IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

IEEE BROADCAST TECHNOLOGY SOCIETY SECTION

Received 1 February 2023, accepted 26 February 2023, date of publication 16 March 2023, date of current version 23 March 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3257856

METHODS

Methodology to Control IoT Smart Devices Driven by Digital TV Video Scene

RODRIGO RIBEIRO DE OLIVEIRA, MAURICIO FERREIRA SOARES, ADRIANA SILVA DE SOUZA, AND KEVEN DE CASTRO GOMES
SIDIA Institute of Science and Technology, Manaus, Amazonas 69055-035, Brazil

Corresponding author: Rodrigo Ribeiro de Oliveira (rodrigo.oliveira@sidia.com)

This work was supported by Samsung Eletrônica da Amazônia Ltda., through the auspice of the informatics law under Grant 8.387/91.

ABSTRACT This paper presents a novel methodology to control Internet of Things (IoT) devices synchronized with digital TV (DTV) signal dynamic video flow. Through this approach, broadcasters can send commands to digital TV receivers and produce multiples physical effects on IoT devices paired with TV sets. Among IoT devices examples, are: smart lamps, robotic seats, smart watches, wearable devices, etc. Those devices will synchronously react with the video scene, according to the IoT command type received by TV set. With this methodology, broadcasters may multiplex multiple IoT commands and be able to produce several effects at same time, through different IoT devices, creating a home multi-dimensional environment. In case there is any command that does not correspond to any paired device on receiver side, or the command is corrupted, the same is simply discarded. The validation of this methodology is based on the development of a receiver embedded system to detect signalization of IoT commands, Data filtering and IoT device control through Smart TV. The IoT devices control is sync with video scene and is based at time stamps added to the DTV signal. Furthermore, to check transceiver channel behavior, a transport stream benchmark, composed by IoT commands data flow multiplexed with audio and video medias in multiple complexity levels, was generated. Lab experiments with transmission tests and IoT devices control results shown technical feasibility to use this methodology in a real digital TV environment. The main contribution of this methodology is to facilitate the creation and dissemination of multi-dimensional entertainment environments, on worldwide households, improving the Smart TV user experience.

INDEX TERMS Internet of Things, digital TV, multidimensional theater, multimedia environments, SBTVD, ISDB-TB, data broadcast, WiFi, communication protocols, smart TV.

I. INTRODUCTION

The recent worldwide pandemic has been raising new digital media trends with greater impact to streaming services [1]. Easy access to digital media is changing people's life in such a way that nowadays they have been opting to stay at home watching movies and other entertainment, more than other leisure options. These trends expanded the TV market, due to Smart TV's major role as an end point for delivering multimedia content through internet. Recent stats shown that in 2022 Smart TV device is already present at 1.7 billion households worldwide with forecast of 51% household worldwide to purchase a new Smart TV until 2026 [2]. With the already wide TV market expansion, in a couple of years, almost every household worldwide will have a Smart TV. Those trends associated with the Internet of Things (IoT) advent may create a huge entertainment market with devices interacting with each other allowing the user to immerse himself with the multimedia content. IoT devices are everywhere, performing important roles to help people perform their tasks or communicate more effectively [3], [4]. These devices use communication protocols to exchange data with each other producing a powerful communication chain [5], and these chains can be connected with internet, collecting data to feed this immense source of knowledge [5].

The associate editor coordinating the review of this manuscript and approving it for publication was Jian Song.

Nowadays, it is common to have multiple IoT devices at home, which may be paired to control or even be controlled by a TV set. Those electronic devices are present at domestic scene being part of people's everyday life. Among a variety of IoT devices, examples are smart lamps, smart air conditioner, smart clocks, smart easy chair, etc. All devices present at the home scene can be part of a multi-dimensional environment, reacting to multimedia content being reproduced in a Smart TV set. Thus, people can enjoy this realistic experience at their own house.

Around the world, few people have the privilege to feel the experience provided by multiple dimensions multimedia environments. Besides that, in general, small cities in emerging countries do not even have a mainstream movie theater available to the local population. The impact is even greater in rural communities where movie theaters play a major role promoting socioeconomic development [6]. Some studies have shown that strategies used to acquire, renovate, and maintain movie theaters provide ideas for rural community development elsewhere [6]. Therefore, multidimensional movie theaters are a privilege of great urban centers and some country capitals.

Normally those expensive multidimensional indoors movie theater environments are composed by control motion based at pneumatic systems developed to react synchronously with media playback [7], [8].

Digital TV environments are defined by complex standards allowing broadcast signal delivering for thousands of digital TV receivers within signal range. Digital TV environment follows digital TV standard adopted by each country. Nowadays, around the world there are four digital TV systems [9] covering all livable areas in the world. Digital TV standards share similarities using the same strategy to data transportation. Theoretically, any data type may be transported through digital TV transceiver channel. The literature offers several works involving data transportation from broadcaster to receivers for different purposes [10], [11], [12], [13]. Fig. 1 shows the main agents involved in IoT devices control through a digital TV signal.

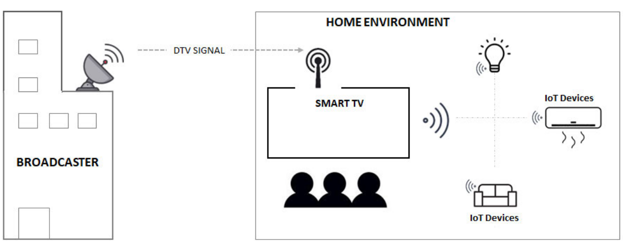


FIGURE 1. Overview involving communication between broadcaster, smart TV set and IoT devices.

In this scheme, the broadcaster is responsible to add a new flow with IoT commands synchronized with audio and video (A/V) elementary streams. On the receiver side (Smart TV) the commands are extracted from digital TV signal, parsed and ready to control devices. It is worth remembering that all controllable IoT devices must be paired in advance with a Smart TV receiver. In short, the main contribution of this work is to facilitate the creation and dissemintaion of multi-dimensional entertainment environments, on worldwide households, which, until now, were extremely limited and not viable.

Throughout this work all details referring to this methodology are thoroughly presented. Beginning with the study and analysis of other approaches, the background necessary to understand the steps of this work, detailing every step of this methodology, broadcast content preparation, reception, and data processing until the control of IoT devices mentioned at digital TV signal. After this, there is a detailed explanation about a benchmark that holds a set of five different transport streams with different level of control of IoT devices and is used to evaluate the technical feasibility to deployment in a real digital TV environment, through digital TV broadcast tests. Lastly, the obtained results and work conclusions are presented.

II. STUDY AND ANALYSIS CONSIDERING DIFFERENT APPROACHES

Related works have been used in the entertainment industry for some decades in major areas, such as multidimensional theaters and Digital TV. On the theaters, there's the notable work of [14], which uses a subtitle file format (SubRip) information to synchronize the physical effects of the *Player5D* along the displayed audiovisual content, to contribute to the viewers immersive experience.

On the DTV scope, there are works like [15] and [16], who overlaps the layers of video and Flash animation, using an XML file to synchronize the object on the television screen with the object in the Flash animation, therefore, promoting their interaction. These works use the features of a local station, so the number of people, per session, benefiting from this kind of service is limited and relatively low. But in a scenario of high public, approaches like these are highly capital demanding [17]. To avoid that, one can take advantage of the vast market presence of Smart TV's and their DTV broadcasts, who allows simultaneous transmission of data alongside the audio and video (A/V) content. As an example, the work of [18] introduces the very notion of *broadcast games*. By taking advantage of the broadcast high range, and combining elements from both television, gaming, and real-time data, such as the car telemetry, camera, and object tracking, into the DTV signal. With this info at the receiver, people could go to a live game mode and play against the pilots of the broadcast, by feeding the simulator game AI the telemetry and tracking received from the broadcast.

The approach presented here uses open digital TV signal plus Smart TV to provide a suitable solution of 3D entertainment environment at households or indoor environments. Nowadays, there is a lack of IoT devices available on the market. Nonetheless, the possibility shown in this work may potentially create those environments and provides entertainment at any place with a Smart TV. Here, the home environment configuration and pairing IoT devices may be under TV user control, and is not dependent on specific devices, such as game consoles or internet services, as already shown by similar solutions.

III. BACKGROUND

This chapter shows an overview referring to the mainstream technology, aiming to providing a better understand about the methodology presented in this work.

A. INTERNET OF THINGS COMMUNICATION PROTOCOLS

Internet of things (IoT) is a well discussed topic in literature contemplating all technology that allows the most different communication devices connection to the internet and interaction with it. Several communication protocols are already available at the market. Those protocols use distinct connecting strategies to perform communication between the most variable kind of devices and this section brings some details of the most common IoT communication protocols.

1) WiFi

Is a network that can send wireless signals to devices, for example smartphones, routers, computers and in this particular case, IoT devices. It can provide internet connection to devices closely located. It works in frequencies range between 2.4GHz or 5GHz to establishing communication through radio waves. The range of a typical Wi-Fi connection is around 100 meters, although the most commonly used range is only 10 to 35 meters. The speed and the range of Wi-Fi connection heavily depends on the environment, which means the speed is higher when the device is closer to the main source and the range depends on the antenna strength and obstacles in the way [19].

2) ZigBee

Is a short-duration wireless communication protocol greatly used in IoT devices because of its low energy consumption. The base of ZigBee protocol is IEEE 802.15.4, that specifies low-rate wireless personal area networks (LR-WPAN) standards. The protocol stack for ZigBee is composed by MAC, network and application layer, supporting three different types of devices: ZigBee coordinator, router and end device. The coordinator is a device responsible for managing the entire network. The router is responsible in tree and mesh networks only. Application object is a software used for controlling hardware units like switches or transducer on devices. These application objects contain a unique identification used for communication with others. The ZigBee Device Object is responsible for managing requests, devices discovery, security, etc [20].

3) Z-WAVE

Is a wireless, interoperable, based on a RF communication technology created for monitoring, controlling and status reading applications for both residential and light commercial environments. Unlike ZigBee, Z-wave supports full mesh networks without the need of a coordinator. It is a Low Powered RF communication that operates at a lower frequency (sub-1GHz), which prevents interference from other wireless technologies, such as Wi-Fi, Bluetooth, ZigBee, etc. It is mainly designed for control and status apps, with data rates of up to 100kbps with AES128 encryption, multi-channel operations and IPV6 [21].

4) PAIRING TECHNIQUES TO P2P COMMUNICATION

Peer-to-peer (P2P) is a computer network architecture where each point or node on the network works as a client/server for the other points on the network. With this, it is possible to provide shared and common access to various resources by excluding a central server, which allows data sharing without the need for a central server. As it is not based on a client-server architecture, where the server is responsible for the execution of all network functions and for being distributed, it divides all the work between the pairs involved in the network, with this in view, it is clear that the main advantage is not being dependent on a central server. As all nodes are interconnected, it is possible for a node to access through any other node in the network, thus providing a high degree of network availability [22].

B. DIGITAL TV TRANSCEIVER CHANNEL

Digital TV transceiver channel is composed by a broadcaster and a digital TV receiver unit, here represented by Smart TV's. Naturally, so that every communication environment works correctly, it is mandatory that both communication sides follow the rules of the adopted digital TV standard. Among the main existent systems around the world are: Digital Video Broadcasting (DVB), Advanced Television Systems Committee (ATSC) and Integrated Digital Services Broadcasting (ISDB). Brazil and other countries from Latin America adopted ISDB standard with the improvements added by Brazil generating the Brazilian Digital TV System (SBTVD). Digital TV environment is composed by complex subsystems allowing broadcast of High-Definition Television (HDTV) content to thousands of receivers within signal range based at a binary transportation flow (digital TV signal). Those subsystems adopt different media coding formats, modulation schemes, strategies of channel coding, techniques of error correction, schemes for data transportation provided by Moving Pictures Experts Group (MPEG), etc. The transportation layer (MPEG2-TS) uses metadata to signalize the transport stream content, such as, Program Specific Information (PSI) tables, System Information (SI) tables and descriptors [26]. Among PSI tables are the program association table (PAT), program map table (PMT), service description table (SDT) and network Information table (NIT). The descriptors, in turn, give more information about the transport stream content and the broadcaster channel. These metadata, known as signalization info, are mandatory to inform service data, such as, stream identifiers, stream type, component tag, PID, etc., and any other info necessary to decode digital TV multimedia content [26]. MPEG2-TS defines two hierarchical data transport levels, sections, for signalization content and packetized elementary stream (PES), to carry audio and video (A/V) media streaming and other data streams such as sign language [11], [13] and closed caption, or even IoT commands.

IV. METHODOLOGY TO CONTROL IoT SMART DEVICES DRIVEN BY DIGITAL TV VIDEO SCENE

The methodology proposed in this work has its usage premise in a complete digital TV environment, composed by a Broadcaster and Receiver (transceiver channel) plus IoT devices previously paired with digital TV receiver. In short, the Fig. 2 depicts a diagram with all necessary interaction to control a single IoT device (i.e. smart lamp).

On the broadcaster side, the IoT command, here represented by a lamp, is encoded in a private format then packaged into PES transfer level structure. Similarly, audio and video (A/V) are encoded then packaged into PES to finally be multiplexed with other contents, such as metadata with signalization info, necessary to inform the receiver about signal content and stream identifiers. In the end of this process a transport stream (TS), composed by 188 bytes packages, is generated. After this, the transport stream is modulated and broadcast through air.

On the receiver side, the digital TV signal is received and demodulated, generating a single transport stream (MPEG2-TS) as output. Then, the transport stream is demultiplexed into elementary streams carrying on different contents, including audio/video (A/V) medias and metadata. The signalization info and descriptors (metadata) are equally obtained from section transfer level. Firstly, the receiver unit uses metadata to check the receiver eligibility (i.e. support to play IoT commands). After this check, the native system proceeds with IoT stream commands decoding and devices control. In this case, in particular, the parsed IoT command correspond to a smart lamp. After parsing this command, the system checks the existence of a smart lamp device paired with Smart TV receiver to finally transmit the action control to it. To better explain each step of this methodology, the subject was split in the following parts:

- **Broadcaster Content Edition**
  Add signalization info through PSI tables and descriptors and the new elementary stream with IoT command controls synchronized with other A/V elementary streams.
- **Receiver's IoT Devices Pairing**
  Scan the IoT devices compatible with this methodology through home network in order perform the pairing with TV device.
- **Receiver's Eligibility Checking**
  Analyses the signalization info to check the receiver eligibility to decode IoT commands.
- **Receiver's IoT Devices Controlling**
  After eligibility checking, the native system starts to extract the IoT commands, from data elementary stream, and control the IoT devices continuously.

A. BROADCASTER CONTENT EDITION

For this methodology, the broadcasters must include the necessary info to guide the receiver's units to detect the content being transported. Among this information are metadata, such as PSI tables and descriptors, and elementary streams (i.e. audio, video, closed-caption, accessibility info, etc). All content added in the signal must be signalized in order to guide the receiver during content decoding. In the IoT Devices Commands case, this signalization is done by labeling the IoT Device Commands data flow as an additional PES private component, at the broadcast PMT section table. For this new stream flow, two fields need to be correctly inserted: stream_type and component_tag.

This first PMT field is needed to define the structure of the data contained within the ES packet identifier. For this work, the used PES private data flow is identified by stream type (0×06), which is also used to carry on subtitles data. Whereas, the component_tag inside the Stream Identifier Descriptor signalizes the content type flowing through this private data. For the IoT commands, the component_tag was defined with the available value 0×95. According to [23], the recommended component_tag values are clustered by component type (i.e. Reserved 0×94 - 0xFF). Although those values were reserved and available for this purpose, it is worth mentioning that the definition of any new PES private data content (IoT data) must be discussed and regulated by the digital TV standard regulatory council.

1) SYNCHRONIZATION DATA INSERTION

For synchronization, broadcasters must include timestamps to guide the receivers during inter-media synchronization, which is responsible to maintain sync between audio, video and subtitles decoding. To that end, MPEG2-TS defines a timestamp metadata field, called Presentation Time Stamp (PTS), in order to guide the receiver when it needs to present the content during media decoding process. This value is given in units related to a program's overall clock reference, usually the Program Clock Reference (PCR), which is transmitted as a time reference in the receiver [24]. The Fig. 3 show how IoT commands are added at PES packet structure.

Starting with the first field showed in (a), packet start code prefix (PSCP), that is a standardized value to indicate the beginning of each PES package, then, the extended header indicating that there is program time stamp (PTS) info for sync and cyclic redundancy check (CRC) value of previous package [25]. And, finally, starts the PES payload area with the IoT commands. The empty PES showed in (b) carries the CRC value refer to previous PES package content. The CRC is a hash value corresponding to whole package content. A hash algorithm accepts variable-length input and produces

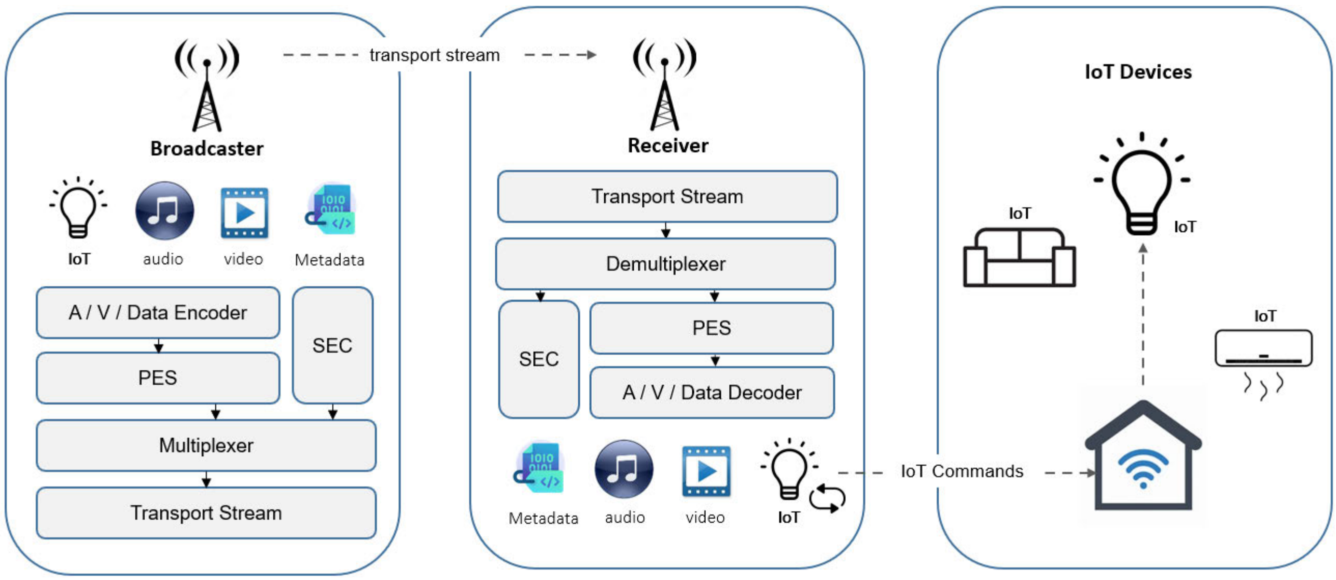


FIGURE 2. Overview about how an IoT device is controlled by digital TV Signal.

a fixed-length output which uniquely stands for the input data. This value is necessary to make sure the content from previous PES package (a) was not received corrupted.

As described in [26], the PES packet supports payloads with variable size, up to 65536 Bytes, leaving it up to application how to use it. Therefore, as the PES packet use the variable size strategy, for bounded contents, as is the case of IoT commands, it is necessary to indicate the beginning and ending of each transported data. Thus, to indicate to receivers that the whole IoT command have been broadcast, the tags "BEGINCMDS" and "ENDCMDS" were defined. It is important to notice that this methodology consider the usage of a single PES package to transport a single IoT command, or a sequence of IoT commands that are also wrapped between the tags (BEGINCMDS and ENDCMDS) based at same PTS value. Besides that, to maintain IoT devices controlling precision on receiver side, it is important to send the commands as close as possible to the presentation time (PTS) value.

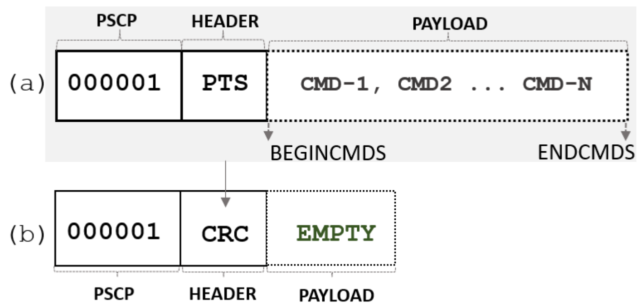


FIGURE 3. (a) PES with PTS and IoT commands, (b) Next PES with CRC from previous package.

B. RECEIVER'S IoT DEVICES PAIRING

Modern smart TVs provide multiple ways to establish connections with external devices. Among those, there are several IoT communication protocols covering a great part of electronic devices present in the home environment (see, chapter III-A). These protocols consist basically in pairing (e.g. connect) two devices establishing a peer to peer (P2P) communication. This process is already well established and performed by most part of electronic devices available at market. This step is crucial for the operation of this methodology considering it works with IoT devices previously paired. At this work, the IoT devices pairing with the TV can be seen at the diagram described in Fig. 4.

To pair the TV with other IoT devices, first the TV must be set to enable the pairing mode and, only then, the IoT device must be enabled to pairing mode. The next statement "TV Receiver searching for IoT devices" starts a pairing process that consists in setting up a linkage between TV receiver and IoT devices to allow communication between them. After that, the receiver tries to search for a device present within the range of the communication protocols. At this point, the found device is either paired with the DTV receiver and enabled to use, or the receiver may define a timeout to the searching engine in order to save processing, in case no device is present. In any case, the execution line checks if there are more IoT devices to be paired and, if positive, the next IoT device enters pairing mode and the subsequent steps are repeated again, until there are no other devices ready to be paired.

C. RECEIVER ELIGIBILITY CHECKING

The signalization on digital TV is essential to incorporate new features at digital TV standards without interfering at

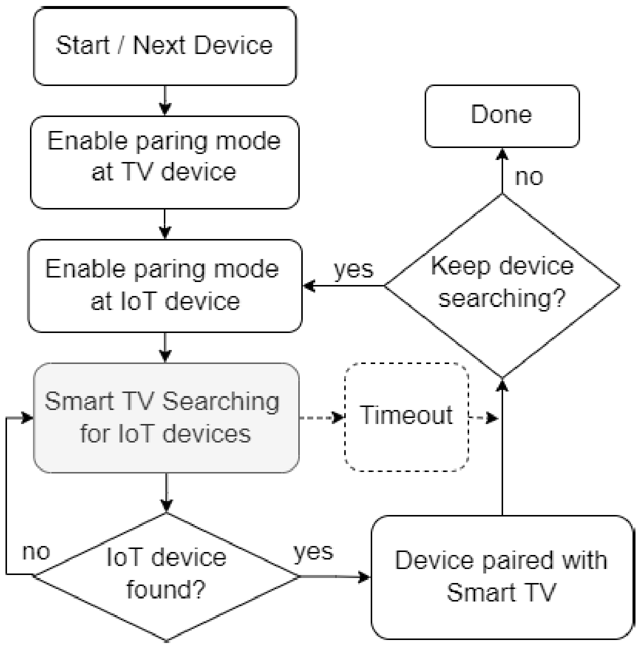


FIGURE 4. Diagram IoT devices pairing with TV set.

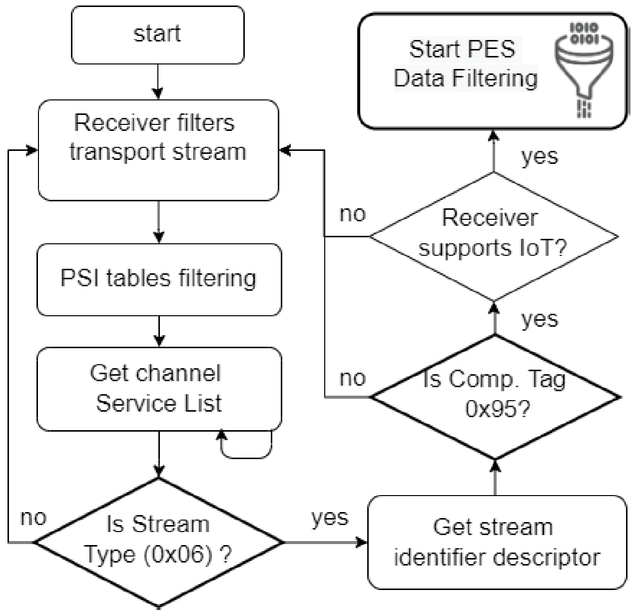


FIGURE 5. Detect receiver eligibility through signaling info provided by broadcaster.

legacy receivers located in field. As an example, the signalization may avoid issues in receivers that do not support some features. Signalization, "guides" receivers about the content being transported at digital TV signal. Thus, receivers can check whether or not they support broadcast content (Eligibility).

The eligibility check is shown at flowchart of Fig. 5 where the receiver consult digital TV tables and descriptors to check if it has support to decode transported content.

The premise before the start of this procedure is that the receiver check is tuned to a valid digital TV channel outputting a transport stream after demodulation step. Thereafter, the receiver control system subscribes at section filters to get the program system information tables, such as, program association table (PAT), program map table (PMT) and service description table (SDT). Throughout those tables, the system get the channel service list. After this, the execution line reach a conditional statement to check the existence of data elementary stream (stream_type $== 0 \times 06$). In negative case, the execution line goes to "Receiver filters transport stream" statement remaining in this checking loop. Otherwise, the execution line reaches statement "Get stream identifier descriptor" that provides information about elementary data stream content.

The next conditional statement checks if component_tag field has the value 0×95, that indicates the presence of IoT device control commands. Soon afterwards, the execution line reach conditional statement "Receiver supports IoT device Control" deciding receiver eligibility. Finally, when the verification is concluded, the system may start the next step and filter IoT commands.

D. RECEIVER's IoT DEVICES CONTROLLING

This step starts after the receiver eligibility confirmation, shown in the previous section. Here, the receiver control system subscribes to a PES filter using component_tag 0×95 to get the IoT commands flow according with flowchart of Fig. 6. It is worth to remember that the receiver system also must constantly extract the program clock reference (PCR) info, which is present at transport stream (TS) packages of tuned channel/service and is responsible to guide the receiver during synchronization tasks. As it is a multitask system, this task occurs in parallel.

At the statement "Filter PES flow using Comp. Tag 0×95" the receiver system filters and remounts PES packages based at component Tag (0×95). Firstly, on the "Extract PES Package" statement the system remounts the PES with IoT commands. Soon afterwards, the system extract CRC value (hash of whole previous package content) from the extended header of next PES package. After this, the system reaches conditional statement "Checksum OK?" that in turn check package integrity performing checksum of received data. If the checksumis is different, the system drops the corrupted PES packet and go back to "Receiver filters transport stream" statement, repeating all process again. Otherwise, the system reaches "Extract/Parse IoT command" statement, extracting the IoT command(s) from PES payload and adding it at CMD (n) Queue processing list. After that, the system reaches "Select IoT device from paired list based at received IoT command and gets IoT device current state to restore it in a given timeout. In parallel with this, the system reach "Extract PTS from PES header" statement that store related PTS (timestamps) necessary to have the exact time to send the command(s) to IoT devices. Finally the end of this process is the "Transmit IoT command to Device (s)" statement that

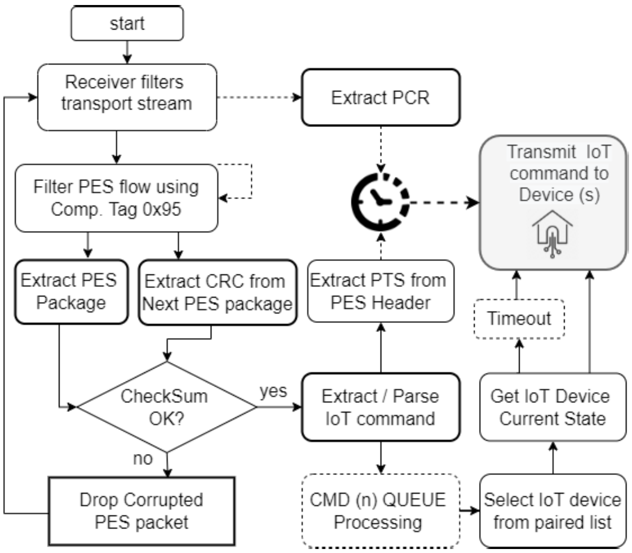


FIGURE 6. PES packet extraction, error checking (checksum), IoT command parsing and IoT device control.

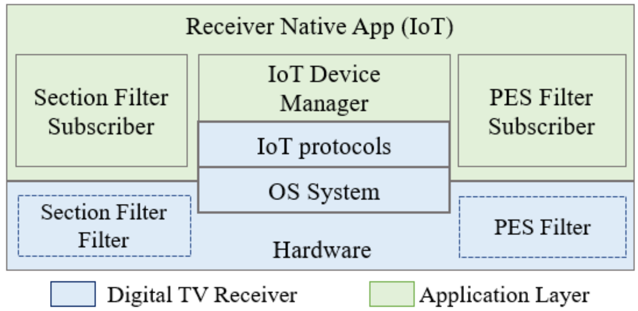


FIGURE 7. Digital TV receiver software architecture.

FIGURE 8. Graphical user interface of native app.

will happen when elapsed time achieve the timestamp. Here, the clock time reference is given by service PCR value. Using the service clock reference (PCR) and the timestamps (PTS) value, the system knows when transmit command to IoT device.

In order to validate the transceiver channel described in this Chapter (Sections A to D), it was necessary to complete all the necessary steps to have a proof of concept using a real digital television environment. To accomplish section (A), some ISDB-T transport streams with IoT commands data flow, were created. The details about these transport streams are show at Chapter V-B. In case of Section (B), the pairing of IoT devices with smart TV procedure is a well-defined task and already accomplished at current smart TV's models available at market. Therefore, the details provided by Chapter III-A and Section (B) are enough to understanding this procedure. Section C, in turn, depict how receiver native application performs signalization info detection and how it verify receiver eligibility for transported content. Finally, Section D showed how the receiver native app filters, parses, and uses timestamps to control paired IoT devices. The whole system architecture involved in this process is shown at Section IV-E.

E. DIGITAL TV RECEIVER ARCHITECTURE

The native app developed in this work confirmed the technical feasibility to deploy this methodology in the field. In short, the system reproduces the flowchart shown in sections C and D. Fig. 7, shows the receiver architecture and the main components involved in this process.

Among the main tasks of hardware layer are, A/V media decoding, transport stream demultiplexing, signal error rate measuring, signal descrambling, etc. This layer must have, at least, two digital signal processing (DSP) dedicated modules available to perform section and PES filtering. Normally, the receiver hardware provides multiple DSP instances to perform those tasks and, to use it, the native app must subscribe to those services. In short, the hardware layer filters sections and PES packages from the transport stream packages to provide it to software layer.

Next is the OS system layer, which is on top of the hardware layer and manages the receiver hardware resources allocation, task scheduling, etc. At last, the IoT protocols module is provided in modern receiver architectures providing support to different IoT communication protocols. This module supports the IoT devices manager, from the native app, that in turn manages pairing and communication with external IoT devices.

To guide users about which IoT device is being controlled by broadcaster signal, the native application provides a graphical user interface (GUI) with several IoT devices. This interface shows an IoT device reacting when some command arrives. At the example shown in Fig. 8, the native application received a smart lamp command and reacted to it by changing the lamp icon to its full opacity value.

It is worth to remember that, to this end, there are actually few types of IoT devices, available at the market and, among those, are the smart lamps, that support different IoT protocols and can be paired with smart TVs. However, to complement the GUI, an hypothetical domestic environment, with many IoT devices that in a few years may be available on the market, was devised.

TABLE 1. IoT devices command list.

Device	IoT Command	Effect
SL	!SL_Flashing[%value]!	SL will flash %value [0-∞]
	!SL_Dimming[%value]!	SL dimming %value [0-100%]
	!SL_Lighten[%value]!	SL lightened %value [0-100%]
	!SL_Off!	SL turn Off
	!SL_On!	SL turn On
	!SL_Color[%R,%G,%B]!	SL color to %R,%G,%B [0-255]
RS	!RS_Shake[%value]!	RS shake %value [0-100%]
	!RS_Stop!	RS stops
	!RS_LetDown[%value]!	RS let down %value [0-90°]
	!RS_LetUp[%value]!	RS let up %value [0-90°]
	!RS_LeanRight[%value]!	RS lean right %value [0-180°]
	!RS_LeanLeft[%value]!	RS lean left %value [0-180°]
	!RS_ReclineUp[%value]!	RS recline up %value [0-90°]
	!RS_ReclineDown[%value]!	RS recline down %value [0-90°]

V. EXPERIMENTS BASED IN THIS METHODOLOGY

To simulate the whole transceiver channel, it is necessary to perform the IoT commands broadcasting, receiving tests, as well as controlling the IoT devices paired with the digital TV receiver. Firstly, was defined a generic IoT command format to fit all possible devices used to this end. The proposed IoT command starts with an exclamation point (!), followed by the device name (DeviceName), plus a separator underscore (_), soon after, comes the action name (ActionName) field, then the optional arguments list enclosed by brackets $[arg_1, arg_2, \ldots, arg_n]$, to finally reach the end exclamation point (!) completing the IoT command:

$$\texttt{!DeviceName\_ActionName}[arg_1, arg_2, \ldots, arg_n]!$$

This simple command definition may cover features from several IoT devices, even those that do not exist yet. A simple parser may identify the device type to be controlled, get the action to be produced on device and the action specific values, given by argument list. As an example, the IoT command list necessary to produce different effects on a generalized smart lamp (SL). Another example, but not limited to, is the IoT command list for a hypothetical robotic seat (RS). Both lists are shown at Table 1.

Despite these devices being completely different, it was possible to define an IoT command list for both, following the generic IoT command format.

A. EXPERIMENTAL SETUP

The experimental setup is composed by auxiliar tools used to prepare broadcast material having IoT data, digital TV signal modulation and reception, and was used for tests and verification of technical feasibility for this methodology. For transport stream generation was developed an auxiliary data inserter and multiplexer tool. This tool was based at OpenCaster toolset version 3.2.2 [27] and is able to multiplex transport streams with IoT commands synchronized with pre-defined video scenes. For signal modulation and broadcast were used a ISDB-T signal USB modulator tool Dektec DTU-215 plus desktop TS playout tool StreamXpress (Fig. 9). Transmission parameters used in the modulator were: RF frequency: 617.143 MHz, Channel: 37, Data output level: 29.958.294 bps, TMCC option checked, Mod. Bandwidth: 6MHz - ISDB-T. On reception side, was used a commercial digital TV receiver running the native application shown at section IV-E. As external IoT device, a smart lamp model Philips Hue White and Color 800 and Philips Hue Bridge, which uses ZigBee protocol to connect to the Wi-Fi and control the device via internet, were used.

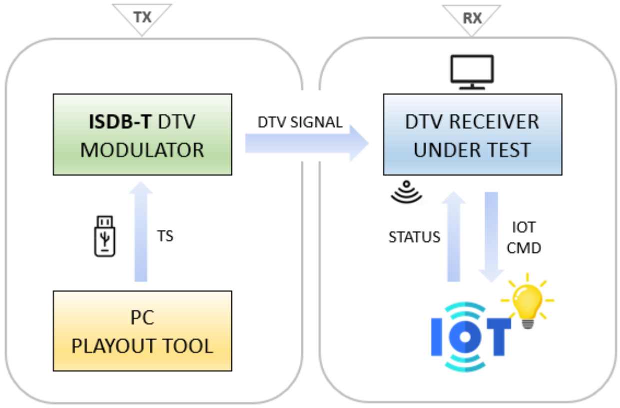


FIGURE 9. Experimental setup flowchart.

TABLE 2. Measures on receiver side.

TS Label	Duration (s)	AWT (s)	ADR (s)
one_cmd_block.ts	187	$4,29 \pm 0,021\sigma$	$6,03 \pm 0,363\sigma$
two_cmd_block.ts	187	$4,32 \pm 0,019\sigma$	$8,30 \pm 0,474\sigma$

B. BROADCASTING TESTS RESULTS

To understand the broadcasting channel behavior, some broadcasting and reception tests were performed. Thus, two ISDB-T transport streams composed by A/V media streams and smart lamps IoT commands (shown at Table 1) multiplexed at a additional elementary stream (PES) according the procedure presented in Chapter IV-A, were generated. It is worth to remember that for each IoT command, or block of commands, there is a single timestamp, needed for synchronization on receiver side. On test file "one_cmd_block.ts" 26 IoT commands to control a smart lamp were multiplexed. However, on test file "two_cmd_block.ts" 52 smart lamp commands, divided in 26 blocks with two commands each, were multiplexed. The transmission experiments show at Table 2 aims to measure the general queue waiting time, which consists of average time since the IoT command arrived at receiver until it is consumed, (Average Waiting Timestamps - AWT) and the general IoT command response interval, given by the average time since the IoT command is transmitted by smart TV until it reaches the smart lamp (Average Device Response - ADR).

The experiments results based at broadcasting and reception of IoT commands are shown at Table 2. The transmission tests using "one_cmd_block.ts" (single command per PES) the AWT was $4,29sec \pm 0,021\sigma$. This result shows the average queue waiting time until the command is used or transmitted from smart TV to IoT device. The ADR result for "one_cmd_block.ts" was $6,03 \pm 0,363\sigma$ showing the average time since the command was transmitted until it reached the IoT device. It is worth remembering that ADR considers the total time since the command was transmitted by TV set and received by IoT device Web server to finally reach the device. Besides that, the test results with stream "two_cmd_block.ts" (two commands per PES) had an AWT result of $4,32 \pm 0,019\sigma$, which is slightly longer than the previous, due to the fact that the composed commands were distributed in blocks and therefore consists of a bigger string to be parsed. The ADR result for "two_cmd_block.ts" was $8,30 \pm 0,474\sigma$, which is noticeable longer then the simple command result. This difference is expected, since the Web server side must process two commands arriving, from the same source, at the same time, to be executed in the same device, also at the same time. For this to be avoided, an approach with peer to peer (P2P) or TV to device direct communication, sharing the same Wi-Fi network connections, may be considered.

VI. CONCLUSION

This work introduced a novel approach allowing to control home IoT devices synchronized with video scenes of digital TV channel. The methodology used to broadcast IoT commands based at time stamps allows a precise sync of device control with A/V media playback. The experiments using a smart lamp showed the precise device response in sync with A/V media playback. Broadcast tests using the ISDB-T transport stream and a commercial TV receiver in application level validated the reception, parsing, remounting, and queuing of IoT commands. The IoT generic command format defined in this work may be used as reference control of new IoT devices control. Finally, it is expected that this methodology may bring to market new home entertainment environments based on new business models for TV manufacturers, TV broadcasters and streaming providers.

ACKNOWLEDGMENT

The authors express their gratitude to the SIDIA Institute of Science and Technology, for all the given support during the development of this work, and mainly to the Visual Display (VD)-Smart Things Team by the dedication and effort to complete this work.

REFERENCES

[1] C. Arkenberg. (2022). *Digital Media Trends—The Future of Movies*. Accessed: May 17, 2022. [Online]. Available: https://www2.deloitte.com/us/en/insights/industry/technology/future-of-the-movie-industry.html

[2] F. Karrer. (2022). *51% of All Households Worldwide Will Own a Smart TV by 2026*. Accessed: May 1, 2022. [Online]. Available: https://mountain.com/blog/51-of-all-households-worldwide-will-own-a-smart-tv-by-2026/

[3] S. Kumar, P. Tiwari, and M. Zymbler, "Internet of Things is a revolutionary approach for future technology enhancement: A review," *J. Big Data*, vol. 6, p. 111, Dec. 2019, doi: 10.1186/s40537-019-0268-2.

[4] R. A. Radouan Ait Mouha, "Internet of Things (IoT)," *J. Data Anal. Inf. Process.*, vol. 9, no. 2, pp. 77–101, 2021, doi: 10.4236/jdaip.2021.92006.

[5] L. Rainie and J. Anderson. *The Internet of Things Connectivity Binge: What are the Implications?* Pew Research Center. Accessed: May 2, 2022. [Online]. Available: https://www.pewresearch.org/internet/2017/06/06/the-internet-of-things-connectivity-binge-what-are-the-implications/

[6] R. Christman, "Movie theaters in the maintenance of rural communities in Kansas," *Great Plains Res.*, vol. 8, no. 2, pp. 299–313, 1998.

[7] K. Y. Dae, "Pneumatic supporting simulator device and portable theater using the device," U.S. Patent 101 361 685 B1, Jul. 19, 2011. [Online]. Available: https://patents.google.com/patent/KR101361685B1/en

[8] A. A. Roberts, "Dynamically orientable seating device," U.S. Patent 10 342 356 B1, Jun. 20, 2018. [Online]. Available: https://patents.google.com/patent/US10342356B1/en

[9] AVEL. (2022). *Worldwide Digital TV Standard*. Accessed: May 24, 2022. [Online]. Available: https://www.aviselectronics.com/articles/worldwide-digital-tv-standard

[10] R. R. D. Oliveira, F. A. S. Guimarães, M. M. D. Lucena, L. C. Cordeiro, E. B. D. L. Filho, and V. F. D. Lucena, "Hardware update through digital TV signals," *Electronics*, vol. 10, no. 24, p. 3072, Dec. 2021, doi: 10.3390/electronics10243072.

[11] R. Ribeiro de Oliveira, J. G. Reiser de Melo, L. Paes, and D. Frossard, "Method for synchronized driving of sign language avatar using a digital TV signal," in *HCI International 2020—Late Breaking Papers: Universal Access and Inclusive Design* (Lecture Notes in Computer Science), C. Stephanidis, M. Antona, Q. Gao, and J. Zhou, Eds. Cham, Switzerland: Springer, 2020, pp. 220–232, doi: 10.1007/978-3-030-60149-2_18.

[12] R. Ribeiro de Oliveira and D. F. Costa, "Method for controlling Internet of Things devices with digital TV receivers using transmission from a broadcaster in a transport stream flow," U.S. Patent 20 200 169 782 A1, Jul. 20, 2020. [Online]. Available: https://patents.google.com/patent/US20200169782A1/en

[13] R. R. de Oliveira, M. De Castro Loebens, A. S. Firmo Filho, and M. V. B. A. Bezerra, "Digital TV based sign language avatar," in *Proc. IEEE Int. Symp. Broadband Multimedia Syst. Broadcast. (BMSB)*, Jun. 2022, pp. 1–7, doi: 10.1109/BMSB55706.2022.9828571.

[14] L. C. Biazon, A. A. G. Martinazzo, M. A. Jose, I. K. Ficheman, M. K. Zuffo, and R. D. Lopes, "Developing an interactive 4D cinema platform with open source tools," in *Proc. IEEE Int. Symp. Consum. Electron. (ISCE)*, Sep. 2016, pp. 127–128, doi: 10.1109/ISCE.2016.7797404.

[15] A. Corrêa, A. A. G. Martinazzo, E. P. Lotto, and M. A. José, "Development of an immersive challenge racing game for 3D-TV," in *Proc. XII Simpósio Brasileiro de Jogos e Entretenimento Digital (SBGAMES)*, 2013, pp. 1–6. Accessed: May 13, 2020. [Online]. Available: http://www.sbgames.org/sbgames2013/proceedings/comp/01-full-paper.pdf

[16] L. C. P. Costa, A. G. D. Correa, M. A. Jose, E. P. Lotto, A. A. G. Martinazzo, L. C. Biazon, I. K. Ficheman, M. Nagamura, R. D. Lopes, and M. K. Zuffo, "3D stereoscopic game development technique for smart TVs," in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, Jan. 2014, pp. 1–2, doi: 10.1109/ICCE.2014.6776141.

[17] C. Giardina. (2013). *CinemaCon: 4DX Targeting U.S. Market*. Accessed: May 17, 2022. [Online]. Available: https://www.hollywoodreporter.com/movies/movie-news/cinemacon-4dx-targeting-us-market-442301/

[18] S. Gibbs, "Broadcast games and digital television," in *Proc. ACM Int. Game Technol. Conf.*, Hong Kong, 2001, pp. 17–21.

[19] AVSystem. (2019). *IoT Standards and Protocols Guide—Protocols of the Internet of Things*. Accessed: May 24, 2022. [Online]. Available: https://www.avsystem.com/blog/iot-protocols-and-standards/

[20] A. P. Singh and P. Tomar, "AI and IoT capabilities: Standards, procedures, applications, and protocols," in *Artificial Intelligence to Solve Pervasive Internet of Things Issues*, vol. 1, G. Kaur, P. Tomar, and M. Tanque, Eds., 1st ed. Cambridge, U.K.: Academic, 2021, pp. 67–83, doi: 10.1016/B978-0-12-818576-6.00022-8.

[21] L. Oliveira, J. Rodrigues, S. Kozlov, R. Rabêlo, and V. Albuquerque, "MAC layer protocols for Internet of Things: A survey," *Future Internet*, vol. 11, no. 1, p. 16, Jan. 2019, doi: 10.3390/fi11010016.

[22] P. Vijayakumar, R. Naresh, L. Jegatha Deborah, and S. Hafizul Islam, "An efficient group key agreement protocol for secure P2P communication," *Secur. Commun. Netw.*, vol. 9, no. 17, pp. 3952–3965, Nov. 2016, doi: 10.1002/sec.1578.

[23] *Digital Terrestrial Television—Operational Guideline—Part 3: Mulitplexing and Service Information (SI)—Guideline for ABNT NBR 15603 Implementation*, Standard ABNT NBR 15608-3:2020, 2020.

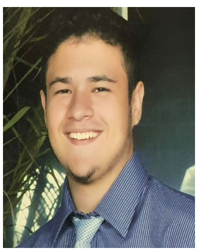
[24] R. Schwarz. (2006). *Comprehensive PCR Analysis Using the R&S DVM Family From Rohde & Schwarz R&S DVM50/DVM100/DVM120/DVM400*. Accessed: May 20, 2022. [Online]. Available: http://scdn.rohde-schwarz.com/ur/pws/dl_downloads/dl_application/application_notes/7bm55/7BM55_0E.pdf

[25] *Digital Terrestrial Television—Video Coding, Audio Coding and Multiplexing Part 3: Signal Multiplexing Systems*, Standard ABNT NBR 15602-3:2020, 2020.

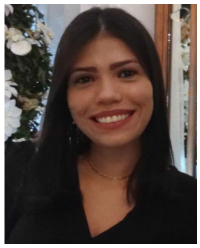
[26] I. technology. *Generic Coding of Moving Pictures and Associated Audio Information—Part 1: Systems*, Standard ISO/IEC 13818-1:2019.

[27] A. P. Singh and P. Tomar, "AI and IoT capabilities: Standards, procedures, applications, and protocols," *ACM Trans. Inf. Syst. Secur. (TISSEC)*, vol. 17, no. 4, pp. 1–27, 2015, doi: 10.1145/2723159.

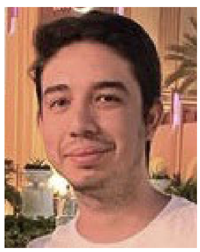
RODRIGO RIBEIRO DE OLIVEIRA received the B.Sc. degree in computer science from Universidade de Cruz Alta (Unicruz), Rio Grande do Sul, Brazil, and the M.Sc. and Ph.D. degrees in computer science from Universidade Federal do Amazonas (UFAM), Manaus, Amazonas, Brazil. Since 2001, he has been working with research and development of embedded systems projects. He is a Specialist in the development of high-complexity solutions involving intelligent architectures, digital TV technology, and embedded systems. He acted as a course coordinator and a teacher of computer science degree. He is currently with the SIDIA Institute of Science and Technology as a Researcher and a Technical Coordinator of the Visual Display (VD) Department, mainly with DTVPlay, digital TV Brazilian middleware, the Internet of Things communication protocols, native apps, and the research of new accessibility features to Samsung TV set. His research interests include microelectronics, hardware reconfiguration, embedded systems, digital TV technology, machine learning, the Internet of Things, and accessibility technology to multimedia environments.



MAURICIO FERREIRA SOARES received the B.Sc. degree in mechatronics engineering from Centro Federal de Educação Tecnológica de Minas Gerais-Campus V (CEFET-MG Campus V), Brazil, in 2021. He was a Research Fellow with Grupo de Estudos em Robótica (GER) and Estudos em Robótica e Automação (ERA), CEFET-MG Campus V, from 2017 to 2018 and from 2018 to 2021, respectively. He has professional experience in software development using standard C and C++, mainly in several projects for Samsung Digital TVs with the SIDIA Institute of Science and Technology (Sidia). His research interests include automatic machines, mobile robotics, embedded systems, and multicore platforms.



ADRIANA SILVA DE SOUZA received the B.S. degree in computer engineering from Fundação Centro de Análise, Pesquisa e Inovação Tecnológica (FUCAPI), Brazil, and the M.Sc. degree in electrical engineering from Universidade Federal do Amazonas (UFAM), Brazil. Since 2011, she has been working with software development. In addition, she has professional experience in software development using Shell Script, ANSI-C, C++, and Python. She participated in several projects, including embedded systems, applications for digital TV with the SIDIA Institute of Science and Technology (Sidia), and research projects developed by UFAM. Her current research interests include formal methods, bounded model verification, compilers, multicore platforms, embedded systems, image processing, multi-threaded model verification, compilers, and neural networks.



KEVEN DE CASTRO GOMES received the B.S. degree in electrical engineering from Universidade do Estado do Amazonas, Manaus, Brazil, in 2020, and the M.B.A degree in computational vision from Instituto de Technology and Management, Belo Horizonte, Brazil. In 2017, he was with companies to automate production lines, using embedded systems alongside with Programmable Logic Controller (PLC). From 2018 to 2020, he was a Developer with Vortice Technology and Innovation (VTInova), where he worked on embedded systems. He is currently a Software Developer with the SIDIA Institute of Science and Technology (Sidia), mainly on projects for Samsung Digital TV. In addition, he has experience working in software development, mainly using C/C++ for embedded systems and Python for automation scripts and machine learning.