The University of Oxford

Engineering Science

Fourth Year Project

PiCom: A Digital Communication Test Bed Based on Raspberry Pi

---

Candidate Number:

CANDIDATE NUMBER OR NAME - ??PAGE COUNT??

Supervisor

Justin Coon

Layout of title page

Trinity Term, 2018

Abstract

RED - Important information to check/change

BLUE - Sections and parts that still need writing/editing

GREEN - Formatting of sections and layout of images/figures

Here we shall have our abstract.

Write Abstract - Do I need an abstract? If so, write this last.

Contents

Todo list

Chapter 1

Introduction

This is what is going on over here.

Figure out how the Intro Chapter will be formatted

1.1 Motivation

Still need to write this whole section

Modern digital communication systems are built upon a solid foundation of modulation and coding theory. Over the years, researchers have successfully developed numerous schemes using pen and paper along with computer models. Any such scheme ultimately must be tested on a suitable hardware/software platform to prove their usefulness in practice. Standard 'software-defined radio' test beds can cost thousands of pounds. Although these test beds provide users with advanced development tools, much of their functionality is superfluous to requirement.

A Raspberry Pi is a simple, affordable ARM-based computer module that is capable of interfacing with external peripheral devices through a bank of IO ports. It is also programmable (using Python), and as such has found many uses by hobbyists and electronics/computer engineers in recent years. The purpose of this project is to develop a basic digital communication test bed using two Raspberry Pi modules (one transmitter and one receiver). The test bed will be affordable and the interested student will need to work to a budget to ensure a successful outcome. The project will require a considerable amount of Python programming as well as knowledge of, and a keen interest in, digital communication theory and techniques.

1.2 Background - Literature Review

Explanation of the existing literature [1].

Chat chat chat.

$$\mathbf{F}(t) = (\mathbf{m}(t) \cdot \nabla)\mathbf{E}(t) \tag{1.1}$$

where $\mathbf{E}(t)$ is the electric field in 150 cT.

1.3 (My) Contributions

```
import numpy as np

def incmatrix(genl1,genl2):
M = "Hello" # To become the incidence matrix
return M
```

Listing 1.1: Hello

Chapter 2

The Raspberry Pi and the Test Bed

Talk about the RPi and test bed.

2.1 Fundamentals

The Raspberry Pi is a simple, affordable ARM-based computing module. It has, which can interface

It is run using a Linux-based operating system called Raspbian which is available for download from the Raspberry Pi official website **?? ??**.

2.1.1 Blablabla

Talky talk talk

2.2 Test Bed Architecture

The test bed comprises the two Raspberry Pis and a number of chips to provide the functions required for more advanced modulation schemes. This is built up as three arrangements with increasing complexity. The first is the Pis connected together by two wires, a serial data line and a clock line (Figure 2.1). This arrangement is similar to that used for an I^2C bus, however the code written for this form of communication doesn't rely on any available modes of serial interfacing because it needs to be extensible to the parallel communication in the next arrangements. The second arrangement is used for Pulse Amplitude Modulation schemes. It uses a single parallel Digital Analogue Converter (DAC) connected to the transmitter which transmits to a parallel Analogue Digital Converter connected to the receiver, allowing for multi-level signals to be transmitted between the two devices. The final arrangement extends the set-up to two DACs and two ADCs. These signals are then multiplied by a sine wave and a cosine wave respectively, and can be separated due to the orthogonality of the two signals at the receiver. This arrangement is used for Quadrature Amplitude Modulation as well as Orthogonal Frequency Division Multiplexing.

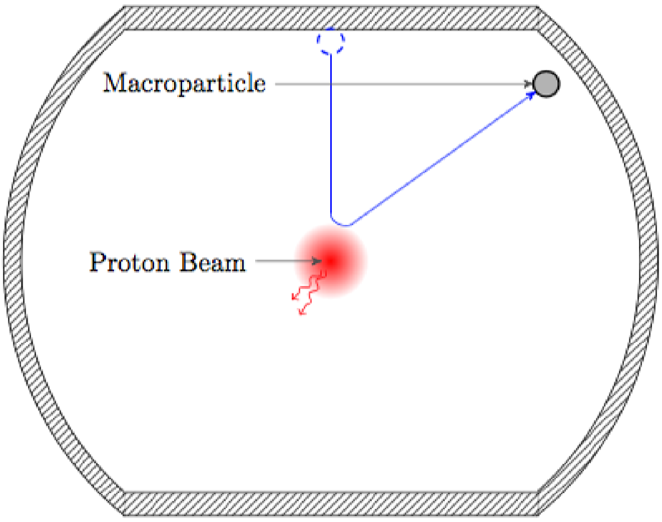

Figure 2.1: Test Bed Set-up For On-Off Keying

2.2.1 Digital Analogue Converter

The Digital Analogue Converter used is the AD5424, an 8-bit CMOS current output DAC with an easy interface to microcontrollers. It has a 17 ns write cycle and a maximum update rate of 20.4 MSPS. The converter is set up with a non-inverting operational amplifier to produce a full-range voltage output. The Read/Write ($R/\overline{W}$) pin is pulled low permanently so the chip is in write mode; read back of the parallel digital outputs is not required. The Chip Select pin ($\overline{CS}$) needs a falling edge and a rising edge to complete a write cycle, where the rising edge loads new parallel data, so this is connected to the clock pin of the Raspberry Pi. Layout of the connections to the Pi are in the Figure 2.2.1.

Missing figure

Layout of Analogue Digital Converter

2.2.2 Analogue Digital Converter

The Digital Analogue Converter used is the AD5424,

2.2.3 Parts Used

It is worth noting that there is a large variety of available options for each component of the test bed. Each possibility has certain advantages and disadvantages, and a lot of the options are not suitable due to the power requirements or ease of interfacing with the Raspberry Pi. As a result of this, the parts used in this project were the most suitable parts which could be found and successfully sourced. However, there may be more suitable chips available given more time or experience, and being aware of this would be useful if this project were to be extended and/or replicated.

Changes which would be made with hindsight, if components with the required qualities could be found, are as follows:

Continue to add to this list as you write the Architecture section - remove redundant information already discussed in earlier sections

- A number of chips used are surface mounted, requiring difficult soldering to solder pads, This would be useful if they were to be used on a printed circuit board for a final design, but on a prototyping breadboard, dual in-line packages would have been easier to use where available
- The Digital Analogue Converter was chosen for its easy interfacing with a microcontroller, but a voltage-output device would remove the need to use additional operational amplifiers at the output
- Analogue Digital Converter
- The Quadrature Sinusoid Generator uses an oscillator chip which outputs 90^o out-of-phase square waves, and the used chip was the only simple one which did this. Ideally the outputs

would already be sinusoidal (one quadrature chip or a sine generator and phase shifter) so that low pass filters with fixed frequency response could be omitted to make changing the carrier frequency purely software-dependent.

Make sure I am consistent with use of Quadrature Sinusoid Generator vs Oscillator in report

- The multiplier is designed to operate around 10 V, and so has a built in 10 V normalisation in the multiple which attenuates the signal and required re-amplification before transmission. A similar chip designed for lower voltages would be ideal.

  Fact check this

- Low Pass Filters

2.3 Programming

The Raspberry Pi is used for its low cost, ease of use, and the fact that it has programmable Input/Output (I/O) pins. The I/O pins can be programmed using different libraries in Python and C. The standard library which comes installed with Raspbian is the RPi.GPIO Python library **??**.

Add bib reference to RPi.GPIO https://pypi.python.org/pypi/RPi.GPIO

This is used for the On-Off Keying part of the communications test bed, and then a C library is used for the pin-level manipulation for all modulation schemes requiring multi-level outputs through Digital Analogue Converters. This is for the improved speed performance of the C library and for the capability of this library to output to multiple pins at once. Section 3.2 goes into a detailed investigation of the differences between these options. All of the code and the report for the project are maintained on GitHub, and may be found at `https://github.com/CamEadie/4YP_PiCom`.

The code is separated into the initial On-Off Keying (Section 2.3.1) which acts as a proof of concept for the Raspberry Pis as a test bed, and the Advanced Modulation Schemes (Section 2.3.2) which are all implemented in the same code. The Advanced section improves the data manipulation and implements a separate compiled module for the actual transmitting and receiving of data.

2.3.1 On-Off Keying

2.3.1.1 Starting the Receiver

Paramiko

2.3.2 Advanced Modulation Schemes

2.3.2.1 Data and Image Handling

2.3.2.2 C Transmitter and Receiver

Chapter 3

Electronic Testing

BLABLABLA.

Write Electro Testing

3.1 Electrical Characteristics of the Raspberry Pi

3.1.1 Maximum Frequency

3.1.2 Impedance

3.2 Comparing Python and C

3.3 Characterising Components of the Test Bed

Electrical Components:

- Analogue Digital Converter
- Digital Analogue Converter
- Quadrature Sinusoid Generator
- Multiplier/Mixer
- Low Pass Filters

3.3.1 Overclocking Components

Chapter 4

Communications Testing

Testing Communications and shizniz.

Write Comms Testing

Easily Attainable: Construct a basic wired unidirectional communication test bed complete with a transmitter and a receiver. These units should be synchronised and an appropriate line code (i.e., baseband modulation scheme) should be exploited to convey test data from one device to another.

Medium Complexity: Characterise the performance of the test bed, identifying bandwidth limitations, noise characteristics, and reliability for different modulation and coding schemes. Test specific state-of-the-art modulation techniques recently published in the research literature. (These will be identified by the supervisor).

Advanced: Develop design enhancements that will enable the test bed to be extended to wireless scenarios, including RF and optical wireless systems. Implement these modifications if the budget permits.

4.1 SNR for Different Modulation Schemes

4.2 Error Rate

4.3 Channel Coding

Chapter 5

Conclusion

Here we shall have our conclusion.

Write Conclusion

Bibliography

[1] Neil Dhir, Adam Roman Kosiorek, and Ingmar Posner. “Bayesian Delay Embeddings for Dynamical Systems”. In: *NIPS Timeseries Workshop*. 2017.