LUÍS MIGUEL
BORGES SANTIAGO

Sistema de Alerta para Estação de Rádio FM Usando Comutação de Áudio

Alert System for FM Radio Station Using Audio Switching

Universidade de Aveiro
2023

LUÍS MIGUEL
BORGES SANTIAGO

Sistema de Alerta para Estação de Rádio FM Usando Comutação de Áudio

Alert System for FM Radio Station Using Audio Switching

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Eletrónica e Telecomunicações , realizada sob a orientação científica do Doutor António Navarro, Professor auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro, e do Doutor Nuno Borges de Carvalho, Professor catedrático do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

o júri / the jury

presidente / president

Prof. Doutor Pedro Miguel Ribeiro Lavrador
Professor Auxiliar da Universidade de Aveiro

vogais / examiners committee

Doutor José Pedro Mateiro Matias Borrego
Diretor-Geral Adjunto de Informação e Inovação da Anacom

Prof. Doutor António José Nunes Navarro Rodrigues
Professor Auxiliar da Universidade de Aveiro

agradecimentos / acknowledgements

Quero agradecer à minha família por me apoiar sempre, independentemente das contrariedades. Quero agradecer aos meus orientadores Prof. Doutor António Navarro e Prof. Doutor Nuno Borges de Carvalho, pela disponibilidade e orientação, e por me terem permitido trabalhar neste projeto. Quero agradecer à FCT/MCTES, pelo projeto Firetec (UIDP/EEA/50008/2020), pelo apoio financeiro e por me ter fornecido todo o material necessário para o desenvolvimento desta dissertação, e por me ter permitido participar no 16º Congresso do Comité Português da URSI, onde um artigo, no qual fui segundo autor, obteve o 3º prémio do *Best Student Paper Award*. Agradeço ao Instituto de Telecomunicações de Aveiro por me ter proporcionado todas as condições de trabalho. Agradeço também à ANACOM pela disponibilidade e esclarecimentos prestados. Quero ainda agradecer a todos os colegas, em particular ao Miguel, que fizeram parte do projeto e me ajudaram ao longo deste percurso.

Palavras Chave

Incêndios florestais, sistema de alerta, comutação de áudio, transmissão de dados, rádio FM, comunicação Ethernet.

Resumo

O problema dos incêndios florestais, recorrentes em cada verão que passa, levanta sérias questões de segurança para as populações, principalmente depois da tragédia ocorrida em 2017 na região de Pedrógão Grande, onde 66 pessoas, na sua grande maioria condutores, morreram a tentar fugir, acabando rodeados pelas chamas. De modo a evitar que se repita, tornou-se fundamental criar um sistema de alerta que possa detetar incêndios e avisar os condutores relativamente a estradas a evitar. Esta dissertação surge com o propósito de criar um sistema de alerta para estações de rádio FM. As estações passam a ter a capacidade de comutar o áudio, de "live" para uma mensagem de alerta. Esta tese propõe uma solução de hardware e software de baixo custo. Tem ainda a capacidade de comunicar com um servidor que é responsável pela criação das mensagens de alerta.

Keywords

Forest fires, alert system, audio switching, data transmission, FM radio, Ethernet communication.

Abstract

The problem of forest fires, which recur with each passing summer, raises serious safety issues for the population, especially after the tragedy that occurred in 2017 in the Pedrógão Grande region, where 66 people, the vast majority of them drivers, died trying to escape, ending up surrounded by flames. In order to prevent a recurrence, it has become essential to create an alert system that can detect fires and warn drivers of roads to avoid. The purpose of this dissertation is to create an alert system for FM radio stations. The stations now have the ability to switch the audio from live to an alert message. This thesis proposes a low-cost hardware and software solution. It also has the ability to communicate with a server that is responsible for creating the alert messages.

Contents

List of Figures

List of Tables

Glossary

AC	Alternating Current
ANACOM	National Communications Authority
ANEPC	National Emergency Service
CAP	Common Alerting Protocol
CMOS	Complementary Metal–Oxide–Semiconductor
DAC	Digital-to-Analog Converter
DC	Direct Current
EN	National Road
FCP	FireTec Communication Protocol
FEMA	Federal Emergency Management Agency
IP	Internet Protocol
IPAWS	Integrated Public Alert and Warning System
MAI	Ministry of Internal Affairs
NC	Normally Closed
NO	Normally Open
OASIS	Organization for the Advancement of Structured Information Standards
PCB	Printed Circuit Board
RF	Radio Frequency
SD	Secure Digital
SPDT	Single-Pole Double-Throw
SPI	Serial Peripheral Interface
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TS	Tip, Sleeve
UDP	User Datagram Protocol
USB	Universal Serial Bus
WAV	Waveform Audio File Format
XLR	External Line Return
XML	Extensible Markup Language

CHAPTER 1

Introduction

1.1 Context and Motivation

Every summer Portugal is visited by millions of tourists looking to enjoy the sunny weather, the beautiful beaches or the historic architecture [1]. Summer, particularly August, has also become a tradition for the return of Portuguese emigrants looking to ease their homesickness [2] [3]. One of the country top priorities is to keep everyone safe, both locals and visitors, but unfortunately, in complete contrast to those joyful visits, summer also marks the return of huge forest fires. Every year this scourge ravages the country, especially in rural areas, destroying forests and even homes. This phenomenon will only get worse with climate change [4]. The fires that start earlier, still during spring, can surprise authorities as the weather danger levels are considered lower [5].

In a group of fires that broke out on June 17$^{\text{th}}$ of 2017, around 53 thousand hectares of land were decimated and, besides this devastation, those fires also caused 66 deaths and over 200 injured people [6]. Among the number of fatalities, 47 were in vehicles that were trying to flee but ended up surrounded by flames on the EN 236-1 or on access roads to that one [6] [7]. The police had difficulties in receiving information, and in operating the road cut, that could alter the course of the events [8]. The aftermath is illustrated in Figure 1.1. Despite the fact that the flames were spreading wildly and at a very high speed, a system that could alert drivers would have been very useful.

It became imperative to develop an alert system that could detect fires and send messages to drivers informing them about which roads to avoid so as not to repeat the aforementioned tragedies. An efficient way of alerting drivers would be to make use of FM radio stations to broadcast audio alert messages. According to the National Communications Authority (ANACOM)[9], it is possible to tune into a local radio broadcast in at least 90% of the Portuguese territory. Thus, the probability of the alert message to reach the drivers tuned into a local radio station is likely 100%. Obviously, in order to comply with the Portuguese laws, the system would have to be regulated by government authorities, and therefore, a connection to the Ministry of Internal Affairs (MAI) is mandatory in order for them to

Figure 1.1: 47 people died on EN 236-1. Image credit: Miguel A. Lopes/LUSA [8].

authorize the switching process to take place in the radio stations resulting in broadcasting the alert messages.

The work developed in this dissertation is, therefore, part of a multidisciplinary project, called *FireTec*, which encompasses: a fire detection system, that makes use of the existing optical fiber networks to detect temperature variations; an alert system that uses audio switching, and which is connected to the Internet to receive alert messages; a server that provides the connection between the detection system and the alert system, which has a link to the MAI for official validation and authorization. This dissertation focuses on the audio switching alert system.

1.2 Objectives

The main objective of this dissertation is to develop and assemble an alert system which allows the switching of the audio signal that a radio station broadcasts, outputting, in case the situation so requires, a custom audio message containing crucial information for drivers traveling.

The system must include internet connection through an Ethernet interface to allow communications with a server, supported by the appropriate firmware to process the data received and save, then play, the audio messages. There should be inputs mounted for the regular radio station audio signals, for the Ethernet cable and for the Universal Serial Bus (USB) cable. The system should also contain the necessary hardware, that will allow to switch between two sources of signals, one being the live audio and the other being the alert audio message. It must incorporate, as well, its own power supply to power the different active hardware parts and keep the system running.

All of these parts must function together as a standalone device with all its components assembled into a closed box, with only the connectors made available to the exterior for connecting the appropriate and necessary cables.

1.3 Structure

- **Chapter 1:** Introduction providing motivation with an everyday context, objectives and structure of this dissertation.
- **Chapter 2:** Relevant information about transmission of data, with some protocols described, and transmission of audio.
- **Chapter 3:** Development of the alert system with hardware description, software organization and the integration of all the parts, together, in an embedded solution.
- **Chapter 4:** Presentation of the results obtained from some tests, carried out to evaluate the proper functioning of the switching circuit and the audio converter, the sound quality and the speed of the software code.
- **Chapter 5:** Conclusions on the work done along with future work suggestions.

CHAPTER 2

Overview on Transmission of Data and Audio

This chapter provides some information on ways of transmitting and processing data. Some options for transmitting audio signals are also presented.

2.1 Unbalanced Versus Balanced Audio

Unbalanced Audio

When an audio signal is transferred to an unbalanced connection, only two wires are required to transmit the signal. So if there are two channels, this implies the existence of two pairs of wires. Each pair consists of a hot, or positive, wire and a ground, or negative, wire. When subject to external electromagnetic interference, it will cause noise to be induced on the wires. In an unbalanced connection, the noise voltage generated on the positive wire is added to the signal voltage, and they are both amplified, as illustrated in Figure 2.1 [10][11].

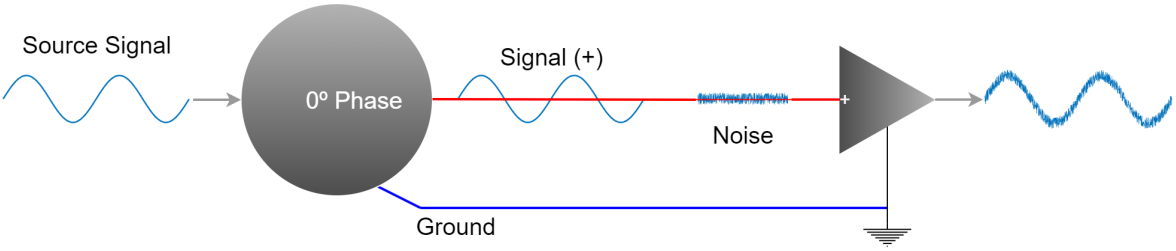


Figure 2.1: Example of an unbalanced connection signal exposed to noise. Adapted from [11].

The most common mono unbalanced connector is the Tip, Sleeve (TS) jack, which comes in 3.5 mm (1/8 inch) and 6.35 mm (1/4 inch) [10]. The tip part of the jack is connected to the positive wire, while the sleeve is connected to the ground conductor that surrounds the positive wire. The two different jack sizes are presented in Figure 2.2.

Figure 2.2: Common 3.5mm (top) and 6.35mm (bottom) TS jacks.

Balanced Audio

A balanced connection has two opposite signal phases per channel, which means there are three wires in each of these channels, a hot, or positive (+) wire, a cold, or negative (-) wire and a ground wire. When an audio source is fed into a balanced connection, the negative signal wire inverts the phase of the positive audio signal. Therefore, there are two identical audio signals traveling over the connection, in opposition of phase (180°difference) with each other. When the wires pick up electromagnetic interference, the noise voltage introduced is identical in both (hot and cold) signals. When the two out of phase signals are input into a differential amplifier, assuming unit gain, the output is a signal with double the amplitude of the hot signal, and with the noise voltages suppressed [11][12]. This is shown in Figure 2.3 as well as demonstrated in the equation:

$$\begin{aligned} V_{out} &= A_V[(V_1 + V_{noise}) - (V_2 + V_{noise})] \\ &= (V_{signal} + V_{noise}) - (-V_{signal} + V_{noise}) \\ &= V_{signal} + V_{signal} + V_{noise} - V_{noise} \\ &= 2V_{signal} \end{aligned} \tag{2.1}$$

where V_1 represents the hot signal voltage, V_{signal}, and V_2 represents the cold signal voltage, $-V_{signal}$, and assuming amplifier unit voltage gain, A_V [11].

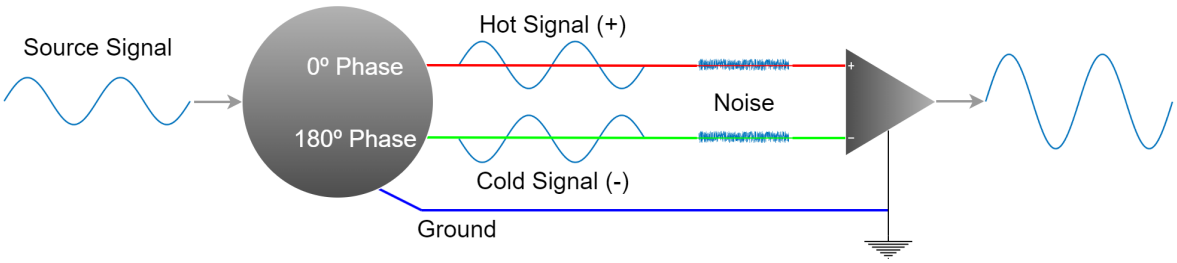


Figure 2.3: Example of a balanced connection signal exposed to noise. Adapted from [11].

The typical connector used for balanced transmission is an External Line Return (XLR) 3-pin jack, which is connected to a 3-wire cable. Each cable has a female and a male plug, and each plug contains one pin connected to the hot wire, one pin connected to the cold wire, and the third pin connected to the ground conductor that surrounds those two wires [11]. Figure 2.4 displays a male and a female XLR plug.

Figure 2.4: Female (left) and male (right) 3-pin XLR connectors.

As practical circuits are not perfect, they do not behave like in equation 2.1, and therefore the noise cancellation is not total. Even so, there is enough suppression of noise to justify the use of balanced connections over unbalanced ones [11].

2.2 TCP/IP Based Socket Communication

TCP/IP Suite

The Transmission Control Protocol/Internet Protocol (TCP/IP) is a suite of communication protocols. It provides a set of rules and conventions that enable computers to communicate and share data with each other over networks. TCP/IP is often referred to as the "language" of the internet because it governs how data is transmitted and received across the global network [13][14].

TCP/IP is not a single protocol but a combination of multiple protocols that work together to enable reliable and efficient data communication. The most important components of the TCP/IP protocol suite include [13][14]:

1. Internet Protocol (IP):
    - IP is an unreliable and connectionless protocol, responsible for routing data packets between devices across different networks. These packets are called datagrams, and its format is presented in Figure 2.5. Because the header's total length field is 16 bits long ($2^{16} - 1 = 65535$), the total length of the IP datagram is limited to 65535 bytes, of which 20 to 60 bytes are the header and the rest is data.
    - It assigns unique IP addresses to each device on the network, allowing routers to determine where to send data packets.
    - IPv4 (Internet Protocol version 4) and IPv6 (Internet Protocol version 6) are the two main versions of IP in use today. They vary the format of the datagram and how it needs to be interpreted.
2. Transmission Control Protocol (TCP):
    - TCP is responsible for ensuring reliable data delivery between two devices on a network.
    - It breaks data into packets, numbers them for sequencing, and includes error-checking information.

- TCP establishes and manages connections between devices, guarantees that data is delivered in the correct order, and retransmits lost or corrupted packets.
- It is a connection-oriented protocol, meaning it sets up a connection before data transfer and closes the connection after data exchange is complete.
- A TCP packet is called a segment and it consists of a header of 20 to 60 bytes, followed by data from the application program. The format of a segment is shown if Figure 2.6. The segment is encapsulated in a datagram for transmission as visible in Figure 2.7.

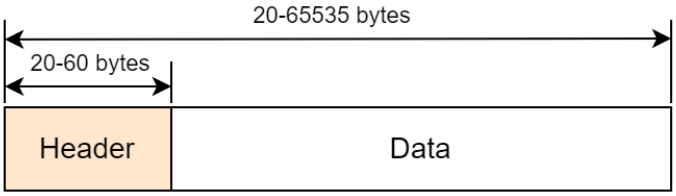


Figure 2.5: IP datagram format. Adapted from [13].

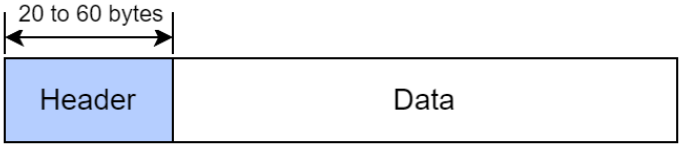


Figure 2.6: Format of a TCP segment. Adapted from [13].

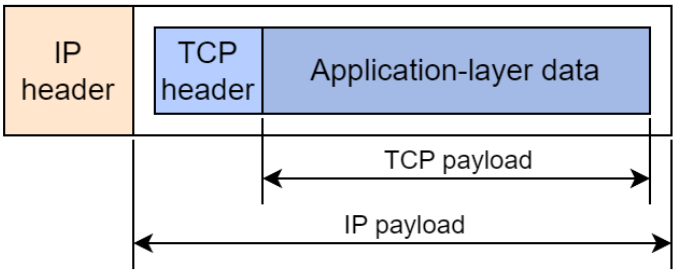


Figure 2.7: Encapsulation of a TCP segment in an IP datagram. Adapted from [13].

Sockets

A socket is a software endpoint that represents one end of a network connection. It serves as interface between an application and the network, allowing applications to communicate with each other over the Internet. Sockets come in two main types: TCP sockets, known as stream sockets, and User Datagram Protocol (UDP) sockets, also called datagram sockets. The latter, while being faster and simpler, do not provide a reliable and connection-oriented communication channel, therefore they are unable to guarantee correct data delivery and order preservation. TCP sockets are the logical choice for applications that require data integrity and order, such as file transfers [13]. TCP/IP based sockets are used in the client/server model of communication. One application, the client, requests a service, and another application,

the server, provides it [15]. A simplified diagram of the aforementioned model is shown in Figure 2.8.

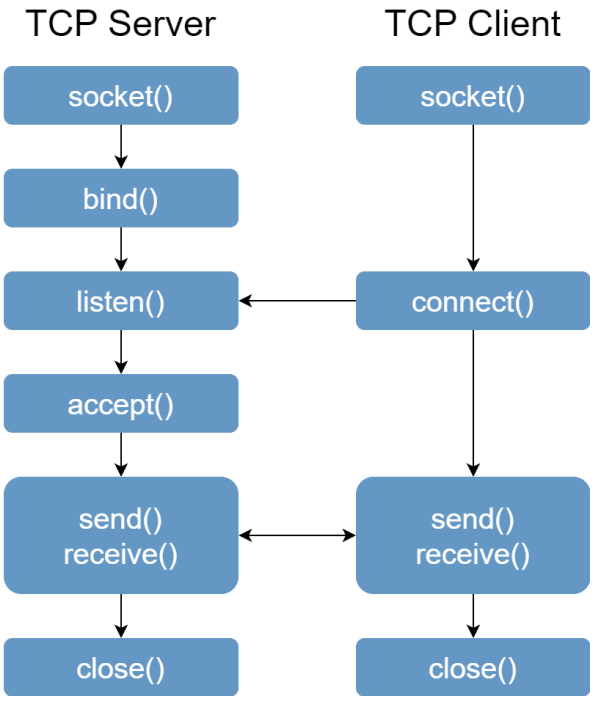


Figure 2.8: Simplified diagram of TCP/IP socket communication. Adapted from [13].

The stages for the TCP/IP socket communication, as presented in Figure 2.8, can be described as [16]:

- socket(): Create a TCP socket.
- bind(): Assign an address and a port number to the socket (for servers).
- listen(): Allow connections from clients to be made to the specified port.
- connect(): Establish a connection with a server (for clients).
- accept(): Accept a client connection.
- send()/receive(): Exchange data between the server and the client.
- close(): Close the connection.

2.3 Common Alerting Protocol

The Common Alerting Protocol (CAP) is a digital format for exchanging emergency alerts and it allows a consistent alert message to be propagated over multiple communications channels. It is used in the United States of America by Federal Emergency Management Agency (FEMA), as part of the Integrated Public Alert and Warning System (IPAWS) to ensure compatibility with existing warning systems [17]. Some of the benefits of using CAP include the possibility of adding of multimedia or using multiple languages. The basic CAP standard was developed by the Organization for the Advancement of Structured Information Standards (OASIS) to ensure that alert messages are consistently structured and can be effectively disseminated across various communication systems and devices. CAP provides

a standardized way for various organizations and authorities to create, disseminate, and interpret alerts, which is crucial for emergency management, public safety, and communication during disasters and crises [18].

Each CAP Alert Message consists of an <alert> segment that provides information about the current message, such as its purpose, its source, its status, and an identifier. which may contain one or multiple <info> segments. The <alert> segment may contain one or more <info> segments, with these being used to describe events in terms of urgency, severity, certainty, while also providing textual descriptions of the event or instructions for appropriate response by the recipients of the message. Each <info> segment may include one or more <area> and/or <resource> segments. The first one, <area> segment, is used to describe a geographic area to which the <info> segment that contains it applies, with geospatial shapes being the preferred representations. The other segment, <resource>, provides an optional reference to additional information related to the <info> segment within which it appears, in the form of an image or audio file [18].

The primary anticipated use of the CAP Alert Message is as an Extensible Markup Language (XML) document, but the format should remain abstract enough to be adaptable to different coding schemes. An example in the form of a CAP XML message is presented in Figure 2.9 [18].

```
<?xml version = "1.0" encoding = "UTF-8"?>
<alert xmlns = "urn:oasis:names:tc:emergency:cap:1.2">
  <identifier>KSTO1055887203</identifier>
  <sender>KSTO@NWS.NOAA.GOV</sender>
  <sent>2003-06-17T14:57:00-07:00</sent>
  <status>Actual</status>
  <msgType>Alert</msgType>
  <scope>Public</scope>
  <info>
    <category>Met</category>
    <event>SEVERE THUNDERSTORM</event>
    <responseType>Shelter</responseType>
    <urgency>Immediate</urgency>
    <severity>Severe</severity>
    <certainty>Observed</certainty>
    <eventCode>
      <valueName>SAME</valueName>
      <value>SVR</value>
    </eventCode>
    <expires>2003-06-17T16:00:00-07:00</expires>
    <senderName>NATIONAL WEATHER SERVICE SACRAMENTO CA</senderName>
    <headline>SEVERE THUNDERSTORM WARNING</headline>
    <description> AT 254 PM PDT...NATIONAL WEATHER SERVICE DOPPLER RADAR INDICATED A SEVERE
THUNDERSTORM OVER SOUTH CENTRAL ALPINE COUNTY...OR ABOUT 18 MILES SOUTHEAST OF KIRKWOOD...MOVING
SOUTHWEST AT 5 MPH. HAIL...INTENSE RAIN AND STRONG DAMAGING WINDS ARE LIKELY WITH THIS
STORM.</description>
    <instruction>TAKE COVER IN A SUBSTANTIAL SHELTER UNTIL THE STORM PASSES.</instruction>
    <contact>BARUFFALDI/JUSKIE</contact>
    <area>
      <areaDesc>EXTREME NORTH CENTRAL TUOLUMNE COUNTY IN CALIFORNIA, EXTREME NORTHEASTERN
CALAVERAS COUNTY IN CALIFORNIA, SOUTHWESTERN ALPINE COUNTY IN CALIFORNIA</areaDesc>
      <polygon>38.47,-120.14 38.34,-119.95 38.52,-119.74 38.62,-119.89 38.47,-120.14</polygon>
      <geocode>
        <valueName>SAME</valueName>
        <value>006109</value>
      </geocode>
      <geocode>
        <valueName>SAME</valueName>
        <value>006009</value>
      </geocode>
      <geocode>
        <valueName>SAME</valueName>
        <value>006003</value>
      </geocode>
    </area>
  </info>
</alert>
```

Figure 2.9: Example in the form of a CAP XML message [18].

CHAPTER 3

Project Design and Implementation

This chapter details how the audio switching alert system was designed and implemented. It provides some information on hardware and software choices.

3.1 Proposed Solution

Before starting to design a solution, it was necessary to figure out where it was going to be placed within the Radio Frequency (RF) transmission chain of a radio station. Figure 3.1 illustrates a typical transmission chain.

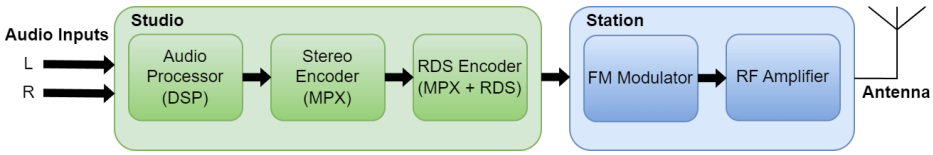


Figure 3.1: Typical FM radio transmission chain.

Since the objective of this solution was to use audio switching, it was decided that the alert system, henceforth also referred to as *FireTec Box*, would be placed, as shown in Figure 3.2, between the Audio Inputs coming from the broadcast source and the block named Studio, which processes the audio signals.

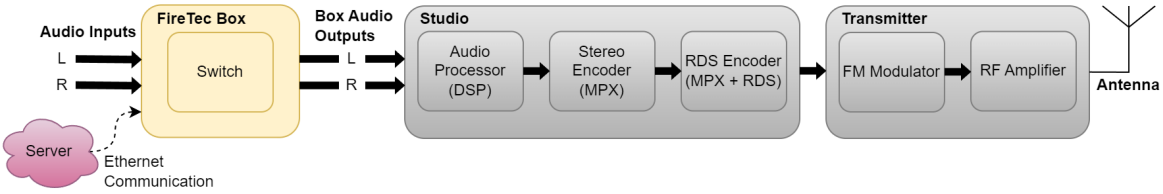


Figure 3.2: Switching system (*FireTec Box*) placement in the FM radio transmission chain.

Taking into consideration the requirements of the project, in which this alert system is part of, it was necessary to incorporate an Internet connection in the *FireTec Box*, in order to receive data coming from a server. This data would then need to be processed by a

microcontroller, creating an audio file with an alert message, to be stored and then accessed and played. Other features needed were two audio inputs, to route the Left and Right audio channels, originated from the radio station studio program, to the box. These audio signals would then pass through analog switches, in order to choose which audio source would be available at the output of the box, which was made up of two audio outputs, one for each channel as well. Lastly, a power supply would be required to provide electrical current to all the active hardware components.

3.2 Hardware

In this section, some of the considerations taken into account when choosing the various hardware parts will be presented, as well as how they all interconnect to build the physical part of the *FireTec Box* alert system.

3.2.1 Microcontroller and Ethernet Connection

To act as the processing piece of the solution, a microcontroller was considered the best option due to it being relatively easy to program and upload code into. The choice fell on the Arduino MKR Zero board, displayed in Figure 3.3, as it was developed with the purpose of allowing audio files to be played through a Digital-to-Analog Converter (DAC), and therefore without the need for extra hardware. It has an on-board Secure Digital (SD) connector with dedicated Serial Peripheral Interface (SPI) interfaces. A microSD card has been used to provide storage space for files. The board is powered by low voltages.

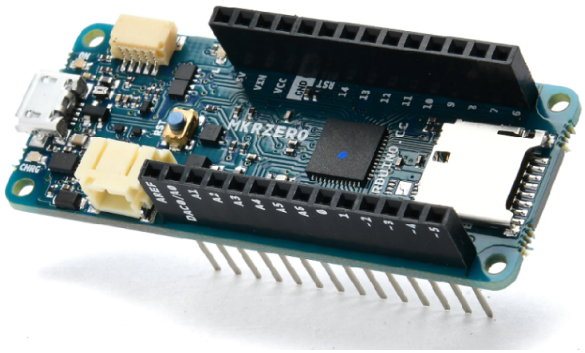


Figure 3.3: Arduino MKR Zero board.

However, this board does not include Ethernet connection. In order to establish an Internet connection, a shield was necessary. The WIZnet W6100 (MKR-)Ethernet Shield, which is pictured in Figure 3.4, was selected over the Arduino MKR ETH Shield because it comes with the more recent WIZnet 6100 chip, whereas the MKR ETH Shield comes with the WIZnet W5500 chip. W6100 suits users in need of stable Internet connectivity, using a single chip to implement TCP/IP communications, and incorporating an SPI interface. The shield has a RJ45 plug to connect an Ethernet cable [19].

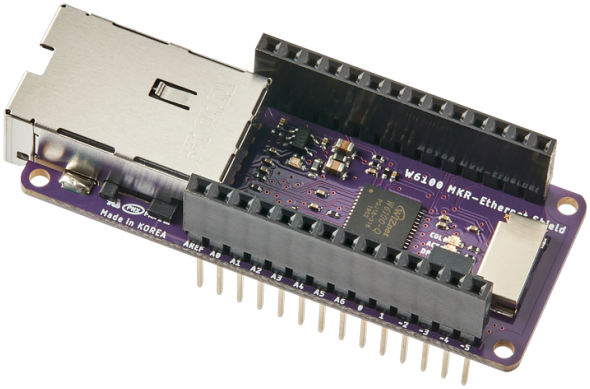
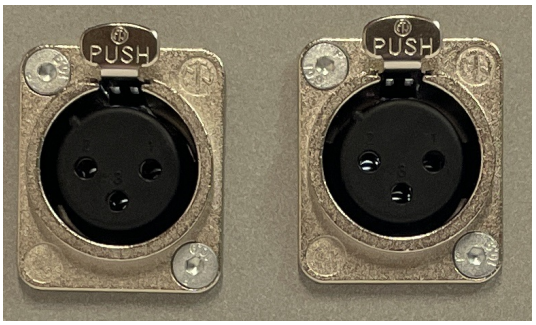


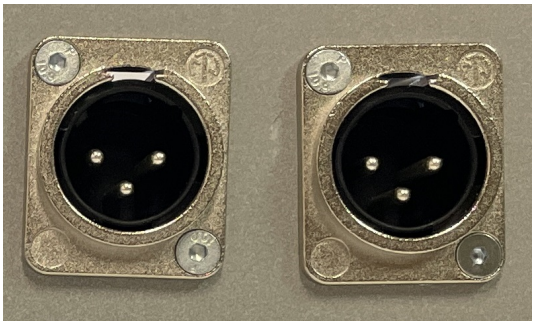
Figure 3.4: WIZnet W6100 (MKR-)Ethernet shield.

3.2.2 Audio Inputs and Outputs

Looking into professional audio processors [20][21], and audio interfaces [22][23], one common denominator was the existence of balanced inputs for analog audio, through XLR female connectors. With this in mind, it was decided to use XLR connectors for the *FireTec Box* audio inputs and outputs, in order to maintain the balanced characteristics, since this box is connected to an audio processor, as previously seen in Figure 3.2. Thus, for the two *FireTec Box* audio inputs, one for the right channel and one for the left channel, two 3-pin female XLR connectors were chosen, as seen in Figure 3.5a. Similarly, for the two *FireTec Box* audio outputs, two 3-pin male XLR connectors were chosen, which are shown in Figure 3.5b.



(a) XLR female (input) sockets.



(b) XLR male (output) sockets.

Figure 3.5: XLR connectors.

3.2.3 Unbalanced to Balanced Conversion

The DRV134 is a differential output amplifier that converts a single-ended input to a balanced output pair. This balanced audio driver consists of high performance op amps with on-chip precision resistors for optimum output common-mode rejection. It is fully specified for high performance audio applications, since it has low distortion of 0.0005% at f = 1 kHz and high slew rate of 15 V/μs [24].

Since the box has two outputs, one for each channel, a module, that is presented in Figure 3.6, was used that comes equipped with two DRV134 amplifiers, as well as other high-quality passive components such as a 7805 voltage regulator, capacitors and resistors.

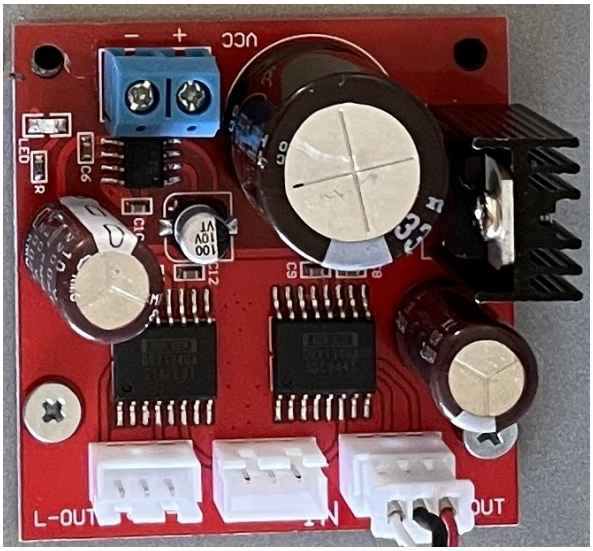


Figure 3.6: DRV134 unbalanced to balanced audio converter module.

It has a 3-pin input connector on the center (IN), with one pin for Ground, and the other two pins for the Unbalanced audio of each channel. The module has two 3-pin output connectors (L-OUT and R-OUT), one for each channel, side by side with the input one. Each connector carries Balanced audio. Each channel output has one pin for Ground, one pin for the positive (+) signal and one for the negative (-) signal. The module is powered by a DC supply voltage of +12 V. An illustrative representation of the inputs and outputs is shown in Figure 3.7, to better understanding of their arrangements.

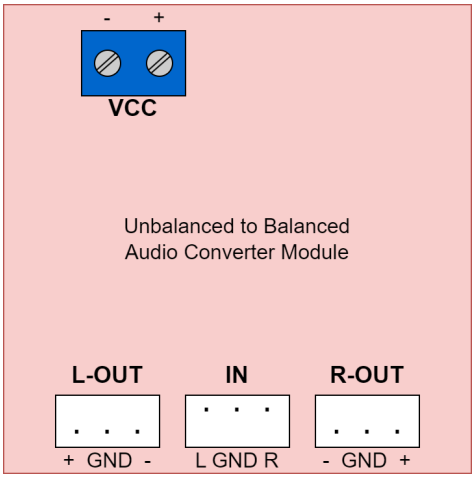
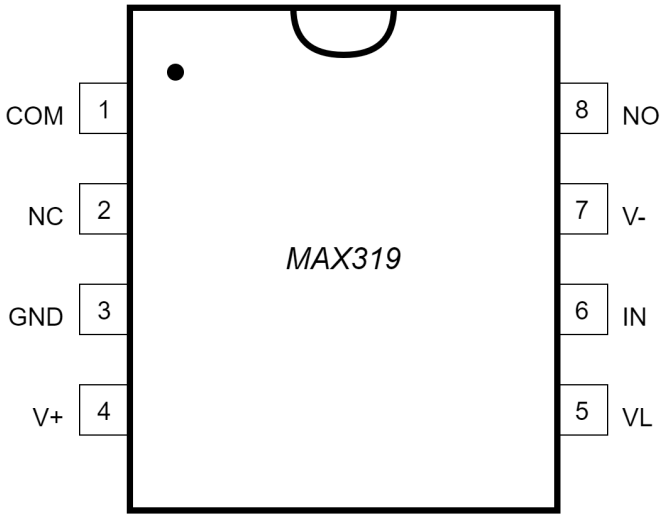


Figure 3.7: Schematic with the inputs and outputs of the DRV134 module.

3.2.4 CMOS Switch and Voltage Regulation PCB

In order to be able to choose which audio source will be provided at the output of the box, it is necessary to use elements that allow this choice to be made. Remembering that we are dealing with analog audio, we chose Maxim's MAX319 [25] Complementary Metal–Oxide–Semiconductor (CMOS) analog switch. This switch is Single-Pole Double-Throw (SPDT), and therefore has one output and two inputs. Figure 3.8 shows the pin

configuration and the truth table for the MAX319 analog switch. The Normally Closed (NC) input is the one which is copied and appears on the output (COM) by default, i.e. when the logic level at the selection input (IN) is 0, thereby at voltage level corresponds to 0V. If the logic level at IN changes to 1, which in this case will correspond to 3.3V, then, the voltage that Arduino places on a digital pin set to logic level 1, the switch output will change to what is on the Normally Open (NO) input. The MAX319 will operate with bipolar supply voltages of +12V at V+ pin and -12V at V- pin. The reference logic voltage (VL) input is connected to 5V. In total, the *FireTec Box* uses four MAX319 switches, due to the fact that each balanced audio channel uses two switches, one for the hot audio signal and one for the cold audio signal.



MAX319		
LOGIC (IN)	NC	NO
0	ON	OFF
1	OFF	ON

Figure 3.8: Pin configuration (top) and truth table (bottom) for the MAX319 switch. Adapted [25].

As a mean to forward the power needed for the active hardware components of the *FireTec Box*, such as the Arduino MKR Zero with the Ethernet Shield, the Unbalanced to Balanced converter module and the CMOS switches, a Printed Circuit Board (PCB) was designed and assembled. This PCB includes the MAX319 switches, a 5V linear regulator which is the L7805CV, and several sockets for wire connections. A schematic of the PCB circuit is shown in Figure 3.9. Several sets of sockets are identified to facilitate analysis, and they are used to connect to the other hardware parts. One connection to supply the Arduino with 5V (Arduino supply), which come from the output of the linear regulator, L7805CV. One connection to Arduino to receive the logic level that will select the state of the four MAX319, and another to receive the analog audio alert message from the Arduino DAC, which were grouped (Audio + Control from Arduino). The analog audio is forwarded to the DRV134 module left and right unbalanced inputs (IN of DRV134). There is also a connection to receive

the left channel of the balanced converted audio (L-OUT from DRV134), and another one to receive the right channel of the balanced converted audio (R-OUT from DRV134). The XLR female connections for the left (LEFT BOX INPUT) and right (RIGHT BOX INPUT) *FireTec Box* input channels are connected to the Normally Closed inputs of the 4 MAX319. On the other hand, the Normally Open inputs receive the balanced signals from the DRV134. Lastly, the XLR output sockets of the box are connected to outputs of the MAX319 switches, for the left channel (LEFT BOX OUTPUT) and the right channel (RIGHT BOX OUTPUT).

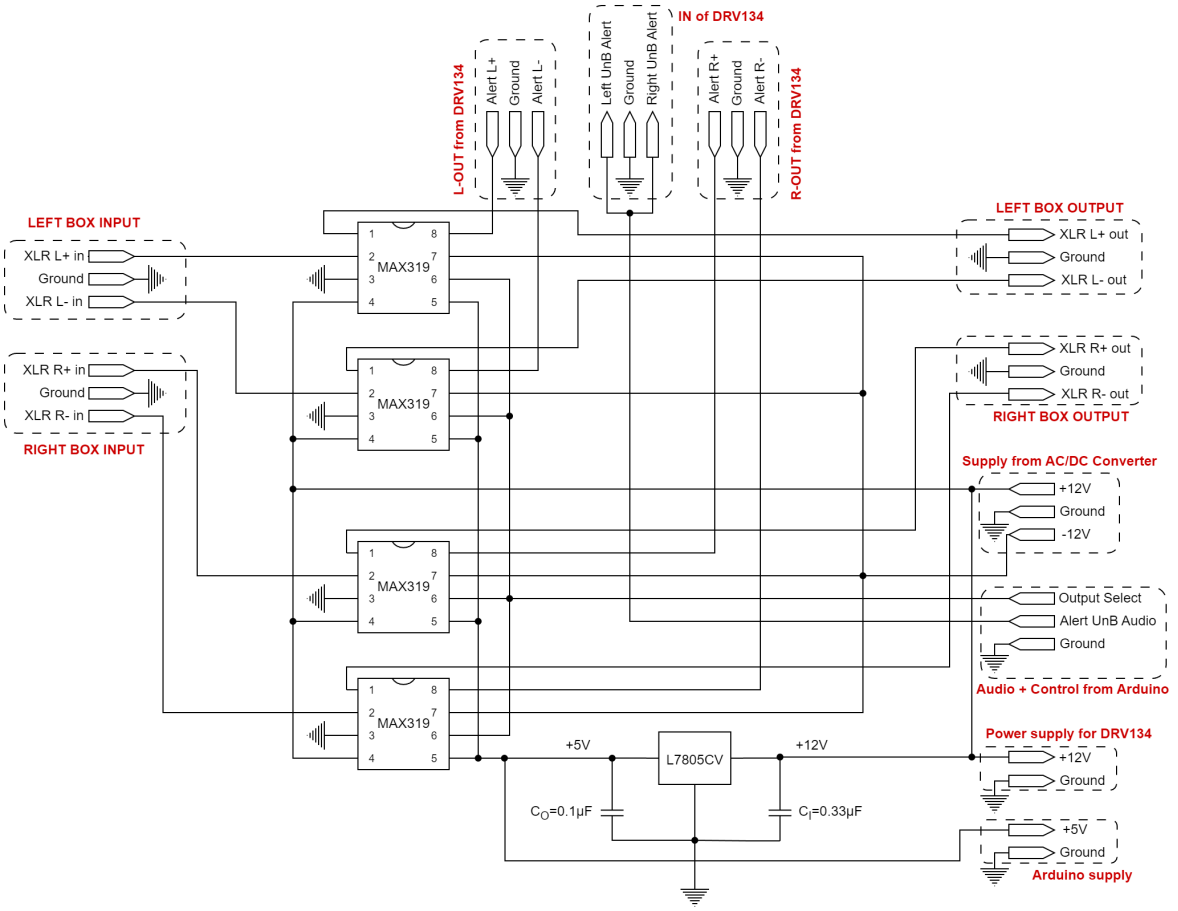


Figure 3.9: CMOS switch and voltage regulation PCB circuit schematic.

After the design of the schematic, the next step was to create a Printed Circuit Board (PCB) layout for the circuit. This layout is presented in Figure 3.10.

A final version of the PCB, with the components already assembled to it, is pictured in Figure 3.11.

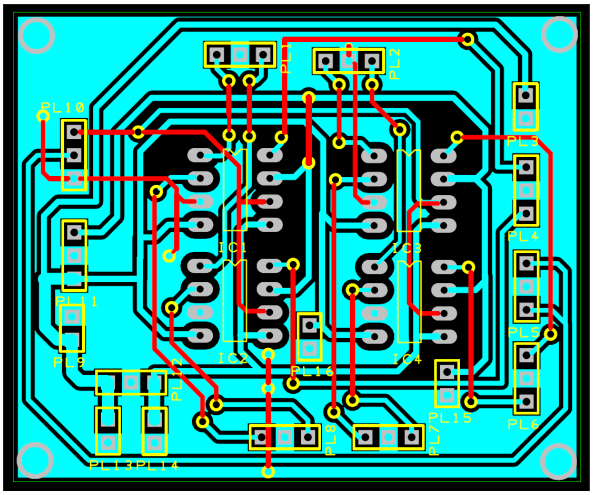


Figure 3.10: CMOS switch and voltage regulation PCB layout.

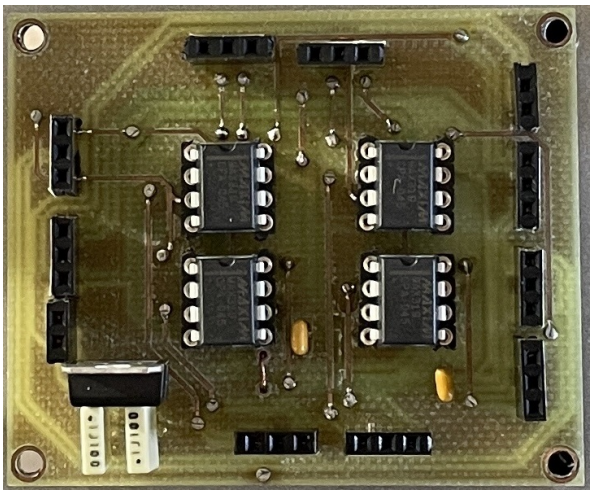
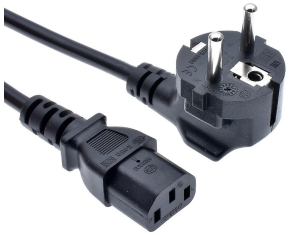


Figure 3.11: CMOS switch and voltage regulation assembled PCB.

3.2.5 Power Supply

Connector

In order to use a common 3-pin power cable, technically known as IEC C13 power cable, as shown in Figure 3.12a, that would connect the box into a power socket, a male connector is needed on the box side. Therefore, a panel mount IEC C14 male connector was selected, as seen in Figure 3.12b. The male connector is also equipped with a protection fuse in order to protect the hardware components inside the *FireTec Box* from power surges or short circuits that could damage them.



(a) C13 female cable.

(b) C14 Panel Mount IEC Male Connector.

Figure 3.12: Cable and respective connector used to plug the box to an electrical socket.

AC/DC Power Supply

Inside the box, an AC/DC converter was placed in order to transform the Alternating Current (AC) into Direct Current (DC), for powering the active hardware elements of the *FireTec Box*. The chosen converter, which is displayed in Figure 3.13, was the XP Power ECL15UD01-T, which has two input pins that receive 220 V AC. The converter has three outputs that are connected to the PCB, and they distribute +12 V and -12 V DC, with the third one being the Ground connection which will serve as reference. All Grounds inside the *FireTec Box* are connected to that pin.

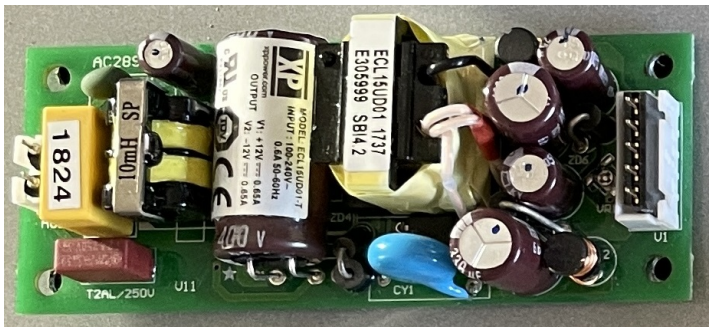


Figure 3.13: ECL15UD01-T open frame chassis mount AC-DC power supply.

3.2.6 Integrated Solution

Having presented the various pieces of hardware that make up the alert system, it is necessary to understand how they relate to each other and how they are interconnected. Figure 3.14 shows the *FireTec Box* block diagram, including the connections between each block and the inputs and outputs, and the voltages used where applicable.

Compared to the previous subsections, the new features introduced here are an ON/OFF switch, so that the *FireTec Box* can be switched off even if the power cable is plugged into the socket, and an informative green LED that indicates whether or not the box is being powered, which is connected to the 3.3 V output pin of the Arduino board.

Figure 3.15 shows a photograph of the *FireTec Box* from above, where the various elements are identified making easier a comparison with the diagram in Figure 3.14.

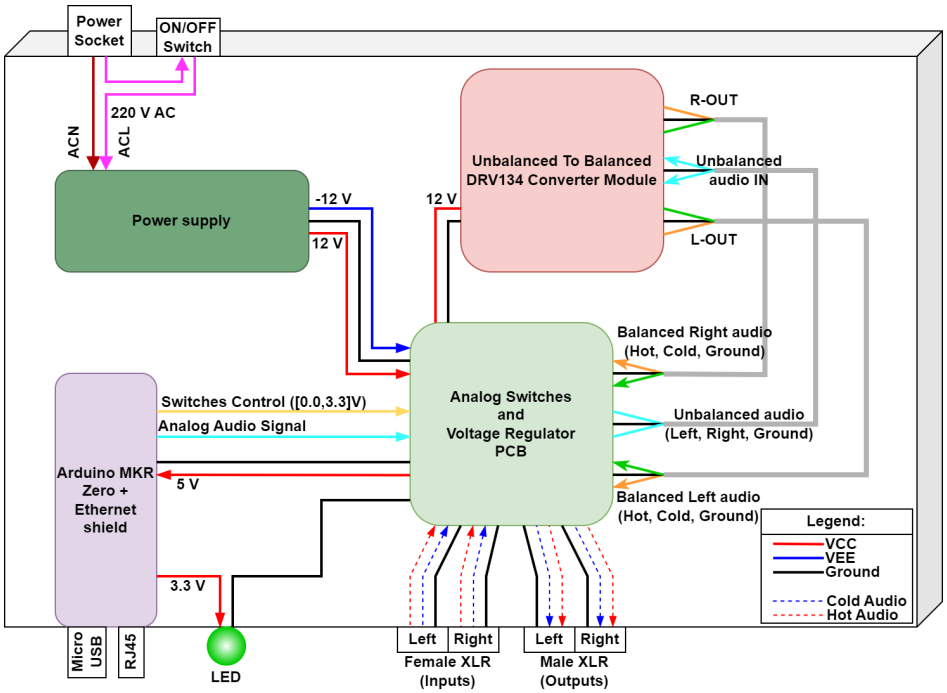


Figure 3.14: Block diagram of the alert system displaying electrical connections, inputs and outputs.

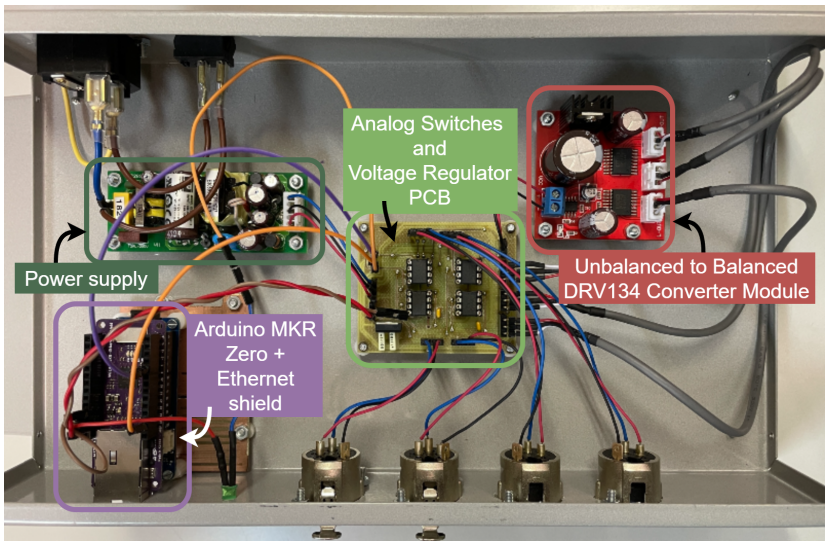


Figure 3.15: Top view of the inside of the alert system box.

Finally, Figure 3.16 shows the front panel view of the *FireTec Box*: 1 - RJ45 plug; 2 - microUSB plug; 3 - LED; 4 - female XLR sockets; 5 - male XLR sockets. And figure 3.17 shows the rear panel view of the *FireTec Box*: 1 - ON/OFF switch; 2 - power socket.

Figure 3.16: Front panel of the *FireTec Box*.

Figure 3.17: Rear panel of the *FireTec Box*.

3.3 Software

This section describes the software uploaded to the Arduino MKR Zero. It also details the different communication protocols implemented in that same software.

3.3.1 Firmware in the Arduino

As previously stated, a fundamental goal of our solution is its autonomy, i.e., no human interaction is needed. To achieve this goal, a piece of code is uploaded to the Arduino and stored in its non-volatile memory. This process guarantees that this piece of code, often referred to as firmware, is saved even when the hardware is turned off or loses its power source [26]. The code was developed, compiled, and uploaded using the Arduino IDE application.

Figure 3.18 depicts how the firmware is designed, showcasing the several steps it goes through. When the Arduino first powers on, the program initializes the configurations required for establishing SPI communications between the Arduino and the microSD card and between the Arduino and the Ethernet shield, as well as the Arduino General-Purpose Input/Output (GPIO) pin configurations needed. Then, a client socket instance is created with the IP address of the server and a port number as parameters. This socket is of the TCP/IP type because it will receive important information that requires all data to be read without errors and in the correct order. If the message gets corrupted, it can lead to the absence of the alert broadcast. Next, the program stays in a waiting loop, until the server accepts a connection. Once the connection is successful, the client starts reading the bytes sent by the server. In order for the correct processing of the incoming data, it is identified which communication protocol is being used to structure the message. There are two possible options, the FireTec

Communication Protocol (FCP) and the Common Alerting Protocol (CAP), which will be further detailed in the next subsection. Regardless of the protocol in use, the next step is isolating the audio message data and save it in the SD card, as a Waveform Audio File Format (WAV) file format. Upon completion of all data transfer, the client terminates the connection to the server. The following step consists of temporarily changing a digital pin of the Arduino board, used for control, from 0V to 3.3V, which in terms of logic level consists of changing from 0 to 1, or *LOW* to *HIGH*, respectively. This setting makes all the MAX319 analog switches output what is on their *Normally Open (NO)* input. This input, NO, will receive the analog alert audio message, in balanced form, coming from the DRV134 module. The audio message is played from the microSD card through the Arduino's integrated DAC to the aforementioned module, therefore making the system output the alert message. After the alert message is finished playing, the Arduino's digital pin for control, previously set as 1, will now be returned to its default *LOW* state, i.e. 0V, switching the output of the system back to the FM station signal which is present in the *Normally Closed* input of the MAX319 switches. This marks the end of the cycle and the program is returned to the initial state of creating a TCP client socket and waiting in a loop for the server's availability to accept a new connection.

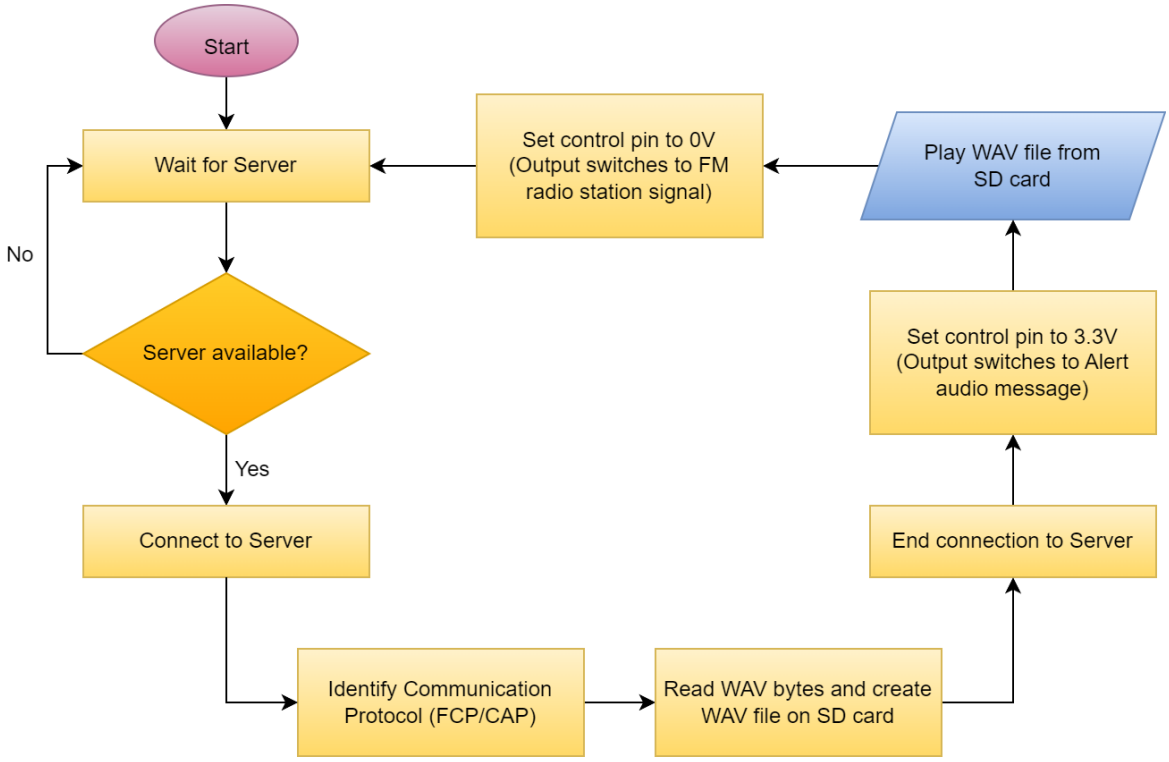


Figure 3.18: Flowchart of the firmware installed in the Arduino.

3.3.2 Communication Protocols

In order to guarantee the correct reception of the data, composed of the audio bytes of the alert message and some other information, two protocols were taken into account. The first one, which was created specifically for this project and named FCP, focuses on ease

of processing, with the data bytes coming right at the beginning. The second protocol was designed to take advantage of the already existing CAP therefore making it more reliable to interact with governmental organizations, such as the MAI and the National Emergency Service (ANEPC). In this case, the audio bytes are inserted between two identifying tags so the Arduino can decode that part of information and generate the WAV file. In both cases the WAV file is stored in a microSD card and, as soon as that is completed, the audio of the alert can start being broadcast.

FireTec Communication Protocol

The main advantage of this protocol, which was created for this project, is its simplicity and how quickly it is processed. The frame format for the FireTec Communication Protocol is portrayed in Figure 3.19, and consists of a variable number of bytes, depending on the duration of the audio message, followed by some data that is discarded in the context of the system presented in this dissertation. The protocol was designed to be suitable for two different systems, with each one of these retaining only the necessary data for them. The software detects when the WAV message data is over, reading the data size on the bytes 40-43 of the WAV file [27]. Upon reaching the end, it stops the reading, moving on to the next steps, which were previously detailed.

Figure 3.19: Structure of the FCP data frame.

Common Alerting Protocol

Despite the fact that the FCP protocol is very simple and fast, it does not provide a versatile solution to include external entities in the chain of communication and authorization. To solve this, it was decided to make use of the Common Alerting Protocol to implement a digital message format that can be used to communicate with the MAI in order to get official governmental authorization, and therefore command the FM stations broadcasts. An example of the key fields, present in a CAP message that was generated by the server, is displayed in Figure 3.20. It is noticeable that the WAV bytes, the key data for this algorithm, are part of a <resource> segment with the description *Audio Message* which is itself inside an <info> segment that describes the event in terms of its type, urgency, severity as well as informing who the sender is along with instructions. Only the audio data is saved by the program, the rest being discarded, similar to the FCP case.

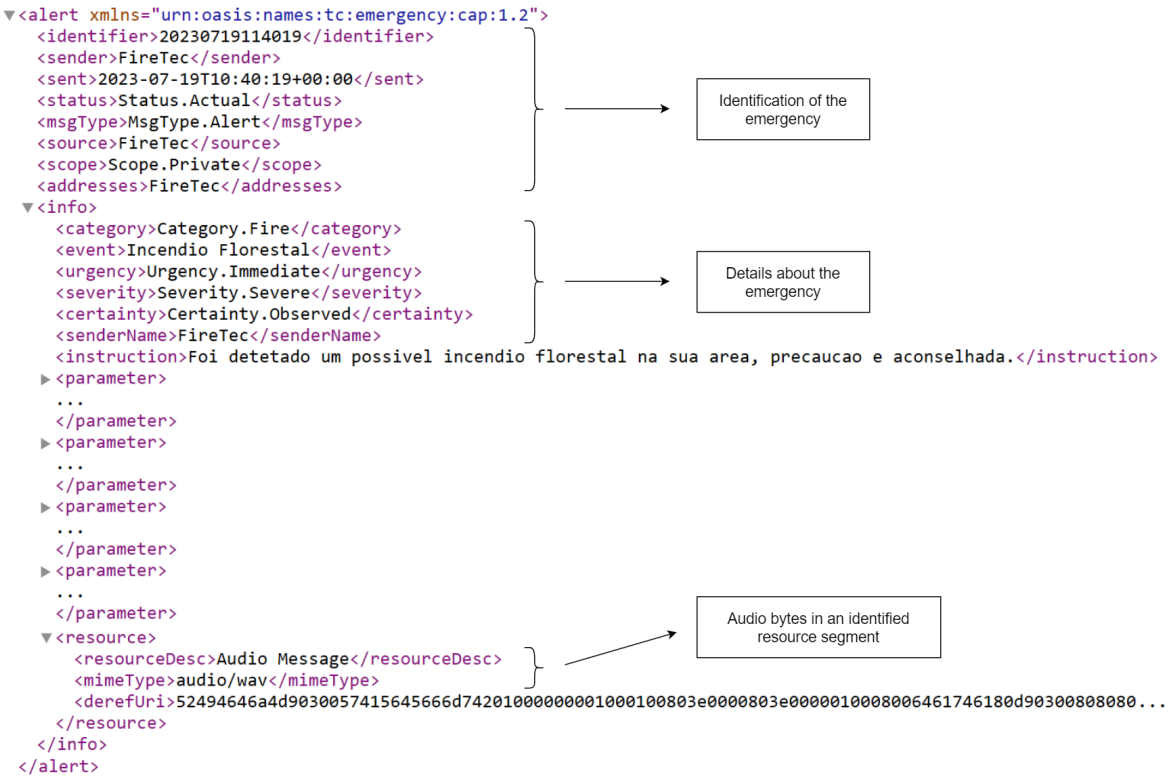


Figure 3.20: Example of a CAP XML message containing the alert audio data.

CHAPTER 4

Results

In this chapter, some graphical results are presented, to assess that the hardware is working as expected. Some values from software profiling are analyzed to evaluate its performance, and some audio quality checks are made.

4.1 Unbalanced to Balanced Converter and Analog Switches

DRV134 Unbalanced to Balanced Converter Module

As seen in subsection 3.2.4, the designed PCB also serves to bypass the supply voltages to the Arduino (MKR Zero board plus the Ethernet Shield) and the DRV134 module. The latter is supplied with 12 V from the power supply. A measurement was made, using an oscilloscope, to confirm that this was indeed the voltage being supplied to the DRV134 module. The result of this test is shown in Figure 4.1, where 11.9 V was measured, which is very close to the reference value of 12 V.

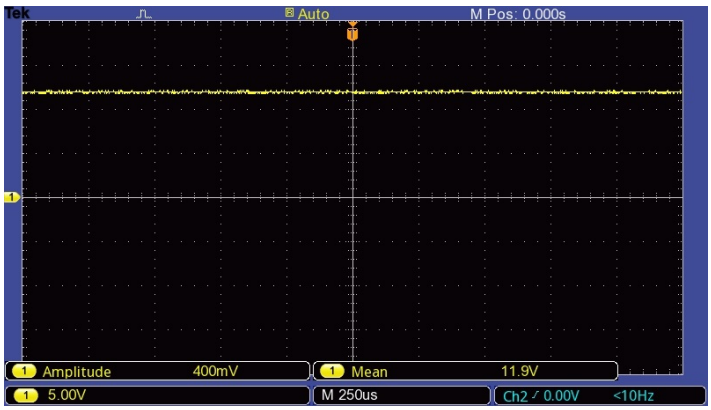


Figure 4.1: Oscilloscope screenshot showing the voltage coming from the power supply.

Next, some results are presented regarding the module's operation in terms of its ability to generate two symmetrical signals from a single one, as well as the relationship between the module's input and output. Figure 4.2, where one of the module's conversion channels

was tested, shows the presence of the input signal (yellow) and the corresponding balanced positive, or hot, output signal (blue).

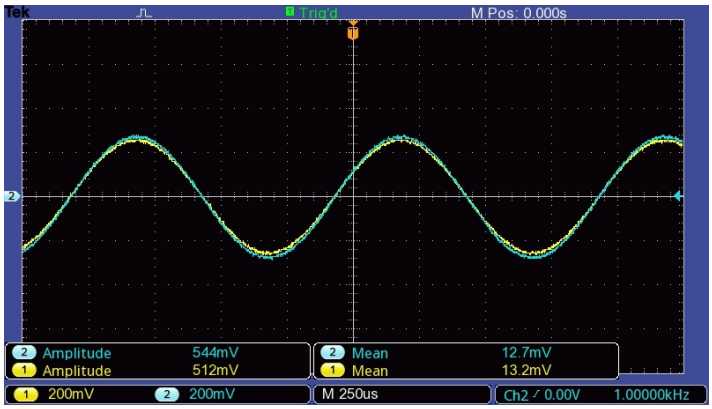
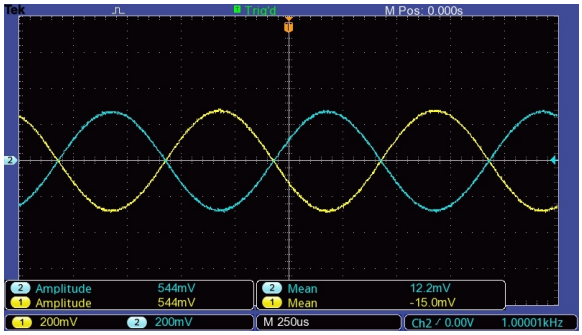
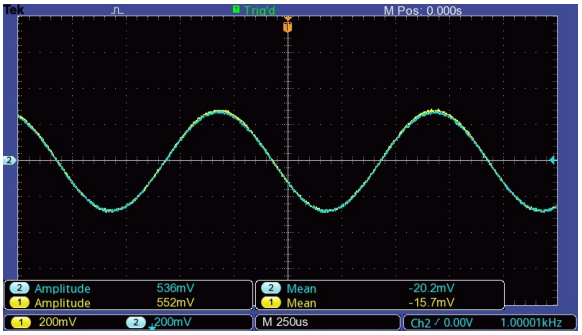


Figure 4.2: Oscilloscope screenshot showing the input (yellow) and corresponding hot output (blue) of the DRV134 converter.

Keeping the same scale on both oscilloscope channels to check for attenuation, it was proved that the output was identical to the input, even showing a subtle increase in amplitude. Note that the input signal was maintained in all the DRV134 module tests. Figure 4.3a shows the two outputs of the converted signal, i.e. the balanced signal, where it can be seen that the signals are, in fact, out of phase, i.e. the phase of the negative balanced signal (yellow) differs from the phase of the positive balanced signal (blue) by 180°. In Figure 4.3b, the positive signal (blue) was inverted, just to check that it would coincide with the negative signal (yellow), and it did.



(a) Screenshot of cold balanced signal (yellow) and hot balanced signal (blue).



(b) Screenshot of cold balanced signal (yellow) and inverted hot balanced signal (blue).

Figure 4.3: Oscilloscope screenshots showing the balanced signals converted by the DRV134 module.

MAX319 CMOS Switches

Another test was to check the output of the L7805CV linear regulator, which feeds the Arduino board and the VL inputs of each MAX319. As shown in figure 4.4, the output of the regulator was indeed 5 V as expected.

It was also necessary to evaluate the operation of the MAX319, namely whether the desired signal was obtained at the output, according to the control logic level. A signal with

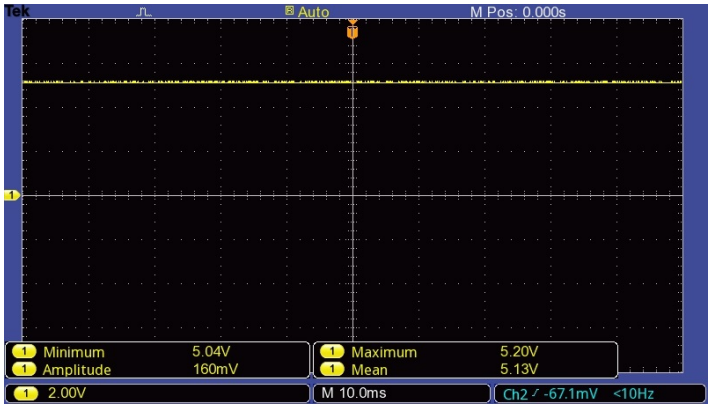


Figure 4.4: Oscilloscope screenshot showing the output of the L7805CV regulator.

a frequency of 2 kHz (yellow) was placed on the Normally Closed (NC) input, and a signal with a frequency of 1 kHz (blue) was placed on the Normally Open (NO) input, in order to distinguish which one would appear on the output (COM). This is shown in Figure 4.5.

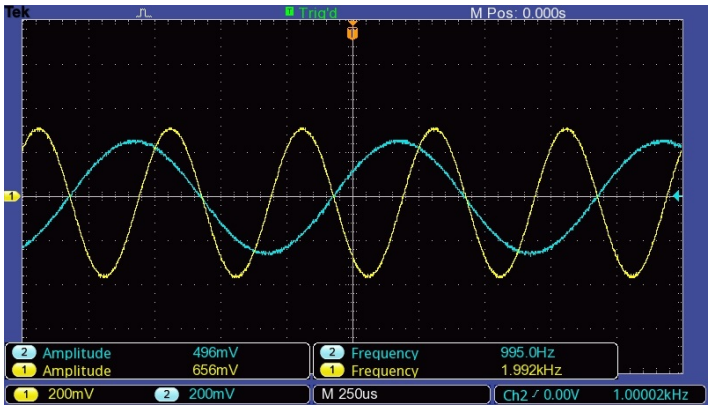


Figure 4.5: MAX319 input signals: 2 kHz (yellow) on NC, 1 kHz (blue) on NO.

Recalling the truth table shown in Figure 3.8, the logic level on the MAX319's IN pin was varied between *LOW*, or 0 (0 V), and *HIGH*, or 1 (3.3 V).

Looking at Figure 4.6, inputting a *LOW* logic level on IN, corresponding to 0 V (yellow), we see that the output, COM, shows the NC signal, which has a frequency of 2 kHz (blue).

In turn, Figure 4.7 shows that for the logic level *HIGH* on IN, corresponding to 3.3 V (yellow), the NO signal, which has a frequency of 1kHz (blue), appears on the output, COM.

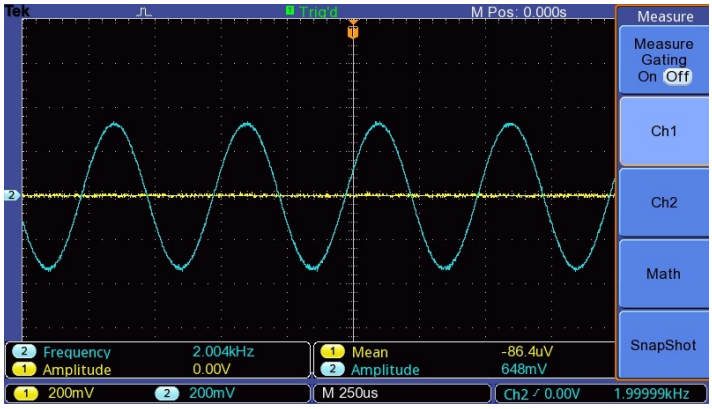


Figure 4.6: Oscilloscope screenshot showing the IN logic level, *LOW* (yellow) and the COM signal (blue).

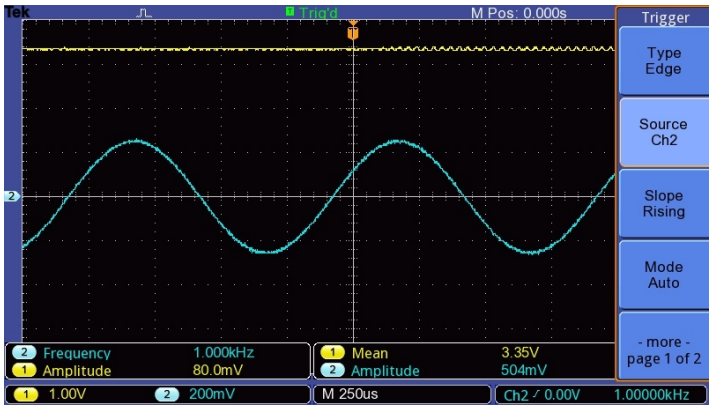


Figure 4.7: Oscilloscope screenshot showing the IN logic level, *HIGH* (yellow) and the COM signal (blue).

4.2 Code Profiling

To analyze and measure the behavior and performance of the firmware, a technique named profiling was used, by including a few lines of code in it. Essentially, time profiling and memory usage profiling were the two types evaluated. The results are presented in Table 4.1.

Measurement	**Protocol**	
	FCP	CAP
Elapsed time (s)	14.85	30.48
Flash Memory Used	12%	12%
Dynamic Memory Used	8%	9%

Table 4.1: Measurements made by the profiling code.

The elapsed time was measured from the moment the connection to the server was established, until the moment the alert message started playing. The difference between the two protocols, CAP taking double the time of the FCP, is that the data size of the former is twice the size of the latter. The Flash Memory used is the same for both protocols, 12% of the total. The Dynamic Memory used by global variables is only 8% with FCP and 9% with CAP, with the rest of the memory available for local variables.

4.3 Output Audio Quality

To analyze the quality of the audio that the *FireTec Box* outputs, the same audio track was recorded twice, first directly from the audio source, and then from the output of the *FireTec Box*. Figure 4.8 shows the audio wave of the recording obtained directly from the audio source. Figure 4.9 shows the audio wave of the recording from the box output. It is visible how identical they are. Thus, in Figure 4.10, the module of the difference between the previous two waves is presented, showing that the amplitude is very low. Therefore, it is possible to conclude that the *FireTec Box* does not introduce noticeable distortion and that the sound quality is maintained even when it goes through the system.

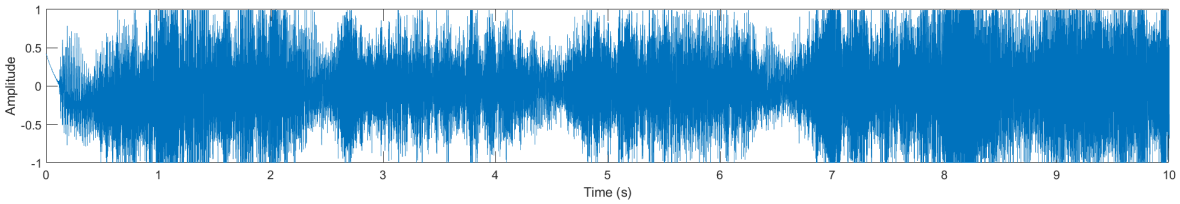


Figure 4.8: Audio wave of the recording from the audio source directly.

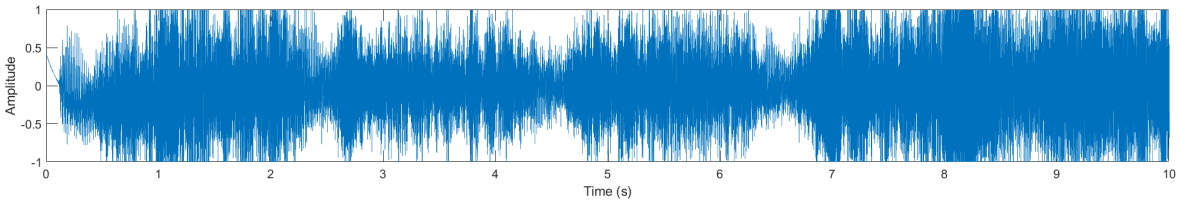


Figure 4.9: Audio wave of the recording from the output of the FireTec Box.

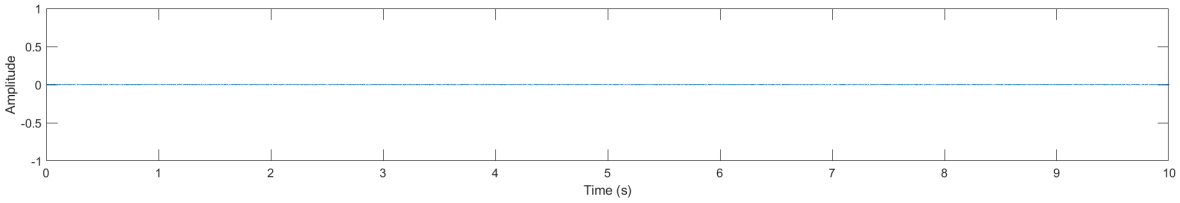


Figure 4.10: Audio wave of the module resulting from the subtraction of the previous two waves.

4.4 Testing the Audio Using an FM Broadcast

In order to emulate an FM radio station broadcast, the setup shown in Figure 4.11 was assembled, following the connection order presented in Figure 3.2, back in chapter 3. A simpler diagram is now shown in Figure 4.12, to showcase how the devices were connected to run the tests.

The FM receiver, presented in Figure 4.13, was tuned to the frequency set on the transmitter, and used to listen to the broadcast. It was possible to attest the proper functioning of the switching feature implemented in the *FireTec Box*. The alert message was successfully received from the server and played, while being perfectly audible, and with all the instructions to

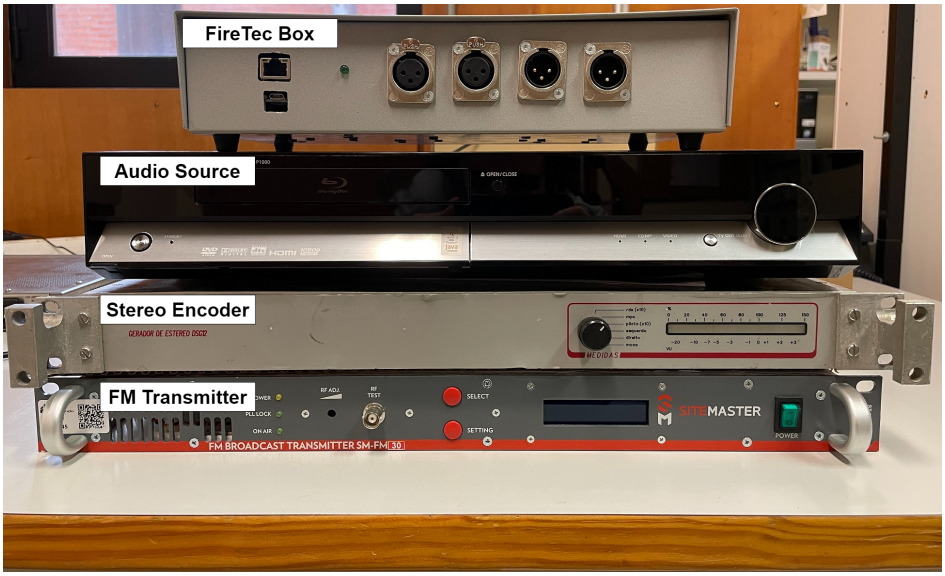


Figure 4.11: Devices used to assemble a setup to emulate an FM radio station.

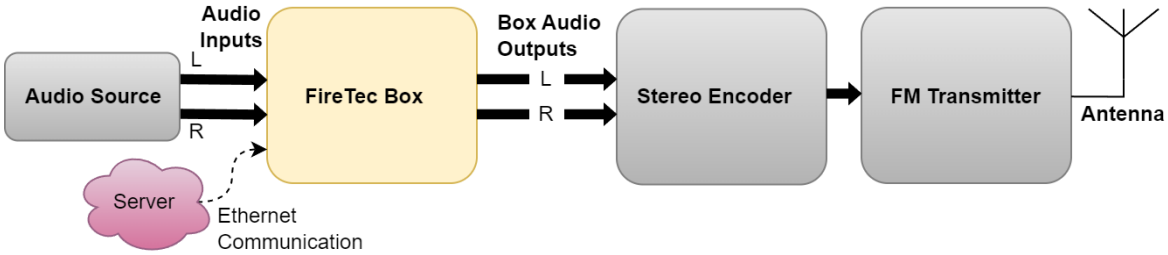


Figure 4.12: Block diagram of the connections of the setup assembled.

be followed being fully perceptible. After the alert message finished playing, the broadcast resumed to the audio playing from the Audio Source.

Figure 4.13: FM receiver used to listen to the broadcast.

CHAPTER 5

Conclusions and Future Work

5.1 Conclusions

This dissertation proposes an alert system for FM radio station. Its goal is to make use of audio switching to broadcast alert messages to warn drivers about which roads to avoid in the case of forest fires break out and block those roads. This may help prevent past catastrophes from happening again and avoid many deaths.

The alert system consists of a box which has: an Arduino microcontroller board, MKR Zero, that incorporates a DAC capable of playing audio files, connected to an Ethernet shield to allow Internet connection; an AC/DC power supply to convert 220 V AC to bipolar DC voltages of -12 V and +12 V that power the different hardware parts; an Unbalanced to Balanced audio converter module; a PCB with a voltage linear regulator and four analog switches, which are controlled to select which audio source is sent to the output of the box, if either the radio station's normal live broadcast or the alert message. The box has two XLR inputs and two XLR outputs to allow the transmission of balanced audio channels. The Internet connection allows the system to receive data from a server, which includes a custom audio message for each specific fire location. The data reception from the server, the audio playing through the DAC and the switch control are made possible by a firmware installed in the microcontroller.

This system was developed to be used in radio stations that do not use Radio Data System in their broadcasts. If the FM station is isolated without overlapping coverage with others, this solution can be used, even with RDS.

Finally, this dissertation has contributed to the writing of the following scientific papers:

- "Transceiver System for Audio Alarming Using Radio FM Stations", written by M. Coelho, L. Santiago, A. Navarro, N. B. Carvalho.
- "A Low-Cost Embedded System to Support Broadcasting Emergency Messages through FM Radio Stations", written by M. Coelho, L. Santiago, D. Araújo, A. Navarro, N. B. Carvalho, accepted for publication in the IEEE Embedded Systems Letters journal.

5.2 Future Work

Despite the fact that the system is functional, it would be important, for future work, to make some improvements. The PCB containing the analog switches can be redesigned to incorporate the DRV134 converter module and the XLR inputs and outputs, thus making the system more compact and with less wires that might pick up interference.

There is also room to improve the software efficiency, speed, and resource usage. This would help decreasing the amount of time it takes for an alert message to start playing. It can also be updated with a sleep mode feature to reduce the power consumption of the system, therefore making it more eco friendly . The software should also be able to, in the future, update the firmware over IP.

The system can be tested and adapted to make it usable in other audio systems, like events or clubs where a lot of people are gathered, and play alerts that would apply to those situations in specific.

References

[1] L. Villalobos. «Turismo bate novo recorde em portugal. agosto com máximos de hóspedes e dormidas». (2022), [Online]. Available: https://www.publico.pt/2022/09/30/economia/noticia/turismo-bate-novo-recorde-portugal-agosto-maximos-hospedes-dormidas-2022408 (visited on 10/16/2023).

[2] O. T. Cordeiro. «Em agosto o regresso dos emigrantes dá outra vida a aldeias do nordeste transmontano». (2016), [Online]. Available: https://www.brigantia.pt/noticia/em-agosto-o-regresso-dos-emigrantes-da-outra-vida-aldeias-do-nordeste-transmontano (visited on 10/17/2023).

[3] P. S. Luz. «Espite, a aldeia que volta a encher-se de gente (e de vida) em agosto». (2022), [Online]. Available: https://www.dn.pt/sociedade/espite-a-aldeia-que-volta-a-encher-se-de-gente-e-de-vida-em-agosto--15092184.html (visited on 10/17/2023).

[4] Reuters. «Explicador: Como as alterações climáticas provocam ondas de calor e incêndios florestais». (2022), [Online]. Available: https://www.publico.pt/2022/07/21/azul/noticia/explicador-alteracoes-climaticas-provocam-ondas-calor-incendios-florestais-2014474 (visited on 10/19/2023).

[5] DN/Lusa. «Fogo de pedrógão grande causado por descarga elétrica e o de góis por raio». (2017), [Online]. Available: https://www.dn.pt/portugal/incendios-fogo-de-pedrogao-grande-causado-por-descarga-eletrica-e-o-de-gois-por-raio-8837678.html#media-1 (visited on 10/16/2023).

[6] «Números negros de pedrógão: 66 mortos e mais de 250 feridos». (2017), [Online]. Available: https://www.jn.pt/nacional/interior/sintese-pedrogao-grande-sessenta-e-seis-mortos-mais-de-250-feridos-e-500-casas-destruidas-8991727.html/ (visited on 10/16/2023).

[7] Lusa. «Um terço da população de pobrais morreu a fugir do fogo». (2017), [Online]. Available: https://www.dn.pt/sociedade/pedrogao-grande-um-terco-da-populacao-de-pobrais-morreu-a-fugir-do-incendio-8574287.html (visited on 10/16/2023).

[8] A. LUSA. «Gnr. incêndio em pedrógão grande "surpreendeu todos" ao atingir en 236-1». (2017), [Online]. Available: https://observador.pt/2017/06/21/gnr-incendio-em-pedrogao-grande-surpreendeu-todos-ao-atingir-en-236-1/ (visited on 10/19/2023).

[9] M. A. Luís and M. Henriques, *Área de consignação de frequências e licenciamentos*, ANACOM presentation, 2021.

[10] T. Wilson. «Balanced vs unbalanced audio connections». (2023), [Online]. Available: https://www.headphonesty.com/2019/05/balanced-vs-unbalanced-audio-connections/ (visited on 10/19/2023).

[11] G. Tait. «Unbalanced versus balanced». (2020), [Online]. Available: https://www.pedalpointsound.com/music-studio/unbalanced-versus-balanced (visited on 10/19/2023).

[12] J. Casper. «Music studio essentials: Understanding balanced vs. unbalanced xlr cables». (2016), [Online]. Available: https://ask.audio/articles/music-studio-essentials-understanding-balanced-vs-unbalanced-xlr-cables (visited on 10/19/2023).

[13] B. Forouzan, *TCP/IP Protocol Suite*. McGraw-Hill Publishing, 2009, ISBN: 9780077414597.

[14] K. Fall and W. Stevens, *TCP/IP Illustrated* (Addison-Wesley professional computing series vol. 1). Addison-Wesley, 2011, ISBN: 9780321336316.

[15] P. Nath and M. Uddin, «Tcp-ip model in data communication and networking», *American Journal of Engineering Research (AJER)*, vol. 4, no. 10, pp. 102–107, 2015. [Online]. Available: https://www.ajer.org/papers/v4(10)/N04101020107.pdf.

[16] M. Donahoo and K. Calvert, *TCP/IP Sockets in C: Practical Guide for Programmers* (TCP/IP Sockets in C Bundle). Elsevier Science, 2009, ISBN: 9780080923215.

[17] FEMA.gov. «Common alerting protocol». (2021), [Online]. Available: https://www.fema.gov/emergency-managers/practitioners/integrated-public-alert-warning-system/technology-developers/common-alerting-protocol (visited on 10/18/2023).

[18] OASIS. «Common alerting protocol version 1.2». (2010), [Online]. Available: https://docs.oasis-open.org/emergency/cap/v1.2/CAP-v1.2-os.html (visited on 10/18/2023).

[19] *W6100 ethernet shield datasheet*, W6100 MKR-Ethernet Shield, Revised February 2019, WIZnet, 2018.

[20] AxelTech. «Falcon d7». (), [Online]. Available: https://www.axeltechnology.com/falcon-d7-2/ (visited on 10/21/2023).

[21] ProFMBroadcast. «Omnia.9sg fm stereo generator». (), [Online]. Available: https://www.profmbroadcast.com/en/product/867/omnia.-9sg-broadcast-stereo-generator/ (visited on 10/21/2023).

[22] Focusrite. «Focusrite scarlett 4i4 4th gen». (), [Online]. Available: https://www.musik-produktiv.com/tr/focusrite-scarlett-4i4-4th-gen.html (visited on 10/21/2023).

[23] MOTU. «Motu m4». (), [Online]. Available: https://www.musik-produktiv.com/tr/motu-m4.html (visited on 10/21/2023).

[24] *Drv13x audio-balanced line drivers*, DRV134, Revised December 2014, Texas Instruments, 1998.

[25] *Precision, cmos analog switches*, MAX319, Rev. 1: 11/94, Maxim, 1994.

[26] R. Bez, E. Camerlenghi, A. Modelli, and A. Visconti, «Introduction to flash memory», *Proceedings of the IEEE*, vol. 91, no. 4, pp. 489–502, 2003. DOI: 10.1109/JPROC.2003.811702.

[27] S. Wilson. «Wave pcm soundfile format». (2003), [Online]. Available: https://ccrma.stanford.edu/courses/422-winter-2014/projects/WaveFormat/ (visited on 10/18/2023).