FM Transmitter for Raspberry Pi on Secure Unix Systems

Kyle Daniel Martin

1 April 2016

Project presented in partial fulfillment of the requirements for the degree of Bachelor of Science (Honours) at the University of the Western Cape

Supervised by:
Dr. Jean-Baka Domelevo Entfellner
Mentor:
Mogammat Waleed Deaney

Contents

Declaration

I, KYLE DANIEL MARTIN, declare that this project titled: “FM Transmitter for Raspberry Pi on Secure Unix Systems” is my own work and that it has not been submitted before to any institution of learning for any degree and/or assessment. All the sources I have used and/or quoted have been indicated and acknowledged by means of complete reference.

Signature:........................ Date:........................

Name in print: Kyle Daniel Martin

Abstract

Frequency Modulation (FM) broadcasting is a Very High Frequency (VHF) broadcasting technology which began with Edwin Howard Armstrong. Armstrong made use of FM to provide high quality sound broadcasts over radio. The term FM band directly dictates the frequency band which is dedicated for FM transmission. Note that the term equates a FM method within a range of frequencies. The Raspberry Pi is a credit card–sized, single-board computer. The Raspberry Pi has on-board hardware that is used to generate spread-spectrum clock signals on the General Purpose Input/Output (GPIO) pins to output FM signal. My proposal is that we will achieve a solution that results in having a FM Transmitter composed solely of the Raspberry Pi as well as source code written in the programming language C.

Acknowledgments

I would like to give thanks to my supervisor Dr Jean-Baka Domelevo Entfellner and mentor Mogammat Waleed Deaney for their continuous support and encouragement during the duration of my Honours year. I would not be where I am today nor be able produce the work I produced without our weekly meetings and constant communication via emails.

I would also like to give thanks to my family for their constant physical and emotional support towards completing my Bachelor of Science (Honours) degree.

Glossary

PWM: Pulse-Width Modulation. PWM is a method for modulating digital signals into a two-level pulsing signal with arbitrary frequency. PWM is notably used in applications controlling LEDs or stopping motors, but it may be used to emulate frequency modulation..

FM: Frequency Modulation. FM is a type of signal modulation used in radio applications. FM receivers are most widespread worldwide.

GPIO pins: General Purpose Input/Output pins. These are the 26 of the 40 physical pins that run along the edge of the Raspberry Pi. These pins are a programmable physical interface between the Pi and the outside world.

DMA: Direct Memory Access. DMA is a feature of most modern computers including the Raspberry Pi that allows certain systems to have direct access to main system memory without issuing memory read/write instructions through the CPU.

CPU: Central Processing Unit. The ”brain” of the Raspberry Pi. This hardware it used for communication between the Raspberry Pi hardware, executing processes and performing calculations. The Raspberry Pi has a Broadcom system-on-a-chip based on an ARM CPU.

SoC: System-on-a-chip or System on Chip is an integrated circuit, i.e. a set of electronic circuits on one small plate of semiconductor material. This semiconductor material; usually silicon, is called a ”chip.” The SoC integrates all components of a computer or other electronic system into a single chip.

1 User Requirements

1.1 Introduction

The following will describe the problem from the perspective of the end-user. It states the problem domain and the functionality of the programme expected by the end-user.

1.2 User's view of problem

The user's will have a simple view of the problem. Users want to transmit FM signals for the transmission of an audio source from their Raspberry Pi whilst making use of little-to-no additional hardware and as low as possible CPU overhead. The audio source may be a file or a stream from the Internet.

1.3 Description of problem

Using the Raspberry Pi to produce a FM signal may seem like a tricky task to perform as we're trying to make use of just the Raspberry Pi and the programming language C to produce the FM signal. In theory, we should be able to transmit the signal solely with the Raspberry Pi without connecting anything to the GPIO pins however, the range of the frequency will be <10 centimeters from the Raspberry Pi. If we make use of the only additional hardware we'll need i.e. a simple wire antenna connected to GPIO Pin 4 of the Raspberry Pi, it will boost the signal to a range of <10 meters which may be improved with additional power to the antenna.

1.4 Expectations of the software solution

The solution to the problem needs to ensure a user is able to transmit the audio as a FM signal that will be able to be tuned-in to by any FM receiver that is tuned onto the FM frequency transmitted by the Raspberry Pi while making use of minimal additional hardware. The audio produced from tuning into the FM signal needs to be clear and free of static noise.

1.5 Not expected of the software solution

The software will not be able to play multiple audio files at a given moment nor will it be able to emit the frequency modulation at multiple frequency bands. The software will also not produce stereo as the audio file must be in a 16-bit mono wav format. Finally not to be expected from the software is being able to adjust the frequency band of the FM signal it broadcasts whilst broadcasting the FM signal.

1.6 Conclusion

The previous section describes how the end-user requires a system that will allow them to transmit FM signals from the Raspberry Pi by describing the expectations of the system to define the scope of the project as well as stating what is not expected of the solution. The following section will explain the solution from the designer's interpretation as to solving the problem.

2 Requirements Analysis Design

2.1 Introduction

In the previous section, the end-user specified the need for a transmitting a FM signal from a Raspberry Pi while making use of minimal additional hardware. The following section will explain how the developer will attempt to meet the user's requirements. This will be done by breaking down the problem into how the C code executes and identifying all applicable details.

2.2 Interpretation and break-down of the problem

The input of the solution will be simply an audio file which the user wishes to broadcast over a FM band. The user will run the solution by entering a single terminal command which will provide the code with the name of the audio file as well as possibly the FM band they want to transmit the signal at provided it remains in valid FM frequency ranges for their specific country. In order to ensure quality of the signal, a simple single wire antenna will be attached to the Raspberry Pi on GPIO pin 4. This will result in the range of the signal being increased from <10 centimeters to <10 meters.

2.3 Complete analysis of the problem

The problem is one of security. It would be relatively simple to make use of existing code for emulating the FM transmitter on the Raspberry Pi 1 but that code makes use of a "hack"; a "hack" that will only work on GNO/Linx operating systems that can read/write to the special file "/dev/mem" and will not work on other Raspberry Pi operating systems such as FreeBSD or NetBSD. The entire objective for this project is to come up with a solution that will allow for the transmission of FM signals on a Raspberry Pi enabling the operating system to make use of two hardware features, PWM and DMA, on the Raspberry Pi in a safe way from **kernel-space**, not making use of a wild and potentially dangerous memory map accessable from user space. This entails writing complex **kernel-drivers** for the PWM and DMA on the Raspberry Pi running any genuine Raspberry Pi operating system; we will begin with NetBSD. To do this we will have to not use the "/dev/mem" special file at all and instead make use of Direct Memory Allocation (DMA) and the Pulse-Width Modulator (PWM) to allow the mapping between the memory and device.

2.4 Other similar systems

Computer Scientists over at Imperial College Robotics Society have designed a simple programme that "hacks" the Raspberry Pi into an FM transmitter. This C programme alters the peripheral bus in physical memory into its virtual address space using the **"/dev/mem"** special file and mmap commands. To do this however, root access is needed. As it makes use of the "/dev/mem" special file, it is seen a security risk as this special file is a direct call to the main memory (RAM) of the Pi from **user-space**; as all information of the current session on the Pi may be accessed using this special file which may include confidential information.

2.5 Suggested Solution

The suggested solution to the problem would be to modify the open-source code of the Imperial College Robotics Society so that it does not make use of the "/dev/mem" special file to perform the FM signal transmission. This will be done by making use of Direct Memory Allocation (DMA) to map information between the Raspberry Pi and its' memory (RAM) without needing to contact the central processing unit.

2.6 Conclusion

This final section stated how the designer interpreted and broke-down the problem. A basic and simple framework for a solution to the problem was provided in order to solve how the Raspberry Pi will be used in order to produce a FM signal using solely the on-board Raspberry Pi hardware, the programming language C and a small wire antenna to act as a signal booster.

References

- Matt Richardson and Shawn Wallace. *Getting Started with Raspberry Pi.* " O'Reilly Media, Inc.", 2012.
- Eben Upton and Gareth Halfacree. *Raspberry Pi user guide.* John Wiley & Sons, 2014.
- Justin Ellingwood. A comparative introduction to freebsd for linux users. *DigialOcean*, 2015.
- Liguo Yu, Stephen R Schach, Kai Chen, Gillian Z Heller, and Jeff Offutt. Maintainability of the kernels of open-source operating systems: A comparison of linux with freebsd, netbsd, and openbsd. *Journal of Systems and Software*, 79(6):807–815, 2006.
- Simon Monk. *Raspberry Pi Cookbook.* " O'Reilly Media, Inc.", 2013.
- John M Chowning. The synthesis of complex audio spectra by means of frequency modulation. *Journal of the Audio Engineering Society*, 21(7):526–534, 1973.
- Timothy L Warner. *Hacking Raspberry Pi.* Que Publishing, 2013.
- Oliver Mattos and Oskar Weigl. Turning the Raspberry Pi Into an FM Transmitter. http://www.icrobotics.co.uk/wiki/index.php/Turning_the_Raspberry_Pi_Into_an_FM_Transmitter, 2015. [Online; accessed 12 April 2016].
- Christophe Jacquet. FM-RDS transmitter using the Raspberry Pi's PWM . https://github.com/ChristopheJacquet/PiFmRds, 2014. [Online; accessed 12 April 2016].