OpenVLC is a renowned and stable platform used by many researchers and practitioners worldwide. In 2020, hardware and software have been fully released and the number of messages in the OpenVLC Google group has duplicated, which indicates the interest for accessing the platform. The messages received have been used to improve and consolidate the guidelines and materials provided to ease the learning curve.

II. State-of-the-Art VLC Platforms

This section presents the VLC platforms available in the community. We classify them into three categories: low-end platforms, high-end platforms and commercial products. They are summarized in Table I.

A. Low-end VLC platforms

Existing research-oriented VLC platforms can be classified based on the computing devices on which they are developed:
1) Arduino/MCU-based: Shine [3] and EnLighting [4] platforms achieve rates of 1 kb/s and 400 b/s at maximum distances of 1 m and 5 m, respectively. SynLight [5] can achieve a data rate of 60 kb/s at 70 cm by using the pulse amplitude modulation scheme.
2) BeagleBone-based: Apart from OpenVLC, introduced in details later in this paper, a few more platforms are based on Beaglebone. SmartVLC [6] focuses on transmitting at dimming mode and achieves around 100 kb/s at a distance of 3.6 m. DenseVLC [7] and Purple VLC [8] are based on OpenVLC, achieving a data rate (distance) of 33.9 kb/s (4 m) and 100 kb/s (6 m), respectively.

B. High-end VLC platforms

High-end VLC platforms use advanced and programmable hardware:
1) FPGA-based platforms: modBulb is a modular platform that can work with FPGA using a clock frequency of up to 20 MHz [9]. Besides, DarkLight employs an FPGA with a clock speed of 100 MHz [10]. It transmits ultra-short pulses to provide dimming.
2) USRP-based platforms: Although these advanced devices increase the cost of the platform, some testbeds have been developed based on them, usually changing the default daughterboards to work with baseband signals [11].

C. Commercial platforms

There are also several commercial products such as LiFi-XC from PureLiFi, Trulifi from Signify, Hyperion from Hyperion Tech, LiFiMax from Oledcomm, LinkRay from Panasonic, and MetaLiFi-1P from HCCL Tech. These products can achieve from tens to hundreds of Mb/s, except for LinkRay that is oriented to advertising (low data rate demand). Other commercial products are LiPHY devices from LiPHY company, and YellowDot from Philips, which are oriented to indoor positioning and location-based content delivery applications. However, these products are not open-source, their implementation details are not available, and they are not targeting IoT applications, which generally require modest data rates.

D. OpenVLC

Against this background, this paper presents OpenVLC, a pioneering platform that was developed starting from late 2013 and it has evolved until now. Fig. 2 shows the samples of different versions, from the pilot to the latest OpenVLC 1.4, and Table II synthesizes their main modifications. Note that the form factor has been decreasing, being more attractive to users. The most recent version slightly increases the size because it achieves a larger transmission range, such that it can be used also for industrial scenarios. Here we present the main prototyping guidelines using the most recent OpenVLC 1.4 version introduced in this paper.

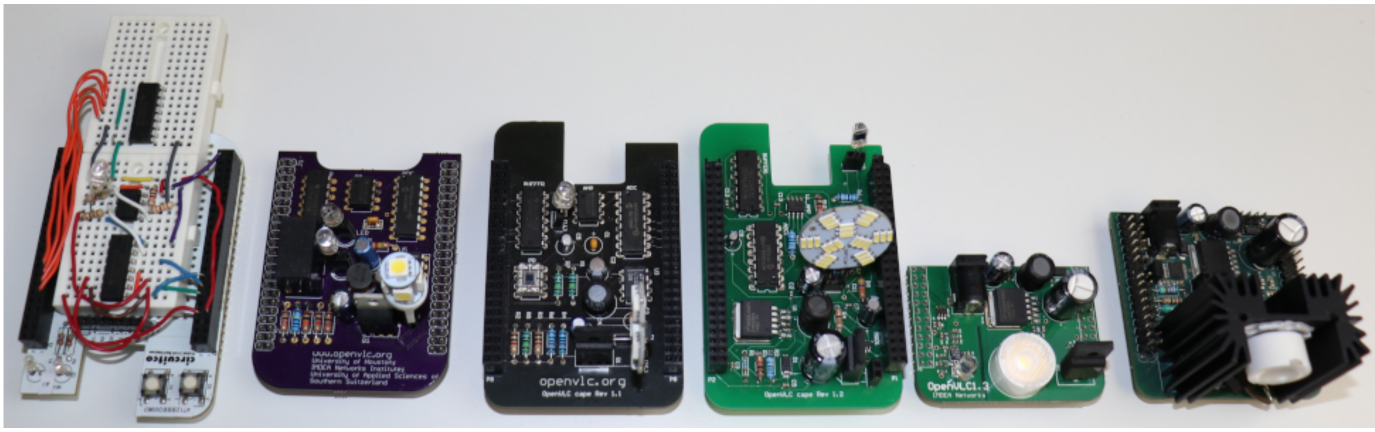


Fig. 2: Capes for the different versions of OpenVLC (from left to right: pilot, 1.0, 1.1 [12], 1.2 [13], 1.3 [14] and 1.4).

TABLE II: Key developments for OpenVLC versions.

	OpenVLC 1.0/1.1 [12]	**OpenVLC 1.2 [13]**	**OpenVLC 1.3 [14]**	**OpenVLC 1.4**
Software (Kernel)	• Whole OpenVLC software code in kernel	• Kernel used for MAC tasks	• No change w.r.t. OpenVLC 1.2	• No change w.r.t. OpenVLC 1.3
Firmware (PRUs)	• No PRU used (real-time problem)	• PRUs used (real-time problem solved) • Faster reception (1 sample/symbol: bit slip problem) • New frame detection (real time)	• Optimization in number of operations • Faster reception (2 samples/symbol: bit slip problem solved) • New frame and symbol detection (real time)	• Modulation with dimming support
Transmitter Hardware	• Two possible LEDs (high and low power)	• LED changed with more powerful and faster	• LED changed by a more powerful 6-9W • External power. • 3-level optical power	• Current-driven design (LED at higher current)
Receiver Hardware	• Low power LED and PD	• Only PD as receiver • External amplifier	• Filters to remove interference and DC • Cape size reduced (pins released for IoT applications)	• No change w.r.t. OpenVLC 1.3
Limitations	• High jitter (no firmware)	• Large form factor compromising BBB GPIOs. • Tx-Rx desynchronization • LED ON only when transmitting	• Temperature issues when working at larger power	• Data rate limited by maximum BBB sampling rate (2.1MHz)

OpenVLC is built around the low-cost BeagleBone Black (BBB) platform, which is a small-sized embedded board. The most recent version of OpenVLC (v1.4) is mainly composed of four parts: *software in the kernel, firmware, receiver hardware* and *transmitter hardware,* as detailed in Fig. 3. The software in the kernel, also called OpenVLC driver, is a module installed in the Linux kernel that is programmed with C language. The firmware is in charge of time-sensitive tasks, implemented in assembly and installed in the Programmable Real-time Units (PRUs) of the BBB, which are fast processors that can run at 200 MHz. PRUs have single-cycle I/O access to several pins and access to a portion of the internal memory and peripherals, thus they are appropriate for managing OpenVLC hardware. Finally, the hardware is the transceiver assembled on a cape that is attached directly to the General Purpose Inputs/Outputs (GPIOs) of the BBB. The open-software and open-hardware models of OpenVLC are at the service of the research community and users, accessible through www.openvlc.org. and www.github.com/openvlc/OpenVLC/.

III. VLC Prototyping: Hardware

A. Fundamentals

A VLC generic transceiver is composed of a transmitter that converts the digital electrical data into an analog light signal, and a receiver that converts the incident light into a readable electrical signal.

Transmitter. A typical VLC transmitter contains a data source, a modulation scheme followed by the LED driver and the LED. The modulation scheme is used to map bits into symbols that result in a real and positive electrical signal. Next, the signal is fed into the LED driver which is in charge of accommodating the electrical signal to the transfer function of the LED. The output signal of the LED is then transmitted through the optical channel. The dual functionality of VLC to provide both illumination and communication places strict requirements on LED operation. For that reason, LED drivers are used in VLC to precisely supply the operating current of LED in order to control the illuminance, LED performance and data modulation of the VLC transmitter.

Receiver. The basic VLC receiver design consists of a PD with output current proportional to the input optical signal, a Transimpedance Amplifier (TIA) that gives an amplified voltage signal as output, filters, an ADC and a demodulator. The optical signal is typically received by a PD because of offering a linear response, low dark current, low resistance and very good spectral response, though Avalanche PDs (APDs) are also very much used for higher sensitivity at the expense of a larger power consumption.

B. OpenVLC hardware

OpenVLC is designed as a plug-in board called cape that can fit on the top of a BBB embedded board making it an easy-to-use platform. The real-time processing capability of the two BBB's PRUs allows achieving the time synchronisation in the order of nanoseconds during transmission and reception of data. The capes are generally used to extend the functionalities

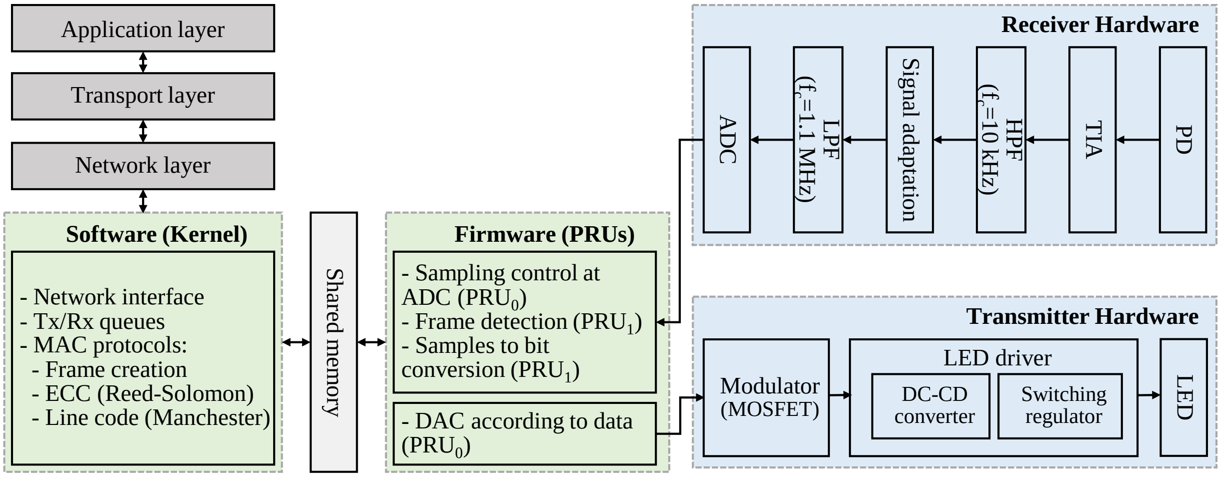


Fig. 3: Diagram of the front-end transceiver and the communication stack of OpenVLC 1.4.

of BBB in a simple manner. The hardware design has two main parts: Transmitter and Receiver, as shown in Fig. 3. The electronic components are selected with the aim to create a low-cost, small size and easy-to-manufacture cape.

Transmitter: In the new OpenVLC 1.4, a current controlled switching regulator is used to modulate the transmitter LED at high frequencies. The LED driver includes: 1) a DC-DC converter to step-up the provided 5 V input to 14 V in order to operate the high-power LED; and 2) a switching regulator that operates in continuous conduction mode to maintain unidirectional current through the LED loop and eliminates the biggest delay in switching the LED on and off. The modulator block is composed of an N-channel MOSFET in parallel to the LED to improve the slew rate of LED for achieving high modulation rates, and a MOSFET driver to feed high current to the gate of N-channel MOSFET to overcome the effect of gate capacitance. The modulation signal is provided by BBB I/O pin to the driver. This approach helps to achieve fast transient response, constant switching frequency, and accurate output current. A high-power LED together with a collimator and a heat sink are used to provide a high illuminance with the average power consumption of 6 W.

Receiver: The receiver converts the incident light into an electrical signal using a PD and process it to readable digital signal. The first amplification stage consists of a low-noise TIA, which presents a low impedance to PD and acts as an isolation between PD and output voltage. Subsequently, the signal is passed through the High-Pass Filter (HPF) to remove the DC and low frequency components coming from other interfering sources, with a cut-off frequency of 10 kHz. Next, the signal is adapted to the dynamic range of the ADC by introducing a DC level and a voltage amplification. The amplified signal is then passed through the Low-Pass Filter (LPF) with cut-off frequency of 1.1 MHz to remove the high-frequency noise components mainly introduced by the overshooting of op-amps. Finally, a digital signal is obtained from the ADC and fed into BBB using the Serial Peripheral Interface (SPI).

IV. VLC Prototyping: Software

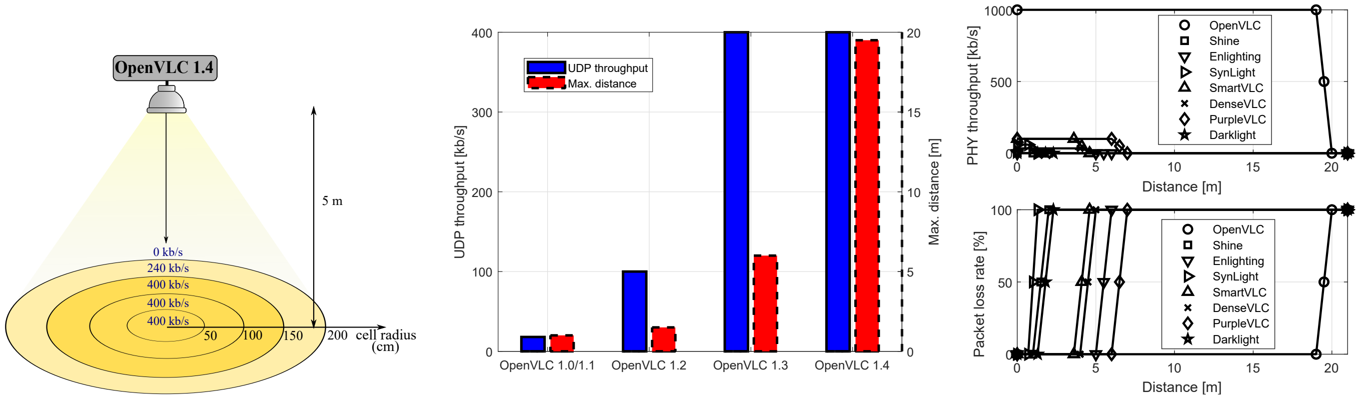
In this section, we present the software design of the OpenVLC platform. Software manages digital signals both in the transmitter and the receiver chains.

A. *OpenVLC Software in Linux Driver*

The first block of the communication pipeline is composed by processes that, as they do not require a high timing precision, do not necessarily require an extremely high priority. For this reason, we place this first group in the operating system and implement a Linux driver. This allows creating an OpenVLC *network interface* in the Linux kernel. As any other network interface, it allows changing its IP address, routing packets using the TCP/IP Linux stack, creating routing rules, opening a socket and writing or listening to it, etc. Besides, we can use the *iperf* tool for network measurement. Note that, for upper layer applications, the use of the OpenVLC channel is completely opaque and it does not require any configuration or special treatment. Any user space application can interact with OpenVLC as with any other network interface, such as Ethernet or WiFi interfaces. As soon as the Linux driver is initialized, it creates two threads, one for transmission and one for reception, which are represented in Fig. 3:

1) Transmission: The thread remains in an idle state until a message is received from upper layers. When a packet is received, the kernel module places it in a *transmission queue* and waits for the event. Once the transmission thread detects that a message is ready to be transmitted, the kernel dequeues it and gets out of its idle state to process it. In the first state of the processing pipeline, the *frame is created* with the data and a header. Then the frame is divided into blocks to apply *error correction codes* (Reed-Solomon). Finally, *Manchester coding* is adopted to provide a constant illumination and the resulting information is placed in a memory shared with PRUs.

2) Reception: The process followed is the mirror to the one in the transmission thread. Now the thread is constantly checking for new data on the shared memory. Whenever it contains new data, it checks the header to compute the number



(a) Performance of the new OpenVLC 1.4.

(b) Evolution among all OpenVLC versions.

(c) Comparison with the state-of-the-art non-commercial VLC platforms.

Fig. 4: Experimental results of the new OpenVLC1.4.

of blocks to be received and decodes them. After that, it prepares the data in a packet and it informs the Linux driver that a packet is ready to be processed by upper layers.

B. OpenVLC Firmware

OpenVLC firmware carries out all the tasks requiring an extreme time precision. BBB has two PRUs that can be used for this purpose, each one having a precision of 5 ns. The firmware has been programmed in assembly. The processing and precision required for reception force to employ both PRUs. For this reason, in OpenVLC we can only transmit or receive at a time (half-duplex), although it could become full-duplex if more PRUs are added or the throughput required is decreased. The tasks carried out in the OpenVLC firmware at transmission and reception are:

1) Transmission: The main task carried out is the *Digital-to-Analog Converter (DAC)*. The transmitter is constantly waiting for a new packet from upper layers. The packets are handled bit-by-bit. Each bit represents a symbol ‘1’ being HIGH and ‘0’ being LOW. The processor reads a value from a register, obtains the correct bit, and controls the GPIO used for transmission. The code requires a different amount of instructions depending on the value of the symbol, the position of the bit or the requirement to read the memory. We have a precise control to guarantee that every symbol transmission takes exactly the same time to avoid de-synchronization issues. OpenVLC transmits at a bit frequency of 1 Mb/s. After finishing the transmission, the PRU places a 0 in the first memory shared with the kernel, allowing it to write a new packet.

2) Reception: As we do not know if the received signal is noise or data until we process it, we must implement a continuous signal processing loop. We divide the required processing into two groups and place each in one PRU:

PRU$_0$ is in charge of the processing that requires hardware control for reception. It *controls the ADC for sampling* the received signal and retrieves values from it using a SPI. This is done at 2.1 MHz. After reading each raw value of ADC, the value is placed in the shared memory.

PRU$_1$ carries out the processing to *detect and demodulate the frame*. It detects if the sample received belongs to a HIGH or a LOW comparing it with the medium value of the ADC. As we use Manchester coding, after receiving two symbols, we perform the *sample-to-bit conversion*. The PRU$_1$ first checks for a certain pattern in the received data. If it matches the preamble and special frame delimiter, it receives the next bits and computes the size of the frame. Every decoded bit is placed in a memory shared with the kernel.

V. Results

A. Experimental Evaluation

The performance of the new OpenVLC 1.4 is shown in Fig. 4a and Fig. 4b. This new version can transmit data up to a distance of 19 m, which enables its use in industrial scenarios. Besides, it provides a cell coverage of around 2 m of radius at a distance of 5 m, then providing a standard illumination area with possible multiple neighbouring fixtures. Note that, as indicated in Table II, OpenVLC 1.4 supports dimming, i.e., the averaged emitted optical power can be adjusted to a certain light percentage depending on the required conditions, still transmitting VLC data. Finally, note that, since the creation of OpenVLC, it has increased the UDP throughput by a factor of 22× (up to 400 kb/s), and the transmitter-receiver distance by 19× (up to 19 m).

Figure 4c compares OpenVLC performance with those of state-of-the-art non-commercial research platforms. OpenVLC provides a much higher PHY throughput (1 Mb/s versus 100 kb/s or even lower) and lower packet loss rate. Besides, the maximum distance achieved by OpenVLC is around 4 times larger than the one provided by the rest of the platforms. Though counterparts use different modulation schemes and front-end elements (see Table I), the key differences that justify such gain are: 1) we provide a current-driven design that allows driving the LED at very high power while having temperature under control; and 2) the code is carefully split in assembly and driver for high and accurate sampling rates, yet efficient packet processing. Its high performance, maturity and open source, position OpenVLC as a suitable platform to do experimental VLC research.

B. Community Impact of OpenVLC

OpenVLC has been used for testing research outputs in real VLC scenarios, which helps to identify and face new challenges. To give some examples, the research community has employed OpenVLC for the creation of new developments in: *New MAC protocols* (CSMA with collision detection and hidden avoidance (CSMA/CD-HA)); *Channel characterization*; *Localization techniques*; *New hardware designs* such as REAL-VLC; *Battery-free systems* such as PassiveLiFi; *Cell-free networking* such as DenseVLC, which shows the potential of OpenVLC for its scalability, IoT scenarios and complex MIMO systems; *Applications* such as video streaming or underwater communications. Besides, new platforms have been created based on OpenVLC, such as: SmartVLC for smart lighting; LocalVLC for IoT oriented-services; DYRP-VLC as a dynamic routing protocol; Purple VLC as a multi-LED and multi-PD platform to work on light polarization; and platforms on VLC side channel. References to all related developments are available in the OpenVLC platform website.

VI. Lessons Learned and Future Opportunities

A. Lessons Learned with OpenVLC Prototyping

OpenVLC was initiated with the objective of making VLC accessible to researchers and developers. We found our place between the users with very high requirements (very high-end platforms), and users with very specific applications where no available system meets the requirements and they need to build it from scratch. During the life of the OpenVLC platform, we learnt some lessons:

Lesson 1: Provide a quick start guide reduces the entrance barrier. Once every component of OpenVLC is built, a VLC link can be created in approximately 10-15 minutes. This allows many users to test if VLC is a technology suitable for their requirements in an easy and inexpensive way.

Lesson 2: Strengthen appealing and generic figures of metric augments the usefulness, rather than focusing on specific features and individual research interests.

Lesson 3: Reduce options of choice for hardware components, as flexibly changing LED and PD causes sub-optimal performance due to mismatches with the baseband hardware.

Lesson 4: Make the development open access whenever it is possible increases the impact. Since 2020 OpenVLC is open-source and open-hardware.

B. Future Opportunities

The VLC prototyping guidelines provided in this paper focus on low-cost design for IoT applications, but it also leads to future opportunities on VLC prototyping from a low-cost perspective, including a range of communication and wireless networking research issues. In the following, we show how researchers can contribute to the OpenVLC ecosystem, both in hardware and software, and how people can leverage this platform for teaching:

1) Hardware

New hardware designs: new components of circuitry could be tested in OpenVLC. For example, the phosphor-coated LED by default could be replaced by other kinds of LEDs, e.g., μLED, OLED, RGB LED, or even lasers and diffusers. Note that the operating current values must be adapted with resistors in the driver to ensure operating within the light source dynamic range. Besides, solar cells can be employed to simultaneously receive data and harvest energy, then making the device self-sustainable.

Merge OpenVLC with RF technology: Since VLC does not interfere with RF systems, both can complement each other for load balancing and line-of-sight blockage avoidance. We can either include an external dongle or exploit the WiFi antenna integrated in some BBB versions.

Modular design: From a design perspective, it is always helpful to fabricate the system in a modular fashion, giving the possibility to the final user of changing whatever block is desired. There is no published approach that satisfies this modular requirement.

Additional material: One more degree of freedom can be included by using light polarization for data transmission. Materials such as polarizers and birefringent elements could be invoked. Also, passive VLC could be developed by using retroreflectors and LC shutters to modulate the light, which is an emerging research topic for low-power IoT systems [15].

Theoretical analysis: Analytical studies can be demonstrated with OpenVLC as support, such as non-linearities of the system or line-of-sight blockages.

2) Software

Modulation schemes: Pulse-based modulations can be developed into a low-end platform such as OpenVLC, or even colour-shift keying modulation if multi-chip LEDs (e.g., RGB LEDs) are employed. This would enable wavelength division multiplexing techniques, then multiplying the overall data rate. Although advanced modulation schemes could also be integrated, they are complex and consume too many resources to be integrated in a low-end platform.

Adaptive schemes: Adaptive schemes in terms of bandwidth, modulation and coding could be developed to adapt to channel conditions.

Multi-hop networks: Relaying techniques could be used for guaranteeing an end-to-end communication. This approach can tackle the shadowing effect in VLC. Suitable protocols for multi-hop networks can be implemented.

3) Teaching

OpenVLC can also be used for teaching in telecommunications, electronics, computer science, and photonics. Students can become familiar with concepts such as Nyquist theorem and its sampling rate, or pulse-based modulation schemes; differently from RF solutions, simple access schemes like listen-before-talk can be visually shown programming OpenVLC; electronics concepts such as a PCB design, elements limitations (3-dB bandwidth of LEDs and PDs), Linux driver design and its interface to the TCP/IP stack and to the PHY layer can be learnt; as well as photonics concepts such as non-linearities of LED transfer function.

VII. Conclusion

This paper provided the basics for prototyping a VLC link, supported by the open-source and open-hardware OpenVLC platform. Besides, state-of-the-art VLC platforms, impact and learned lessons from OpenVLC prototyping and future opportunities were detailed. We have shown the performance and features of various generation of OpenVLC platforms over the years and the design choices as a platform for research purposes in IoT applications. Besides, we presented the latest OpenVLC version, its design and performance results.

Acknowledgment

The authors would like to thank the collaboration of Daniele Puccinelli (SUPSI) and Omprakash Gnawali (University of Houston) in the early stages of OpenVLC development.

References

[1] F. Tariq, M. R. A. Khandaker, K. Wong, M. A. Imran, M. Bennis, and M. Debbah, “A speculative study on 6G,” *IEEE Wireless Commun.*, vol. 27, no. 4, pp. 118–125, 2020.

[2] K. Mekki, E. Bajic, F. Chaxel, and F. Meyer, “Overview of cellular LPWAN technologies for IoT deployment: Sigfox, LoRaWAN, and NB-IoT,” in *Proc. IEEE PerCom Workshops*, 2018, pp. 197–202.

[3] L. Klaver and M. Zuniga, “Shine: A step towards distributed multi-hop visible light communication,” in *Proc. IEEE MASS*, 2015, pp. 235–243.

[4] S. Schmid, T. Richner, S. Mangold, and T. R. Gross, “Enlighting: An indoor visible light communication system based on networked light bulbs,” in *Proc. IEEE SECON*, 2016, pp. 1–9.

[5] Y. Yang, J. Luo, C. Chen, W.-D. Zhong, and L. Chen, “SynLight: synthetic light emission for fast transmission in COTS device-enabled VLC,” in *Proc. IEEE INFOCOM*, 2019, pp. 1297–1305.

[6] H. Wu, Q. Wang, J. Xiong, and M. Zuniga, “SmartVLC: Co-designing smart lighting and communication for visible light networks,” *IEEE Trans. Mobile Comput.*, vol. 19, no. 8, pp. 1956–1970, 2020.

[7] J. Beysens, Q. Wang, A. Galisteo, D. Giustiniano, and S. Pollin, “A cell-free networking system with visible light,” *IEEE/ACM Trans. Netw.*, pp. 461–476, 2020.

[8] S. Yin, N. Smaoui, M. Heydariaan, and O. Gnawali, “Purple VLC: Accelerating visible light communication in room-area through PRU offloading,” in *Proc. EWSN*, 2018, pp. 67–78.

[9] K. Hewage, A. Varshney, A. Hilmia, and T. Voigt, “modBulb: A modular light bulb for visible light communication,” in *Proc. Workshop on Visible Light Communication Systems*, 2016, pp. 13–18.

[10] Z. Tian, K. Wright, and X. Zhou, “The darklight rises: Visible light communication in the dark,” in *Proc. ACM MobiCom*, 2016, pp. 2–15.

[11] L. Shi, W. Li, X. Zhang, Y. Zhang, G. Chen, and A. Vladimirescu, “Experimental 5G New Radio integration with VLC,” in *Proc. IEEE ICECS*, 2018, pp. 61–64.

[12] Q. Wang, D. Giustiniano, and D. Puccinelli, “An open source research platform for embedded visible light networking,” *IEEE Wireless Commun.*, vol. 22, no. 2, pp. 94–100, 2015.

[13] A. Galisteo, D. Juara, Q. Wang, and D. Giustiniano, “OpenVLC1.2: Achieving higher throughput in low-end visible light communication networks,” in *Proc. WONS*, 2018, pp. 117–120.

[14] A. Galisteo, D. Juara, and D. Giustiniano, “Research in visible light communication systems with OpenVLC1.3,” in *Proc. IEEE World Forum on Internet of Things*, 2019, pp. 539–544.

[15] S. K. Ghiasi, M. A. Z. Zamalloa, and K. Langendoen, “A principled design for passive light communication,” in *Proc. ACM MobiCom*, 2021, pp. 121–133.

Borja Genoves Guzman [M] (borja.genoves@imdea.org) is a Postdoctoral researcher at IMDEA Networks. He received a Ph.D. degree from the UC3M in 2019 (Extraordinary PhD award). His current research focuses on LiFi, IoT and next generation wireless networks.

Muhammad Sarmad Mir (sarmadmir2003@gmail.com) is a Ph.D. researcher at UC3M, Spain. He was a recipient of MSCA ITN scholarship, and his research interests include low-power communication and battery-less devices.

Dayrene Frometa (dayrene.frometa@imdea.org) is a Ph.D. researcher at UC3M and IMDEA Networks Institute, Spain. She is a recipient of MSCA-ITN scholarship, and her research interests include VLC, RF communication, and ML techniques.

Ander Galisteo (andergalisteo@gmail.com) is currently a Senior Firmware Engineer at Dojo Five, USA, and received a Ph.D. from UC3M and IMDEA Networks, Spain. He is working on cybersecurity for industrial embedded systems and applications.

Qing Wang [SM] (qing.wang@tudelft.nl) is a tenured Assistant Professor in the Delft University of Technology, the Netherlands. His research interests include visible light communication & sensing, and artificial intelligence of things.

Domenico Giustiniano [SM] (domenico.giustiniano@imdea.org) is Research Associate Professor (tenured) at IMDEA Networks Institute and leader of the Pervasive Wireless System Group. His current research interests cover battery-free IoT systems, large-scale spectrum-based analytics, and 5G+ localization systems.