UNIVERSITEIT•STELLENBOSCH•UNIVERSITY
jou kennisvennoot • your knowledge partner

Time synchronisation of physically separate audio recorders

Kobus Meyer
20790988

Report submitted in partial fulfilment of the requirements of the module Project (E) 448 for the degree Baccalaureus in Engineering in the Department of Electrical and Electronic Engineering at Stellenbosch University.

Supervisor: Prof T. R. Niesler

November 2020

Acknowledgements

I would like to thank the following people:

- Prof Niesler for his skripsie supervision and helpful feedback throughout the project.
- Christiaan, Tanvir, Fran, Francois and Gary for graciously lending me components.
- All my friends both within Eendrag and without, for helping me through these "unprecedented times".
- My parents for their continuous support, especially during the lockdown period. I couldn't have done it without them.

UNIVERSITEIT•STELLENBOSCH•UNIVERSITY
jou kennisvennoot • your knowledge partner

Plagiaatverklaring / *Plagiarism Declaration*

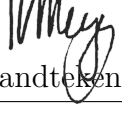
1. Plagiaat is die oorneem en gebruik van die idees, materiaal en ander intellektuele eiendom van ander persone asof dit jou eie werk is.
   Plagiarism is the use of ideas, material and other intellectual property of another's work and to present it as my own.

2. Ek erken dat die pleeg van plagiaat 'n strafbare oortreding is aangesien dit 'n vorm van diefstal is.
   I agree that plagiarism is a punishable offence because it constitutes theft.

3. Ek verstaan ook dat direkte vertalings plagiaat is.
   I also understand that direct translations are plagiarism.

4. Dienooreenkomstig is alle aanhalings en bydraes vanuit enige bron (ingesluit die internet) volledig verwys (erken). Ek erken dat die woordelikse aanhaal van teks sonder aanhalingstekens (selfs al word die bron volledig erken) plagiaat is.
   Accordingly all quotations and contributions from any source whatsoever (including the internet) have been cited fully. I understand that the reproduction of text without quotation marks (even when the source is cited) is plagiarism.

5. Ek verklaar dat die werk in hierdie skryfstuk vervat, behalwe waar anders aangedui, my eie oorspronklike werk is en dat ek dit nie vantevore in die geheel of gedeeltelik ingehandig het vir bepunting in hierdie module/werkstuk of 'n ander module/werkstuk nie.
   I declare that the work contained in this assignment, except where otherwise stated, is my original work and that I have not previously (in its entirety or in part) submitted it for grading in this module/assignment or another module/assignment.

20790988 Studentenommer / *Student number*	 Handtekening / *Signature*
K Meyer Voorletters en van / *Initials and surname*	09/11/2020 Datum / *Date*

Abstract

English

In this report a system that allows time synchronisation of physically separate, commercially available audio recorders is designed, built and tested. The motivation for this is to allow acoustic localisation using widely-spaced microphones in order to track the movement of elephants. This can be useful for wildlife conservation, for averting human-elephant conflict in rural areas, and for the field of nature science in general. GPS is used as a time source, and the time information is encoded in one channel of a stereo audio signal using a modified IRIG-B 124 format. A maximum time difference of 21.3 µs was achieved between separate systems, and the decoding symbol error rate was 1.17×10^{-6}. Time-difference of arrival between recorder pairs based on simple cross-correlation was used to localise a sound source. In field tests consisting of two recorders spaced 10 m apart, 88.9% of 158 cases were localised to within 1 m of the expected location.

Afrikaans

In hierdie verslag is 'n sisteem ontwerp, gebou en getoets wat kommersieële klank-opnemers in tyd sinkroniseer. The motivering hieragter is om akoestiese lokalisering met wyd vespreide mikrofone te bewerkstellig, met die doel om die beweging van olifante op te neem. Dit het toepassings vir natuurbewaring, die vermindering van konflik tussen olifante en mense in landelike gebiede, en die veld van natuurwetenskap in die algemeen. GPS is as tydbron gebruik, en die tyd inligting is in een kanaal van 'n stereo oudio-sein enkodeer in 'n IRIG-B 124 formaat. 'n Maksimum tydverskil van 21.3 µs is gemeet tussen twee aparte sisteme, en die foutkoers by simbooldekodering was 1.17×10^{-6}. Die tydverskil — gebaseer op eenvoudige kruiskorrelasie — tussen die aankoms-tye van 'n geluid by opnemers is gebruik om die klankbron te lokaliseer. In veld-toetse met twee opnemers wat 10 m uit mekaar is, is 88.9% van 158 gevalle gelokaliseer binne 1 m van hul verwagte posisie.

Contents

List of Figures

List of Tables

Nomenclature

Variables

c_0	Speed of sound in air (343 m/s).
D	Difference.
d	Distance between recorders.
F_n	Focal Points.
f_{source}	Frequency of clock source of micro-controller.
k	Fraction of timecode stored in memory.
μ	Mean.
N	Counter period value.
n	Number of samples per period of timecode carrier.
R	Prescaler value.
R_n	Receivers.
r_n	Receiver positions.
S	Hyperbola.
σ	Standard Deviation.
t_d	Time difference.
V_{DDA}	Micro-controller positive voltage reference.

Acronyms and abbreviations

ADC	Analogue to Digital Conversion
BCD	Binary Coded Decimal
Ch	Channel
CPU	Central Processing Unit
DAC	Digital to Analogue Conversion
DHR	Data Holding Register
DMA	Dynamic Memory Allocation
DOR	Data Output Register
ECSA	Engineering Council of South Africa
GLONASS	GLObal NAvigation Satellite System
GPIO	General Purpose Input/Output
GPS	Global Positioning System
HPF	High-Pass Filter
IRIG	Inter-Range Instrumentation Group
LTC	Linear or Longitudinal Timecode
NMEA	National Marine Electronics Association
PC	Personal Computer
PCB	Printed Circuit Board
PPS	Pulse Per Second
RAM	Random Access Memory
RX	Receive
SMPTE	Society of Motion Picture and Television Engineers
TIM	Timer
TTL	Transistor-Transistor Logic
TX	Transmit
UART	Universal Asynchronous Receiver-Transmitter
US	United States
USB	Universal Serial Bus
UTC	Coordinated Universal Time
VITC	Vertical Interval Timecode

Chapter 1

Introduction

1.1. Background

The World Wildlife Foundation lists the African Elephant as being vulnerable and the Asian Elephant as endangered. They are also classified as being a "keystone" species, since they have a pivotal role in structuring both animal and plant communities, as well as forming the largest share of mammal biomass in their habitats [4]. They are noted for having advanced social and cognitive behaviours; an article in Nature describes how traumatic events such as the violent death of a mother early in the life of elephants leave them at high-risk for mental disorders. Responses usually associated with human post-traumatic stress disorder such as abnormal startle response, depression, unpredictable asocial behaviour, and hyper-aggression have been noted in wild elephants. [5]

Much information can be gathered from tracking the movement of elephants, such as determining their home range, tracking periodic or seasonal long distance movements, the relationships between darts, herds and individuals, foraging behaviour and spatial use of resources. [6]

Tracking the movement of elephants is typically achieved by equipping them with radio transmitting devices, and triangulating their position based on terrestrial antennas. The usage of satellite tracking through Global Positioning System (GPS) collars is prevalent, as well as using orbiting satellites such as the ARGOS system. [6]

It is not only to the benefit of science to track these animals. In rural areas both in Africa [7] and Asia [8], human-elephant conflict presents a danger to both parties involved. When situations turn violent, more often it results in the death of elephants than that of humans. This conflict also negatively affects local attitudes towards the animals involved as well as attitudes on conservation, especially in areas bordering on protected spaces. In Kenya, interactions with elephants make up the majority (61.6%) of reported incidences over the period 1995 - 2017. This is mostly attributed to the increasing competition for resources due to growth of both the human and elephant populations in the areas. [7]

An additional benefit is that continuous monitoring of elephant activity could give indication of poacher activity in the area [9]. If this should prove to be the case, it would confer benefits to both elephants and other vulnerable species such as Rhinoceros.

Elephants are known to communicate over long distances using low-frequency vocalisations or "rumbles", with fundamental components below 30 Hz. These calls have been detected over 10 km distances in favourable conditions. However, elephants are only known to extract social information from these vocalisations over distances up to 2.5 km. Recognition of whether a call belongs to a family or bond group member is more frequently achieved for distances in the 1 - 1.5 km range [10]. There has been some research into the tracking or localisation of elephants by using these vocalisations [8, 9].

1.2. Problem Statement

One approach to tracking elephants by means of their vocalisations involves having several microphones or recorder units separated by large distances that make running cables impractical. These units need to be time synchronised in order to produce useful output, since losses in time accuracy translate to losses in tracking accuracy.

Previous approaches have made use of an array of closely spaced microphones attached to the same device, removing this need for synchronisation [9]. In addition to the multiple microphones, the device used in this work also has a programmed micro-controller or computer, storage, analogue to digital converters (ADC's) and a GPS. Although it is a convenient self-contained system, it becomes difficult to maintain and develop further, especially in the absence of the original developer. This is often the case with longer-term projects with changing participants.

In this report the aim is to design, build and test a time synchronisation unit that can interface with commercial field recorders, thereby resulting in a more modular system and making maintenance easier. The time information must be encoded on an audio channel with sufficient accuracy to localise a sound source.

1.3. Scope

The following fall within the scope of this project:

- Identify ways in which time synchronisation between separate audio audio recordings can be achieved.
- Design and prototype a device that can interface with commercial field recorders.
- Write software than can decode the time information from the recorded audio signal.
- Perform field recordings for evaluation purposes.
- Use the field recordings to confirm that the added synchronisation information allows localisation and to what accuracy.

The following do not fall within the scope of this project:

- Optimal component selection. The focus of this project is to develop a working prototype as quickly as possible, minimising cost where possible.
- Localisation software. The focus of any localisation experiments will be to show that the synchronisation is sufficient to in principle allow localisation. Accurate localisation itself is not a goal of this project.
- Power sources for long term field usage. The developed devices need only be powered long enough to gather data for testing. The design of a power supply allowing long-term field use, for example by solar energy harvesting, is not within the scope of this project.
- Beamforming approaches to acoustic localisation need not be considered, the design will assume the presence of single omnidirectional sensors only.

1.4. Roadmap

First information is gathered on the topics of accurate time sources, acoustic localisation techniques and timecodes. A system is then designed that addresses the problem of recorder synchronisation. The constituent parts of this system, their design and implementation are discussed. These parts include the voltage regulation circuitry, the GPS modules, the micro-controller as well as the encoding and decoding of the chosen timecode. Practical measurements and results that directly relate to the synchronisation capabilities of the system are given, as well as a field test to determine the localisation capabilities. A conclusion is then reached with regards to the achievement of project goals and recommendations for future work are made.

The project planning and demonstration of ECSA level outcomes are discussed in the appendices.

Chapter 2

Literature Review

This section aims to show what relevant and related research has been done in the area of acoustic sensor synchronisation, as well as to present the material required to understand the problem and solution techniques.

Important aspects that will be considered in related research include: how sensor synchronisation is achieved, the signal processing techniques used, and the localisation approaches that are used.

2.1. Sources of Accurate Time Information

The primary goal of synchronising clocks in a network is to ensure that all non-faulty clocks are synchronised to within δ time units, where δ is called the synchronisation precision. This is a non-trivial task. Even if clocks are initially synchronised, they tend to drift away from this reference at different rates [11]. How quickly this occurs depends on the implementation of the time source at the node. Oscillating circuits are dependent on component tolerances, and even those using quartz crystals are dependent on temperature. Quartz oscillators are commonly used in consumer electronics such as computers, micro-controllers and wrist-watches. It is possible to go directly to the definition of a second as derived from the ground-state hyperfine frequency interval of caesium-133 atoms by using an atomic clock, but this is too expensive and cumbersome a solution for most applications [12].

Another source of inaccuracy is the delays in message travel time due to either the physical constraints of the medium or the software implementation aspects such as queuing and input-output node throughput [11].

Two approaches can be defined in the synchronisation of clocks in a system, namely internal synchronisation — keeping nodes synchronised with each other, and external synchronisation — keeping all nodes synchronised to an external reference such as coordinated universal time (UTC) or GPS. [11]

We will focus on external synchronisation, since the goal is to have the nodes physically separated. Possible time references include broadcasts by terrestrial radio which are received by all nodes with minimal delay, or the time information that is available from

the GPS network [13].

2.1.1. GPS

A GPS module receives signals from the GPS satellites in medium-earth orbit. These signals contain the following information [14]:

- Ephemeris data, which is needed to determine the satellite's postion. This includes the current date and time, the health of the satellite.
- Almanac data, which conveys the position throughout the day of every satellite in the system and orbit information.
- Pseudorandom identification code, which identifies the satellite transmitting the information.

Each satellite has an atomic clock onboard, which is updated twice per day from a terrestrial reference. As discussed in Section 2.1 this is one of the most accurate clock types.

In addition to calculating its position using trilateration, a GPS module's internal clock is synchronised to the clock on the GPS satellites. The module needs to receive data from at least four satellites simultaneously to calculate its time and position accurately [13]. This time information is available in the form of a pulse per second (PPS) TTL level logic signal. The δ between the PPS rising edge of all modules is stated to be less than 30 ns [15].

In a sensor synchronisation application using an Adafruit Ultimate GPS v3, the time between rising edges was found to be maximum 300 ns, and the jitter between the rising edges of different GPS modules was found to be maximum 400 ns. [13]

2.2. Acoustic Localisation

Acoustic localisation is the determination of the position of a sound-emitting source using only the sound it makes. In our context it involves the usage of several arbitrarily-positioned audio receivers or recorders, and the processing of recordings either in real time or afterwards, to determine the position of one or several sources.

There are generally two steps to such localisation algorithms. The first is to move from the audio to some numerical input, and is often based on the time differences of the sound being detected between receiver pairs. The second is to use numerical input to calculate or estimate the position of the source or sources.

Note that all of these techniques assume that the source can be successfully detected by multiple receivers. Therefore if the source is highly directional, these approaches might not work.

2.2.1. Time Difference of Arrival

The time difference of arrival method (sometimes abbreviated to TDoA) involves calculating the difference in the time of which sound is detected at different receivers, and using this to calculate the position.

A prerequisite for this technique is that the receivers are time-synchronised. This can be done by emitting a known sound at a known position and calculating the time offset between receivers [16]. In [8] this is achieved by equipping each receiver with an FM radio transmitter that transmits to a central receiver and recorder.

The time difference of arrival of a sound can be calculated by visually inspecting the recordings, but this is time-consuming and becomes difficult when the signal to noise ratio is low or the sound does not contain sharp peaks [16]. This manual method can only be performed after the fact, and is therefore not possible for continuous monitoring. A more automated approach is to find the time delay corresponding to the maximum value of the cross-correlation.

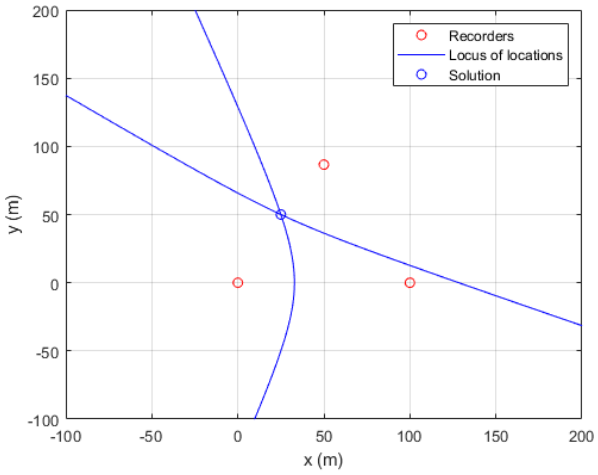
For every pairwise time difference between recorders, a hyperbola can be drawn of the possible positions of the source. This follows directly from the definition of a hyperbola as a line S of constant difference D, between two focal points F_1 and F_2 in Equation 2.1. Here $|\cdot|$ denotes the Euclidean distance.

$$|F_1S| - |F_2S| = D \tag{2.1}$$

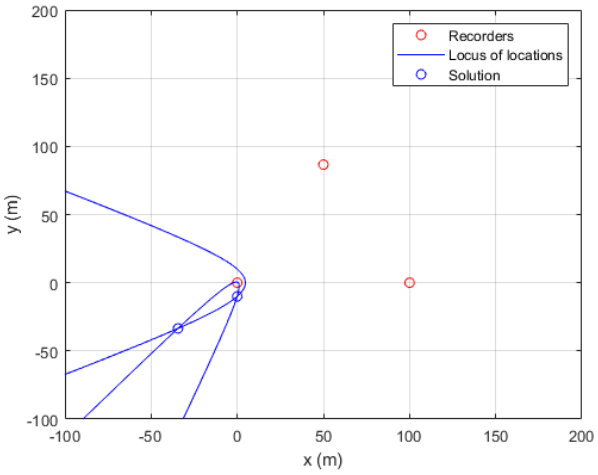
$$|R_1S| - |R_2S| = t_d c_0 \tag{2.2}$$

If the focal points are at the receivers, we can write Equation 2.2 where R_n the positions of the receivers, t_d the time difference of arrival and c_0 the speed of sound. In a two-dimensional case, the intersection of these hyperbolas can be solved for three or more recorders in some cases, giving the position of the source. For the 3-dimensional case, the hyperbolas turn into hyperboloids, and a minimum of four receivers are required. Figure 2.1a shows a source successfully localised at the intersection of the hyperbola loci. In Figure 2.1 the hyperbolas intersect twice, and the source cannot be localised, since real sound sources are not normally in two places at the same time.

For a real-valued hyperbola it is necessary for the difference D in Equation 2.1 to be smaller than the distance between the two focal points. Any real-world sound source will give real-valued loci. In order to eliminate incorrect results from the pairwise cross-correlation, the range of delays can be limited to $\pm\frac{d}{c_0}$ where d is the distance between the pair of recorders [17].



(a) Single solution at (25,50)



(b) Multiple solutions

Figure 2.1: Plot showing source at intersection of hyperbola loci.

2.2.2. Microphone Array Approach

Instead of having several wide and irregularly spaced microphones, one could have several precisely positioned microphones on the same device. This approach rests on the principle that an array of audio receivers can be steered in the same way that the array of microwave antennas can be steered by exciting the elements out of phase. The array then acts as a single highly directional sensor. [9]

This is more computationally intensive to implement and excluded from the project scope, and therefore not discussed further.

2.2.3. Sources of Uncertainty

The accuracy of localisation is negatively influenced by several factors including wind, multipath-effects, echoes, background noise, non-uniform absorption of sound due to foliage, and non-constant temperature variation across the testing space [8, 16]. In [16] a model of the wind speed and temperature distributions are calculated using a tomographical reconstruction process: estimating the properties of the 3-D area from 2-D sections given by the lines of travel from the audio source to the receivers. A more recent approach in [8] uses test sound sources that emit a known 'chirp' sound in order to obtain the travel times in the area of interest, presumably at regular intervals.

2.3. Timecodes and Time Stamping

A timecode consists of time information encoded in a signal. The time is defined at specific on-time instants as shown in Figure 2.2. The duration between these instants is

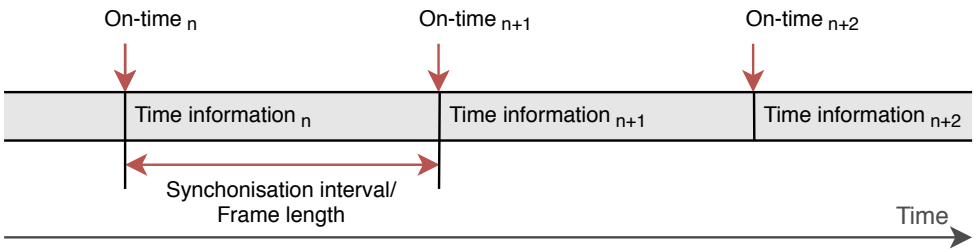


Figure 2.2: Diagram of general timecode structure.

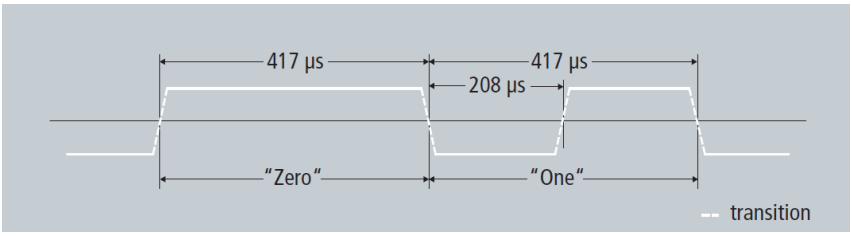


Figure 2.3: Bi-phase encoding timing for 30 frames per second. Reproduced from [1].

the synchronisation interval or frame length. The time defined at the on-time instant is encoded in the timecode immediately following it.

There are several established timecode formats, with varying synchronisation intervals and modulation types. Although we are not limited to standard encodings for this project, it is useful to consider existing standards, notably Linear Time Code and IRIG codes.

2.3.1. Linear Time Code

Linear or Longitudinal Time Codes are a standard originally set by the Society of Motion Picture and Television Engineers (SMPTE). For this reason they are also sometimes referred to as SMPTE codes. There also exists another format called Vertical Interval Time Code (VITC), but this is closely tied to the blanking time interval of analogue video standards, and will therefore not be discussed further. The main use case of this code is to have a single master time code generator connected to multiple follower devices, thereby ensuring they are all synchronised. [18]

Linear time codes are defined for a finite number of synchronisation intervals, including 41.7 ms, 40 ms, 33.4 ms and 33.3 ms, corresponding to framerates of 24, 25, 29.97, 30 frames per second. This standard uses bi-phase encoding and is designed to fall within normal audio bandwidth [1]. Bi-phase is a binary scheme where information is encoded in the zero crossing of a signal. Intervals are delimited with zero crossings, a binary zero is indicated by the absence of zero-crossings in an interval, and a binary one is indicated by a single zero-crossing in an interval. An example is given in Figure 2.3. This code is then directly recorded on an audio channel.

The time information is encoded in an 80-bit message. This message consists of the current time, expressed in hours, minutes, seconds and frames, along with 32 user-defined

bits and a 'sync-word'. The on-time instant is given as the first zero-crossing after the sync-word.

2.3.2. IRIG

IRIG is a standard containing a number of timecode formats created by the Inter-Range Instrumentation Group (IRIG), part of the Range Commanders Council of the US Army. This standard is set out in [2].

Table 2.1: IRIG code synchronisation intervals and symbol rates, adapted from [2].

Type	Synchronisation interval	Symbol rate (as symbols per second)
A	0.1 s	1000
B	1 s	100
D	1 h	0.0167
E	10 s	10
G	10 ms	10
H	1 min	1

There are different formats of IRIG, with different synchronisation intervals and symbol rates. These are summarised in Table 2.1.

All IRIG formats use pulse width encoding. Position reference pulses have a length of 80% of the index interval. These delimit different sections of the code. Binary zeros consist of a pulse that is high for 20% of the index interval, and the pulse indicating a binary one is high for 50%. The 'on-time' instant is on the rising edge of the position reference marker at the start of the frame.

Time is specified using years and the time of year. There are 18 user-defined or control bits available in an IRIG-B message. An example of a frame is given in Figure 2.4. Note that all values are encoded in binary decimal code (BCD) format: the units are encoded as a binary number, the tens as a seperate number and where applicable the hundreds also.

IRIG-B with a 1 kHz carrier was historically recorded on voice-grade channels on multi-track cassettes [19]. In [20], IRIG-B is output by a GPS device and used to synchronise clocks at two different sites.

2.4. Existing timecode generators

There are several existing timestamping devices on the market, for various applications. One of the most common is the timecode generators for the film and audio industry. The use case here is to have one main time code generator, and have all other devices directly connected to this. This is to ensure that all cameras and microphones are time synchronised. Lower end devices use an on-board oscillator, but the more expensive devices

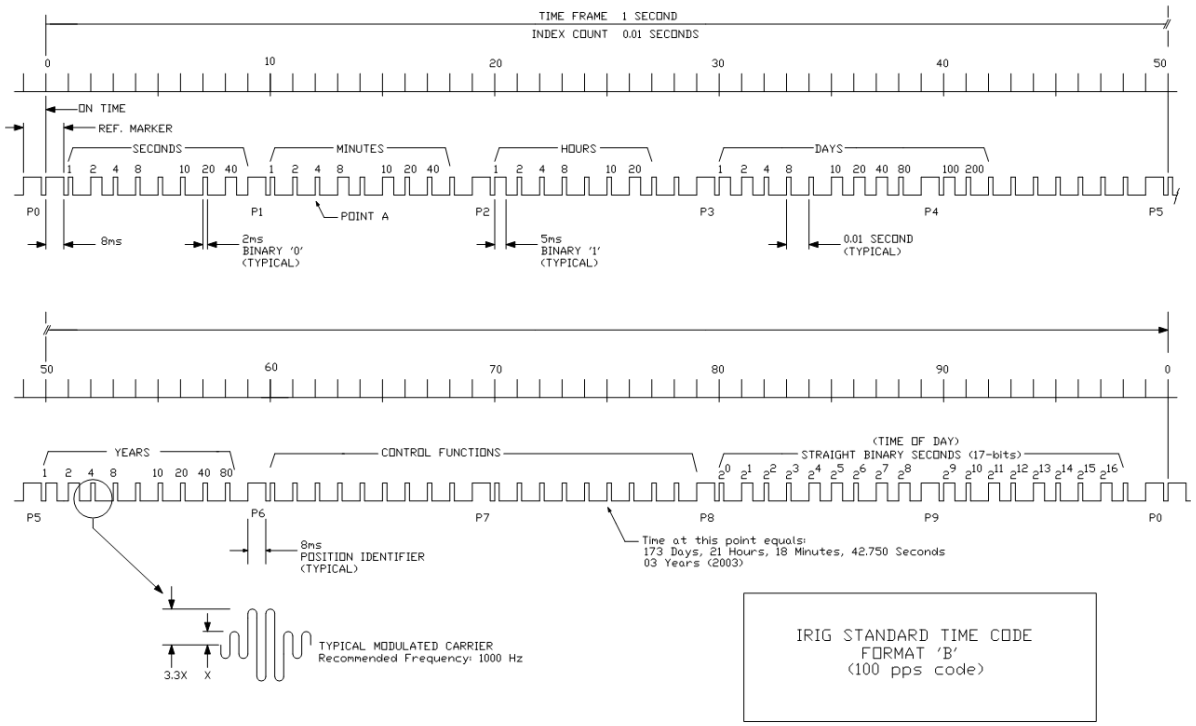


Figure 2.4: Contents of an IRIG-B message. Reproduced from [2].

obtain date and time information from a GPS module. These devices are for the most part large, power intensive and expensive.

Other typically more expensive devices are for network synchronisation. These obtain time information from one or several satellite networks, and output a continuous information carrying signal.

Chapter 3

System Design

There are four conceptual parts of the system that need to be implemented, namely the time source, the encoding of this information in an audio channel, the decoding of this information from the audio channel information, and using the time information in conjunction with audio for localisation. This chapter will detail the design from conceptual pieces to actual components, as well as the interfaces between them.

A GPS was chosen as the time source, due to the ease of use and its known high accuracy.

A micro-controller was chosen to do the processing.

Time difference of arrival with general cross-correlation was chosen as the localisation method due to its simplicity.

3.1. System Overview

The system is shown in Figure 3.1. Power is obtained from a fixed bench power supply during laboratory development and a 9 V battery during testing. This is then converted to 5 V for the micro-controller, and 3.3 V for the GPS module and signal conditioning circuitry.

The micro-controller decodes UART messages from the GPS module which are sent once per second. The contents of these messages are in the RMC format described by the

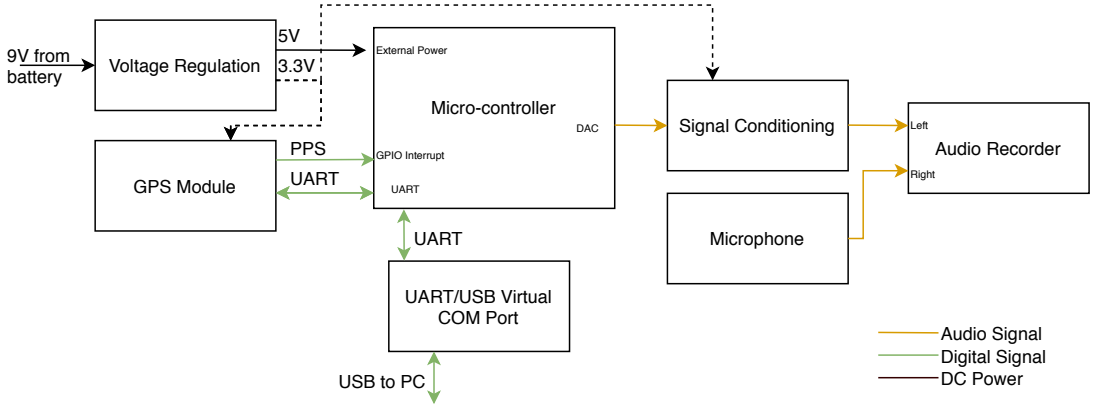


Figure 3.1: Overall system block diagram.

NMEA 0183 standard, and contain the date, time and position. Before the rising edge of the PPS signal, this information is encoded as an audio signal in the IRIG-B 124 format in memory. On the rising edge of the PPS signal, this is output through the DAC of the micro-controller using DMA. The signal is conditioned and input on the left channel of the audio recorder. The microphone is directly connected to right channel of the audio recorder.

The UART/USB virtual COM port creates an interface between the micro-controller and a serial terminal on a PC. This was mostly used for debugging purposes and for changing settings on GPS modules.

After this system has been deployed, and data simultaneously recorded on 2 or more recorders, this captured data need to be processed to deliver the localisation results. This was achieved by transferring the audio files from the recorders to a PC, and processing them in MATLAB to extract the decoded data. This data, combined with the audio from the microphone are used as inputs for the localisation algorithm to give the location of a sound source at specific moments in time.

3.2. Voltage Regulation

There are three components that require power: the micro-controller, the GPS module, and the op-amp in the signal conditioning, as shown in Table 3.1. It was decided to use a 5 V input for the micro-controller, and a 3.3 V input for the GPS and op-amp. This was done to ensure that as little as possible thermal strain is placed on the onboard regulators of both the micro-controller and the GPS, since external components are much easier to replace in case of failure. The op-amp was biased between 0 V and 3.3 V.

Table 3.1: Voltage levels of components.

Component	Voltage input range
GPS	3.3 V - 5 V [15]
Microcontroller	7 V - 12 V or 4.75 V- 5.25 V [21]
op-amp	-8 V to -2.2 V or 2.2 V to 8 V [22]

The power source for the system was selected to be a 9 V battery, in order for it to be portable. Energizer 9 V batteries have a capacity of 500 mAh (Alkaline) or 750 mAh (Lithium) at a 100 mA discharge rate. The current draw of the full system was measured to be 80 mA. For proof of concept test this 5.25 - 9.4 hour battery life is sufficient, but is not necessarily a solution for long term field deployment. The alkaline battery was chosen for its lower cost.

In Figure 3.2 the circuit for voltage regulation is shown. The diode D_{in} protects the circuitry against accidental negative voltages on the input and has a forward voltage of

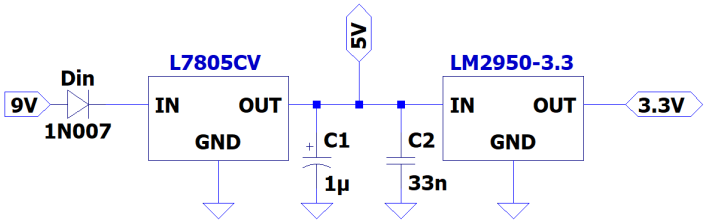


Figure 3.2: Voltage regulation circuit diagram.

0.6 V to 0.75 V for the forward current range of 10 mA to 100 mA. Since the dropout voltage of the 5 V regulator is 2 V, this leaves a 1.25 V downward tolerance for the input.

Both the 5 V and 3.3 V regulators are linear regulators. Although these have much lower efficiency than switching regulators, they do have the advantage of only requiring additional capacitors and no inductors to ensure stability. The noise on the output of linear regulators is usually lower and uncorrelated/incoherent, due to the absence of kilohertz frequency switching. High noise levels on the GPS power could affect the accuracy negatively [15]. Noise on both the power supply and electromagnetic noise caused by large inductors could affect and couple to the audio input. This could in turn cause spurious peaks in the cross-correlation between different recorders, which is undesirable since it would negatively affect the localisation.

The capacitors used in Figure 3.2 are to ensure stability and to filter the output. The values are directly from the relevant component datasheets. The 3.3 V output has no capacitor; this is placed close to the opamp input.

3.3. GPS Modules

Two different GPS modules are used in this design. The selection criteria were: (1) it had to be available as a break-out board, to remove the need for a printed circuit board (PCB) design, which would have added cost and time, and (2) it had to have an accessible PPS pin, of which the significance is discussed later in this section. Further considerations were short shipping times and low cost. The Adafruit Ultimate GPS v3 was used in initial prototyping since it satisfies the above-mentioned criteria and was already available.

Later the GOOUUU NEO M8N Mini GPS Module was identified as having all the necessary features at a cost 4 to 5 times lower than the Adafruit. However, this module had significantly less supporting documentation available. One of each was used, since a second GOOUUU module could not be obtained in the project time frame.

The Adafruit has an built-in passive antenna, and a larger number of output pins, but can only track GPS satellites. The GOOUUU requires an external antenna and has the minimum required output pins, but can communicate with GPS, GLONASS and Bidou satellite networks.

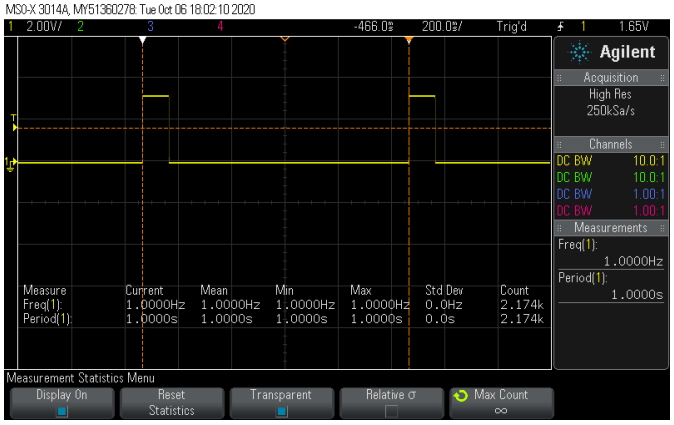


Figure 3.3: Measurement of time difference between PPS rising edges.

These different GPS networks can all provide the same data, but their implementation of UTC is not necessarily the same. The networks could also implement leap seconds at different times [23]. Both modules were set to use only the GPS network to ensure that both are synchronised to GPS UTC.

The PPS output is a very important part of the design. When the GPS module has a 'fix' or is successfully tracking more than 5 satellites, the PPS outputs a 2.8 V pulse once every second. As discussed in Section 2.1.1, the rising edge of this signal is synchronised with UTC, and differs by no more than 400 ns between different modules. The signal is high for approx 100 ms. The accuracy was measured from 2174 two-second windows in Figure 3.3.

To consider whether this is a accurate measurement, we need to consider the sampling frequency and the oscilloscope's internal clock accuracy. The sampling frequency of the oscilloscope is 250000 samples per second, giving 4 µs sampling intervals, meaning that we can specify to the nearest 10 µs with confidence. Assuming the oscilloscope is 6 years old, the accuracy of the internal clock is specified to be 75 ppm. This gives an error of 75 µs per second, meaning we can specify to the nearest 150 µs. Thus, we can say with confidence that the PPS has a period of 1.000 s. This is nowhere near the specified accuracy of 300 ns [13], but it is the most accurate we can measure it with the available equipment.

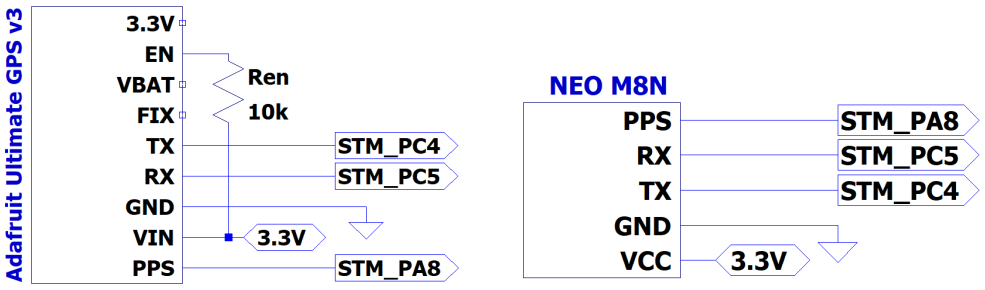
GPS modules also output universal asynchronous receiver-transmitter (UART) messages at configurable intervals and baud-rates. These messages are all defined by the NMEA 0183 (National Marine Electronics Association) specification. These messages contain the date, time, position, altitude, and velocity of the module. There are many different types of messages, each containing a different combination of the aforementioned information. It was decided to use the RMC (Recommended Minimum specific GPS/transit data) messages following the information presented in [9]. These include the date, time (UTC), latitude and longitude in the format specified in Table 3.2.

The widths of the fields are not fixed, but the maximum length is of the message specified as 84 characters. The precision of the coordinates is also not fixed, and differed

Table 3.2: GPRMC message contents.

`$GPRMC,131403.000,A,3355.70625,S,01851.99143,E,0.00,0.00,191020,,,A*71<cr><lf>`

Message contents	Meaning
`$`	Start of message character
`GPRMC`	Identifies network as GPS, and the type of message as RMC
`131403.000`	UTC in format hhmmss.sss
`A`	Validity of message as (A)ctive or (V)oid
`3355.70625`	Latitude in degrees and minutes in format ddmm.mmmmm
`S`	North/South
`01851.99143`	Longitude in degrees and minutes in format dddmm.mmmmm
`E`	East/West
`0.00`	Groundspeed in knots
`0.00`	Course over ground
	Magnetic variation in degrees
	East/West
`A`	GPS mode, (A)utonomous, (D)ifferential or (E)stimated
`71`	Checksum
`<cr><lf>`	End of message characters, carriage return and linefeed



(a) Adafruit Ultimate GPS v3 **(b)** GOOUUU NEO M8N Mini

Figure 3.4: Circuit diagrams of GPS modules.

between the two modules. The number of field separators (commas) is constant however, and this was used to parse the message. A custom parsing algorithm was written, in order to precisely determine when the decoding takes place and to which variables the data are written. The checksum is the bitwise exclusive or of all characters between the $ and the *, presented in hexadecimal with the most significant digit first. The magnetic variation, ground speed and course are part of the standard, but not all modules provide this information, and these are not used in our application.

In Figure 3.4 the connections of both of these modules are shown. The enable line of the Adafruit is tied high as suggested in [15]. Both breakout boards have their own voltage regulators and decoupling capacitors, and no additional ones were added in Figure 3.4.

In Figure 3.5 the UART Transmit (TX) pin output and PPS output are shown. Here

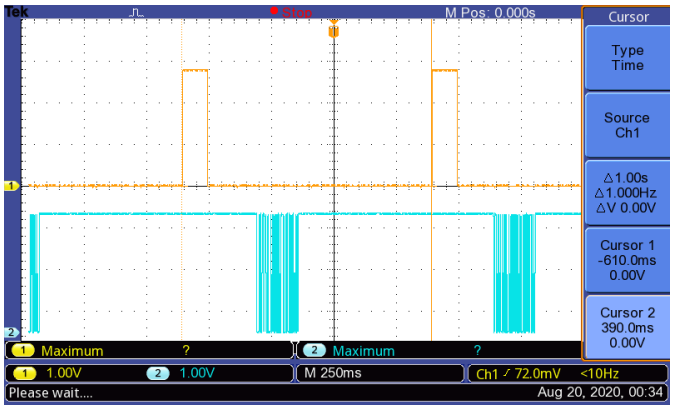


Figure 3.5: Measurement of PPS output (Ch1) and UART TX (Ch2) of Adafruit GPS module.

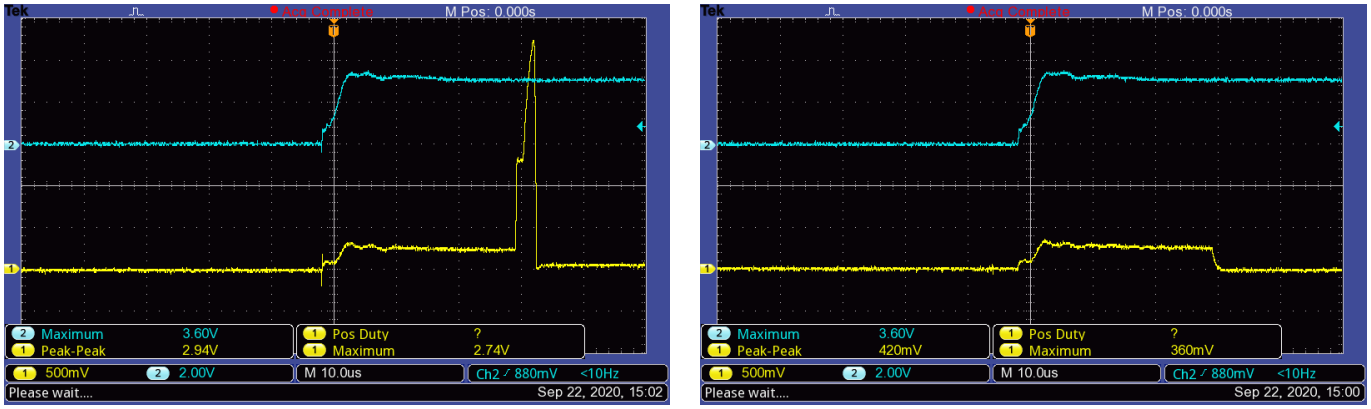
it can be seen that the UART messages are not synchronised with the PPS signal.

3.4. Micro-controller

The STM32F344R8 micro-controller on the Nucleo-64 development board was chosen for the project. Leading factors in this choice were the decreased development time due to an existing familiarity with the board and associated development software, the large number of available peripherals as well as the lower cost to similar alternatives such as the Arduino Mega. The board consists of two parts, the micro-processor and surrounding circuitry, and the ST-Link part. The ST-Link has a separate micro-controller which manages interactions between the PC and the micro-processor, allowing loading programs into flash memory, real-time debugging, and presenting a virtual communication port (COM port) to the PC. This virtual port is wired to the UART2 peripheral and further discussed in Section 3.4.3.

In order to load a program onto this board, it must first be written and compiled on a PC in the STM32CubeIDE program. This environment allows the user to initialise pins and peripherals of the micro-controller through a graphical interface. The C-code the user writes does not have to manipulate register values directly; instead interacting with Hardware Abstraction Layer functions. These are presented as an application programming interface (API) described in [24]. The compiled program is then sent to the on-board flash-memory through the ST-Link interface.

Several different on-board peripherals of the micro-controller were used. These include the General Purpose Input/Output (GPIO), Digital to Analogue Converter (DAC), UART, Dynamic Memory Allocation (DMA), and Timers (TIM).An overview now follows of how these form part of the design.



(a) Internal op-amp enabled.

(b) Internal op-amp not enabled.

Figure 3.6: Measurement of DAC output (Ch1) and PPS (Ch2) with differing internal op-amp settings.

3.4.1. DAC

The micro-controller has two Digital to Analogue Converters (DACs). DAC1 was used due to its higher number of features, although none of these features ended up being useful to the final design.

The DAC was enabled in 8-bit mode (as opposed to 12-bit mode) in order to conserve the STM32's limited memory (RAM). DAC1 has an on-board op-amp that can be switched in, that allows it to directly drive a 25 kΩ load. This would have allowed the DAC to directly drive the recorder input, which has an input impedance of 25 kΩ. A test was conducted by enabling the DAC after the PPS interrupt is triggered both with the op-amp enabled and not enabled. It can be seen from Figure 3.6a that a 2.7 V spike is present on the output when the op-amp is enabled. This spike is undesirable because it was larger than the recommended maximum voltage input of the recorder [25]. This necessitated the signal conditioning between the DAC output and the recorder input.

$$V_{out} = \frac{\mathrm{DOR}}{256} V_{DDA} \tag{3.1}$$

In order to convert a digital value to an analogue voltage, it must first be loaded into the Data Holding Register (DHR). The value is then transferred to the Data Output Register (DOR) which happens on the next rising edge of the system clock by default. Another clock can be specified as trigger for this operation. The value in the DOR is the the value that is output as a analogue voltage, with the conversion formula in Equation 3.1. Here V_{DDA} is 3.3 V [21].

In order to output a sequence of values to form a waveform it would have been necessary for the CPU to transfer each value from memory to the DHR, placing non-trivial strain on the single core/threaded CPU. To avoid this, DMA was used.

3.4.2. Direct Memory Access

The Direct Memory Access unit (DMA) transfers data between memory and any peripheral without CPU involvement, except to initiate the process. A DMA instance requires three parameters: a trigger, a target register and a source register. The trigger is the signal that controls when the data is transferred and source and/or target addresses incremented or decremented. The target in our case is the DOR register of the DAC1, and the source is the pointer to the start of the array of 8-bit numbers representing the timecode. The target register was set to remain fixed, and the source set to 8-bits wide, and to increment after each transfer. This works because arrays in C are stored as contiguous blocks of memory.

The trigger was set to TIM6 of which the configuration is discussed in Section 3.4.4.

3.4.3. UART

The STM32 has 3 independent UART modules, of which 2 were used. UART2 is wired to the ST-Link interface on the board by default and the UART1 was connected to the GPS module's UART interface. Both were set to use a baud-rate of 9600 bps with format 8N1 (8 bits, no parity, 1 stop bit), in full-duplex mode, allowing simultaneous receiving and transmission of messages.

The ST-Link software presents a virtual COM port to a PC through the same physical USB port with which the board is programmed and debugging info is sent. This ST-Link micro-controller converts the UART data to USB format, which can be read in a terminal on Linux, or on terminal software such as Termite on a Windows PC.

A set of messages was created for communication from the PC to the micro-controller, and are detailed in Table 3.3. The $ indicates the start of message and the * the end of message. The ping-pong message is used to confirm both that the connection is still open, and that the board is still responding.

The code on the micro-controller is designed to be compatible with two different types of GPS modules. Although the GPRMC message format is standard, the configuration protocol of the two modules are different, and the rest of the messages are used to set the module to the correct type and change the GPS message configurations.

3.4.4. Timers

Counter-intuitively, the clock source of the micro-controller does not directly come from a crystal oscillator circuit. Instead the ST-Link micro-controller (STM32F103CBT6) lies between the crystal clock source and the micro-controller. The frequency of the micro-controller is equal to the resonant frequency of the crystal, so it would not be unreasonable to assume the same accuracy for both. The crystal has a resonant frequency of 8.000 MHz

Table 3.3: Message protocol used for PC to micro-controller communication.

Message	Action
$ping*	Transmit 'pong' back to PC.
$NMEAset*	Transmit message to GPS that sets message type to GPRMC only.
$NMEAreset*	Transmit message to GPS that hard-resets the module.
$GPSsetneo*	Sets internal variable that connected GPS module is the GOOUUU NEO.
$GPSsetada*	Sets internal variable that connected GPS module is the Adafruit.

and an accuracy of 20 ppm.

All timers have up to two counter registers connected in series to the 8 MHz source. These count upward from zero to a specified value and then reset to zero once this value is reached. The specified value of the first is known as the prescaler R and the second the counter period N. The output frequency of the timer is the frequency at which the second counter reaches its max value. This is given by Equation 3.2, where f_{source} is the input to the first counter, 8 MHz in our case.

$$f = \frac{f_{source}}{(N+1)(R+1)} \tag{3.2}$$

N and R are chosen to create the desired output frequency. These are limited by the width of the integer that can be stored, 16-bits in this case. We require a 5000 Hz signal for the DMA trigger and a 1 Hz output for a surrogate debugging PPS signal. TIM6 was initialised with $R = 7$, $N = 198$. This results in a frequency of 5025 Hz, just higher than the required 5000 Hz. It was chosen slightly higher to account for variations across different devices, ensuring that it is always 5000 Hz or higher. This timer is connected to the DMA, which has to trigger the transfer of 5000 bytes per second. If the timer is slower, the complete message would not be transferred, resulting in lost data.

TIM2 was initialised with $N = R = 1999$ for a 2 Hz signal, and set to generate an interrupt on every period. Every time this interrupt is handled, the state of GPIO pin PB5 was inverted. This results in a square wave with a frequency of half the frequency of the clock source: 1 Hz at the output of the pin.

3.4.5. GPIO

The General Purpose Input/Output (GPIO) pins are used for two purposes: detecting the rising edge of the PPS signal, and generating a simulated PPS signal for debugging.

Pin PA8 was enabled in external interrupt mode, with rising edge detection. An internal pull-down resistor was activated, in order to prevent the pin from floating when not connected. When the pin is floating the interrupt can trigger on the random fluctuations on this pin.

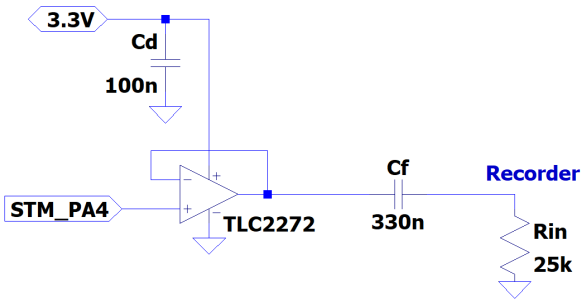


Figure 3.7: Signal conditioning circuit diagram

Pin PB5 was enabled in output mode, in a push-pull configuration. This allows it to drive the pin both low and high. This pin was connected to TIM2 to generate a 1 Hz square signal.

Almost all of the micro-controller pins can be enabled for GPIO. The interrupt and simulated PPS pins were deliberately chosen to be physically far from the DAC output pin, to avoid electromagnetic coupling between these, since the DAC output was typically small and sensitive to interference.

3.5. Signal Conditioning

The signal conditioning circuitry consists of an op-amp voltage follower and a high pass filter (HPF).

The op-amp serves as a buffer between the output of the DAC and the input to the recorder. The high input resistance of this configuration means that the DAC does not need to provide high current, and it also protects the micro-controller from faults on the recorder side.

The TLC2272 rail-to-rail precision op-amp was used. It has a maximum input offset voltage of 2.5 mV which is acceptable for outputs in the range of 250 mV. The typical slew rate is 2.6 V/µs which gives a rise time of 96 ns for 250 mV, the peak-to-peak value of the timecode. This rise time is acceptable since 96 ns << 20.8 µs which is the sampling interval of the recorder. The rail-to-rail aspect was also useful: the minimum value of the output is 0.09 V for output currents less than 50 µA.

The passive HPF has two purposes. Firstly it removes the DC offset present in the DAC output, and it prevents any DC voltage present on the recorder input from interfering with the buffer op-amp and micro-controller.

Figure 3.7 shows the circuit diagram. The positive input of the op-amp is connected to the output pin of the DAC, PA4. C_d is the decoupling capacitor. R_{in} is the internal resistance of the recorder. C_f and R_{in} form a passive HPF.

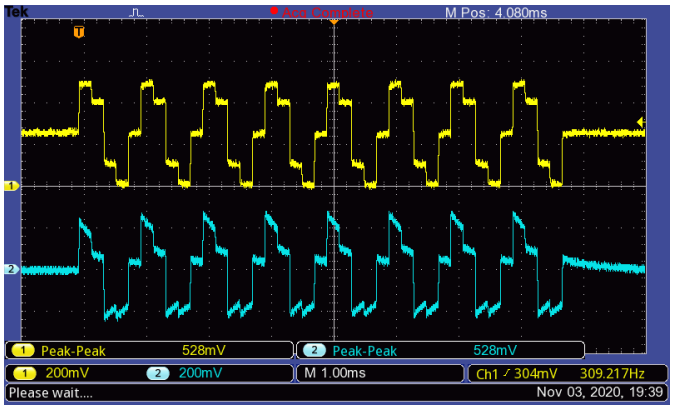


Figure 3.8: Measurement input (Ch1) and output (Ch2) of the high pass filter.

$$f_c = \frac{1}{2\pi RC} \tag{3.3}$$

The cutoff frequency of the filter is given by Equation 3.3. f_c was chosen as 20 Hz since that is the lower limit of the recorder's frequency range. With $R = R_{in}$ fixed at 25 kΩ, $C = C_f$ was calculated as 318 nF, and the closest capacitor value of 330 nF was chosen.

The cutoff of the filter was tested by generating a 500 $\text{mV}_{\text{peak-to-peak}}$ sinusoidal signal with a 250 mV DC offset with a signal generator, and measuring the peak-to-peak value of the output. The recorder was connected and switched on to complete the filter. Low frequencies were attenuated and high frequencies are let through as expected. The cutoff of the filter was verified to be between 180 Hz and 220 Hz, nearly 10 times higher than designed for.

This means that the input impedance of the recorder is closer to 2.5 kΩ than the specified value, or the internals of the recorder influence the cutoff. The latter is more likely, since there should be an additional capacitor to enforce the 20 Hz cutoff inside. If this is then in series with C_f the capacitance is lower and by Equation 3.3, the cutoff is higher. A possible improvement here would be to have an active filter, but this would need either a negative voltage rail or level shifting capacitors to place the required offset on the output and then remove it before it is input to the recorder. A further study of the properties of recorder line-in interface could also improve this design.

The signal from the DAC has some fast rising edges, which contain high frequency components. The sampling rate of the recorder is 48 kHz, meaning that all frequency components higher than 24 kHz will not be recorded. We assume the recorder has anti-aliasing filters in place, so no low pass filters were built.

Figure 3.8 shows the input and output of the filter. The 250 mV DC offset on the input has been replaced with a 5 V offset on the output, which is not visible on the AC coupled channel 2. The shape of the waveform has also changed; the straight tops of the signal have been characteristically sloped downwards.

3.6. Audio

3.6.1. Microphones

Good quality audio equipment is frequently expensive. The leading factors in audio-equipment choice was therefore cost and availability.

For microphones, RS PRO Omni-Directional 6mm condenser microphones (part number KECG2240PBJ) were used. While the common frequency ranges for condensers are 50 - 20 000 Hz, the chosen microphone extends down to 20 Hz. It also has a voltage supply range of 2 V - 10 V, for which the 5 V plug-in power supplied by the recorder is suitable. These microphones were attached to 3.5 mm male audio jacks in order to be easily removed and exchanged with other microphones.

One Rødelink Lav Professional Wearable Microphone was acquired for comparison purposes, at a cost more than 300 times that of the RS PRO. Despite this large cost, the specifications are very similar [26, 27]: the Rødelink has a sensitivity of -33.5 dB, compared to the -30 dB of the RS PRO, signal to noise ratio of 67 dB to 58 dB, and identical frequency ranges. The dynamic range of the RS PRO is not specified.

It must be noted that both of these microphones are condenser types. This means that they need to be biased at a DC offset to function. This is also called phantom power.

3.6.2. Recorders

TASCAM DR05 2-channel recorders were used. These were available from a previous project. They can record two channels simultaneously and provide "plug-in-power" for condenser microphones, which is nominally 5 V [25]. This is useful since this obviates the need to provide phantom power to the condenser microphones from the voltage regulation circuitry in Section 3.2. Data transfer between a PC and these recorders was easy and did not involve the removal of micro-SD storage cards.

Another recorder that was considered was the Zoom H1, but this displayed strange artifacts on the recorded waveforms during testing. Combined with the reduced feature set and interface, lead to the TASCAM to be chosen for the project.

The right channel of the recorder was directly connected to the microphone as shown in Figure 3.1, and the left channel to the signal conditioning output. The built-in microphones of the recorder could unfortunately not be used, since the options for recording were either both built-in microphones, or both line-in channels. If a 4-channel recorder were to be used in a future application, no external microphones would be needed.

3.7. Timecode encoding

The time and date information from the GPS needs to be encoded in an audio signal. The information from the GPS has a synchronisation interval of 1 s.

3.7.1. Format

The encoding decided upon is IRIG-B 124. The 1 means that it is modulated with a sine wave, the 2 indicated that the frequency of the sine wave is 1 kHz, and the 4 means that the message contains the time and date in binary coded decimal (BCD) format, 18 control bits, and 'second of day' encoded in binary.

IRIG-B was selected because the other researched alternative, Linear Time Code (LTC), does not contain the date information, meaning that it repeats every day. In continuous or long-term monitoring this will not suffice. IRIG also has a predefined format with a synchronisation interval of 1 s, which the LTC lacks.

IRIG-B is a ternary pulse-width modulated encoding, consisting of reference pulses, (P), zeros (0), and ones (1). There are also index pulses (I), which are encoded the same way as zeros are, but do not represent any data. Each symbol has a length of 10 ms the information is encoded in the duration of which the signal is high. A duration of 8 ms indicates a P, 5 ms a 1, and 2 ms a 0. This pulse width modulated signal is multiplied by a 1 kHz sinusoidal carrier in order for it to better fit in the audio bandwidth which attenuates components lower than 20 Hz.

The reference pulses delimit parts of the timecode. The on-time instant is given by the rising edge of the latter of two consecutive reference pulses, which occur at 0.3091 s and 1.3091 s in Figure 3.10. This on-time instant marks the start of the second encoded in the following timecode. It contains current time specified by the seconds, minutes, hours, day in year and year encoded in BCD format, and seconds of day encoded in binary, all with the least significant bit first. The seconds of day is redundant, and serves as a checksum for the hours, minutes and seconds. In Table 3.4 the encoding of the timecode is explained, along with example data from the timecode shown in Figure 3.10. The reader is referred to Appendix C for an complete description of the message format.

3.7.2. Implementation

The flow diagram of the encoding process which is implemented on the micro-controller is given in Figure 3.9. The main loop in Figure 3.9a starts when the micro-controller exits the reset state after power on or when the reset button is pressed. If a complete message from the PC on UART2 is received, the message is handled as described in Table 3.3.

After receiving a message from the GPS on UART1, the date and time information is extracted from the message and stored in local variables if the message is a valid GPRMC

Table 3.4: Time and date "08:30:56 27 Oct 2020" encoded in IRIG-B format

Message	Information contained	Decoded information
`P0110I101P`	Units of seconds (4 bits), Tens of seconds (3 bits)	56
`0000I1100P`	Units of minutes (4 bits), Tens of minutes (3 bits)	30
`0001I0000P`	Units of hours (4 bits), Tens of hours (2 bits)	08
`1000I0000P`	Units of days (4 bits), Tens of days (4 bits)	01
`11IIIIIIIP`	Hundreds of days (2 bits)	+300 = 301
`0000I0100P`	Units of year (4 bits), Tens of year (4 bits)	20
`000000000P`	Control bits	all zero
`000000000P`	Control bits	all zero
`000000111P`	Straight Binary Seconds of day 2^0 to 2^8	448
`11011100IP`	Straight Binary Seconds of day 2^9 to 2^{16}	+30208 = 30656

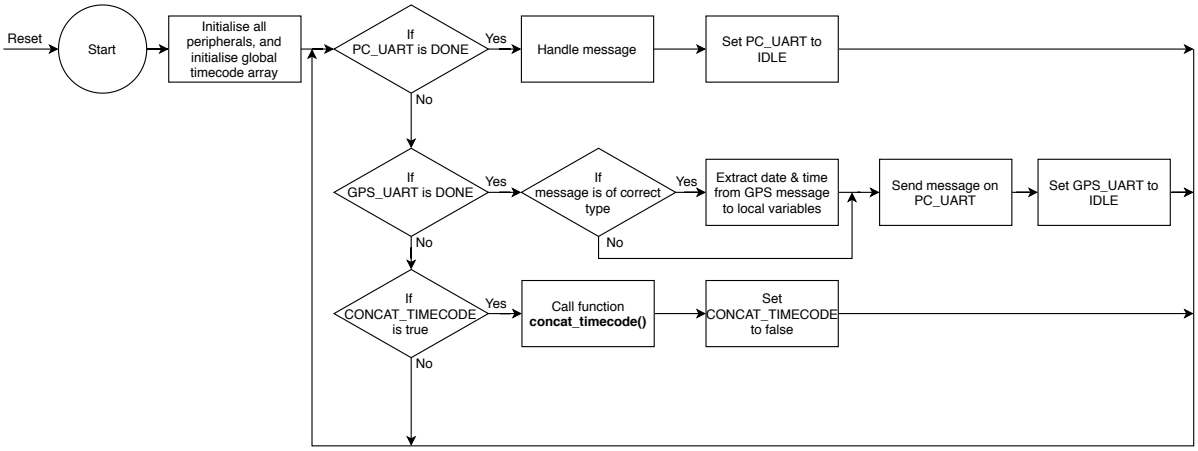
message. On the next rising edge of the PPS signal the PPS interrupt routine (Figure 3.9c) starts. If it is the first time this routine is called the DMA is started, otherwise the DMA would already have been started after completion of the previous DMA finished. The DMA clock is the started, triggering the transfer of the global timecode array from memory to the DAC DHR.

After the DMA has transferred the entire timecode, the DMA transfer complete interrupt is raised, starting the DMA transfer complete interrupt routine in Figure 3.9d. Here the main loop is notified that the global timecode array needs to be re-concatenated, the DMA and DMA clock are stopped. Immediately afterwards the DMA is started, but the data will only start transferring once the DMA clock has been started. Although the timecode at this time is still unchanged, the DMA has a reference to this array. The reference stays the same, but the data will be changed when the main loop calls the `concat_timecode()` function (Figure 3.9b).

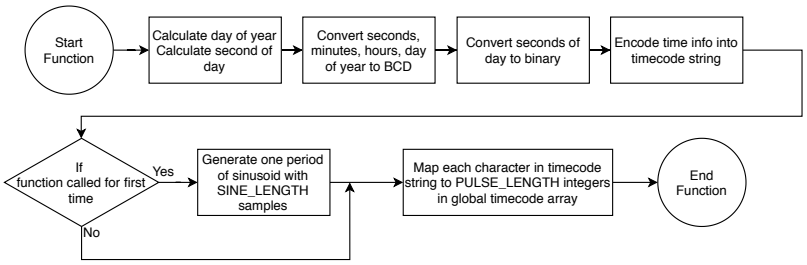
3.7.3. Memory Usage

The micro-controller has 12 kB memory (RAM).

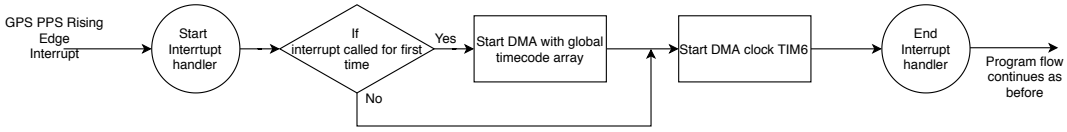
The timecode is stored in both string and integer format. The string representation is 100 characters in length and uses 800 Bytes. The size of the integer representation depends on the sample frequency with which the timecode is constructed. Each 10 ms symbol of the IRIG-B message consists of up to 8 ms of a 1 kHz carrier and 2 ms of silence. If a single period of the carrier is to be represented by n samples, each symbol consists of $10n$ samples, and the timecode consists of $1000n$ samples. The samples are represented as 8-bit integers, which gives $1000n$ Bytes. $n = 5$ was chosen, which uses 5 kB, just under half the total memory. This also means that the clock speed at which these samples should be output is 5 kHz, of which the implementation is discussed in Section 3.4.4.



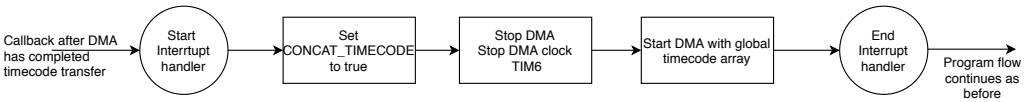
(a) Main loop



(b) Function concat_timecode()



(c) PPS interrupt routine



(d) DMA transfer complete interrupt routine

Figure 3.9: Flow diagram of timecode encoding process implemented on micro-controller.

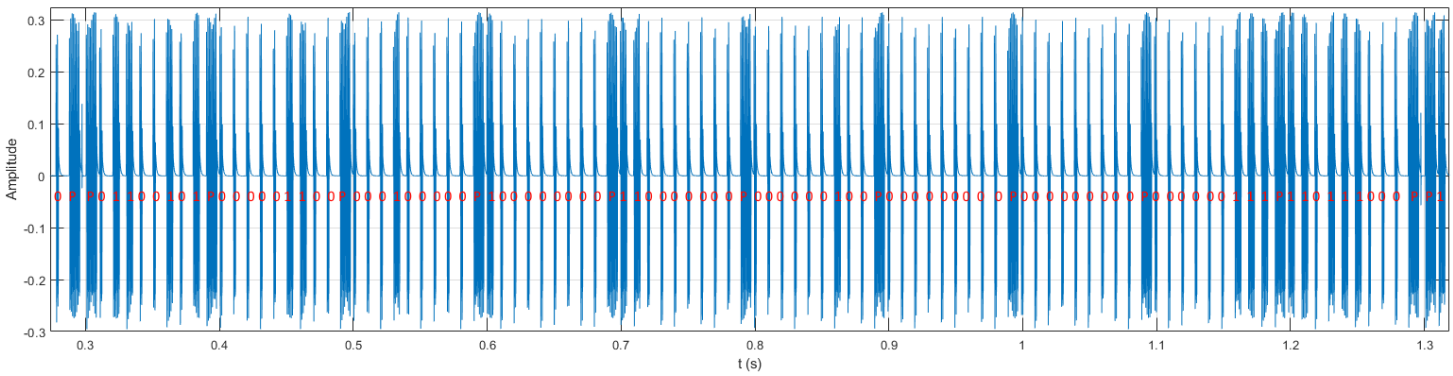
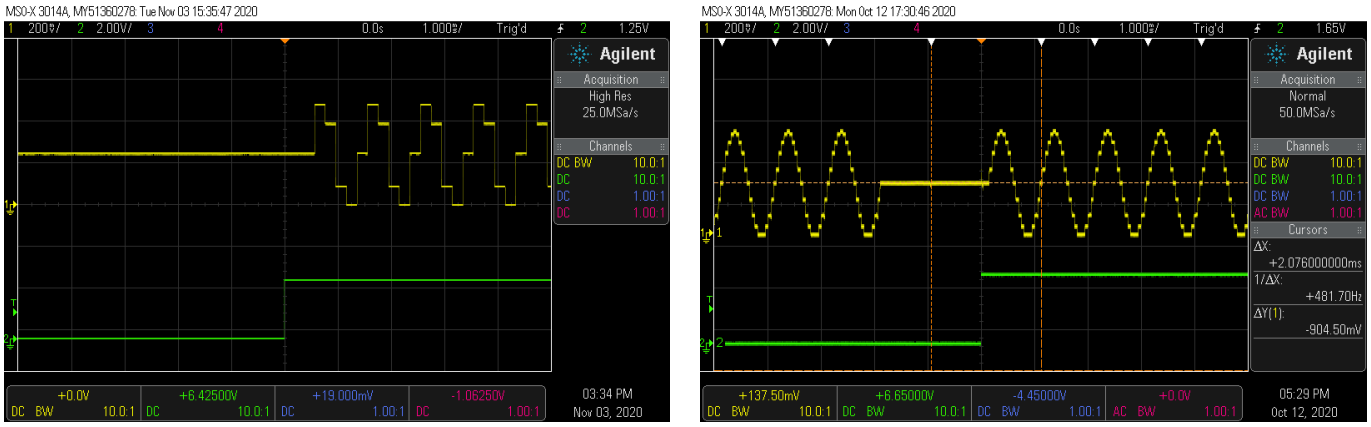


Figure 3.10: Timecode on left channel of recorder with symbols annotated.



(a) 5 Samples per period

(b) 20 Samples per period

Figure 3.11: Measurement of timecode (Ch1) and PPS (Ch2) with differing samples per period

In the above approach the entire timecode is assembled in memory, and the DMA is started once. This places minimal strain on the CPU, since it only has to handle the DMA once per second. An alternative approach was tested to reduce the required memory. By storing only a fraction $\frac{1}{k}$ of the timecode in memory, the array length is changed to $1000\frac{n}{k}$ Bytes, but the clock speed is still $1000n$ Hz. $n = 20$ and $k = 100$ were implemented and tested in Figure 3.11b. This places much greater stain on the CPU, since it has DMA transfer complete interrupt triggers once every 1 ms. This strain causes it to miss UART interrupts, meaning that data from the GPS cannot be read.

It was decided to rather stored the complete timecode in memory, due to the smaller CPU load. An additional advantage of the low amount of samples is that the rising edges of each 1 ms period are clear and easy to detect. This was used in the decoding process in the next section.

3.8. Timecode decoding

The decoding of the timecode occurs on the PC. The audio data provided by the recorder consists of two audio channels, encoded as a 16-bit stereo .wav file. As mentioned before, the right channel contains audio from the microphone and the left contains the timecode. The timecode is detected using the following algorithm:

1. Detect edges and store boolean array of where audio signal rises above a certain THRESHOLD value. See Figure 3.12.

2. Starting from the beginning of array of edges, find the largest number of edges separated by no more than TIMEOUT samples and store these as integers, together with the index or sample number of the first edge

3. Replace every integer with its corresponding character:

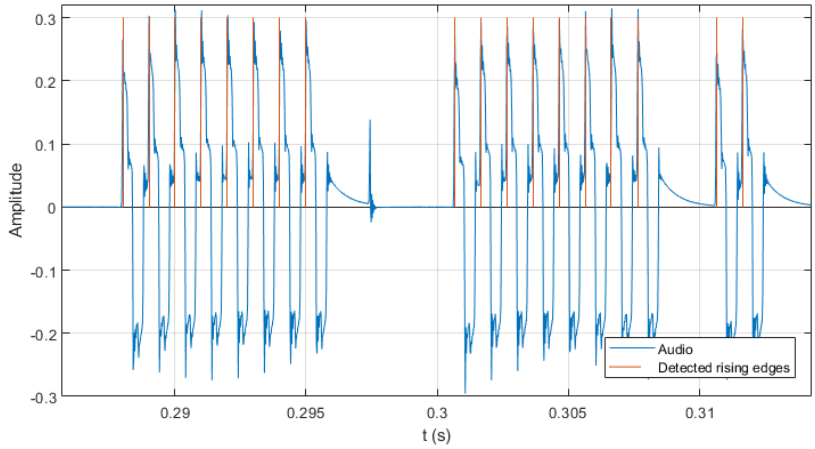


Figure 3.12: Rising edges detected in step 1 of decoding algorithm.

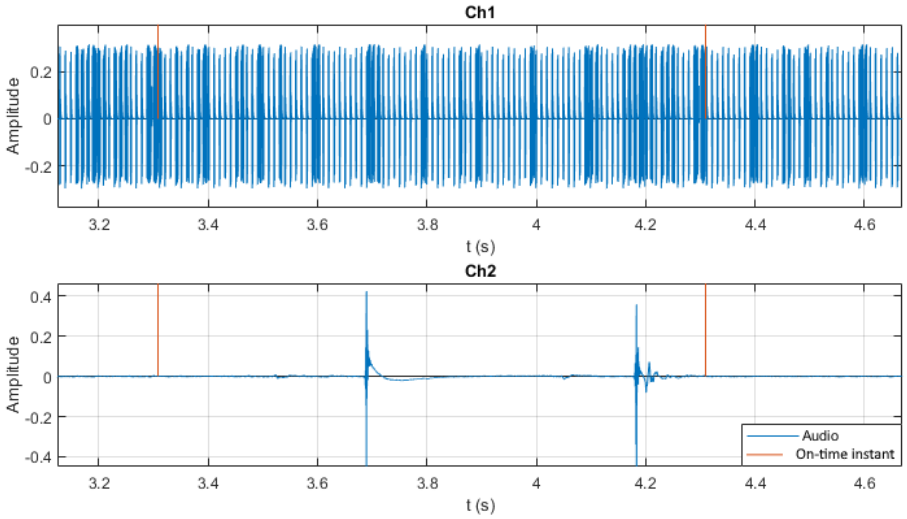


Figure 3.13: Audio recorded on both channels of recorder with on-time instants indicated.

- 8 with 'P'
- 5 with '1'
- 2 with '0'
- anything else with 'E'

4. Split this character string into messages. Criteria for a message is that it must with the latter of two consecutive P's, and must be 100 characters in length. The index of the first rising edge of the message is also stored.

5. Parse all time information from each string, and save it with the index of the first rising edge of the message to a .csv file.

Figure 3.12 shows an excerpt of the recording in Figure 3.10, with rising edges indicated. Note that this is the audio normalised to the maximum 16-bit recorded integer value. The rising edge at 0.309 s is an on-time instant, since it marks the start of the latter of

two consecutive P's. The THRESHOLD used here and for all decoding was 0.175, which we can see excludes the spike at 0.975 with amplitude 0.1. The TIMEOUT delay is 72 samples, or 1.5 ms which is larger than the expected time between edges of the same symbol of 1 ms, but is shorter than the minimum time between symbols which is 2 ms after a P.

The outputs of this decoding algorithm are the date and time information for each second, as well as the sample number or sample index corresponding to all on-time instants. Because MATLAB was used, these sample numbers start at 1. Given the wired connection between the signal conditioning output and the recorder input, the signal to noise ratio of the recording should be very high. Therefore, a very low error rate is expected of this decoding technique. The symbol error rate and synchronisation accuracy are tested in Chapter 4.

3.9. Localisation

The localisation technique that was decided upon was the time difference of arrival method described in Section 2.2.1.

The hyperbola in Equation 2.2 can be rewritten for the 2-dimensional case. The locus of source positions x, y given the positions of the receivers r_1, r_2, the time difference of arrival t_d and the speed of sound in the medium c_0, is shown in Equation 3.4.

$$\sqrt{(x - r_{1x})^2 + (y - r_{1y})^2} - \sqrt{(x - r_{2x})^2 + (y - r_{2y})^2} = t_d c_0 \tag{3.4}$$

For three receivers, there are three pairwise time differences and three hyperbolas. Assuming no errors are introduced, the time difference between receivers 2 and 3 contains no new information and can be calculated from the time difference between receivers 1 and 2, and receivers 1 and 3. This gives two non-linear equations with two unknowns — the x and y coordinates of the source.

In our case only two recorders were built, which gives one equation with two unknowns. In order to be able to localise the source, it was decided to constrain it to one dimension, by setting $x = 0$. Practically this meant that we assume the source is always on the line passing through both devices, and also between the two recorders. An plot of this configuration is shown in Figure 4.6d.

In software the solution was found by setting up the two equations in MATLAB and using the numeric solver `vpasolve`.

This procedure was followed to localise a sound source based on two audio recordings:

1. Copy audio files from both recorders to PC.

2. Import audio files into MATLAB as Nx2 arrays.

3. Run decode script (which is described in Section 3.8) on both files, save output .csv files.

4. Run analyse script to check for skipped seconds or decoding errors.

5. Take first 'second of day' that to common in both audio files, plot both from this second to end of recording.

6. Visually inspect recordings and find 'seconds of interest' and store as array.

7. Provide localise script with these seconds, and run localise script.

8. For each 'second of interest', view the audio from both recorders, the cross-correlation and predicted location.

The localise script mentioned above is now described:

```
audio1 = audio from recorder1
audio2 = audio from recorder2
second_0 = highest of first second of day of each recording

max_delay = distance between recorders/speed of sound

for each second of interest
    snippet1 = 1.5 s duration section of audio1 starting at second_0+second of interest
    snippet2 = 1.5 s duration section of audio2 starting at second_0+second of interest
    calculate the cross-correlation of snippet1 and snippet2
    find the maximum value of the cross-correlation for delays in the range ±max_delay
    calculate the position of the source by solving the intersection of hyperbola and x=0
end
```

Chapter 4

Practical Measurements and Results

4.1. Timecode

4.1.1. Timing Accuracy

For multiple separate recorders to be synchronised the difference between the on-time instants of timecodes need to be as small as possible. The difference between on-time instants can be divided into two parts: difference in PPS signals from GPS modules and difference in delay between input and output of the designed system.

The tests were conducted by connecting the two signals of interest to an oscilloscope. This oscilloscope has the ability to calculate the statistics of a number of measurement types. Here the time delay between the rising edges of the two signals was measured. This unfortunately does not allow us to view the actual distribution of measurements, only the mean, minimum, maximum and standard deviation. Based on short term observations, the difference between PPS signals (Table 4.1) was mostly close to the mean, but drifting over time to be slightly either above or below it. It is assumed that this measurement distribution is approximately Gaussian. For the other measurements however (Tables 4.2 and 4.3), most values would be close to the mean with sporadic outliers near the minimum and maximum, indicating a non-Gaussian distribution. The standard deviation is therefore not used in evaluating these measurements.

Table 4.1: Statistics of difference between on-time instants PPS signals from different GPS modules.

Mean	Minimum	Maximum	Standard Deviation	# of Measurements
−30.1 ns	−1092.5 ns	213.0 ns	134.6 ns	8547

The difference in on-time instants of PPS signals from different GPS modules is measured in 4.1. Here it can be seen that the mean difference is the same as the 30 ns specified in [15]. However, the largest absolute difference is 1.09 µs, which is larger than that found by [13].

The difference in delays between between the on-time instants of the PPS signal of the GPS module that of the timecode output by the micro-controller is measured in Table 4.2.

Table 4.2: Statistics of delays between on-time instants of GPS PPS signal and timecode output of system.

	Mean (μ)	Minimum	Maximum	Standard Deviation	# of Measurements
System 1	568.46 µs	μ−3.64 µs	μ+19.74 µs	1.12 µs	14.18×10^3
System 2	566.21 µs	μ−3.59 µs	μ+19.53 µs	1.03 µs	12.79×10^3

Table 4.3: Statistics of synchronisation accuracy of system 1 and system 2.

Mean	Minimum	Maximum	Standard Deviation	# of Measurements
−2.154 µs	−21.304 µs	17.322 µs	1.510 µs	3698

System 1 is equipped with the GOOUUU GPS and system 2 with the Adafruit GPS, but are otherwise identical. The mean delay differs by 2.25 µs, the minimum by 0.05 µs and maximum by 0.21 µs.

Table 4.3 gives the differences of the on-time instants of the timecodes. This is equal to the the synchronisation accuracy of the two systems.

At a sampling rate of 48 kHz, the sampling period is 20.83 us. The synchronisation error of the two recorders was always smaller than 2 samples and (based on short term observation) mostly smaller than 1 sample.

4.1.2. Decoding Accuracy

In order to test the decoding accuracy, the two systems were powered on, but with no microphone attached. The recorders were then switched on for more than an hour. The audio was decoded as described in Section 3.8, but instead retaining only the resultant time and date values, the message was stored as a character string `P01100101P01...`. Given a message, the content of the next message is known; it should contain the same time information incremented by one second. The algorithm below was executed to calculate

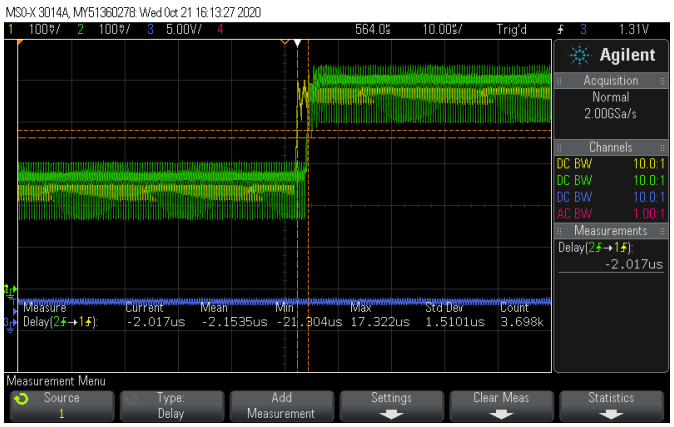


Figure 4.1: Measurement of synchronisation accuracy between System 1 (Ch2) and System 2 (Ch2).

the error rate.

```
for i = 2 to number of messages
    decode i-1'th message
    increment decoded time by 1 second
    encode predicted message with time
    compare i'th message with predicted message
    if any characters differ between messages
        count number of differences and log to file
    end
end
```

In an hour long test, 428600 symbols encoded by system 1 were decoded without fault, and 429100 symbols from system 2 were decoded with one fault. The fault was that the voltage spike that occurs between timecodes (visible in Figure 3.12) crossed the threshold used for rising edge detection as part of the decoding algorithm. If we interpret this as a single error we arrive at a symbol error rate of $\frac{1}{857700} = 1.17 \times 10^{-6}$.

For the practical localisation, the most important information is the seconds per day and the on-time sample, indicating the start of each second. If the GPS loses its fix and does not output a PPS pulse, no timecode is encoded. It was verified that no seconds were dropped or encoded more than once by counting the number of samples per second in the recorded audio. We expect this to be 48000, since that is the sampling frequency of the recorder. In Table 4.4 we can see that the results for both systems are similarly distributed. 47999 samples per second occurred the most frequently (79% for system 1 and 76% for system 2), with a number of deviations therefrom. It was verified that the spacing of different values occur sporadically, and are not grouped together.

Table 4.4: Measured number of samples per second for $N_1 = 4286$, $N_2 = 4291$ seconds.

Samples per second	Occurrence in System 1	Occurrence in System 2
47998	10	10
47999	3384	3241
48000	888	1037
48001	4	3

4.2. Localisation

A field test was conducted during which a sound source was localised from data gathered by two systems spaced 10 m apart in an courtyard. 158 measurements were taken over two days with little wind, and the testing procedure was as follows:

1. Place both devices in the open 10m apart.
2. Connect power to both devices.
3. Connect microphones and position them to point directly toward each other.
4. Connect recorders.
5. Wait 3 mins to ensure both systems have a GPS fix.
6. Connect PC to both, use serial communication to set GPS modules to correct mode and to verify both are receiving messages.
7. Start both recorders.
8. Stand close to recorder 1, make sound, move 1 m towards recorder 2, repeat until recorder 2 is reached.
9. Stop both recorders, make note of type of test performed.

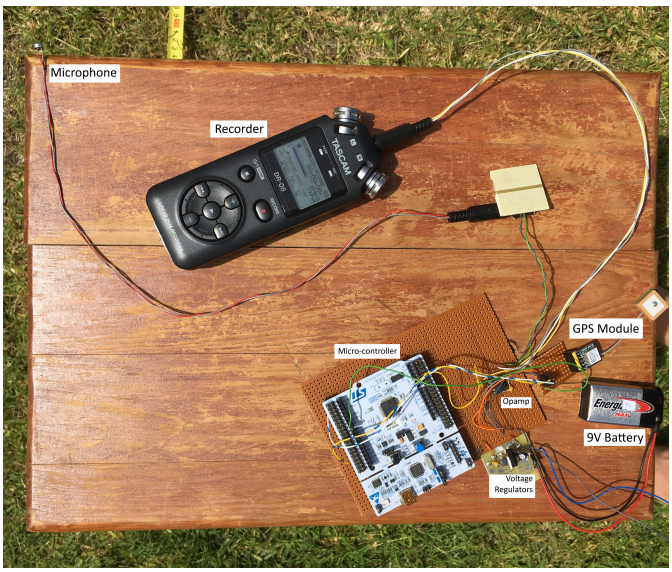
Figure 4.2a shows a photo of the test setup. In Figure 4.2b the complete system is shown with annotations. On the recorder in this photo it is possible to see that there is a large input on the left channel and no input on the right. The timecode is constantly recorded on the left channel and the microphone on the right channel is not picking up any sound. Figure 4.2c shows the microphone pointing towards recorder 1 in the distance, with the measuring tape lying on the line between the recorders.

Two experimental conditions were varied across tests. Firstly, for the sound source either 2 hand-claps in quick succession or hitting a spoon to the sides a coffee mug were used (Figure 4.4). Secondly, the microphones were tested in two configurations, either two RS PRO microphones, or one RS PRO microphone and one Rødelink microphone (Figure 4.5). The reason for variation in sound source is to see whether one type of sound source is localised much more accurately than the other. The microphones were varied for the same reason. The reason both microphones were not changed was that only one Rødelink was available.

Figure 4.3 shows the localisation errors made for all microphone and sound source configurations. It can be seen that in 88.9% of the 158 cases, the localisation distance errors were smaller than 1 m.

In Figure 4.4 it can be seen that the localisation errors were smaller for the spoon-in-mug sound than for clapping twice. This is likely because of the longer duration of the sound, which gives a larger peak in when they are cross-correlated than the short impulse-like peak of the clapping. See Figure C.2 in Appendix C for an incorrectly localised source. The influence of the detected sound on the localisation accuracy should further be investigated, but this is outside the scope of this project.

(a) Recorders spaced 10 m apart.



(b) Annotated system.

(c) Microphone and measuring tape.

Figure 4.2: Photos of testing setup and system.

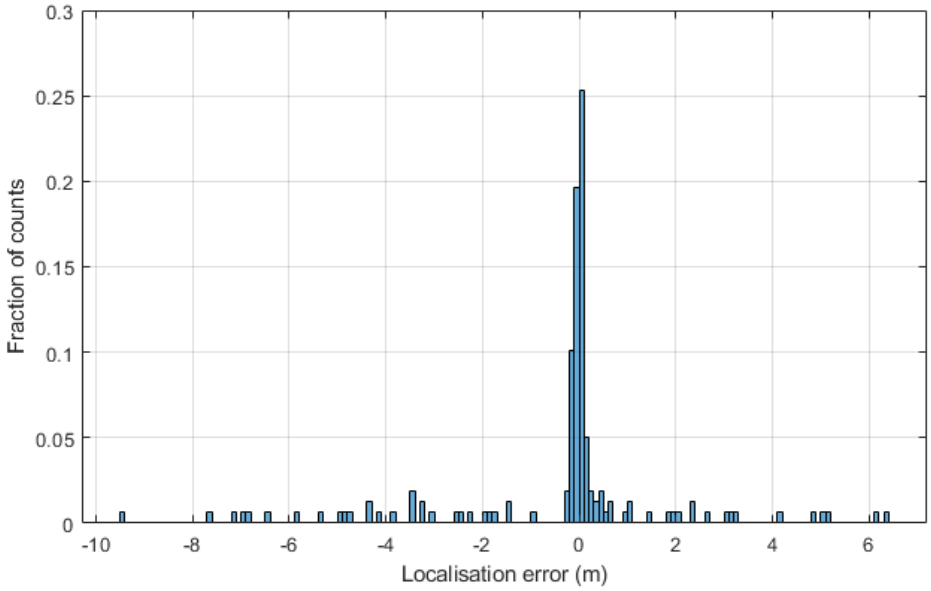
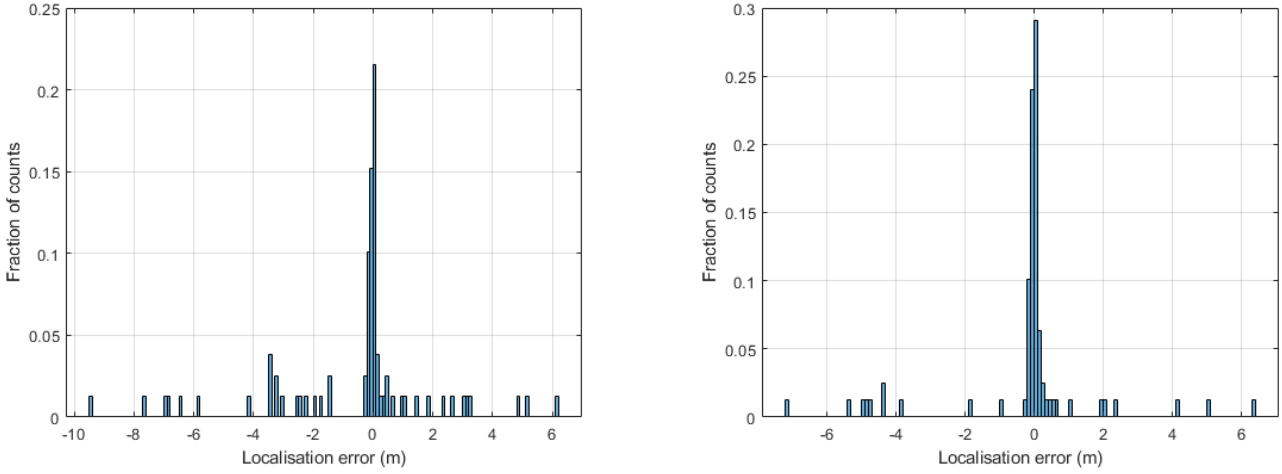


Figure 4.3: Normalised distribution localisation error of all measurements.

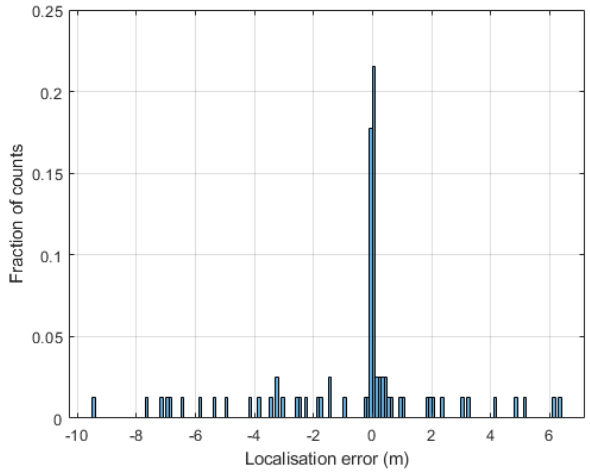
In Figure 4.5 it can be seen that the localisation errors were smaller for the configuration with the Rødelink microphone than without.



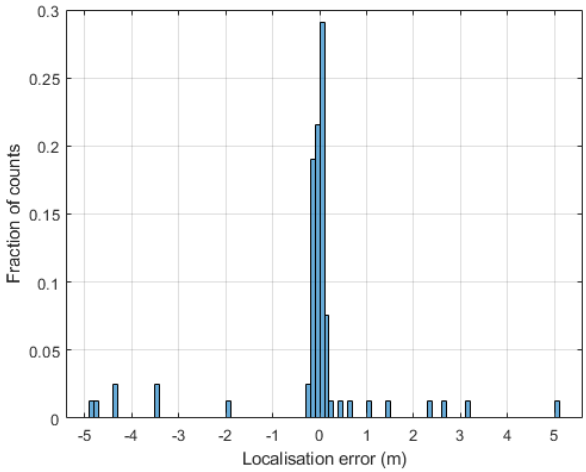
(a) Two claps ($\sigma = 2.65$)

(b) Spoon-in-mug ($\sigma = 1.97$)

Figure 4.4: Normalised distribution localisation error different sound sources with standard deviation σ.



(a) Two RS PRO microphones ($\sigma = 2.96$)



(b) One RS PRO microphone, one Rødelink ($\sigma = 1.44$)

Figure 4.5: Normalised distribution localisation error different sound sources with standard deviation σ

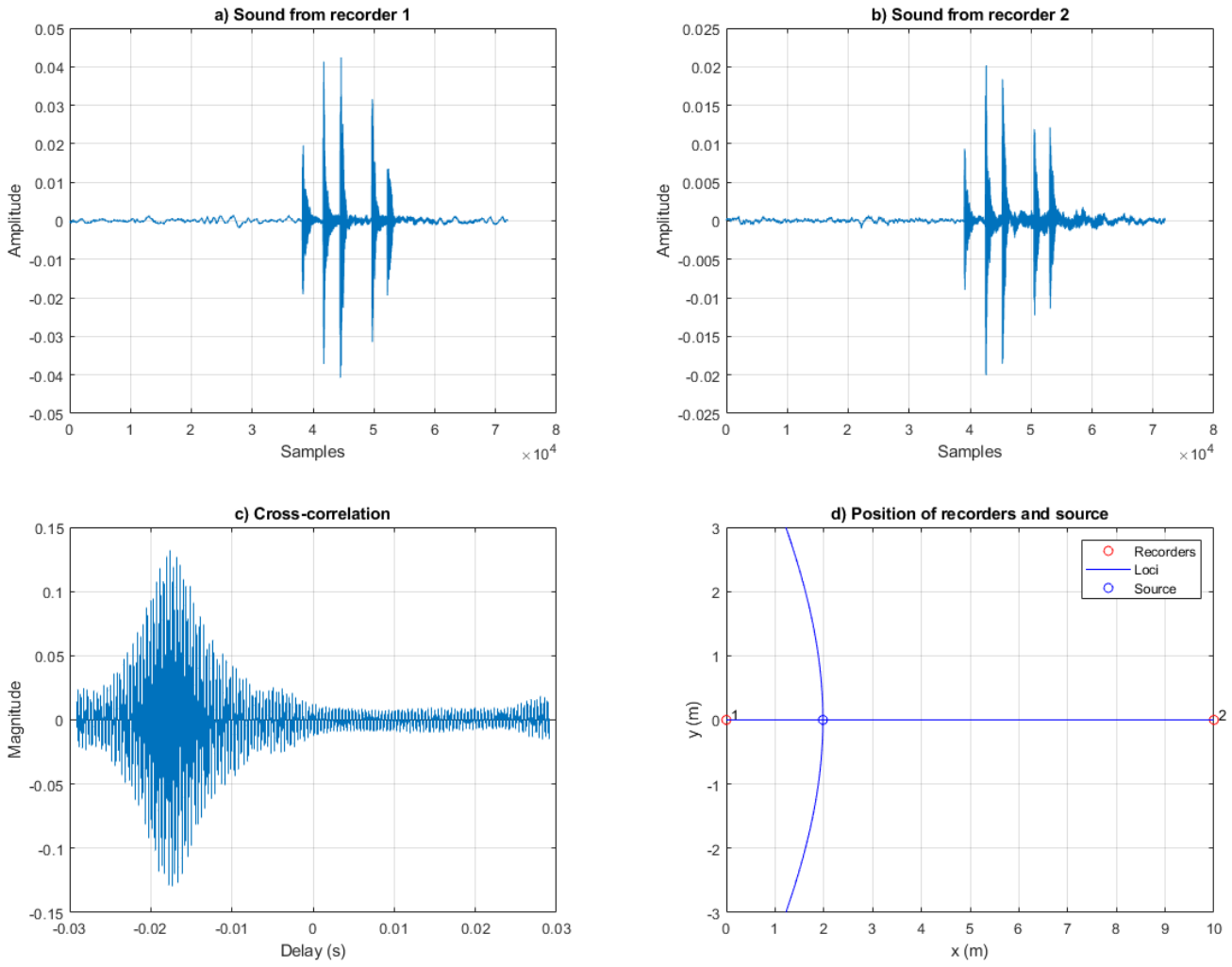


Figure 4.6: Successful localisation of source at (2, 0)

Chapter 5

Summary and Conclusion

5.1. Summary

In this report we have designed, implemented and tested a method allowing the embedding of accurate time stamps in an audio recording made using a commercially available field recorder. This method was developed to allow the accurate time synchronisation of two or more recordings made by separate recorders at different locations with a view to the localisation of elephants based on their vocalisations. The hardware and software was designed to output synchronised time information on an audio channel, and software to decode the embedded timing information was written. Successful encoding and decoding was confirmed experimentally, as was the feasibility of localisation using signals captured in this way using two physically separate recorders.

5.2. Conclusion

The maximum synchronisation interval between two separate devices with different GPS modules was found to be 21 µs. At a sampling rate of 48kHz this is at most two samples. The decoding algorithm was able to decode 857700 symbols with one error. In a field test with two recorder units, 88.9% of 158 measurements were localised to within 1m.

From this we conclude that the main aim of the project is achieved; it is possible to synchronise two separate audio recorders with sufficient accuracy to localise a sound source to within 1 m. Furthermore, it is possible to implement this solution using inexpensive hardware that can in principle be powered indefinitely using battery and solar power if the appropriate energy harvesting system is used.

5.3. Future Work

Specifically relating to the system itself, in order to be suitable for field use this system still needs to be built on a PCB and placed in some sort of protective container. The expected operation time is 6.25 h, which could be improved by redesigning the voltage

regulation and choosing a different power source. The PCB design will also make the design more easily repeatable when manufacturing multiple units.

The process of choosing which parts of the audio to localise could be more automated than the manual selection which is currently used. A more advanced approach to both the cross-correlation and localisation algorithm can be considered in order to improve the accuracy. The maximum clock speed of the micro-processor was not used, and increasing the clock-speed from 8 MHz to 32 MHz could improve the timing accuracy of the timecode encoding. Optimising the embedded code for speed could also help, but will likely need more stringent enforcement of access permissions to variables.

From the project objective of localising elephants, it will need to be verified whether audio equipment has a sufficient sensitivity and frequency range to detect these localisations, especially in the presence of ambient noise and wind. Due to the standard 3.5mm stereo interfaces with both the microphone and the recorder it will be easy to change to a different configuration.

It is also expected that field tests with real elephants will come with its own challenges, which will need to be solved before it can be said that the system can successfully localise elephants based on their vocalisations.

Bibliography

[1] *Application Note DK9222-0213-0063: Reading SMPTE timecode information*, Beckhoff Automation, February 2013.

[2] *IRIG STANDARD 200-16: IRIG Serial Time Code Formats*, Range Commanders Council Telecommunications and Timing Group, August 2016.

[3] *TN-102: Overview of IRIG-B Time Code Standard*, Cyber Sciences, 2016.

[4] J. J. Blanc, R. F. W. Craig, H. T. Dublin, C. R. Thouless, I. Douglas-Hamilton, and J. A. Hart, *Elephant Status Report 2007, An update from the African Elephant Database*, 1st ed. Gland, Switserland: International Uninon for Conservation of Nature and Natural Resources, 2007.

[5] G. Bradshaw, A. N. Schore, J. L. Brown, J. H. Poole, and C. J. Moss, "Social trauma: early disruption of attachment can affect the physiology,behaviour and culture of animals and humans over generations," *Nature*, vol. 433, p. 307, 2005.

[6] K. Kangwana, Ed., *Studying Elephants*, 1st ed. Nairobi, Kenya: African Wildlife Foundation, 1996.

[7] J. M. Mukeka, J. O. Ogutu, E. Kanga, and E. Røskaft, "Spatial and Temporal Dynamics of Human–Wildlife Conflicts in the Kenya Greater Tsavo Ecosystem," *Human-Wildlife Interactions*, vol. 14, pp. 255–272, 2020.

[8] C. M. Dissanayake, R. Kotagiri, M. N. Halgamuge, and B. Moran, "Improving Accuracy of Elephant Localization Using Sound Probes," *Applied Acoustics*, vol. 129, p. 92–103, January 2018.

[9] R. W. Byker, "Infrasoniese, GPS-gesinkroniseerde klankopnemers vir die opsporing van olifantdreungeluide," Master's thesis, Stellenbosch University, June 2016.

[10] K. McComb, D. Reby, L. Baker, C. Moss, and S. Sayialel, "Long-distance Communication of Acoustic Cues to Social Identity in African Elephants," *Animal Behaviour*, vol. 65, pp. 317–329, 2003.

[11] G. Alari and A. Ciuffoletti, "Implementing a Probabilistic Clock Synchronization Algorithm," *Real Time Systems*, vol. 13, pp. 25–46, 1997.

[12] E. F. Arias and G. Petit, "The Hyperfine Transition for the Definition of the Second," *Annalen der Physik*, vol. 531, 2019.

[13] K. Y. Koo, D. Hester, and S. Kim, "Time Synchronization for Wireless Sensors Using Low-Cost GPS Module and Arduino," *Frontiers in Built Environment*, vol. 4, no. 82, 2019.

[14] Garmin, "About GPS," 2020, Accessed on: 07/11/2020 [Online]. Available: https://www.garmin.com/en-US/aboutgps/.

[15] *Adafruit Ultimate GPS*, Adafruit Industries, 06 2020.

[16] J. L. Spiesberger and K. M. Fristrup, "Passive Localization of Calling Animals and Sensing of their Acoustic Environment Using Acoustic Tomography," *The American Naturalist*, vol. 135, pp. 107–153, January 1990.

[17] N. Zhu and T. Reza, "A Modified Cross-correlation Algorithm to Achieve the Time Difference of Arrival in Sound Source Localization," *Measurement and Control*, vol. 52, pp. 212–221, 2019.

[18] *SMPTE Made Simple*. TimeLine Vista, 1996.

[19] B. Dickerson, *IRIG-B Time Code Accuracy and Connection Requirements with comments on IED and system design considerations*, Arbiter Systems, 2006.

[20] H. G. Berns and R. J. Wilkes, "GPS time synchronization system for K2K," in *1999 IEEE Conference on Real-Time Computer Applications in Nuclear Particle and Plasma Physics. 11th IEEE NPSS Real Time Conference*, 1999, pp. 480–483.

[21] *UM1724 User manual: STM32 Nucleo-64 boards*, 12th ed., STMicroelectronics, December 2017.

[22] *TLC227x, TLC227xA: Advanced LinCMOS Rail-to-Rail Operational Amplifiers*, Texas Instruments, March 2016.

[23] Masterclock, "GPS vs. GNSS: Understanding PNT Satellite Systems ," January 2019, Accessed on: 27/08/2020 [Online]. Available: https://www.masterclock.com.

[24] *UM1786 User Manual: Description of STM32F3 HAL and low-layer drivers*, 7th ed., STMicroelectronics, July 2017.

[25] *TASCAM DR-05 Reference Manual*, TASCAM, N.d.

[26] *Electret Condenser Microphone KECG2240PBJ*, Kingstate Electronics, August 2005.

[27] *RØDELink LAV*, RØDE, N.d.

Appendix A

Project Planning Schedule

Week of	Work planned	Work done
13 Jul	Background research	Started reading up on timecodes
20 Jul	”	”
27 Jul	Design prototype	Started writing of literature review
3 Aug	Design prototype	Planning, investigating acoustic localisation, micro-controller selection, conceptual design
10 Aug	Design/Build prototype	Acquire recorders, designed GPS to micro-controller interface, achieved micro-controller to PC communication
17 Aug	Build prototype	Investigated acoustic localistion, wrote parts of literature review, powered up GPS module
24 Aug	”	Got DAC working with DMA and triggering on external interrupt
31 Aug	Evaluate prototype, improve design	Selected microphones and other components
7 Sep	”	Re-evaluate project progress and planning
14 Sep	”	Planned structure of report, selected GPS modules
21 Sep	Finalise design	Ordered additional GPS, built and tested completed system
28 Sep	Test and debug design	Started working in the DSP lab, improved system performance, designed and built voltage regulation circuit
5 Oct	”	Tested full system, received ordered components, wrote decoding and analysing script
12 Oct	Write documentation, PCB Manufacture, Field tests	Tested different ways of storing timecode in memory, wrote localisation script, designed and built HPF, improved timecode accuracy, designed experiments to show functioning of system
19 Oct	”	Tested final timecode accuracy, performed preliminary field tests
26 Oct	Write report	Improved localisation script, performed first round of field tests, wrote system design section of report
2 Nov	”	Report writing
9 Nov	Report hand in	Report hand in

Appendix B

Outcomes Compliance

In this appendix it is stated how each required ECSA ELO was achieved during execution of the project, with reference to the relevant report sections.

ELO 1: Problem Solving

Identify, formulate, analyse and solve complex engineering problems creatively and innovatively.

The problem of recorder time-synchronisation was divided into sub-problems, and this problem was solved. This problem required information from various sources as discussed in the literature review (Chapter 2) and referenced throughout.

During the process many other minor problems were identified, analysed and solved, for example the DAC on-board op-amp voltage spike in Section 3.4.1.

ELO 2: Application of scientific and engineering knowledge

Apply knowledge of mathematics, natural sciences, engineering fundamentals and an engineering speciality to solve complex engineering problems.

Knowledge of different types was gathered in the literature review (Chapter 2). Mathematical and engineering knowledge was used throughout to design and solve problems (e.g. Section 3.9) through physical circuits and code and to evaluate results in Chapter 4.

ELO 3: Engineering design

Perform creative, procedural and nonprocedural design and synthesis of components, systems, engineering works, products or processes.

The system and all subsystems were designed in Chapter 3.

ELO 4: Investigations, experiments and data analysis

Demonstrate competence to design and conduct investigations and experiments.

Several intermediate experiments were performed during the design process in Chapter 3. The data obtained from experiments in Chapter 4 were analysed and illustrated using different plots.

ELO 5: Engineering methods, skills and tools, including information technology

Demonstrate competence to use appropriate engineering methods, skills and tools, including those based on information technology

Several different software was used including MATLAB for data processing, STM32CubeIDE for writing embedded C code, GitHub for version control of code and this report, TexMaker for writing this report in LaTeX.

ELO 6: Professional and technical communication

Demonstrate competence to communicate effectively, both orally and in writing, with engineering audiences and the community at large.

This report of approximately 15 000 words was written to meet this requirement in part. The poster will be submitted on 13 November 2020.

ELO 8: Individual work

Demonstrate competence to work effectively as an individual.

This work described in this report demonstrates this outcome: the building and testing were done solely as an individual.

ELO 9: Independent learning ability

Demonstrate competence to engage in independent learning through well-developed learning skills.

The gathering of knowledge from several different sources such as academic articles, datasheets and the internet is illustrated in Chapter 2. An awareness of the context, social and ethical implications of the work is illustrated in Section 1.1.

Appendix C

Additional Material

C.1. GitHub Repository

All code in this project is publicly available on GitHub at

https://github.com/20790988/SkripsieCode

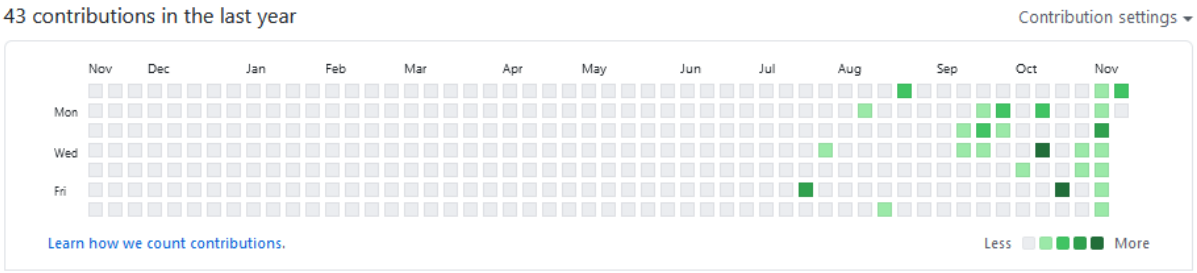


Figure C.1: Activity Map of GitHub commits.

C.2. Additional Figures

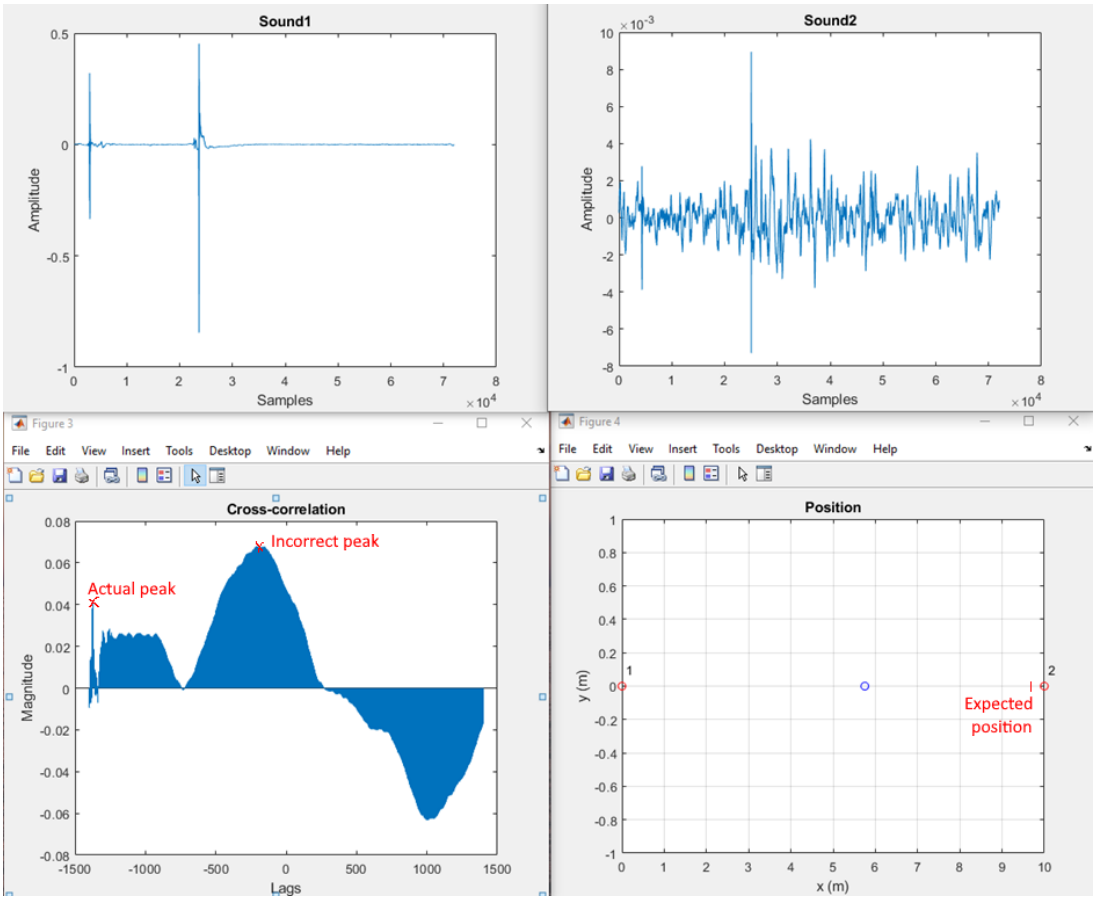


Figure C.2: Incorrectly localised sound source.

Table 5-4. Format B, Signal B000

BCD Time-of-Year Code (30 Digits)

Seconds Subword			Minutes Subword			Hours Subword			Days Subword				
BCD Code Digit No.	Subword Digit Wt Seconds	Bit Time[1]	BCD Code Digit No.	Subword Digit Wt Minutes	Bit Time	BCD Code Digit No.	Subword Digit Wt Hours	Bit Time	BCD Code Digit No.	Subword Digit Wt Days	Bit Time	BCD Code Digit No.	Subword Digit Wt Days	Bit Time
Reference Bit		P_r	8	1	P_r + 100 ms	15	1	P_r + 200 ms	21	1	P_r + 300 ms	29	100	P_r + 400 ms
1	1	P_r + 10 ms	9	2	P_r + 110 ms	16	2	P_r + 210 ms	22	2	P_r + 310 ms	30	200	P_r + 410 ms
2	2	P_r + 20 ms	10	4	P_r + 120 ms	17	4	P_r + 220 ms	23	4	P_r + 320 ms	Index Bit		P_r + 420 ms
3	4	P_r + 30 ms	11	8	P_r + 130 ms	18	8	P_r + 230 ms	24	8	P_r + 330 ms	Index Bit		P_r + 430 ms
4	8	P_r + 40 ms	Index Bit		P_r + 140 ms	Index Bit		P_r + 240 ms	Index Bit		P_r + 340 ms	Index Bit		P_r + 440 ms
Index Bit		P_r + 50 ms	12	10	P_r + 150 ms	19	10	P_r + 250 ms	25	10	P_r + 350 ms	Index Bit		P_r + 450 ms
5	10	P_r + 60 ms	13	20	P_r + 160 ms	20	20	P_r + 260 ms	26	20	P_r + 360 ms	Index Bit		P_r + 460 ms
6	20	P_r + 70 ms	14	40	P_r + 170 ms	Index Bit		P_r + 270 ms	27	40	P_r + 370 ms	Index Bit		P_r + 470 ms
7	40	P_r + 80 ms	Index Bit		P_r + 180 ms	Index Bit		P_r + 280 ms	28	80	P_r + 380 ms	Index Bit		P_r + 480 ms
Position Ident. (P_1)		P_r + 90 ms	Position Ident. (P_2)		P_r + 190 ms	Position Ident. (P_3)		P_r + 290 ms	Position Ident. (P_4)		P_r + 390 ms	Position Ident. (P_5)		P_r + 490 ms

Year and Control Functions (27 Bits)

Control Function Bit	Bit Time	Control Function Bit	Bit Time	Control Function Bit	Bit Time
1	P_r + 500 ms Units of Year 01	1	P_r + 600 ms	10	P_r + 700 ms
2	Units of Year 02	2	P_r + 610 ms	11	P_r + 710 ms
3	Units of Year 04	3	P_r + 620 ms	12	P_r + 720 ms
4	Units of Year 08	4	P_r + 630 ms	13	P_r + 730 ms
Index Mark	P_r + 540 ms	5	P_r + 640 ms	14	P_r + 740 ms
5	Tens of Year 10	6	P_r + 650 ms	15	P_r + 750 ms
6	Tens of Year 20	7	P_r + 660 ms	16	P_r + 760 ms
7	Tens of Year 40	8	P_r + 670 ms	17	P_r + 770 ms
8	Tens of Year 80	9	P_r + 680 ms	18	P_r + 780 ms
Position Ident. (P_0)	P_r + 590 ms	Position Ident. (P_7)	P_r + 690 ms	Position Ident. (P_6)	P_r + 790 ms

Straight Binary Seconds Time-of-Day Code (17 Digits)

SB Code Bit	Subword Digit Weight	Bit Time	SB Code Bit	Subword Digit Weight	Bit Time
1	2^0 = (1)	P_r + 800 ms	10	2^9 = (512)	P_r + 900 ms
2	2^1 = (2)	P_r + 810 ms	11	2^{10} = (1024)	P_r + 910 ms
3	2^2 = (4)	P_r + 820 ms	12	2^{11} = (2048)	P_r + 920 ms
4	2^3 = (8)	P_r + 830 ms	13	2^{12} = (4096)	P_r + 930 ms
5	2^4 = (16)	P_r + 840 ms	14	2^{13} = (8192)	P_r + 940 ms
6	2^5 = (32)	P_r + 850 ms	15	2^{14}=(16384)	P_r + 950 ms
7	2^6 = (64)	P_r + 860 ms	16	2^{15}=(32768)	P_r + 960 ms
8	2^7 = (128)	P_r + 870 ms	17	2^{16}=(65536)	P_r + 970 ms
9	2^8 = (256)	P_r + 880 ms	Index Bit		P_r + 980 ms
Position Ident. (P_9)		P_r + 890 ms	Position Ident. (P_8)		P_r + 990 ms

[1]The bit time is the time of the bit leading edge and refers to the leading edge of P_r.

Figure C.3: IRIG-B format description. Reproduced from [3].