University of New Hampshire
University of New Hampshire Scholars' Repository

Master's Theses and Capstones Student Scholarship

Winter 2007

Prototype mixed-signal hardware for public safety radio interoperability

Timothy Edwood Bond
University of New Hampshire, Durham

Follow this and additional works at: https://scholars.unh.edu/thesis

Recommended Citation
Bond, Timothy Edwood, "Prototype mixed-signal hardware for public safety radio interoperability" (2007). *Master's Theses and Capstones*. 323.
https://scholars.unh.edu/thesis/323

This Thesis is brought to you for free and open access by the Student Scholarship at University of New Hampshire Scholars' Repository. It has been accepted for inclusion in Master's Theses and Capstones by an authorized administrator of University of New Hampshire Scholars' Repository. For more information, please contact nicole.hentz@unh.edu.

PROTOTYPE MIXED-SIGNAL HARDWARE FOR PUBLIC SAFETY RADIO INTEROPERABILITY

BY

TIMOTHY EDWOOD BOND

BSEE, University of New Hampshire, 2005

THESIS

Submitted to the University of New Hampshire

in Partial Fulfillment of

the Requirements for the Degree of

Master of Science

in

Electrical & Computer Engineering

December, 2007

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

UMI Number: 1449578

UMI®

UMI Microform 1449578

Copyright 2008 by ProQuest Information and Learning Company.
All rights reserved. This microform edition is protected against
unauthorized copying under Title 17, United States Code.

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

This thesis has been examined and approved.

[illegible]

Thesis Director, Andrew L. Kun,
Assistant Professor of Electrical Engineering

[illegible]

W. Thomas Miller, III,
Professor of Electrical Engineering

[illegible]

William H. Lenharth,
Research Associate Professor of Electrical Engineering

12/12/05

Date

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

DEDICATION

This thesis is dedicated to my mother for her consistent love, help, and support throughout all levels of my life and education.

"... that in all things he [Christ] might have the preeminence."

- *Colossians 1:18*

"... whatsoever ye do, do all to the glory of God"

- *I Corinthians 10:31*

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

ACKNOWLEDGEMENTS

I would first like to thank my thesis advisor, Dr. Andrew L. Kun, for the time he has devoted to helping me with this project and the guidance he has provided in the writing of this thesis.

I also would like to express my gratitude to Dr. W. Thomas Miller, III and Dr. William Lenharth for the additional assistance that they have provided to me on the project and for serving on my thesis committee, offering both their time and ideas for its revision and completion.

I would also be remiss were I not to express my appreciation to the entire CATLab team for their assistance and collaboration during my time on this project.

Finally, I would like to thank my family and friends for their prayers and support while I have been at UNH.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

TABLE OF CONTENTS

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

LIST OF TABLES

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

LIST OF FIGURES

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

LIST OF ACRONYMS

AC	Alternating Current
ADC	Analog to Digital Converter
AGC	Automatic Gain Control
APCO	Association of Public Communications Officers
ASCII	American Standard Code for Information Interchange
CAB	Configurable Analog Block
CMOS	Complementary Metal Oxide Semiconductor
COTS	Commercial Off-the-Shelf
CPLD	Complex Programmable Logic Device
CPU	Central Processing Unit
DC	Direct Current
DCE	Data Circuit Terminating Equipment
DIP	Dual In-line Package
DSP	Digital Signal Processing
DTE	Data Terminal Equipment
DTR	Data Terminal Ready
EMI	Electro-Magnetic Interference
EOF	End-of-File
FCC	Federal Communications Commission
FM	Frequency Modulation

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

FPAA	Field Programmable Analog Array
FPGA	Field Programmable Gate Array
GSM	Global System for Mobile Communication
GUI	Graphical User Interface
I/O	Input / Output
IC	Integrated Circuit
IDB	Intelligent Transportation System Data Bus
IF	Intermediate Frequency
ISP	In-System Programming
ISR	Interrupt Service Routine
JTAG	Joint Test Action Group
MIC	Microphone
OPAMP	Operational Amplifier
PC	Personal Computer
PCA	Programmable Capacitor Array
PCB	Printed Circuit Board
PLD	Programmable Logic Device
PROM	Programmable Read Only Memory
PSTN	Public Switched Telephone Network
PTT	Push-to-Talk
RAM	Random Access Memory
RC	Resistor-Capacitor
RF	Radio Frequency

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

RTS	Request-to-Send
RX	Receive
SDR	Software Defined Radio
SFR	Special Function Registers
SMT	Surface Mount Technology
SNR	Signal-to-Noise Ratio
SOC	System-on-a-Chip
SVF	Serial Vector Format
TAP	Test Access Port
THD	Through Hole Device
TX	Transmit
USART	Universal Synchronous Asynchronous Receiver Transmitter
VAD	Voice Activity Detection
VHDL	VHSIC Hardware Description Language
VHSIC	Very High-Speed Integrated Circuits
VoIP	Voice Over Internet Protocol
VOX	Voice Operated Switch
WTC	World Trade Center
XSVF	Xilinx Serial Vector Format

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

ABSTRACT

PROTOTYPE MIXED-SIGNAL HARDWARE FOR PUBLIC SAFETY RADIO INTEROPERABILITY

by

Timothy Edwood Bond
University of New Hampshire, December, 2007

In performing their required duties public safety personnel from differing departments often need to communicate with one another using their in-car radios. However, in many cases, especially involving small departments, this interoperability doesn't exist.

This research develops a low-cost, low-complexity, yet high performance solution to the public safety radio interoperability problem. This is done through custom, mixed-signal hardware that can be placed in a command vehicle. The analog front-end routes audio signals between radios while the digital portion manages user interaction and controls the analog switching matrix.

A prototype circuit design has been developed and tested within the laboratory using two common radio systems: EFJohnson and Motorola. The preliminary results have shown successful operation as a system gateway between the two radio systems with good performance regarding audio signal latency and minimizing the push-to-talk signal generation delay.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER I

INTRODUCTION

CATLab - Project54 Overview

Project54, a project of the Consolidated Advanced Technologies Laboratory (CATLab), is a collaborative research and development effort between the University of New Hampshire and the New Hampshire Department of Safety supported by the U.S. Department of Justice. The project entails the introduction of advanced technologies to the New Hampshire State Police and other public safety and law enforcement agencies [1]. The goal of this effort is to allow officers to execute their responsibilities more efficiently and in a safer manner [2].

The Problem of System Gateways

In performing their required duties it is often necessary that public safety personnel, including police, fire, ambulance, and others, from different departments, be able to communicate with one another while still being able to use their familiar, in-car radio systems. However, this interoperability between divergent radios systems, frequently from different manufacturers, often does not exist, especially in cases involving smaller departments. Public Safety Radio Interoperability is an important issue as multiple agencies often need to have this ability to communicate but have

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

incompatible communications equipment which can lead to detrimental communication breakdowns, as illustrated in Figure 1.1. However, solutions the like purchasing of new inter-compatible/interoperable systems is costly and impractical as training would additionally be required. Further, others that involve the use of common channels would severely limit the band space available to each party in non-interagency activities. For these and other reasons the System Gateway general solution presents a number of tantalizing opportunities.

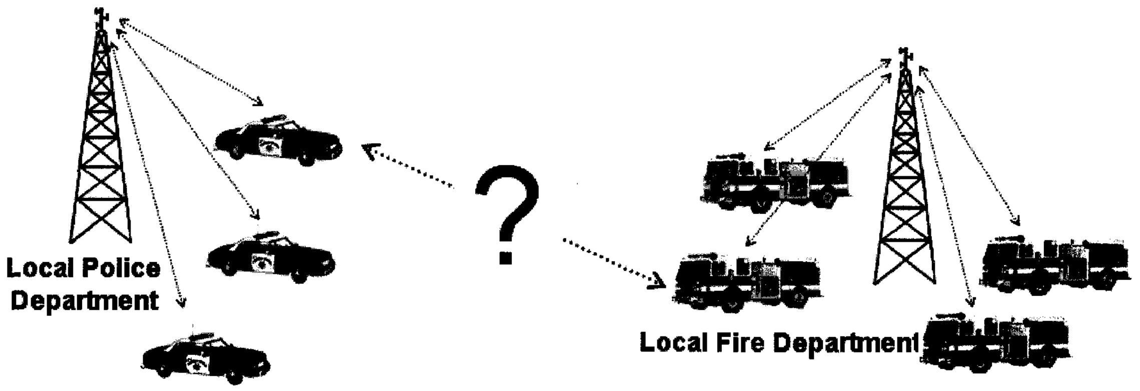


Figure 1.1 – Lack of Radio Interoperability in Public Safety

However, in the solutions that are currently available for system gateways numerous problems do exist. In general they are often too expensive, overly complicated to operate, have limited performance, or have some combination of all three of these shortcomings. For example, previous attempts using Voice-Over-IP (VoIP) have largely been unsuccessful due to the failures of the systems to function well in real time and the inflexibility of the systems, particularly in field-based operations. Note that a Personal Computer (PC), which adds to the cost and complexity, may not always be available or in operation at the field site. Further difficulties arise from the lack of dependability in solutions involving commercial network systems, such as the internet, for audio transmission.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Another shortcoming that is common to some of the products available is that, in the attempt to design a high performance system, the result can be cumbersome and ill-fitted for use in the center console area of a small command vehicle. The ACU-1000 for example is 5.25" by 19" by 11". This is much too large for a typical in-vehicle installation. Note similarly that solutions like VoIP, Software Defined Radio (SDR), and others have additional computational and processing hardware (such as Push-to-Talk generation circuitry) that can be quite large when taken as a whole system. This again can present problems for their use and field-deployment.

Also, for those few systems that have been created with a lower price tag for smaller departments, key features are often left out or the performance is unacceptable. This includes aspects such as ease of controllability which, as a hidden cost, then requires additional financial expenditures for training and/or system set up and configuration. Performance, such as the latency of the audio signal exchange, is major area that is hit in some of the available prototype designs that are coming to market. The biggest effect is found in digital software based approaches such as VoIP and SDR which involve the use of signal buffers that can significantly impair the course of a conversation.

The cost and complication issues in particular, of the majority of available systems, mean that small, local departments are often unable to capitalize on modern interoperability technologies. Instead, these more common solutions tend to focus on advanced features that, while essential for large-scale disasters, are superfluous for local issues and coordination on the small scale. There is therefore a real dearth of inexpensive, yet moderately high performance, public safety gateway solutions targeted for applications involving temporary patches to be used in major, locally-defined crises.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Goals of the System Gateway Interoperability Solution

The main goal of this work is to develop and test a System Gateway prototype communications bridge as a solution to the broader public safety radio interoperability problem. The solution must also meet a number of basic demands and solve some of the typical system gateway issues that are present in previous attempts.

Figure 1.2 shows the basic interoperability goal that the system will be designed to achieve. Note that in the original mega-system there are four distinct radio systems that are unable to communicate with each other. However, these are then bridged through the interoperability solution operated from a command vehicle.

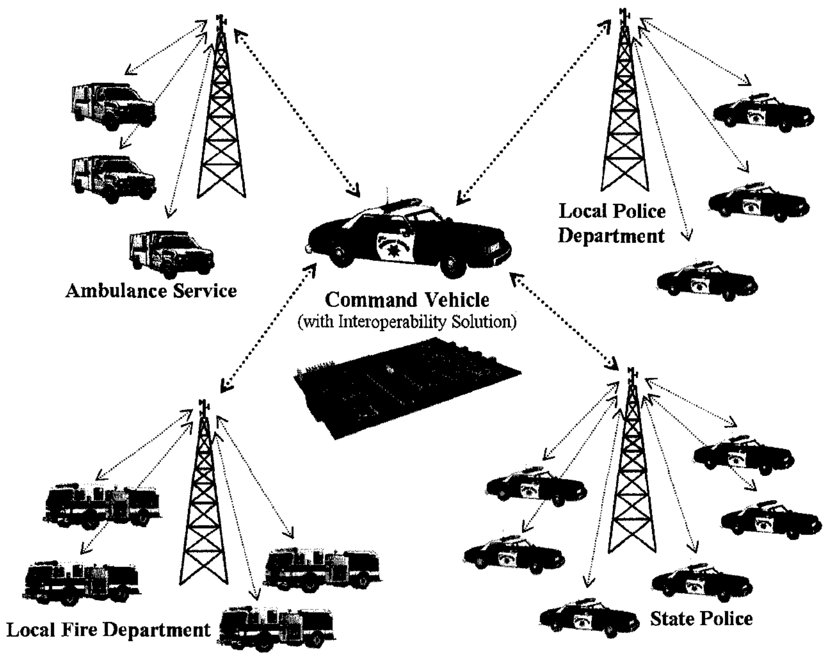


Figure 1.2 – Four Radio Interoperability Mega-System Block Diagram

In order to properly meet the needs of a small, public safety department it was decided that the interoperability solution must support four separate radio systems that can be connected as both audio inputs and outputs. The reason for selecting the number four is that this would enable support for the local police department, the ambulance

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

service, the local fire department, and the state police or some other assisting agency thereby fulfilling the most commonly needed configurations. This four system arrangement is also shown in Figure 1.2

The command vehicle, which is the only unit that has the radios for all four radio systems, is able to communicate with each individually. The individual radio system connections that the operator has in the command vehicle are referred to in this work as radio audio channels or simply **Channels**. Through the use of the interoperability device the operator is able to connect the radio systems or channels together so that they are able to communicate with each other as well as the operator. When channels are connected together they form groups of radio systems that are called, in this thesis, **Meetings**.

Note that in practical terms the maximum number of meetings that are necessary for a four radio system interoperability device is two. Therefore, an additional goal of the design is that it supports the simultaneous operation of two meetings. This will allow four major configuration modes. The first three configuration modes are similar involving two, three, or four channels all connected to a single meeting with the other meeting unused. In this way, the operators using the radio systems of these channels will be able to communicate with each other. The fourth case entails using both meetings at the same time with two channels connected to each. Note that there will be no cross communication or interference between the meetings and their respective channels.

Another important goal for the system gateway is that the final solution be adaptable and flexible to various situations that might arise. This is in terms of both the connection of the radios and in the control of the overall solution. As mentioned previously, the system must be designed to support up to four radios and two meetings

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

simultaneously. The arrangement of connections between the four channels must be completely configurable to allow for each channel to be connected to each meeting.

With regards to how the system is controlled, note that some upper end models currently available on the market can be controlled through manual control heads, RS232 serial ports, IP connections, and even phone line interfaces. Therefore, a goal of this work is that the system be controllable through two primary means: a manual control head allowing direct interaction with the interoperability system and remote software control through an RS232 serial communications port. The added control flexibility of two separate control mechanisms will improve the ability of the operator to establish the necessary communications bridges with little or no training and also to have options available should one of the control methods fail due to lost infrastructure.

With both manual and serial port control options working in parallel, very little technical expertise or training should be required. Furthermore, by interfacing with a main Project54 software program, on an optionally connected PC or laptop, the device should be capable of quickly tapping into the benefits of the common interface approach used by the Project54 system thus allowing for a unified control of all the connected radio systems.

It is also critical that the safety personnel in charge of the system still be able to maintain communication with all the various parties that are being connected together via the interoperability solution. This should include both the ability to address any given radio system specifically and independently at any time and also to be able to actively communicate with a group of radio systems. All this must be done without disrupting the interoperability connections for the radio systems users and also in such a manner as to

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

minimize the interface complexity for the operator. Thus, in the interests of simplifying the interface for the officer, the car should only have one microphone for the entire interoperability system. This microphone must be able to alternate between addressing each of the individual radios or the groups of radios that are assigned to meetings via a simple rotary switch or similar means. The microphone will also need to automatically bypass the circuit board during a power down or system crash to prevent it from tying up the attached radios. The basic, single radio system communication options for the operator must also still function even if the interoperability solution used is not in operation, either due to failure or shutdown.

The system also must meet a number of constraints for use by public safety. First, in the interest of being applied for used by the smaller, local department market the design must be low cost. In the modest production volume of 100 units the price should be in the couple of $100 range, fully assembled. This low cost constraint also includes the idea that there must be a minimal amount of overhead required for its use, that the solution be designed for ease of installation, and that little operator training be needed. This can be a major issue for small departments who often do not have their own technical staff nor the funding to support such operations. Therefore, being simple to set up and control is a very important feature.

Another constraint on the system is that it be designed for mobile, in-car applications. As such it must be relatively small, i.e. able to fit in the center console area of a vehicle. A target size would then be a maximum of 6" square with a low profile. Also, with the in-car application it must be able to be powered from the car's battery power supply, around 13.8 V DC, with some flexibility on the input voltage range. It

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

must also be fit for operation in a real-time, field based setting. This includes consideration that operational performance delays due to using the system should be minimized so as not to negatively impact the performance of public safety personnel. (Note that the requirement that it operate in a small setting allows limitations to be placed on the number of individual radios that must be supported at one time.)

Also, as part of a larger in-car system it is important that the solution be robust and produce only a minimal amount of its own electro-magnetic interference (EMI) that could severely impact co-located radio transmitters/receivers. As part of being a robust design it is also important how the solution acts in the event of its own failure. Therefore, some form of automatic isolation must be included in the design so that a crash does not cause all of the attached radios to jam. This can again be especially critical for the small department market where system redundancy is not always available and there is therefore a susceptibility to a single point of failure.

Mixed-Signal Hardware Audio Baseband Switch Approach

In this research we proposed the development of a small scale, mixed-signal, hardware-based solution to the interoperability problem, in the form of an audio baseband switch, system gateway. By working at the audio baseband, this allows the proprietary aspects of the radios to be largely avoided and brings the design down from radio frequency (RF) signals to the lower audio frequency level. The basic idea is that the designed solution will act as an analog bridge between the audio signals from each radio system, connecting outputs from some systems to the inputs of others. A basic diagram of the system design approach is shown in Figure 1.3. Note that *Chapter II - Background*

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

provides some Background Information on the more general Public Safety Interoperability Problem and works that have attempted to deal with it in the past.

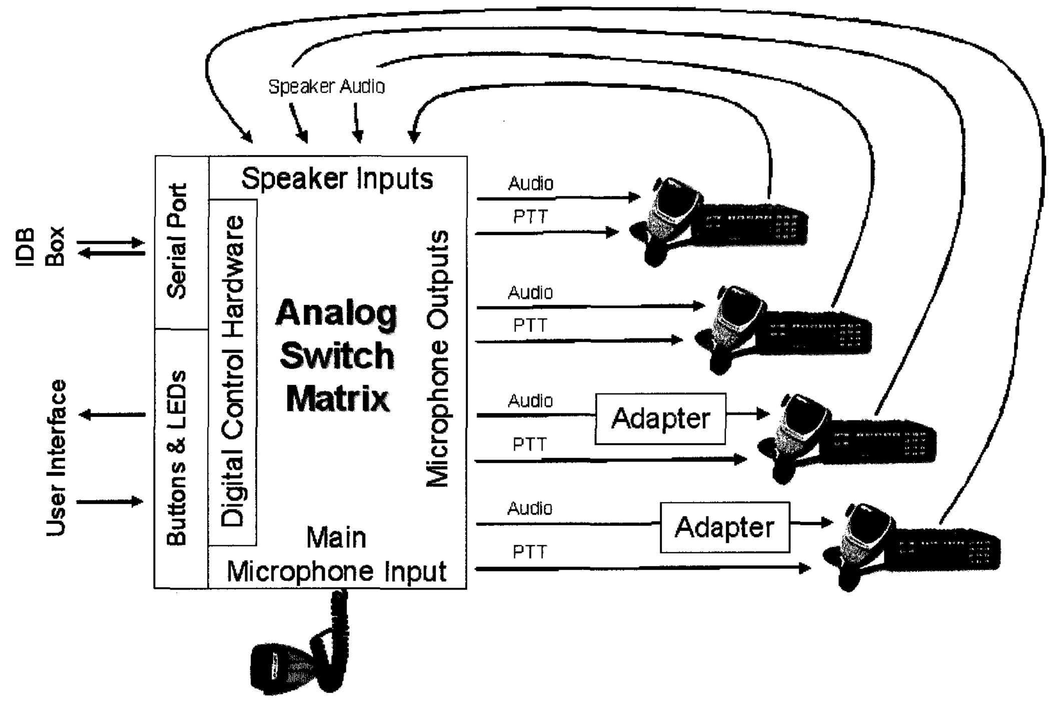


Figure 1.3 – Interoperability Device Diagram

In the first step we proposed the use of dedicated hardware for the implementation of the system gateway design. It was hypothesized that by pursuing a dedicated hardware solution the overall latency of the system would be reduced and the performance level thereby improved. It was also surmised that there would be a greatly reduced cost overhead with the hardware approach when compared to other solutions making it more desirable for smaller departments that heretofore have been largely ignored in the market. This is the case as many PC based implementations still need some specialized external interface hardware, such as PTT generation and Analog to Digital Converter (ADC) circuitry; all of which consume financial resources and space. This approach would also limit the design to a single printed circuit board (PCB) with some small adapter cables.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The resulting small size would make it ideal for positioning within a cruiser or similar command vehicle. A high level description of the hardware utilized is given in *Chapter III – System Hardware Design.* Note that it will encompass the fundamentals of all of the various hardware aspects involved in the design.

The second step taken involved making a decision on the form of hardware that would be employed in order to implement the front-end routing of the audio signals. We proposed that the routing of the audio signals be done using analog circuitry. By doing this there should be virtually no latency when compared to the more common digital approaches which utilize ADCs and often have internal data buffers and processing times based on the internal processing unit speeds. Note that through the use of general purpose analog components, instead of the more expensive digital general processors, such as PCs with sound cards, the cost benefit also should further be realized. The reduced cost, with an increase in the latency performance, should make this a much better fit for the general public safety community. Details on the design of the analog hardware circuitry are contained in *Chapter IV - Analog and Switch Matrix Hardware Design.* It provides circuit design and operation details and also discusses some of the decisions made in selecting between design alternatives.

Also critical to the success of this project, considering its target audience of small departments in need of immediate interoperability solutions, is the fact that the system is designed to be both easy to set up and easy to control. However, committing to analog hardware will reduce the flexibility of the system. To compensate for this, and thus add some extra abilities to the operator's arsenal, we proposed that the system would be controlled by digital, microcontroller-based, back-end hardware. It was believed that this

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

would accomplish the desired goal of making the solution adaptable to a myriad of situations. In addition, all of the back-end digital control hardware that is run by firmware was designed to be field upgradeable, if necessary, via a standard RS232 communications port. *Chapter V - Digital PTT Mapping and Switch Control* is concerned with the implementation of the digital PTT mapping and PTT control signal generation circuitry within a Complex Programmable Logic Device (CPLD). Details on the microcontroller that forms the heart of the digital hardware used to control the analog switching matrix, as well as the firmware it employs, are covered in *Chapter VI - Digital Control Hardware & Firmware.* This chapter also provides details on the in-system programmability options that are available for most of the digital hardware.

The next step involved the actual creation of the prototype PCB. With the presence of both digital and analog sub-circuits in the completed design it was essential that proper care be taken to manager the interaction and interference between the high speed logic and sensitive audio paths. *Chapter VII – Mixed-Signal PCB Design Considerations* addresses these issues, and broader ones associated with general PCB design. It also examines the steps taken to correct them both in this work and in previous mixed-signal designs.

Having gone through the development steps of the prototype design it was then necessary to provide basic operational procedures as well as to test the performance of the system. *Chapter VIII - Device Operation & User Interfaces* presents the necessary information for controlling the radio connections through the various means available. This includes a manual, hardware-based control head and a software-based serial communications port.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Finally, having a fully operational prototype, it was necessary to test the system's operation as an interoperability system gateway device. *Chapter IX - Prototype Design Analysis* looks at the basic performance of the device and also notes some areas for future development. This also includes some design enhancements that could be used to further improve the system. *Chapter X – System Mockup Testing* presents some of the interoperability testing scenario results for the complete interoperability system.

Concluding remarks are made in *Chapter XI - Conclusion* on the use of the system, involving the designed prototype circuit board and accompanying firmware, for solving the Public Safety Radio Interoperability Problem and it eventual field deployment.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER II

BACKGROUND

Public Safety Radio Interoperability and the Need for it

Interoperable Communications Systems, in the context of public safety, have been defined as "communications systems which enable public safety agencies to share information amongst local, State, Federal, and tribal public safety agencies in the same area via voice or data signals" [3]. Prompt, reliable communication, or interoperability, between the police, fire, rescue, emergency support personnel, and others is well documented as an essential component of the public safety sphere in modern times. In many cases however public safety agencies remain unable to communicate with each other using their own radio systems thereby severely limiting this desired goal of full interoperability.

Further, the strict definition of public safety, "any State, local, or tribal government entity, or nongovernmental organization authorized by such entity, whose sole or principal purpose is to protect the safety of life, health, or property", can also be expanded to encompass others who have a vested interest in or who are critical to the operation of the public safety functions. This might include public/private transportation personnel, corrections facility staff-members, private business security/safety personnel, and even political leadership who may need to communicate with the public or make critical decisions as quickly as possible – consider the recent tragedy in the Tallmansville,

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

West Virginia Saco Mine mining accident on January 2nd, 2006 where the death of 11 miners was initially falsely reported by company heads and the state's governor resulting in significant public outcry [4, 5].

While disasters obviously cannot be completely avoided or eliminated, the speed and efficiency with which they are responded to can greatly reduce their impact. Interoperability is one of the major factors controlling the response to the situation. By allowing all aspects of public safety to communicate with one another coordination can be increased and the correct people notified and in position when and where it is most beneficial. This interagency cooperation is especially helpful in large-scale events/disasters which may involve agencies from wide geographical areas.

The 'National Task Force on Interoperability', in its report 'Why Can't We Talk? Working Together To Bridge the Communications Gap To Save Lives', outlined five principle reasons for this failure to communicate [6]:

- Incompatible and aging communications equipment
- Limited and fragmented funding
- Limited and fragmented planning
- Lack of coordination and cooperation
- Limited and fragmented radio spectrum

Engineering solutions to the wider interoperability problem typically focus their attention on the first, second, and fifth obstacles. The planning and coordination/cooperation encompassed by the third and fourth issues must rather be dealt with on an individual level through interacting local, state, and/or federal leadership, administrative personnel, and other forms of management. The engineering solutions

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

designed are principally accomplished through a hardware addition/modification remedy that can be implemented to facilitate the physical aspect of interoperability between users while minimizing the cost that is often associated with such a venture. This role is largely secondary to and contingent upon the success of the administrative actions and solutions.

Some Current Interoperability Solutions and Shortcomings

Common Radio Systems

The AGILE Interoperability Strategies for Public Safety program, of the National Institute of Justice, outlines in its report, 'Guide to Radio Communications Interoperability Strategies and Products', three primary means for establishing interoperability between organizations and agencies [7]. The first of these involves the usage of the same radio system by all agencies/jurisdictions concerned. This, while arguably the most complete method available, allowing for both the exchange of voice audio and data, is also the most costly, both in financial terms and in the use of the available spectral bands. Note that currently agencies, even within the same jurisdiction, may utilize completely different radio systems. For one or more groups to change over to match the others would require the investment of significant capital, both for in-car radios and for such infrastructure as additional radio towers in certain deficient regions. This cost proves too high a burden, particularly in the context of widely scattered agencies in rural localities. In fact, on the state-wide scale, such an attempt has been described as "prohibitive and unrealistic" [8].

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Further challenge to this arrangement comes from the limited and severely fragmented frequency bands that the FCC has currently allocated for public safety agencies. Spectral bandwidth in the public safety domain is highly desirable and there is significant competition for what is available. Also, there are currently 10 major bands available for public safety usage: 25-50 MHZ (HF and VHF), 150-174 MHz (VHF), 220-222 MHZ, 450-470 MHz (UHF), 470-512 MHz (UHF), 764-776 MHz (700 MHz), 794-806 MHz (700 MHz), 806-824 MHz (800 MHz), 851-896 MHz (800 MHz), and 4.94-4.99 GHz [6]. However, the two 700 MHz frequency bands are still occupied by Analog Broadcast Television channels which are to be cleared out with the transition to Digital TV[1]. Therefore, while designated as public safety spectra, they are not currently available for public safety use nor are there many radio systems on the market currently capable of operating in the 700 MHz bands [8].

Additional complications arise through the concept that by uniting all agencies to work with the same type of radio the frequency bands in which that equipment operated would undoubtedly become significantly over-crowded. Concerns could also appear in cross-jurisdictional interoperability where certain radio frequencies may perform better in the rural vs. urban geographic areas [7]. The solution chosen by many departments to reduce the effects of these two spectral problems is simply to use two or more different radios in each car at the same time. This is obviously very financially expensive as it requires redundant systems. However, perhaps more critical is the additional workload

[1] This transition will not be enacted until February 18th, 2009 at which time the FCC will have authority under 'The Digital Television Transition and Public Safety Act of 2005' (Conference Report: 109-362 in Congressional Record H12641-12737) to reclaim the frequency bands and, in addition to auctioning some of the reclaimed spectrum off, allocate some for use by public safety [9].

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

that it places on the officer who in many cases already must deal with an over-crowded cruiser. With no common interface to operate all of the divergent radio systems, the driver must focus a proportionately large amount of their attention on the radios. This can be quite distracting during some of the more consuming actions an officer is faced with in the execution of their duties. Due to the diverse radio interfaces few multi-radio interfacing devices exist and those that do are often either quite expensive or in early prototype stages. Some preliminary work also being carried out by Project54 is an effort involving the creation of a software-based audio baseband patch. Some of the early design implementation and testing, which is outlined in [10], entailed the creation of a unified interface for an officer to control multiple radio systems. Thus the officer is able to selectively broadcast over all of the connected radios using a single Project54 microphone and also hear the audio coming from any radios via a common set of speakers. In this prototype the different radios did not have the ability to rebroadcast on each other. Note that this early solution has since been extended to a more useful, complete system gateway patch.

Use of the shared, common radio system methodology exclusively for major emergencies has been implemented in some recent events. For example, in the September 11th, 2001 attacks at the Pentagon the 'Greater Metropolitan Washington Area Police and Fire/Rescue Services Mutual Aid Plan' (COG Mutual Aid Plan) and the 'Northern Virginia Mutual Aid Agreement' (NOVA) were quickly activated in order to deal with the situation. What made this event less of a disaster was that there had been preplanning and that the departments were able to communicate with one another via the emergency radio system – in other words, because of interoperability. Even agencies

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

responding from outside the mutual aid region, without appropriate equipment, were able to communicate. This is because “agencies responding from outside of the area covered by the agreements were given portables from a limited supply maintained by Arlington County and, … Motorola, a private vendor” [11]. This is obviously an adequate solution for large disasters; however it does require significant investment of capital to create these radio stashes, a cost that may make it too difficult to enact in some areas.

This success is in stark contrast to the events that unfolded at the World Trade Center (WTC) in New York City on the same day. While the attacks at the Pentagon were met by an interoperable public safety task force, those in New York had no such benefit. Instead, the fire department in particular was faced with “little reliable radio communication that morning, could not keep track of all the firefighters who entered the towers, and were unable to reach them as the threat of a collapse became unmistakable” [12]. As a result of these communications faults it seems that “non-interoperability was at least partially responsible for the loss of 343 firefighters” [11]. Such catastrophic losses may have been prevented with even a basic interoperability system structure.

Commercial Services

The problem with the ideal solutions have already been described and may prohibit some from working towards interoperability, at least for the time being. For this reason, other stop-gap solutions to interoperability have also been designed. One is the use of commercial services, at least at command levels. This however can also prove expensive and may not be expandable to large scale events with many personnel. Furthermore, the greatest flaw comes in the fact that it depends on commercial services

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

which may more easily become congested or damaged. Such was also the case in the WTC attack "with telecommunications equipment destroyed in some cases, or simply too congested" [13]. Public safety cannot risk using equipment that can be overloaded by other customers and thus made unavailable but rather must have priority access to wireless communications that are available at all times, ideally with built in redundancy, for mission critical communications. The result is that generic commercial services, while useful as an augmentation, cannot be solely relied upon by public safety. Those commercial services that are used must have equipment dedicated to public safety and that can also be deemed highly reliable [6].

Software Defined Radio

Another even more promising and potentially permanent solution is SDR with the inherent ability to be able to insure interoperability simply by uploading a new software program to a set of universal and default hardware. However, it faces many of the common challenges of requiring some, often expensive, external hardware. The main difference from other approaches is that SDR actually can replace an existing radio system completely and connect directly to the antenna. The result is that rather than dealing with low frequency audio-band signals, the SDR is working with high frequency radio signals that can range from 25 to 900 MHz and even some in the gigahertz range. The biggest advantage to this is that a complete, well developed SDR solution essentially represents the ultimate in public safety radio interoperability by allowing officers to quickly and easily transition between disparate radio frequency bands, modulation schemes, and even proprietary encryption/encoding (with the availability of proper

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

firmware/software). In fact, in the FCC's recent loosening of constraints on SDR systems it cited two of the primary reasons for doing this as being that "Software defined radios could allow more efficient use of the spectrum by facilitating spectrum sharing and … their ability to be programmed could also enhance interoperability between different radio services" [14]. Furthermore, the nature of SDR permits the use of over-the-air downloads in order to upgrade systems to the latest version instead of enacting costly hardware change-outs through maintenance personnel [15].

However, the SDR device does have a major disadvantage in that it must operate well above the carrier frequency in terms of the analog to digital conversion (or at least an IF frequency for systems using front-end, software controlled hardware instead of connecting directly to the antenna). This means that it requires an extremely high sampling rate and a fast processing unit/path capable of handling significant amounts of data in a very short time. This leads to some distinct problems involving tradeoffs between desirable features when working in a real-time application. These relate to the price, power consumption, and performance of the SDR system [16]. (The size of, or real-estate taken up by, the system is also important but is often incorporated into these other tradeoffs.)

First consider the tradeoff between system performance and power consumption. To operate well in real-time (i.e. have an acceptable level of performance) the SDR must operate at high frequencies. However, operating at higher frequencies typically entails higher power consumption due to switching of CMOS circuits. Such may not be an issue with a base station but can become of critical importance when dealing with handheld devices such as portable radios [15, 16]. Power consumption even bears consideration in

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

mobile (car-based) applications where battery usage is less of a concern but can still be an issue when it becomes excessive.

Another performance tradeoff occurs with the cost of purchasing the system. To obtain a better systems often cost significantly more. This is especially true in developing technologies like SDR where businesses are still trying to justify and recoup research and development expenditures. Considering that funding is a concern for public safety, replacing all pre-existing radios is impractical. However, SDR does present a slight cost advantage feature to public safety. Public safety radio systems are traditionally more expensive than commercial models (such as Cellular and Personal Communications Services) largely due to economies of scale and the fact that public safety is a niche market [8]. However, with the arrival of SDR it is postulated that the public safety industry, and others, should be able to capitalize on the flexibility of the system and use the same base hardware with only minor variations for licensing and encryption [15].

One attempt to complete a real-time SDR solution has been made by the Vanu Corporation with its 'Vanu Software Radio Virtual Patch Prototype' [17]. As the name implies the goal of the prototype was not to produce a complete radio replacement but to develop a switch that worked at radio frequencies instead of the audio baseband. This type of interoperability solution belongs to a more general class of System Gateways which are discussed in further depth in The System Gateway Approach section.

Other systems have also been made that attempt to use reconfigurable hardware for the signal processing of an SDR to improve throughput. In this arrangement there are two principle means of implementing the front-end design. One entails the use of "soft-

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

hardware" or Field Programmable Gate Arrays (FPGA) while the other, more traditional SDR, uses true software through high speed embedded processors, most often designed specifically for Digital Signal Processing/Processor (DSP) applications [18]. (Note that the Vanu solution is much closer to the traditional processor solution but is at a higher level than an embedded solution.) The possibilities of faster computational speeds due to the soft-hardware paths of FPGAs have given this solution significant advantages. In addition, technological advancements in recent years have led to a modest decrease in the physical size and power consumption of these circuits which has further increased their appeal. However, before widespread application can be made, some improvements are still necessary.

Sandbridge Technologies, Inc. has recently been working on the development of a specialized DSP processor, the Sandblaster Convergence Platform [16]. Their goal is to push the envelope in designing a DSP chip for SDR applications by minimizing power consumption while also improving through-put and operational speed. In order to accomplish such an SDR solution they have chosen, instead of building on a generic base system as Vanu did, to operate with a specialized, reprogrammable hardware structure. The attempt appears to have been successful with significant power savings over other DSP processor based solutions.

The SDR solution obviously presents a number of significant benefits including widespread device interoperability, field upgradeability, and cost sharing with wider markets. Its main faults, relating to power consumption, cost, and implementation size for a given level of performance, that in the past have limited its release in a real-time, practical platform, are also rapidly being overcome through technological advancements.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Particularly promising opportunities have also arisen with the use of modern FPGAs, DSP Processors, and System-on-a-Chip (SoC) implementations. However, further development is still needed before implementations using SDR can be performed on the desired scale.

The System Gateway Approach

Audio Baseband Switch

The final modern interoperability solution to be discussed is the use of System-to-System Gateways. The most common of these is the Audio Baseband Switch which, in its most primitive form, simply takes the audio output of one radio and feeds it to the audio input of another. In this manner the communications on each are shared despite differences in spectrum, proprietary coding, and further obstacles that may confront other solutions. Similarly, the cost of the hardware involved can be much smaller than the alternatives of replacing entire systems as the preexisting infrastructures are left undisturbed. This includes the radio systems in each cruiser or safety personnel vehicle which then eliminates the need, not only for new radio equipment, but also additional training in that new equipment. The only modification would occur at the designated base station or control vehicle where radios from each system must be provided for the implementation of the radio patch. Note that with a true audio baseband switch, signals on one radio system are automatically rebroadcast on the other radio system(s) and each officer need only manage a single radio, while the command/base station vehicle or location is the only place where all the radio systems must be represented.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

There are, however, some distinct disadvantages to use of a gateway interoperability solution. The most significant is that since each group still transmits on its original system rather than only on a special, shared channel, more spectral bands are occupied by one meeting group [7]. In fact, each new agency needs to bring in its own radio frequency band to use or else go on that of some other group, with which they are already compatible. But this solution would still be helpful in linking a smaller agency's network to a large interoperable structure without any prior system overhead. This is especially true for cases where a low-cost, temporary solution is needed in a major, locally defined crisis.

A second flaw is that the use of a baseband audio gateway will inherently mean that digital data sent over the radio waves, which typically has a higher bandwidth, will not be able to be exchanged between systems over the audio-band limited radio speakers and microphones. Thus, the interoperability will only be in terms of the lower frequency audio [7]. This might be avoided with some larger systems that supply an extra port to capture/transmit the digital video, image, and alternate data that is required between systems. Note however that such a device may also need to cope with proprietary data signaling as not all radio devices hold to the same universal standard. In any case, the need to at least supply the public safety community with a low-cost audio interoperability solution is significant enough to outweigh concerns that such a system might not allow the exchange and transference of digital data.

One audio gateway already on the market is the ACU-1000 designed by JPS - Raytheon. The ACU-1000 has a number of advanced features including the capability to interface up to 12 radios, telephones, and other diverse communications equipment (24

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

with the connection of a second ACU-1000 chassis). It also can be controlled manually, via RS232, or over an IP connection thereby facilitating ease of use and application during any public safety event [19]. It should also be noted that the ACU is the flagship product in a family that includes the ACU-T Tactical Intelligent Interconnect Unit, the MCK-1000 Mobile Communications Kit, and the TRP-1000 Transportable Radio among others. This gives the public safety management force some options in the selection of needed equipment [20]. The largest fault for such devices, however, is the significant cost. By way of example, the ACU-1000 is in the range of $100,000; outside the reach of many small agencies and beginning to approach the cost of simply upgrading current equipment [21]. Other concerns, such as required technical expertise requirements and latency are also important in considering a local department level solution to the interoperability problem. These will be mentioned further in discussion on the course taken in this thesis.

Voice-Over-IP

Another recent attempt to implement the basic System Gateway design approach has arisen with VOIP. By leveraging existing internet-type standards, it holds the possibility of radios that can be automatically set to work together simply by designing a universal interface to already existing standards.

The VOIP solution has however presented a number of flaws including the dependence on an internet network structure which may or may not be dependable in an emergency. Alternative, dedicated networks could be developed but would largely be impractical with the increased infrastructure overhead that would have to be borne solely

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

by the controlling public safety agency. Issues have also been examined that explore the effects of dropped packets, jitter, and other IP constraints that limit its use in real-time applications. In typical internet transactions delays or dropped packet, which may need to be resent after detection, present little problem to the end user. However, in mission critical applications, particularly during events when public infrastructure might already be overloaded (as in the World Trade Center attacks on 9/11), these delays and failures are unacceptable. Furthermore, there is a significant amount of pre-incident overhead planning that would be needed for proper implementation. This includes details as to who would control the VOIP main server and who could connect to it. Lastly, the VOIP solution requires computational hardware overhead. This usually includes some form of digitization board. It is more common to use a PC with a sound card, however this can be somewhat expensive as a channel of the sound card is required for each radio or otherwise some form of analog switching hardware must be included. Other issues also arise in the need for special hardware to generate the Push-to-Talk signals needed by the radios [22]. Combined with set up difficulties this option may not be reasonable for most public safety interoperability needs.

Software Defined Radio as a System Gateway Patch

The SDR solution to radio interoperability also has an application within the broader System Gateway approach. While it does require an additional change in hardware in the short term, it would not be difficult to have the controlling software unit (for example a PC) map the audio signals between various software radios configured to work with the pre-existing infrastructures. This has the added benefit of making the need

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

for radios at the connection site unnecessary and also allows for greater versatility in connecting systems together on the fly.

The aforementioned 'Vanu Software Radio Virtual Patch Prototype', being developed by the Vanu Corporation as a complete real-time SDR patch solution, is one such SDR based System Gateway [17]. However, while the attempt appears to have been reasonably successful it does present some of the limitations that are typical for SDR methods that utilize true software to perform all signal processing. First, the software program is coded to only support FM, APCO Project 25, and GSM waveforms. Other signaling schemes, including proprietary forms, still need to have their protocols applied to the structure. This could present coding difficulties especially for small departments that utilize obscure/obsolete radio systems. However, this is a challenge that all SDR solutions face [23].

The second issue relates to the aforementioned cost and power consumption vs. performance issues. The Vanu solution is designed to operate using adaptable, software-controlled, front-end hardware followed by processor boards having analog to digital converters (ADC). The outputs of the processors are fed into a computer for processing in terms of modulation and coding. All DSP is then handled in pure software, versus the traditional approach of using digital signal processors and FPGAs [17]. While this does work, and it has the advantage of placing the full processing power of a computer at the designer's disposal to increase flexibility, it has some significant drawbacks. First is the need for somewhat expensive, "wideband and waveform-independent" radio frequency (RF) and digital front-end hardware. Secondly, the attempt to obtain flexibility by using a full computer processor further adds cost to the system. (For the prototype designed,

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

which supported only two radios, the system required a 1.2-GHz Intel Pentium III processor.) Both of these factors also mean the system utilizes significant amounts of power (especially if the system was ever to be used as an independent SDR solution) and takes up a great deal of space [23].

Another shortcoming is that in using sampling hardware it was necessary to operate with buffered data blocks 1024 elements long. This created delay problems resulting in latency of approximately 21.3 ms just on the front-end hardware side. As a result the round trip latency through the interoperability system alone will most likely be a minimum of 50 ms and possibly more depending on the performance of the computer's processor. This creates problems for communications equipment which may need to be interfaced to the system, such as a Global System for Mobile Communications (GSM) device which is specified to have a 42 ms maximum round-trip latency [23].

Finally, the system designed was only able to support one connection between two radios. This is largely due to the computation intensity required of the PC for DSP processing. The solution is therefore both expensive and cumbersome making it prohibitively large for rapid field deployments that may be required for a faster interoperability solution. Also, that the system is scalable to more than two channels with the addition of supplementary RF hardware and processing units was assumed and not attempted. Moreover, it is not known whether the PC can adequately handle the added loads [23]. Thus, while the Vanu SDR solution is optimal in a number of ways and presents significant opportunities for future research, its current application through the use of a computer for processing is somewhat limited.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER III

SYSTEM HARDWARE DESIGN

System Hardware Elements

In order to improve the latency performance, while also reducing size and cost, it was decided that the system would be designed mostly in dedicated hardware. The hardware is divided into thee main parts, each of which are shown in Figure 3.1. They are the Analog Hardware, the Digital PTT Circuitry, and the Digital Control Circuitry. A high level summary of the fundamental aspects of each is given here but analyzed in more depth in the following chapters.

The main interaction with the four radios of the system is performed through the Analog Hardware. It serves three primary purposes. The first of these is the buffering and appropriate scaling of the input and output audio signals. Another purpose of the analog circuitry is the detection and generation of the PTT signals that are needed by the Digital PTT Circuitry. Lastly, and as it primary function, portions of the analog hardware form the Analog Switch Matrix that actually connects the various radios together to create conditional interoperability. The Analog Hardware passes audio signals between each radio and receives audio from the In-Car Microphone. It also sends raw PTT signals to the Digital PTT Circuitry and receives control signals from the Digital Control Circuitry hardware.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

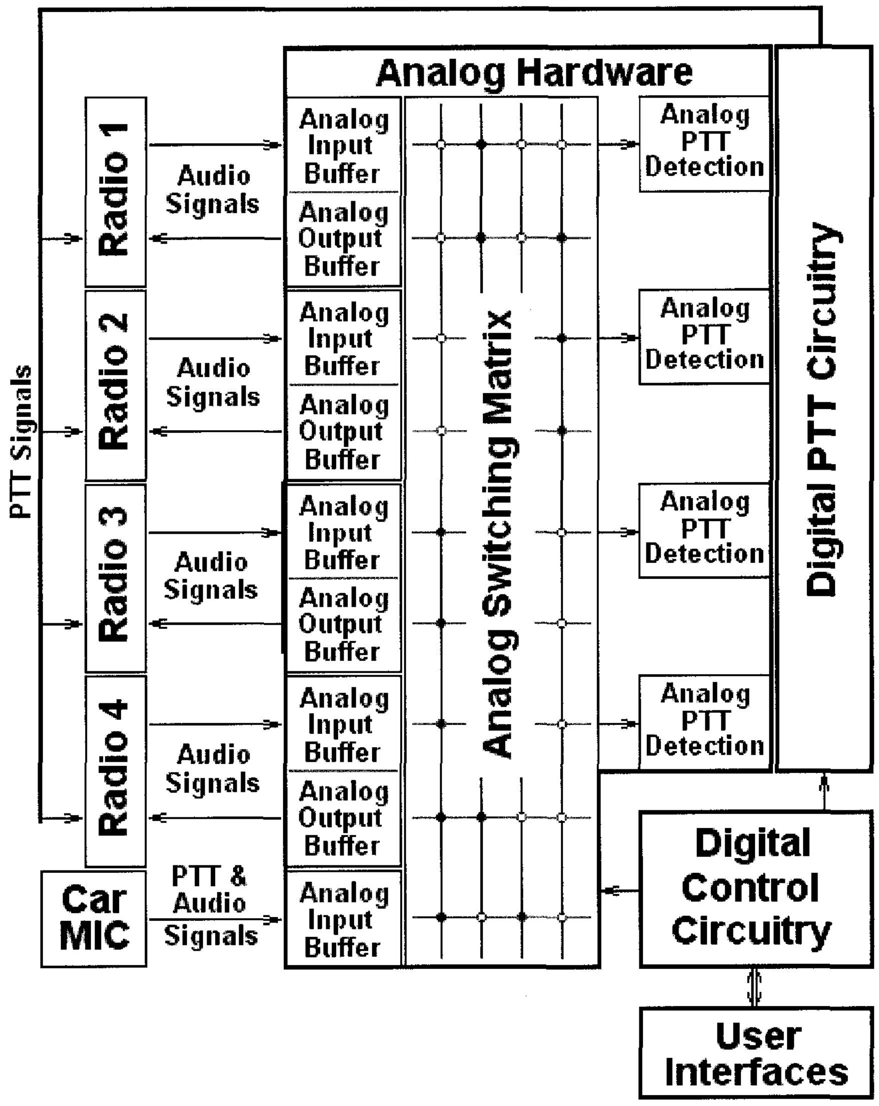


Figure 3.1 – System Hardware Block Diagram

The primary function of the Digital PTT Circuitry hardware is to route the digital PTT signals received from the In-Car Microphone and the Analog PTT Detection circuits to the appropriate radio microphone PTT inputs. Determination of the appropriate routing is determined by the conditions specified by the connection control signals received from the Digital Control Circuitry. The PTT mapping hardware can also be reprogrammed through an interface with the control hardware which greatly increases its flexibility.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The final hardware section of the system is the Digital Control Circuitry. Its main aspect is the digital microcontroller which manages the execution of two primary functions. The first is the control of the Analog Switch Matrix and the Digital PTT Circuitry in the mapping of the audio and PTT signals. This involves the management of control signals for the switching elements and digital logic variables. The second function of the Digital Control Circuitry is the management of the User Interfaces. The User Interface is broken down into two parts. One is the manual, hardware-based control head (which is largely implemented in the Digital Control Circuitry). The other is the (optional) software interface which connects to the system via the serial port communications hardware. (Note that the serial port communications hardware is actually discussed in the context of the user interfaces in *Chapter VIII – Device Operations & User Interfaces*.)

The In-Car Microphone

Rather than have four in-car microphones, as well as any meeting microphone(s), that the operator must juggle in order to communicate with all the channels, the approach taken utilized only a single, main system microphone in order to simplify operations. The single microphone can be physically switched between five settings - one for a direct connection to each of the channels and one for connection to an interoperability meeting. When connected as a channel microphone the audio will only be broadcast to that individual channel. When activated it will also temporarily break the channel's audio feed from any meeting it might be connected to and thereby allow the operator to use it like any normally operating microphone.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The selection of the microphone to act as the Master Microphone is the more versatile setting that will allow the operator to use it within the interoperability context. It broadcasts directly onto either Meeting A or Meeting B and thus out to any channels that are connected to them. Selection of the meeting is done by connecting the Master Microphone to the desired meeting just as any channel would be connected. (The Master Microphone input can really be treated as another speaker audio input to the system in addition to the four from the channels.) Note that the Master Microphone has override priority over any audio arriving on a meeting from one of the channel sources.

The Master Microphone has a manually keyed push-to-talk button, however, it is also equipped with a firmware controlled PTT in parallel with the manual one. This will enable the Project54 program, or some similar software attached through a serial port, to engage the Master Microphone. This permits the system to use text-to-speech synthesized voice directly over the airwaves if desired by a future P54 application or to use the Project54 in-car microphone in place of the generic interoperability device microphone.

The microphone employed by the system should be a passive microphone. This means that no power need be supplied to the microphone and that the signal output levels will be quite low (on the order of 200-300 mV). From an interface perspective the microphone contains both a single-ended audio signal and an active low PTT signal. A ground reference line for one or both of the audio and PTT signals should also be provided. (Note that the microphone used in the laboratory and system testing was a Whelen model that was passive and very simple with only four lines – Audio, PTT, and a reference 'ground' for each.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The Radios

The System is capable of supporting up to four different radios. In order to make the design as universal as possible very few assumptions were made as to what signal and control lines are available on the radios. The base line signals that it was decided would always be available for any given radio were: the speaker audio output (potentially differential), the microphone audio input, and the microphone active-low PTT input. Note that in some cases this may entail the removal of a radio's normal microphone and/or speaker in order to connect it up to the interoperability system. However, in other cases, auxiliary ports that provide the same signal functions might also be available. Note that the microphone audio input and speaker audio output of the radio connect respectively to the analog output buffers and analog input buffers of the interoperability circuit board. The radio microphone PTT input similarly connects to the channel PTT outputs of the Digital PTT Circuitry.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER IV

ANALOG AND SWITCH MATRIX HARDWARE DESIGN

Introduction to the Analog Prototype Hardware Design

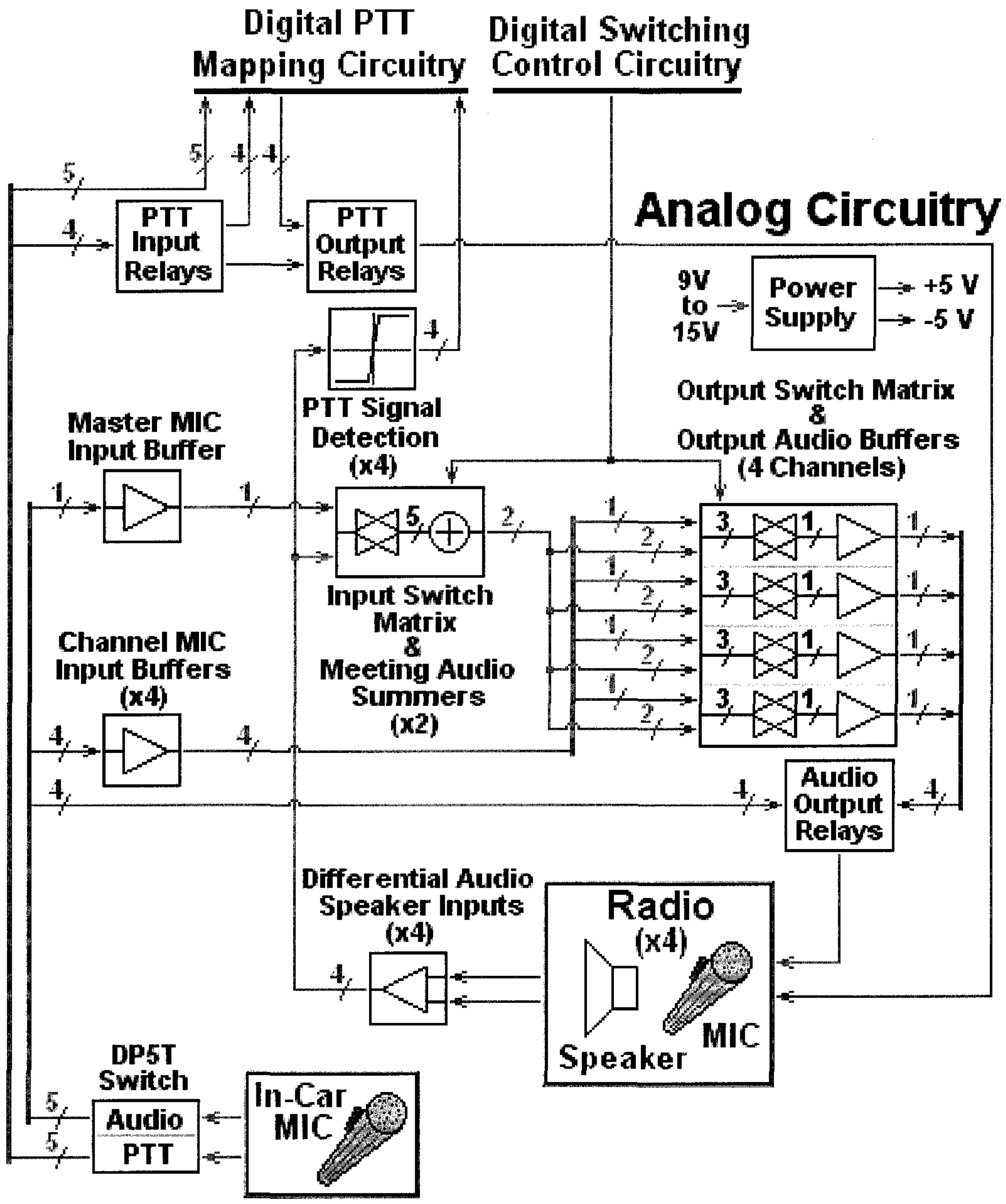
The main functional aspect of the prototype design is the analog front-end and switching matrix hardware. These analog aspects entail the actual manipulation and controlled routing of the audio signals that are needed by the radio inputs and outputs. Figure 4.1 shows a block diagram of the analog circuitry that was used. Note that the control interface with the digital microcontroller and the digital PTT mapping circuitry as a signal source and destination are also included in the diagram to show the interaction with the rest of the prototype system.

The main input of the analog audio is provided via the four Differential Audio Speaker Inputs that each come from one of the radios. These provide initial buffering of the signal converting it from a potentially differential signal to a single-ended, circuit board ground referenced one. There is also the option for scaling of the input signals. Connections on these inputs can be made to either an auxiliary audio output or the speaker connection itself of the radio.

From here the audio signals are passed into the Input Switch Matrix and the two Meeting Audio Summers. The input switch matrix optionally connects none or any number up to four of the radio audio sources to the summers that generate the meeting audios. This allows for the actually unification of audio signals from the various sources

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

onto the shared meetings. Control of the input switch matrix configuration is by the digital control hardware. After this audio combination stage, two primary audio signals remain for potential output to the radios – the Meeting A audio and the Meeting B audio. Of important note is that the Input Switch Matrix, while supplied from the four channel speaker inputs, is also supplied by the Master Microphone.



Digital PTT Mapping Circuitry
Digital Switching Control Circuitry
Analog Circuitry
PTT Input Relays
PTT Output Relays
9V to 15V
Power Supply
+5 V
-5 V
PTT Signal Detection (x4)
Output Switch Matrix & Output Audio Buffers (4 Channels)
Master MIC Input Buffer
Input Switch Matrix & Meeting Audio Summers (x2)
Channel MIC Input Buffers (x4)
Audio Output Relays
Differential Audio Speaker Inputs (x4)
Radio (x4)
Speaker
MIC
DP5T Switch
Audio
PTT
In-Car MIC

Figure 4.1 – Analog Hardware Block Diagram

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The Master Microphone audio is one of five signals that are generated from the In-Car Microphone. These signals are buffered at the input side of the circuit board. The four channel microphone audio signals, with the Meeting A and Meeting B audio signals, are passed to their respective channel portions of the Output Switch Matrix for buffered output to their respective channels. Note that the individual channel microphones uniquely address only their individual channels while the Master Microphone is able to address particular meetings.

The Meeting A, Meeting B, and individual channel microphone audio signals are all combined at the Output Switch Matrix and Output Audio Buffers. The Output Switch Matrix operates in a similar manner to the Input Switch Matrix by optionally connecting one or none of the meeting audio lines or the channel microphone audio to individual channel audio outputs for rebroadcast on the radio system attached to that channel. Simple buffering is performed to insure the ability to consistently drive a load on the output. Note that this output is single-sided and designed for connection to the microphone inputs on the individual radios. Alternately, if an auxiliary port is available, these can be connected to the auxiliary inputs of the radio.

Not discussed up to this point is the analog hardware that deals with the PTT signals used to queue the radios at the appropriate time. The audio signals that come from the in-car microphone are also accompanied by PTT signals. Therefore, these signals, which are really digital signals, can be passed directly to the Digital PTT Mapping Circuitry. However, the audio that comes from the Differential Audio Speaker Inputs is not so accompanied. It is then necessary to perform some detection on these audio signals to generate the PTT signals for transmission to the PTT mapping circuitry.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

This operation is accomplished by the PTT Signal Detection circuits (one for each channel). These circuits are one of the most critical aspects of the system and discussion of their design and performance comes up at numerous times.

In order to insure a robust design and that a crash of the system doesn't bring down the attached radio systems, three automatic shutdown relays were designed into the Input / Output (I/O) portions of the circuit board. These operate at the absolute limits of the system audio and PTT I/O in order to insure a complete by-pass and form a boundary around the whole interoperability interconnection solution. A complete description of these relays is given along with an explanation of their additional control circuitry overhead.

Finally, details of the power supply used for powering the entire circuit board off of a standard vehicle battery level voltage are provided. Note that both positive and negative 5 V power supply rails are used. Linear voltage regulators are used to generate the +5 V rail and a low noise switching voltage inverter creates the -5 V from it.

Each of these analog subsystems is fully detailed in the following chapter subsections. The physical implementation of the design as a Printed Circuit Board (PCB) is provided in *Chapter VII – Mixed-Signal PCB Design Considerations.* Also, the detailed schematic for the Revision C design, as well as the physical implementation layouts, are shown in Appendix E.

Differential Audio Speaker Inputs

The main audio inputs to the system are the audio speaker inputs. A sample circuit is shown in Figure 4.2 for a single channel. Many modern radio systems have

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

devoted audio inputs/outputs that can be used and connected directly to these inputs on the interoperability board. However, with some systems, particularly aging equipment, these additional ports are not available. In these cases the speaker output of the radio can optionally be used, in parallel with the speaker if necessary. For this reason the circuit is designed as a differential input which is then converted down to a circuit board ground referenced, single-ended signal. Note that if a single-end signal is available on the input it should be connected to one of the two input terminals. (These input terminals are pins 1 and 2 of PL1 in Figure 4.2 and labeled together as the Speaker Audio Input. They are individually labeled V_{in+} and V_{in-} respectively). The other terminal input should then be connected to ground or some other signal that is a reference for the input. If it is left disconnected, the input will float to ground due to the 1 MΩ pull down resistor. However to avoid noise issues it is recommended that it still be grounded when unused.

Both input pins of the Speaker Audio Input are AC coupled through a 1 μF capacitor and have a resistive load of 1MΩ. They are interchangeable with the only difference being the final polarity of the AC audio signal. This will have no impact on the sound quality or perception.

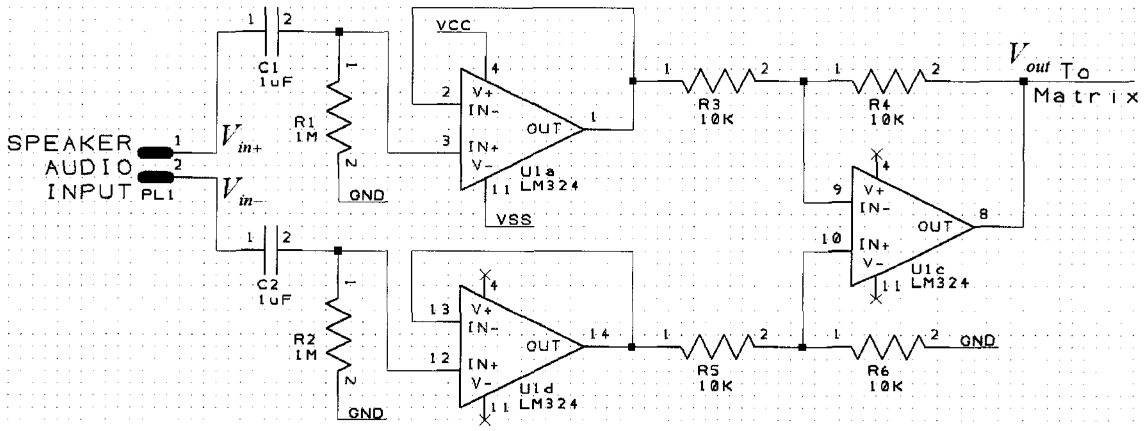


Figure 4.2 – Differential Audio Speaker Input Circuit

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The circuit is designed for a gain of 1 although it is possible to adjust this by changing resistors R3, R4, R5, and R6. In general, however, this should not be done if possible but rather that signal attenuation/amplification be done prior to the circuit board connection using an external operational amplifier (OPAMP). (Note that scaling can also be performed at the meeting summing amplifiers where it is much less complex and involves only changing a single resistor for each meeting. This is mentioned in the Meeting Audio Summers and Switches section.)

The basic equation governing the output voltage, using the component labels shown in Figure 4.2, is given by:

$$V_{out} = -\frac{R_4}{R_3} V_{in+} + \frac{(R_3 + R_4) R_6}{R_3 (R_5 + R_6)} V_{in-}$$

Note that V_{out} is the voltage that will be sent to the switching matrix and the PTT detection circuit. Also, V_{in+} is pin 1 of the speaker audio input terminal while V_{in-} is pin 2. (In the PCB layouts, pin 1 has a square pad shape while all others will be round.)

The simplification can then be made that $R_3 = R_5$ and $R_4 = R_6$ which insures a parallel circuit. This is something that should always be checked whenever changing the values of these resistors. The equation for the output voltage then reduces to:

$$V_{out} = -\frac{R_4}{R_3} (V_{in+} - V_{in-})$$

Assuming that V_{in-} is the reference for the signal on V_{in+} (implying it is at ground if V_{in} is single-sided or that it is at $-\frac{1}{2}V_{in}$ if V_{in} is differential with $V_{in+} = \frac{1}{2}V_{in}$) then V_{in} is simply V_{in+} - V_{in-}. Therefore V_{out} can be expressed in its most simplified form, with a symmetric circuit, as:

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

$$V_{out} = -\frac{R_4}{R_3} V_{in}$$

Note that the inversion on the input is not significant as mentioned previously. Incidentally, the use of the inverting summer for the meeting groups will re-invert it anyway.

Meeting Audio Summers and Input Switch Matrix

The main functional part of the designed hardware solution is the switch matrix that is used to map audio signals from some radios to others. This is accomplished through the use of analog switches that connect or disconnect the audio lines coming from the radios to one of the two radio group meetings (Meeting A or Meeting B). This section deals with the input switch matrix and some of the general aspects of the entire matrix. It also covers the summing amplifier used for each of the meetings.

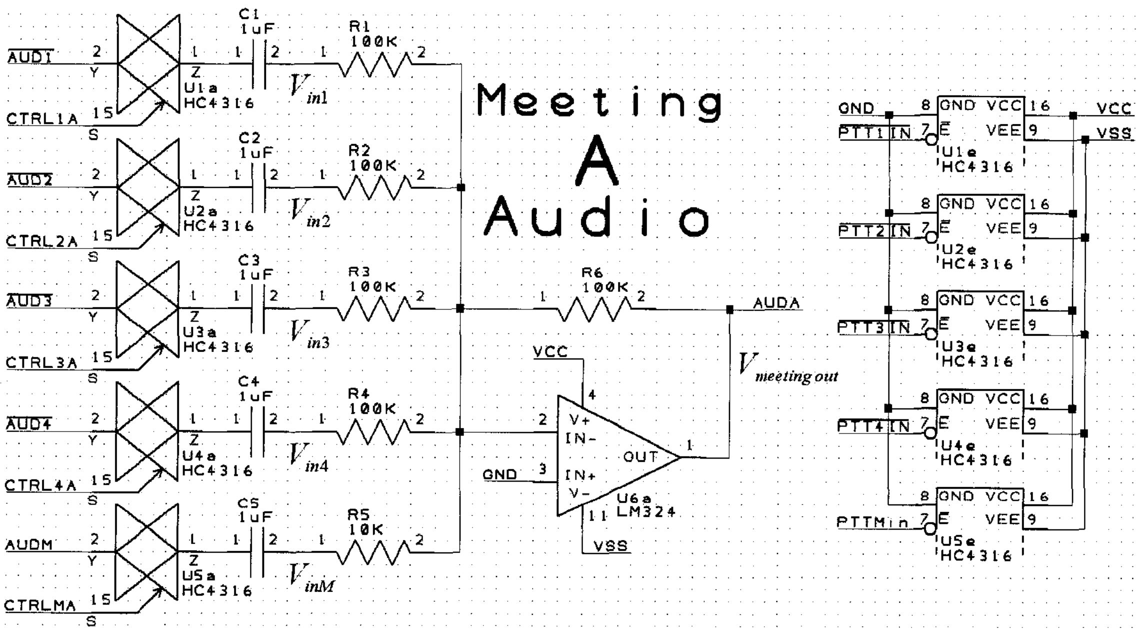


Figure 4.3 – Input Switch Matrix and Inverting Meeting Summer Circuit

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Figure 4.3 shows the input side of the switch matrix associated with Meeting A and the Meeting A inverting summer. Note that the connection of each channel (radio) is controlled by the analog switch, an HC4316 integrated circuit (IC). When the enable signal is high the circuit is closed and the channel becomes connected thereby feeding the meeting audio. Each channel/meeting input switch is controlled by its own control line that is buffered off of the Central Processing Unit (CPU) or microcontroller. Note for example that when CTRL2A is high Channel 2 becomes connected to Meeting A and will thus feed any channels that are connected at the output of Meeting A. The output switch matrix also uses the same control lines so a channel that feeds a meeting will also receive that meeting's audio. (It is also important to observe that a channel cannot be connected to both meetings at once.)

Another important aspect of the switch matrix circuit that must be discussed is the main IC enables and how they are used. Each individual switch in the switch matrix is controlled by its own control line, which the input and output connections to a particular meeting and channel share, as already mentioned. However, each IC, which is a quad package of switches, also has a master switch (active-low) enable that has the ability to disable all four switch gates. This allows the execution of a very important function with regards to the Master Microphone. Note that the four switch gates that exist within each IC package are all arranged so that they belong to the same channel (1, 2, 3, or 4). Then, the master enable of the IC belonging to each channel is connected to the inverted PTT of the microphone for that channel (as shown on the right half of Figure 4.3). Thus, when the operator has the main system microphone connected to a specific channel and queues the microphone, that channel's microphone PTT will go low (PTT is an active low

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

signal). Then, the inverted PTT for that channel will go high thereby disabling the four switch gates associated with the channel. This then causes a temporary isolation of that channel from all the meetings (and thereby all the other radios) which permits the operator to directly communicate with only one radio system if it is so desired. (Communicating with multiple channels can be accomplished by setting the main microphone to operate as the Master Microphone and selecting the Master Microphone to be connected to the desired meeting.)

The summing amplifier for each meeting is a simple DC isolated, five-input, inverting summer designed using a single OPAMP as shown in the center region of Figure 4.3. Note that the capacitors are used to pass the AC signals but block any DC component that may arise from the switch or any other circuitry. Resistors R_1, R_2, R_3, and R_4 are the feed resistors for the audio signals coming from the Differential Audio Speaker Input circuits. The R_5 resistor is the feed for the audio signal from the Master Microphone. The gain resistor is R_6. The equation for the meeting output voltage is given by:

$$V_{meeting\,out} = -R_6\left(\frac{V_{in1}}{R_1} + \frac{V_{in2}}{R_2} + \frac{V_{in3}}{R_3} + \frac{V_{in4}}{R_4} + \frac{V_{inM}}{R_5}\right)$$

Note that R_1, R_2, R_3, and R_4 can be used to control individual channel signal amplitudes however currently the design assumes that they are all the same. Thus, the above equation reduces to:

$$V_{meeting\,out} = -\frac{R_6}{R_1}\left(V_{in1} + V_{in2} + V_{in3} + V_{in4} + \frac{R_1}{R_5}V_{inM}\right)$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

This is an inverting summer with a gain of R_6/R_1 for the four audio channels and a gain of R_6/R_5 for the Master Microphone. The gain differential for the Master Microphone allows for it to be a passive microphone which is typically at a very low signal level. Based on initial testing the microphone was found to have a maximum signal around 250 mV while the test radio signals were around 2.5 V (peak-to-peak). Since the 2.5 V was a good system voltage to work with the speaker audio signals from the radios were given a gain of one ($R_1 = R_2 = R_3 = R_4 = R_6 = 100$ KΩ). The microphone then needed a gain of 10 to put it on par with the other signals. Thus, the selection was made for $R_5 = 10$ KΩ. These values can be adjusted in a final design.

Note that early in the design stages it was determined that a non-inverting summer, as shown in Figure 4.4, was not an acceptable choice for the meeting summers. One reason for this is that the gain equations are much more complicated, as the gains for each input depend not only on the feed resistor for that input but for the other inputs as well.

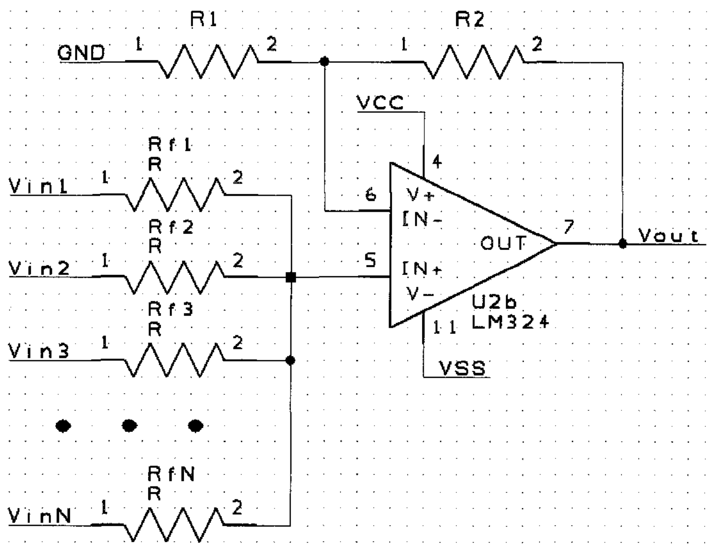


Figure 4.4 – Rejected Non-Inverting Summer Circuit

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

However, the primary reason for rejecting the non-inverting design is that each of the inputs will not necessarily always be connected. When a switch is turned off that line's feed resistor is essentially changed to an extremely high impendence. The result will then be a shift in the gains for the remaining channels! For example, look at the simple case with N inputs where all the inputs use the same feed resistor and hence have the same gain:

$$V_{out} = \left(\frac{R_1 + R_2}{R_1}\right)\frac{\left(V_{in1} + V_{in2} + V_{in3} + \ldots + V_{inN}\right)}{N}$$

If a channel is switch off the matrix, then N will be changed (to N-1) resulting in new gains for the remaining channels that are scaled from the original by a factor of N/(N-1). This is obviously unacceptable for the desired application and should not be used.

Microphone Input Buffers

There are audio input connections for each of the four radios and for the Master Microphone. These are however connected to a rotary switch that allows a single microphone to act as the microphone for each of the individual radios and the Master. The microphone's PTT signal is also passed through the 5-way rotary switch and directly into the five microphone PTT inputs on the CPLD, with a pull-up resistor attached. Each radio microphone input buffer feeds directly into the output portion of the switch matrix for that radio. The input buffer for the master microphone feeds into the front end of the switch matrix and can be connected to either the Meeting A or Meeting B audio summer circuits. A typical microphone buffer circuit is shown in Figure 4.5.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

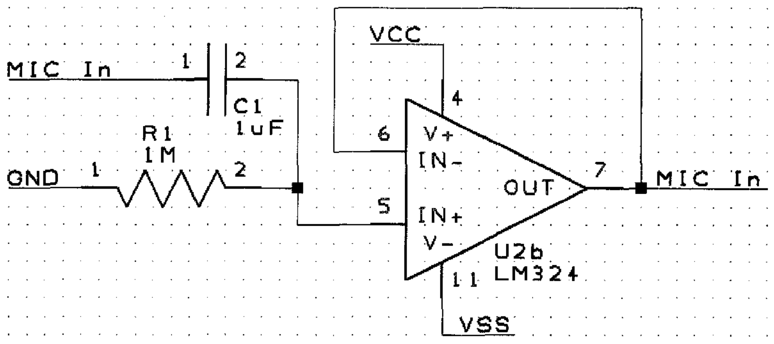


Figure 4.5 – Microphone Input Buffer Circuit

Note that C_1 and R_1 create a high-pass Resistor-Capacitor (RC) input filter that creates AC coupling while isolating the DC. Also the microphone input is assumed to be single-ended with a ground reference line (as is the case for a typical, passive microphone). The resistive input impedance is approximately 1 MΩ.

Output Audio Buffers and Output Switch Matrix

Figure 4.6 shows the output audio buffers and the output switch matrix portion relating to the first audio channel. Note that four such circuits are in operation on the prototype, one for each output channel.

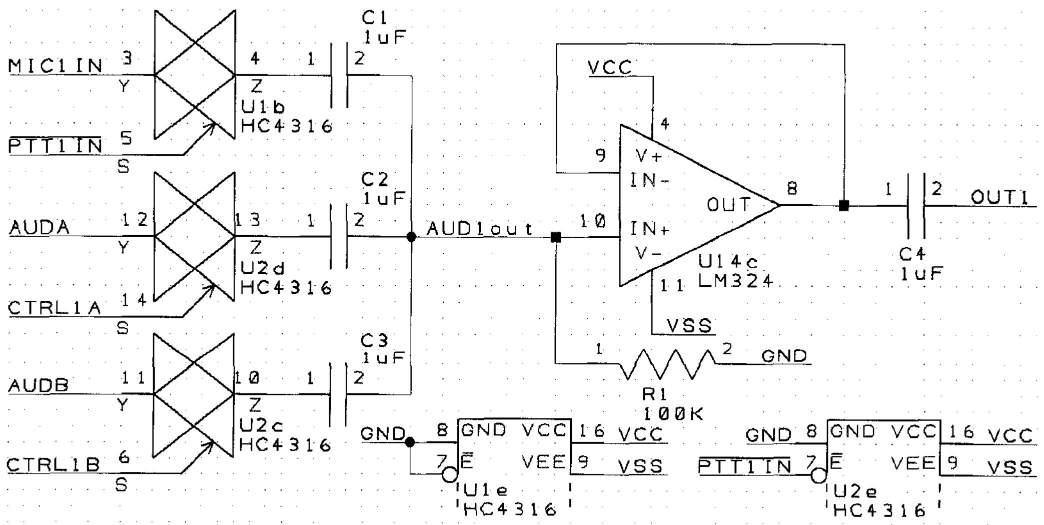


Figure 4.6 – Output Audio Buffers and Output Switch Matrix Circuit

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The output audio (AUD1out in the circuit diagram) is buffered using a unity gain buffer. At some point it may be advisable to create a variable gain output stage for signal level scaling however for the prototype design this was not necessary. The output audio buffers are each fed directly from three different sources. The Meeting A Summer output, the Meeting B Summer output, and that specific channel's microphone buffer (AUDA, AUDB, and MIC1IN respectively for Channel 1 as depicted in the diagram). Over-driving between the three inputs is avoided by the presence of the output half of the switch matrix, the operation of which is explained in the following.

The output switch matrix works in the same manner as the input switch matrix. The control lines buffered from the microcontroller act as the enables for the individual switches (U2d and U2c) that pass the audio signals from the two meetings. Again note that both of these gates also have a common master enable for the entire IC package that is controlled by the inverted PTT for Channel 1. The third input on the matrix is from the Channel 1 microphone. Its individual gate enable is also controlled by the inverted PTT for Channel 1, however all individual gate enables are active high enabled. This means that when this switch is ON (PTT1in is low and its inverse is high) due to its individual gate enable the other two switches that feed the Channel 1 output are automatically OFF because their quad package IC enable (which is active low) has been disabled. This prevents a drive conflict between a meeting and the channel's microphone audio circuitry. Note that all four of the channel microphone switches that feed each of the individual outputs use the same quad switch IC and thus share a common package level switch enable. This however is grounded so that it is effectively not used and the only

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

thing controlling each microphone audio switch is the unique, inverted PTT signal for that channel's microphone.

In the output audio buffer circuit diagram shown in Figure 4.6, C_4, just prior to the output terminals, is used for DC isolation. R_1 is used for the purpose of bleeding off any DC voltages that have crept into the audio paths which could have negative effects. Capacitors C_1, C_2, and C_3 also serve as DC isolators from the feeding audio sources.

Push-to-Talk Signal Detection

PTT outputs are not always available from radio systems. Furthermore, in the case where the radio's speaker output is used as the audio input to the interoperability system, an output port on the radio may not even exist. However, it is still necessary for routing control purposes, as well as for radio broadcast triggering, that PTT signals exist for each of the input audio signals. This mandates that some form of PTT detection/generation circuitry be used.

Previous, similar public safety interoperability solutions that have entailed software based solutions with limited COTS hardware have utilized commercially available specialized amplitude scaling and PTT signal generation hardware [10, 24]. An example device is the SignaLink Model SL-1+ Sound Card – Radio Interface produced by TigerTronics (http://www.tigertronics.com). While its Auto-PTT circuit functions correctly its use for this design would require the purchase and installation of four additional "boxes" which adds cost and consumes a significant amount of physical space. (It also has a delay in the PTT signal generation which can be reduced in this design.) It was determined therefore that as a custom circuit board was already being fabricated, a

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

custom hardware solution would be optimal in meeting the original cost, simplicity, and space constraints.

Another, custom, hardware solution was proposed by Mock in [22] as an interface between each radio and the PC used for his VoIP solution to public safety radio interoperability. A circuit diagram of Mock's Mobile Radio/PC Interface is shown in Figure 4.7.

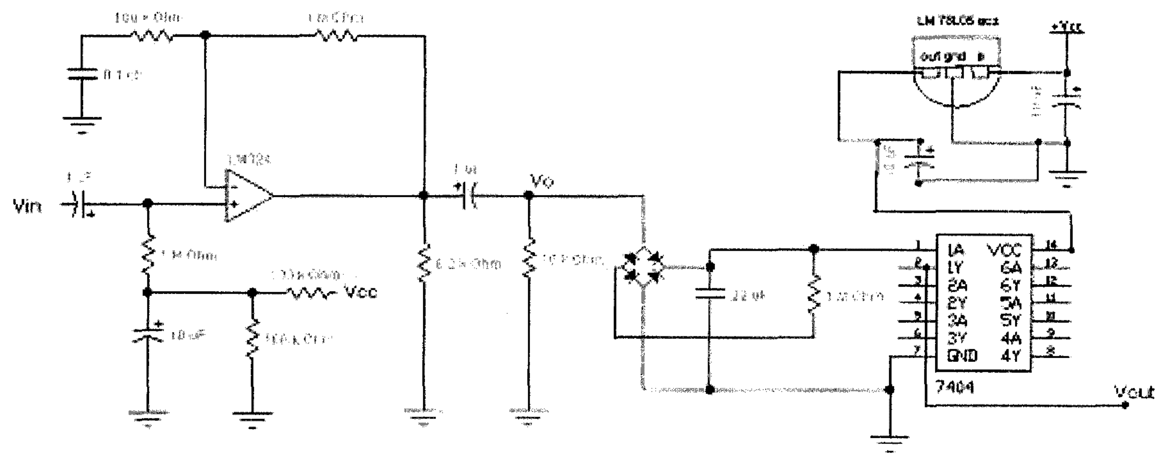


Figure 4.7 – John Mock's PTT Generation Circuit

The designed circuit appears to have functioned well, however, it only performed detection for a single audio line and was slightly over-complicated, entailing the use of a large amount of hardware considering the relatively simple task required of it. While potentially sufficient for a prototype design this would eventually result in a higher component cost and greater real estate requirements, all of which have negative impacts for the approach taken in this thesis. For this reason the circuit was only used as a reference starting point with modifications made to strip away superfluous aspects. Figure 4.8 shows the modified circuit design used in this work.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

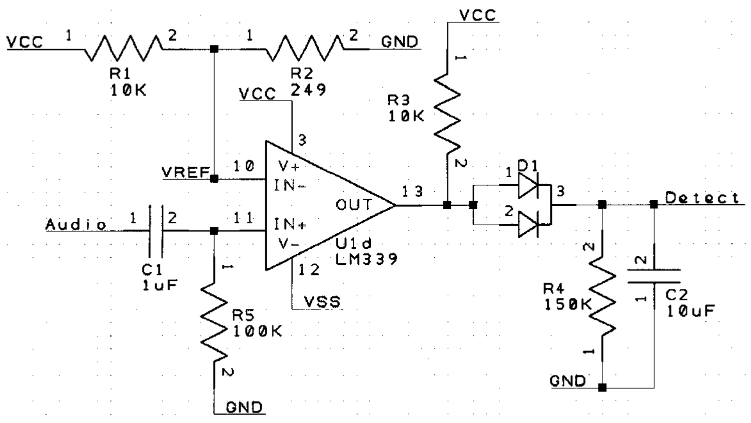


Figure 4.8 – Modified PTT Signal Generation Circuit

First note that Mock's design calls for a DC biasing of the circuit to permit 0V/+5V operation. As this project supplies a -5V/0V/+5V rail system some biasing resistors could be removed. Secondly, instead of using an OPAMP set to a gain of 10 with predicted saturation (which is not an effective design), this project utilizes a comparator with a reference threshold. The reference voltage is set by R_1 and R_2 in Figure 4.8 and the equation for this simple voltage divider is given by:

$$V_{ref} = \frac{R_2}{R_1 + R_2} V_{cc}$$

Whenever the voltage on the audio line exceeds V_{ref} the output of the comparator is pulled high (by R3). Note that with $R_1 = 10K\Omega$, $R_2 = 249\Omega$, and $V_{cc} = 5V$ this equates to approximately 125 mV, which exceeds the typical noise threshold observed on the circuit board. (Note that C1 provides DC isolation and that R5 acts as a sink to bleed off any DC voltage that is able to get into the signal path.)

Another design variation is that instead of performing full-wave rectification on the signal it was determined that only the positive phase of the signal was required. This eliminates the need for a full-wave bridge rectifier (four diodes). Instead, a simple diode

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

is placed at the output of the comparator in order to prevent current backflow. This is then followed with the RC low-pass filter similar to that used in the reference design. The RC circuit is designed to hold the PTT signal on for a brief amount of time to allow for pauses in an officer's speech. The time constant is given by:

$$\tau_{RC} = RC$$

For the base-level design this time constant was set to 1.5 seconds with a 10 μF capacitor and a 150 KΩ resistor. (Note that the time constant is not necessarily the time that the PTT will be held after the signal is terminated. This value depends on the digital transition level on the inputs to the CPLD. It does however provide a rough approximation for the prototype design.) This can be easily adjusted in testing simply by changing the resistor or capacitor values (increase either value to increase the hold time). It is however much easier to deal with changes in the resistor while leaving the capacitor constant. Note, if R becomes too large, consideration will also need to be made of the input impedance of the port pins on the CPLD (at a very minimum 500 KΩ [25]).

The last major variation in the PTT signal generation circuit has already been alluded to and is the use of the CPLD instead of a digital inverter (i.e. an HCT7404 inverter). This was done since the CPLD was already needed for PTT signal routing and therefore saves one additional component from being needed in the circuit. The digital inversion actually serves two important purposes. First, the inversion converts the signal from an active high PTT signal to the industry standard active low PTT signal (0 V signal when a voice transmission is detected and 5V at all other times). Less obvious however is that it acts as a second threshold detector that will convert the analog RC filter output into the digital PTT signal. Therefore whenever the PTT detected signal is above 2.0 V

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

the CPLD will treat it as high (+5 V). Below 0.8 it will be treated as low (0V, PTT inactive). Treatment of the range of input voltages from 0.8 to 2.0 V is indeterminate [25].

Robust System Design Constraint and the Shutdown Relays

One of the most important design criteria is that the system be robust and resistant to failure. This is extremely important with regards to any equipment in the public safety setting. By using dedicated hardware instead of more complex devices, such as a PC, certain measures have already been taken to reduce the likelihood of failure. However, some process should also be in place that can cope with a full system crash without requiring activity by the operating personnel.

For this reason shutdown relays were added to the front and rear ends of the device as depicted in Figure 4.9. (The relays chosen are three of the G6A-434P-ST-US-DC5 4PDT relays by Omron Electronics Inc.) This figure uses the original Revision A prototype (which didn't have the relays) as the MRCM Board block module in the diagram.

Note that on the front end there is only one relay. It takes the PTT signals from the five microphone connections on the rotary switch and either maps them into the PTT pins on the five microphone inputs or by-passes the whole board and goes to the output PTT relay. On the output side one relay, the output PTT relay, ties the connections that go to the four radio PTTs to either the four PTT output signals on the circuit board or to the input PTT relay. The third relay, the output audio relay, maps either the audio signals

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

from the microphones or the output audio signals from the board to the audio inputs on each of the connected radios.

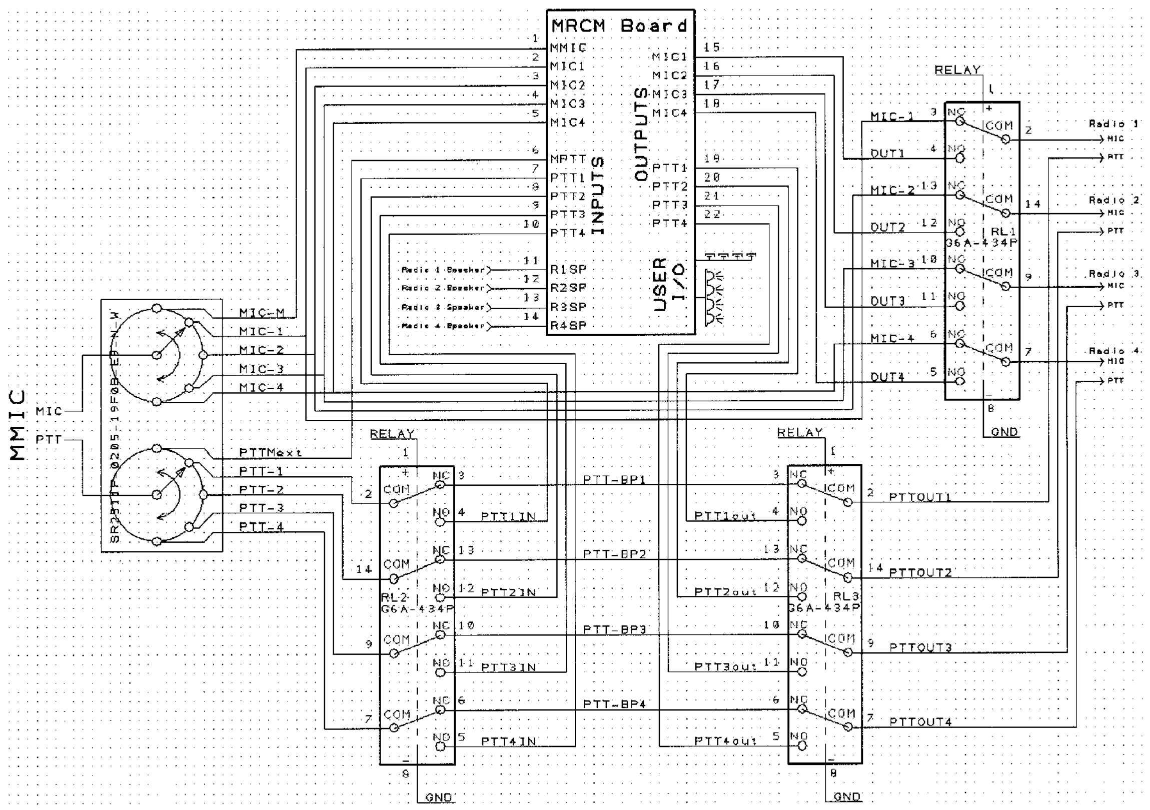


Figure 4.9 – Shutdown Relays for System Bypassing

By default, with the relays un-powered, i.e. in the normally closed position, the system is configured with the signals all bypassing the circuit board, meaning that the system is totally unused. However, if the relay control signals are set high (+5 V) the contacts all close and the signals are now routed through the circuit board. The control signal is operated directly by the microcontroller that controls the entire circuit board and its interfaces. The port on the CPU is by default in the relay off position (signal high or at +5V as the control is active low) on power-up. Once the control setup operations are complete the relays are closed and the status is checked at multiple times within the

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

firmware to insure that nothing is wrong. If something does occur the relays are abruptly opened.

Note that if the on-board CPU were to freeze, the relays would be latched in the previous state, which might be the ON state. However, in the case where everything fails the personnel in charge of the equipment need only shut the board off (at the main power switch), automatically opening the relays and essentially disconnecting the board from all the radios. Without the relays powering down, the circuit board could ground all of the PTT signals on the radios, thereby ordering them to transmit. This would then jam all the channels that were being used by the radios, again an unacceptable response to a system failure.

Due to the fact that the CPU port pins default to high at start-up it is desirable that the control signal from the CPU that is used be active low. However, to insure that the relays are open when the board is powered off the relays themselves have to be active high (as the G6A-434P relays selected for the task are). Coupled with the fact that the relays each need 72.50 mA of continuous control current to close the mechanical contact, while each CPU pin can only sink 10 mA maximum when low and can only source around 10-50 μA in the high state, some form of buffering is needed. The resulting solution is shown in Figure 4.10.

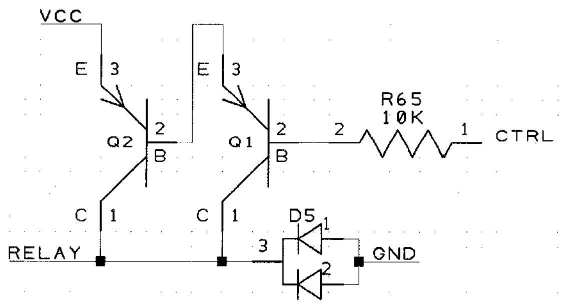


Figure 4.10 – Shutdown Relays' Control Drive Circuitry

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

All three relays use the same control line, however since there are three and each needs 72.5 mA to turn ON, a total of around 220 mA is needed. In order to accomplish this, two PNP transistors (MMBT4403) were placed in a Darlington pair arrangement. This provides sufficient gain to the signal from the CPU and also operates at a current that is well within the specification of the transistors (maximum of 600 mA). Note that diode D_1 is used to protect the relays from negative voltage spikes and resistor R_1, with a value of 10 KΩ, limits the drive current from the CPU port pin to a maximum of 500 μA.

Power Circuits and Voltage Supplies

The entire system is designed to be contained, if necessary, within a car. For this reason the power supply takes a DC input voltage of around 13.8 V and uses it to create +5 V (V_{CC}) and -5 V (V_{SS}) power supply rails. The basic power supply diagram is shown in Figure 4.11.

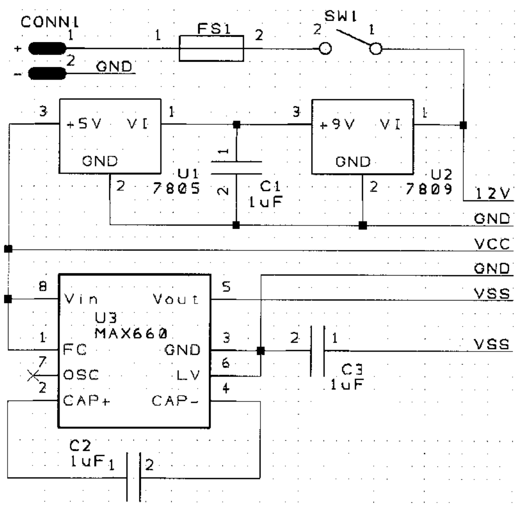


Figure 4.11 – Power Supply Circuitry

The two voltage regulators bring the input voltage down to +9 V and then +5 V. The MAX660 Switched Capacitor Voltage Converter is then used to generate the -5 V

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

supply from the +5 V supply. This is done by placing it in the low-noise switching voltage inverter configuration shown. The actual voltage conversion is accomplished using a simple charge pump circuit involving two external capacitors. The +5 V rail is used in both the digital and analog portions of the circuit board however the -5 V rail is only used in the analog portion.

The reason for also including a negative voltage supply for the analog sections of the board was to simplify the circuits by eliminating the need for a constant DC offset which might need to be supplied by numerous offset references (in order to adequately supply the whole circuit board), essentially requiring another power rail anyway. Furthermore, use of the negative rail increases the range of the circuits with little additional overhead.

Component	**Comments**	**Current**
LED	10 mA each × 6 (max)	60 mA
CPU	peak, active mode	14 mA
XC9536		~30-40 mA
Analog Circuitry		~20-40 mA
Relays	72.5 mA each × 3	217.5 mA
TOTAL:		**~371.5 (max)**

Table 4.1 – Original Supply Current Estimates

Lastly note that the circuit board is equipped with a power switch and with an automatic fuse. The fuse chosen is the MF-MSMF075/24-2 by Bourns Inc. which is designed to automatically reset itself. It is specified for 750 mA and 24 V which should exceed the specifications for the system. Note that the estimate on total system current

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

(for the Rev. B and C circuit boards) was around 371.5 mA. The recommended safety margin of a factor of two then brings the maximum supply current to around 743 mA. The original estimates for the current breakdown are shown in Table 4.1.

Actual measured consumption for the circuit board in laboratory testing is contained in the prototype design. Due to the high amount of current it is recommended that the voltage regulators also be equipped with a heat sink. (The parts, as laid out in the Revision C PCB design, are such that they can share a common, dual voltage regulator heat sink. Currently, however, the bottom fins must be bent upward to avoid contact with a relay.)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER V

DIGITAL PTT MAPPING AND SWITCH CONTROL

Overview of PTT Signals and Their Origins

The Push-to-Talk (PTT) signal is a signal employed in half-duplex communications systems where reception of a signal is precluded by an active transmission, meaning data can only travel in one direction at a time (in contrast to full-duplex systems, such as telephones, where audio/data can be transmitted and received simultaneously.) This is commonly the case with two-way radio systems including radios used in public safety. The PTT signal is activated using a momentary button that can be pressed by the radio operator (or through some digital control device) to switch from reception mode to transmit mode. Upon release the PTT signal switches off and the radio automatically reenters receive mode. For this design it is important to note that received audio will be muted if the PTT is keyed; therefore, when an active radio's audio and PTT signals are being routed to the connected radios it is important that the original sender not receive the PTT queuing as well. This must be handled by the digital mapping logic. (Note that presentation of the audio on the output should have no negative impact on performance as the radio ignores audio on the microphone so long as the PTT is not queued. The PTT signal is typically active low for circuit design reasons, meaning that when the PTT signal line is low (0 V) the radio will be placed in transmit mode.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The PTT signals used in this project originate from two different areas. The first are the PTT signals that are generated by the use of the microphone. Most microphones have a PTT signal along with the voice audio and the voice audio is passed to the radio for transmission only when the PTT button is keyed (pressed). Note that for this case the five microphones used by the system, one for each of the four radios and the master microphone for direct communication on a meeting group, are in fact only one with a simple rotary switch attached to the output to select the destination input to be used. The PTT signals for each are then fed directly into the CPLD IC for control and distribution as needed. Note that these signals are active only from a direct queuing of the in-vehicle microphone by the personnel in charge of this device.

The second set of PTT signals originate from the speaker audio inputs themselves and are generated using the analog circuitry discussed previously. These signals are created whenever an attached radio is observed to have received some form of audio input. This means that a sister radio has been queued by another officer at a remote location. The device needs to now pass that PTT signal to the radios of other systems that have been connected to the queued radio and pass the audio signal along as well. Note that these PTT signals are active high and not yet digitized at the input to the CPLD. However, the CPLD will perform the necessary inversion and digitization as it distributes the signals for use by the switching matrix and radio PTT outputs.

Discrete Logic versus a CPLD

Early in the design phases for the project the use of discrete digital logic was planned for implementation of the PTT mapping. However, it quickly became clear that

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

this option had a number of drawbacks and the original Revision A design used a programmable logic device (PLD) instead.

The first reason for rejecting the use of discrete components in favor of a CPLD was that, as this was a prototype design, it could be assumed that changes would need to be made to the PTT mapping equations. This flexibility could only be accomplished through the use of some reprogrammable logic.

Closely related to this is the idea of routability of the equations. As the PCB was manually routed, implementation of the equations using the digital logic was quite involved, especially as multiple results within the circuit needed to be fed back as inputs to others. This also would have had to have been repeated each time a new revision was made to the PTT equations.

Other concerns relating to the cumulative delay time, significant amount of physical space required for all the discrete components, financial cost entailed from using multiple ICs, and the assumed power consumption increase that they would entail all supported the decision to transition to a PLD.

CPLD Digital Logic PTT Mapping Implementation

The device selected for implementing the PTT mapping equations was the XC9536 CPLD by Xilinx. It has an estimated typical supply current of I_{CC} = 30 mA. The pin-to-pin delay can also be selected as low as 5 ns to minimize the PTT delay. (Note that the XC9536 comes in five different delay time variations of 5, 6, 7.5, 10, and 15 ns. Based on the nature of the PTT mapping equations no more than two delay periods should be needed for propagation of a PTT transition. With a maximum delay

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

time of 30 ns any of these chip versions would be acceptable.) Most importantly, the chip also has the feature of being in-system programmable via a Joint Test Action Group (JTAG) port which meets the design requirement of flexibility.

The digital mapping logic implemented in the CPLD was written in VHDL; however, the following provides a basic outline of the code's operation. The mapping of the PTT signal is done in a manner similar to that conducted for the audio signal. The main difference is that the PTT signal on a specific input cannot also trigger that channel's corresponding output PTT.

As with the audio signals, intermediary Meeting A and Meeting B PTT signals are formed (PTT_A and PTT_B respectively). A meeting will have its PTT active if one of the PTT detection circuit outputs (signals Det_1, Det_2, Det_3, and Det_4) on a channel connected to it goes high. (Note that the meeting PTT signals are active high.) Also, in the notation used for the PTT equations the C_{nm} terms correspond to the control lines from the microcontroller. A high control line signifies a connection between Channel *n* and Meeting *m* while a low control line means that there is no connection. Channel M corresponds to the Master Microphone. The impact therefore of a channel *n* on the PTT of meeting *m* is given by:

$$PTT_{m-n} = \left(C_{nm} \cdot Det_n\right)$$

It also will be active if the Master Microphone is queued and it is connected to that meeting. This is described for meeting *m* by the equation:

$$PTT_{m-Master} = \left(C_{Mm} \cdot \overline{PTTM_{in}}\right)$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The complete PTT for a meeting will then include the impact from each individual channel and the Master Microphone. The resulting equations, given for both the PTT of Meeting A and Meeting B, are then:

$$\begin{aligned} PTT_A = {} & (C_{1A} \cdot Det_1) + (C_{2A} \cdot Det_2) + (C_{3A} \cdot Det_3) + \\ & (C_{4A} \cdot Det_4) + \left(C_{MA} \cdot \overline{PTTM_{in}}\right) \end{aligned}$$

$$\begin{aligned} PTT_B = {} & (C_{1B} \cdot Det_1) + (C_{2B} \cdot Det_2) + (C_{3B} \cdot Det_3) + \\ & (C_{4B} \cdot Det_4) + \left(C_{MB} \cdot \overline{PTTM_{in}}\right) \end{aligned}$$

The equations for the individual channel PTTs are again similar in basic structure to the audio signal mapping. However note that there must be a special exception so that a channel's PTT isn't active for the case where that channel's PTT detection/generation circuit detects an audio signal. Thus, if a channel is connected to a meeting, and that meeting's PTT signal is active (high), then that channel should also have an active PTT output (low) so long as its own detect signal is also low. Also, if the officer manually queues the microphone PTT when the rotary switch selects that channel, the channel's PTT should automatically queue that radio without regard to other activities (so that it behaves similar to a normal radio). This is described for all four of the channels by the equations:

$$PTT_1 = \left(\overline{(C_{1A} \cdot PTT_A) + (C_{1B} \cdot PTT_B)} + Det_1 \right) \cdot PTT_{1in}$$

$$PTT_2 = \left(\overline{(C_{2A} \cdot PTT_A) + (C_{2B} \cdot PTT_B)} + Det_2 \right) \cdot PTT_{2in}$$

$$PTT_3 = \left(\overline{(C_{3A} \cdot PTT_A) + (C_{3B} \cdot PTT_B)} + Det_3 \right) \cdot PTT_{3in}$$

$$PTT_4 = \left(\overline{(C_{4A} \cdot PTT_A) + (C_{4B} \cdot PTT_B)} + Det_4 \right) \cdot PTT_{4in}$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The complete VHDL code can be found in Appendix A along with the pin assignment constraints file for the inputs and outputs. Figure 5.1 shows the pin connection diagram used in Revision C.

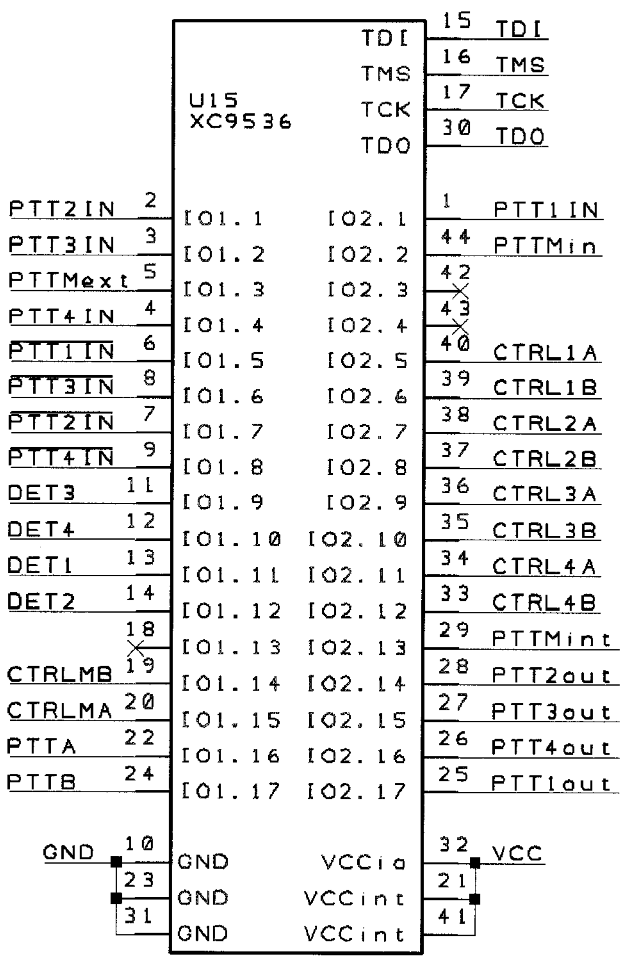


Figure 5.1 – CPLD Pin-out Circuit Diagram

Note that the meeting PTT signals, PTT A and PTT B, while not explicitly needed, have been supplied output ports on the CPLD so that they can be used for troubleshooting if necessary. Also, the CPLD is used for other miscellaneous logic tasks that include inverting the input microphone PTT signals for each channel (one through four) for use in controlling the analog switching matrix. It also performs the function:

$$PTT_{Min} = PTT_{Mext} \cdot PTT_{M\,\mathrm{int}}$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

This operation provides a special feature that allows a connected PC to take the role of the Master Microphone and implement the Master Microphone PTT function via software. Note that PTT_{Min} is the PTT signal used for all aspects that relate to the Master Microphone PTT in both the switch matrix and in digital PTT mapping equations. PTT_{Mext} is the external Master Microphone PTT signal that actually comes from the microphone via the 5-way rotary switch. PTT_{Mint} is the internal Master Microphone PTT signal that is generated by the microcontroller. It can be activated (active low) by a PC or other device issuing a serial software command to the microcontroller via the attached serial port. Thus, if the main microphone is connected to the audio output of the PC, then the PC can actually be used to supply the officer's audio input to the system. This can also allow the main microphone to be completely eliminated and the PC's Project54 microphone used instead (although direct, individual radio communications would no longer be supported as the PTT trigger is not available for each channel microphone input; a feature that should rarely be used in any case).

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER VI

DIGITAL CONTROL HARDWARE & FIRMWARE

The Main Digital Control Hardware

In order to meet the desired goals for the system it was decided that some of the hardware circuits would be implemented in reconfigurable digital hardware. The digital control hardware, shown in Figure 6.1, serves three primary purposes. First, it controls the analog switching matrix through Channel-Meeting control lines. Secondly, it provides a means of reprogramming the digital PTT mapping CPLD. And lastly, it provides interface to the user, through both the manual and serial port control methods.

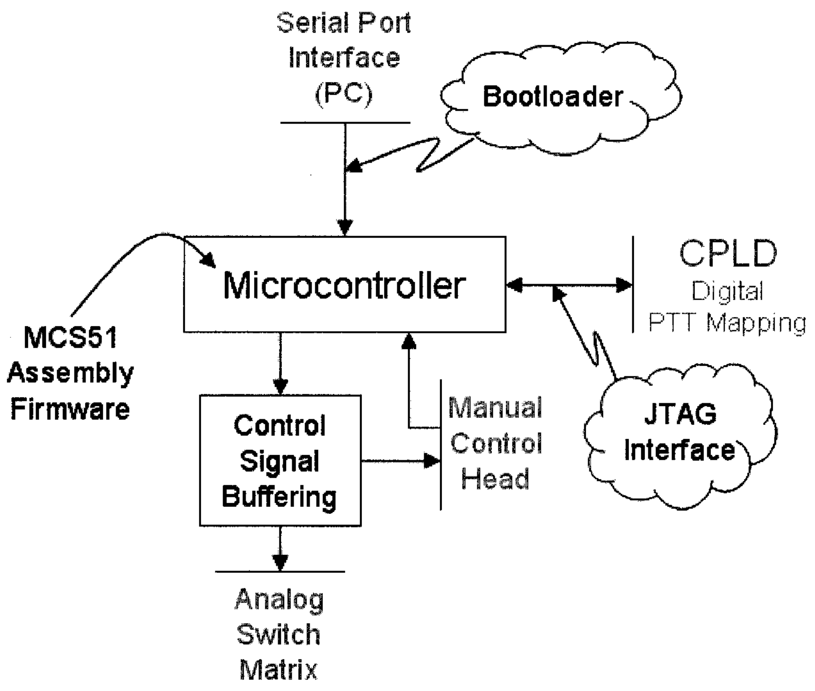


Figure 6.1 – Digital Control Hardware Block Diagram

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The heart of this digital control hardware is the reprogrammable main system embedded microcontroller shown at the center of the figure. Details on its attributes and basic characteristics needed to perform the necessary functions are contained in the following embedded microcontroller section. The operation of the microcontroller is determined by the firmware that is loaded onto it via the serial port during the bootload processes. For added system flexibility this bootloading can be performed in-system meaning the microcontroller does not need to be taken out of the circuit. Both the firmware and the method for loading it into the microcontroller are also discussed in sections within this chapter.

Another critical portion of the digital control hardware is its ability to reprogram the CPLD for correct PTT signal mapping. This is accomplished through a JTAG interface between the CPLD and the microcontroller. Details on the generic interface and the specific procedures used in this design encompass another section of this chapter. Also covered is the microcontroller firmware used for implementing the interface and the PC level files that need to be transferred.

The chapter concludes with section describing the digital hardware used for the buffering of the control signals that need to be distributed to the analog switching matrix as well as to the hardware user feedback LEDs.

The Embedded Microcontroller

Usage of a Microcontroller

While front-end analog hardware is used for the routing of the actual analog audio signals and high-speed digital logic is used for mapping the PTT signals some flexible

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

method is needed to control the whole system and provide interfaces for the user. This includes both the decision making involved in the alterations to the switch matrix control lines as well as responding to serial commands and manual control head button presses.

Initially the examination was made to see if it would be possible to perform this task using a Project54 IDB Interface device as a serial controller for a series of latches. This approach was however rejected for a number of reasons. First, the newest revision of the IDB Interface device will no longer have a general purpose parallel output port that could be used to control the analog hardware. Furthermore, the use of the IDB box would require some special firmware modification to it which would be impractical. A second major problem observed was that the device would no longer be stand alone. Without some form of on-board processing the possibility of using manual control buttons in parallel with the serial interface was remote. Lastly, limited by the IDB Interface device, the system would be very inflexible and it would be difficult to add any advanced features at a later time. For these reasons it was decided that a general purpose microcontroller would be selected as the primary digital control hardware. This would meet the needed computing power demands for the application while provide the required flexibility and still being fairly simple and low cost. For this purpose the 80C51FA, manufactured by Intel, was used for the Revision A design.

Introduction to the 8051 Microcontroller Family

The 80C51FA is a member of the Intel 8051 Microcontroller family which includes a range of 8-bit microcontrollers with a number of additional features from the base 8051 core. The family has been around since its inception with the 8051 in 1980

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

and currently has a wide range of applications and numerous alternate vendor clones. This ensures that it is available at a relatively low cost. The basic 8051 core includes 64 KB each of program and data memory, 128 bytes of RAM, 32 bi-directional I/O lines, a full duplex UART, as well as internal/external interrupt sources and 2 timers. It also has a large assembly instruction set (MCS-51 Assembly) including significant single bit manipulation capabilities and hardware multiply/divide commands. The 80C51FA selected for the design has the additional features of a third timer, a total of 256 bytes of RAM, 7 interrupt sources, 4 interrupt priority levels, and a Programmable Counter Array - features that increase it usefulness in the necessary control operations. The design was then implemented with this CPU in the early stage prototype model, Revision A of the PCB.

However, at the time of the re-design for Revision B, which was to add some additional features to the system, Intel released its End-of-Life notice for the 80C51FA and its entire embedded product line (starting in 2007). For this reason an alternative was sought among the various other vendors for implementation in Revision B. The decision was made to use the Atmel AT89C51RC2, which still uses the MCS-51 Assembly and all the related support infrastructure, for the main control functions.

The decision in favor of this microcontroller, which is very similar to the original design's 80C51FA, was based on a number of considerations. First, the Atmel variant has a Keyboard Interrupt Interface port. This eliminated the need for the external OR gate (implemented with a CD4068 IC in Revision A) on the button signal lines which was needed to allow the use of only one interrupt source for triggering of a user input. This also allowed for the pull-up resistors on the buttons to be removed as the buttons

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

were now directly tied to only the CPU port pins, which exhibit a mild pull-up characteristic in the input mode. With direct CPU access the HCT541 read latch was also no longer needed.

The other major feature of the Atmel AT89C51RC2 microcontroller is that it includes 32 Kbytes of internal FLASH program memory that can be reprogrammed via the serial port. This is accomplished through the use of a special embedded bootloader algorithm in an additional section of memory. Details are given in the following section. Note however that this allows the removal of the external 28-DIP PROM and the HCT573 program high address byte latch, which results in the saving of further PCB space.

AT89C51RC2 Microcontroller Connections

The main control for the entire system is provided by the microcontroller. Figure 6.2 shows the pin connections diagram for the AT89C51RC2 used. Note the external oscillator with a value of 22.1184 MHz that is used by the microcontroller as the main internal clock cycle. Also note the Reset and Program 2-pin header connectors, PL2 and PL1 respectively, which are used in the ISP programming processes.

Capacitor C_1 is used to automatically generate a power-on reset for the microcontroller. This is accomplished, in tandem with an internal resistor, as the reset pin will be pulled high when power is first applied to the board (the capacitor transmits the transient from V_{CC} into the reset pin). This then bleeds off through the internal resistor over a short period of time and the microcontroller then exits the reset state and begins operation at address 0000h in program memory.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

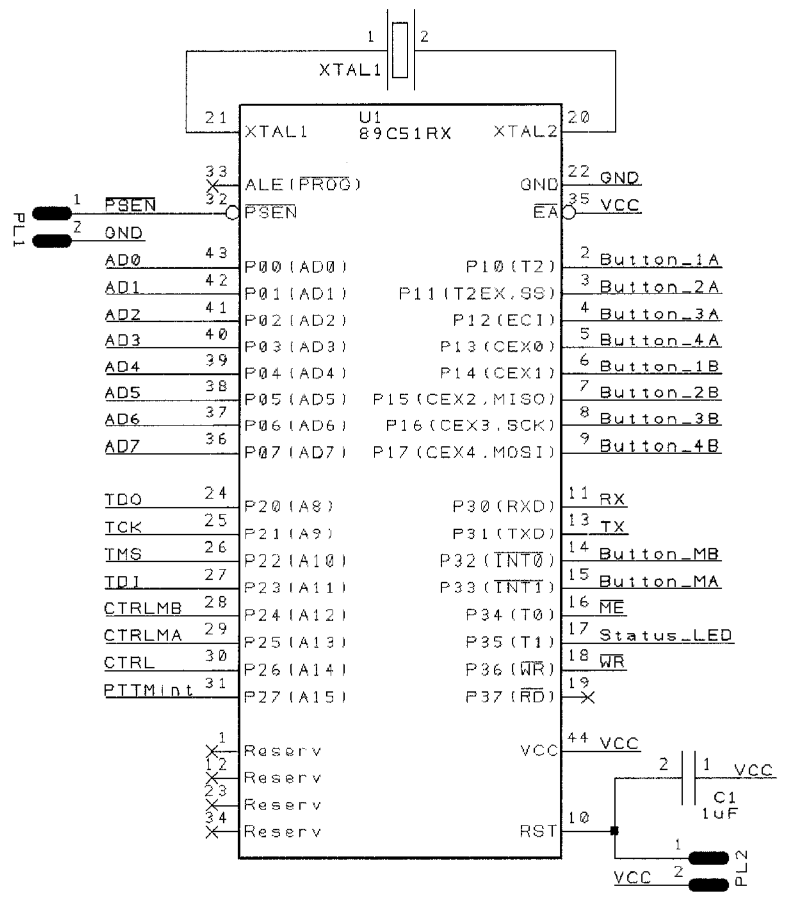


Figure 6.2 – Microcontroller Pin Connection Diagram

Note that the ten port pins on the microcontroller that are named starting with Button (the whole of Port 1 with two from Port 3) are inputs from the ten buttons on the manual control head. The Port 0 pins AD0-AD7 are used to interface to the control line buffering circuitry, as are CTRLMA and CTRLMB from Port 2. TDO, TCK, TMS, and TDI, all on Port 2, serve to provide the JTAG interface to the CPLD. The other pins perform miscellaneous task relating to general microcontroller operations.

General MCS51 Firmware Implementation Overview

A block diagram of the main assembly code firmware that is running on the 80C51 microcontroller is shown in Figure 6.3. Upon reset the program performs some

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

initialization operations that set up the output control lines, serial port, and other program registers. The main system LED is then toggle to notify the operator that the device is now in operation and then the main idle state is entered.

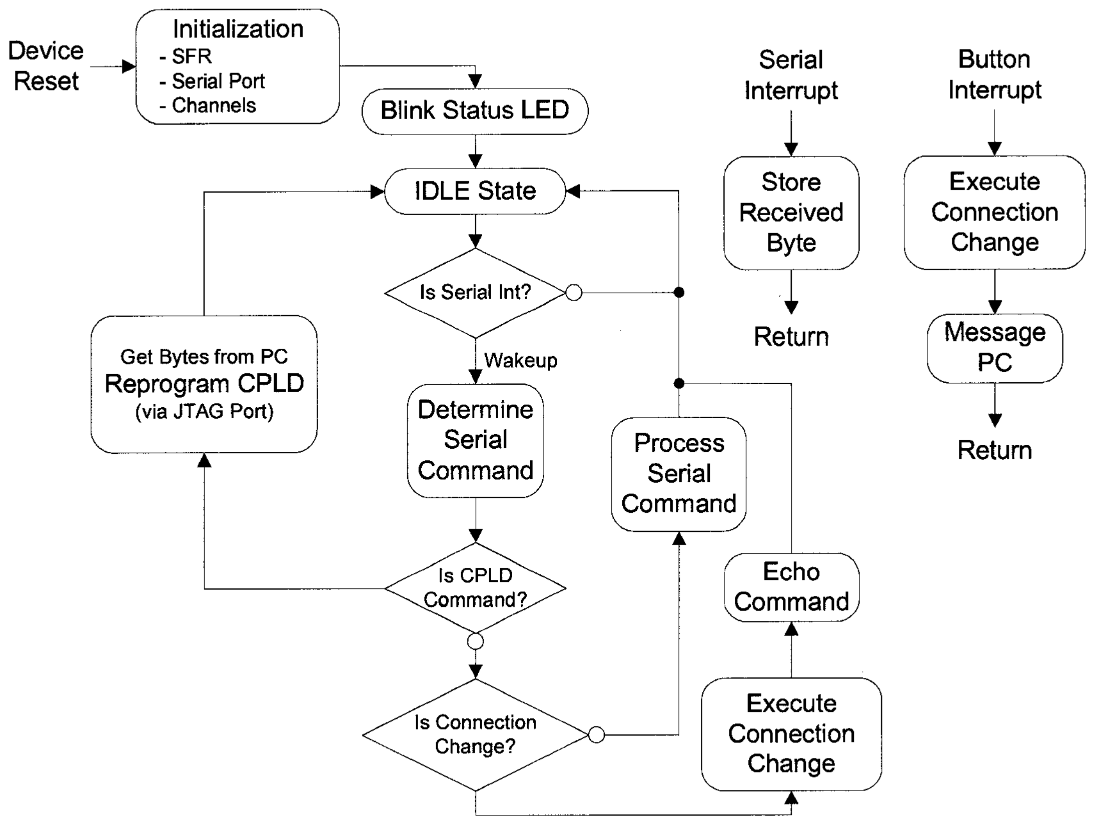


Figure 6.3 – Block Diagram of Microcontroller Firmware

Note that the program is dominated by a main control loop accessed after the occurrence of an interrupt, either from the serial port or from a button press on the manual user interface. Once woken from its idle state at the start of the loop the program checks to see if the interrupt was a serial interrupt. This is done by checking the head and tail serial port circular buffer pointers. If they are equal the queue must be empty while if they are not it must contain received serial data that needs to be processed. If it is not a serial interrupt then the microcontroller returns to the idle state. Otherwise, the program determines which serial command it was. If it is the command to reprogram the CPLD

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

that control loop is entered (which will only terminate after resetting the entire system). If the command is a non-connection changing command then it is immediately processed, often by sending out a response via the serial port. A full list of all the available commands is contained in the Remote/Software User Interface section of *Chapter VIII - Device Operation & User Interfaces.* If the command orders a connection change that change is made and echoed back to the PC via the serial port, again to insure proper synchronization. An attached software program should not change its internally maintained connection states until it receives this return notification of a successful change from the hardware. After finishing a cycle through the main loop the program will automatically causes the microprocessor to enter the idle state so long as no additional byte are present within the serial port data stack. This consumes significantly less power and can be exited only in response to an interrupt (or a full device reset).

The button interrupt is entered when the user presses one of the ten channel-meeting connection buttons. The button interrupt determines which button was pressed and then makes the correct change to the control lines that lead to the CPLD and switch matrix, if needed. Following this the program will send out a notification over the serial port detailing the connection change so that synchronization can be maintained with any attached PC-based software.

The serial port interrupt is extremely simple. It takes the byte that caused it from the serial port register and stores it to the serial port circular data buffer or queue. It then updates the head pointer accordingly and insures that there is no imminent buffer overrun. The received serial command is actually processed by the main loop. (The serial queue is circular meaning that there are pointers to both the high address and low

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

address. Data is processed at the location specified by the low address or tail pointer and new data received from the serial port is placed at the location specified by the high address or head pointer. Additional serial port details are covered in *Chapter VIII – Device Operation & User Interfaces.*

It was critical that care be taken with the firmware designed for this project so that inadvertent connections not be made between channels. Also, the control byte must be checked to confirm that a channel is not connected to two different meetings, before any writes to the connection control port. (Note that the actual output byte is written to a 74HCT574 8-bit latch. This also helps to insure that the data is valid, particularly at startup, and additionally buffers the outputs from the CPU which are not capable of driving all the analog switches in the matrix.) This could cause both meeting summer OPAMPs to drive the same node with only decoupling capacitors and analog switches between. The results are both unpredictable and could possibly cause damage to some of the hardware. Additional details on the full MCS51 assembly firmware program that is used in the microcontroller are provided in Appendix C.

Bootloader for In-System Programming Flexibility

The biggest advantage of the transition to the Atmel AT89C51RC2 microcontroller used in Revisions B and C is that it can be reprogrammed at any time by using the internal bootloader for In-System Programming (ISP). The CPU supports both parallel and serial ISP. However, only the serial method is needed for this design as serial port access is available and the most desirable form. Also, the ISP operation does not require any additional high voltages but can work using only the standard +5 volt

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

V_{CC}. The basic hardware level procedure for preparing to program the internal program FLASH via the serial port is shown in Figure 6.4.

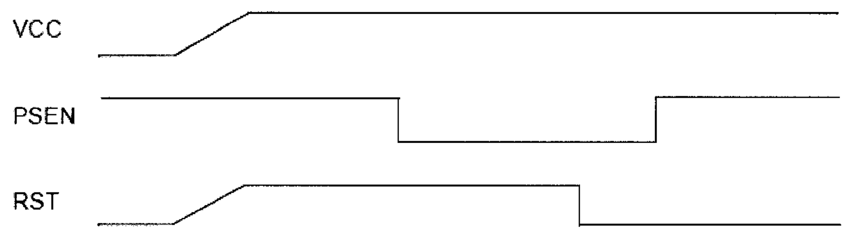


Figure 6.4 – Hardware Conditions for Bootloader Activation

Note that with the device in reset (RST held high) the PSEN pin should be grounded (0 V). Then the RST is released to allow execution, at which time the microcontroller will begin to run the bootloader embedded program. The PSEN line can be released shortly after exiting reset and left to float naturally to V_{CC} [26]. (Note that this discussion assumes that pin EA is also held at +5 volts which is the case for the current PCB design.)

Details of the process involved in performing the ISP operation on the Revision C PCB from a PC by using the Atmel FLIP utility are provided in Appendix B. This is the approach taken with the prototype. However, in order for the microcontroller to be programmed while in-vehicle it is desirable that the programming be done over the IDB network. To accomplish this task some additional details on the software end of the bootloading operation are necessary.

First note that the reprogramming via this method is conducted over the UART of the microcontroller. Transmission is conducted with the parameters shown in Table 6.1. Note that the bootloader program is designed to automatically determine the communications baud of the connected PC. This is done by the PC initially transmitting

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

the character 'U' to help the microcontroller determine the baud used. This is then echoed back to the PC to insure that the baud is correct.

Character:	8-bit data
Parity:	none
Stop:	1 bit
Flow Control:	none
Baud:	Autobaud to match PC

Table 6.1 - UART Communications Port Settings

The PC then can begin transmission of the HEX programming file to the bootloader on the microcontroller. This is done on a frame by frame basis. The format of the data frame sent via the serial port follows that of the Intel Hex records as shown in Table 6.2. This is the file format of the code that is generated by the assembler used for this project and contained within the MRCM.hex file. **Record Mark** starts the frame and should always be a ':'. This is followed by the **Reclen** byte which specifies the number of data/info bytes that follow the **Record Type** byte. **Load Offset** is the 16 bit starting load offset of the data bytes. The **Record Type** byte specifies the type of command that this frame contains. The **Data/Info** element is variable in length with zero or more bytes encoded as pairs of hexadecimal digits. Lastly, the **Checksum** byte is used for error detection. To determine its value each pair of ASCII hexadecimal digits in the frame elements **Reclen** through the end of the **Data/Info** is converted to a single binary byte and the sum taken. The **Checksum** byte is then the least significant byte of the two's complement of that sum. (Note that the sum of all ASCII pairs in the record from the **Reclen** field through the **Checksum**, after conversion to binary, will be zero.)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Record Mark ':'	Reclen	Load Offset	Record Type	Data/Info	Checksum
1 byte	1 byte	2 bytes	1 byte	n bytes	1 byte

Table 6.2 - Intel HEX Frame Format

The PC will transmit each character and wait for an echo by the bootloader. Once a frame is complete the bootloader will process it. If successful it will return '.', followed by CR (Carriage Return) & LF (Line Feed). A return of 'X', CR, LF represents a check sum error and 'P', CR, LF is returned for a security bit error. The programming software running on the PC should check each echo for the occurrence of an error. This frame transmission process is continued until the entire HEX file has been sent. Other features available via the bootloader should not be necessary for basic programming. They are however detailed in the Atmel manual [26].

This ISP feature of the microcontroller helps to accomplish the desired goal of having the system be fully flexible as now both the CPU and CPLD can be reprogrammed via the serial port. Note however that access to the Reset (RST) and Program (PSEN) headers on the PCB is currently required which may be difficult in-vehicle. This obstacle can be removed as the Atmel FLIP program utility, used in the ISP operation, also supports automatic generation of the required RESET and $\overline{PSEN}$ signals through the use of the RS232 serial port DTR and RTS lines, respectively (called AutoISP). These were not implemented in the current PCB revision due to the fact that the interface with the Project54 IDB box would also involve these RS232 lines, which are usually reserved for RS232 hardware data flow control purposes. This is an area that can be examined further with a more detailed analysis of the IDB interface however with the

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

end result being that the headers may be able to be eliminated and the whole system made in-car programmable via the IDB network.

JTAG Interface Programming of the CPLD

TAP Interface and the JTAG Protocol

In order to increase system flexibility and to further facilitate changes that may arise in the prototype design, the CPLD selected for implementation of the digital PTT mapping logic was selected so that it was also in-system reprogrammable. The task of reprogramming is handled by the 8051 microcontroller with a simple multi-line interface. The interface type could be extremely flexible due to the significant options available through the programmability of the microcontroller.

The Xilinx XC9536 Complex Programmable Logic Device (CPLD) used for the digital PTT signal mapping is programmed via a standard 4-wire Test Access Port (TAP) that can be used for ISP and testing. The specific connection type used is referred to as a JTAG port and is also defined by the IEEE 1149.1 boundary scan standard. The option of in-system testing is currently not used as the performance requirements of the CPLD are quite limited. However, as this is a major feature of the standard it might be possible to implement this in a future revision as a means of testing board functionality and correct assembly [27].

The IEEE 1149.1 Boundary Scan (JTAG) interface protocol specifies a minimum four pin connection. These pins are TDI (Test Data In), TDO (Test Data Out), TCK (Test Clock), and TMS (Test Mode Select). Note that the in/out reference is from the perspective of the device to be programmed, therefore TDI, TCK, and TMS are all

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

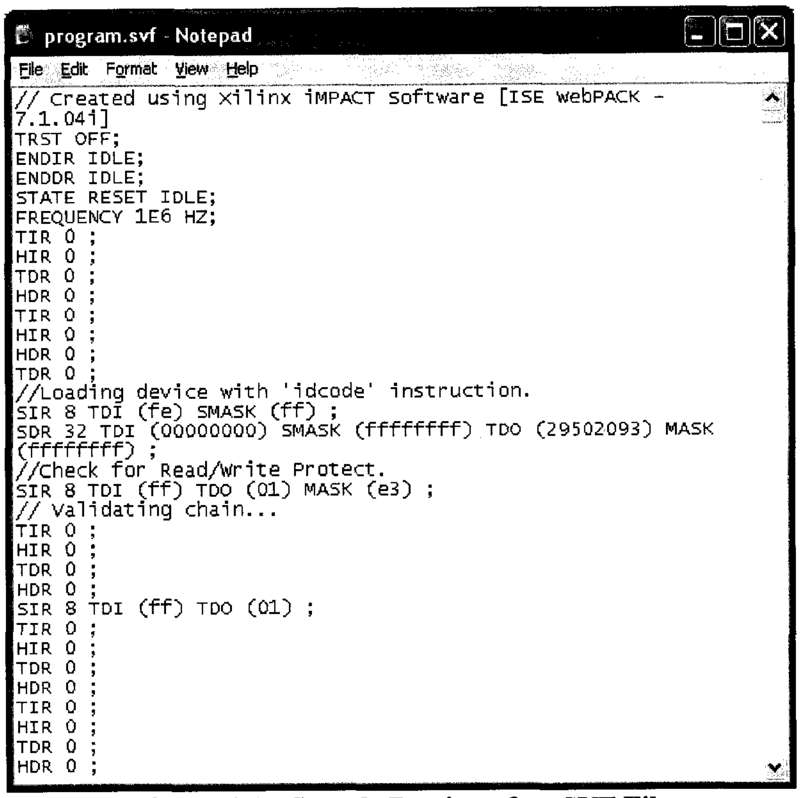
outputs of (driven by) the microcontroller while TDO must be configured as an input. The optional TRST (Test ReSeT) connection is not used as the selected Xilinx CPLD did not have that option.

XSVF versus SVF Programming Files

In general when using a JTAG port for the programming of a device it is standard to use Serial Vector Format (SVF) stimulus files that define the necessary settings and connections that are to be made internal to the device. However, since in this application the SVF programming file is being sent over a "slow" data transmission path of a serial port operating at only 9600 baud it was determined that the use of the XSVF file format would be more appropriate. The information contained in the two file formats are essentially the same however the means used for encoding those instructions for use by the JTAG programmer are completely different [28, 29]. Figures 6.5 and 6.6 show the starting section of some sample code used in this thesis work for programming the CPLD. Note that the two files are in the SVF and XSVF programming formats respectively.

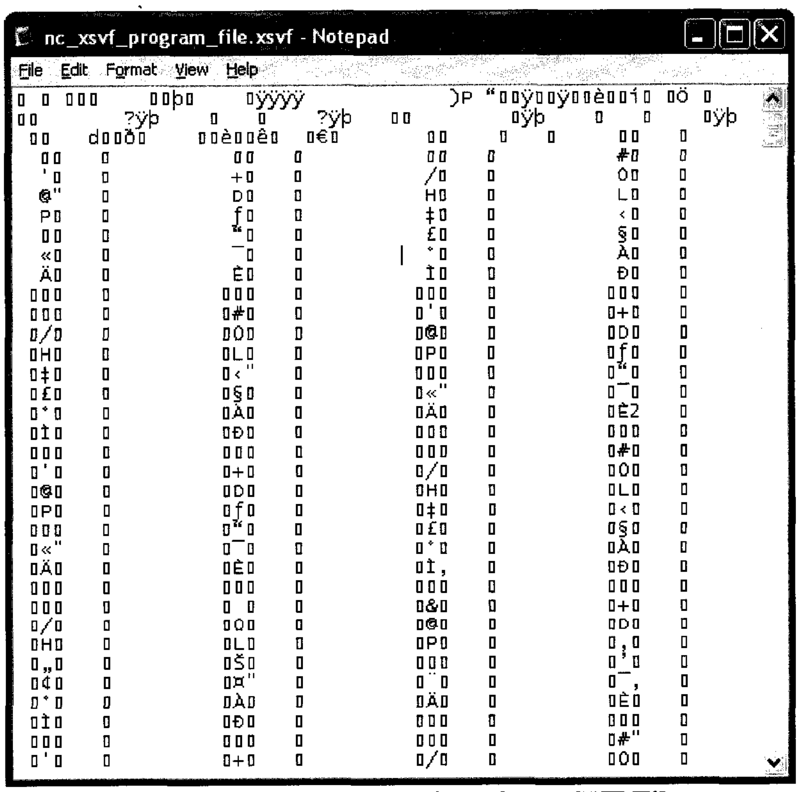
Note the first, and most obvious, distinction between the data format is that the SVF is in a readable ASCII text format. The XSVF however is a compact binary file that is completely not human-readable. This helps to maximize the data transmission rate. By way of comparison, the SVF file is 147,765 bytes long while the same file in XSVF is only 22,864 bytes long meaning it can be transmitted much more quickly over the PC to microcontroller connection.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.



```
// Created using Xilinx iMPACT Software [ISE WebPACK -
7.1.04i]
TRST OFF;
ENDIR IDLE;
ENDDR IDLE;
STATE RESET IDLE;
FREQUENCY 1E6 HZ;
TIR 0 ;
HIR 0 ;
TDR 0 ;
HDR 0 ;
TIR 0 ;
HIR 0 ;
HDR 0 ;
TDR 0 ;
//Loading device with 'idcode' instruction.
SIR 8 TDI (fe) SMASK (ff) ;
SDR 32 TDI (00000000) SMASK (ffffffff) TDO (29502093) MASK
(ffffffff) ;
//Check for Read/Write Protect.
SIR 8 TDI (ff) TDO (01) MASK (e3) ;
// Validating chain...
TIR 0 ;
HIR 0 ;
TDR 0 ;
HDR 0 ;
SIR 8 TDI (ff) TDO (01) ;
TIR 0 ;
HIR 0 ;
TDR 0 ;
HDR 0 ;
TIR 0 ;
HIR 0 ;
TDR 0 ;
HDR 0 ;
```

Figure 6.5 – Sample Portion of an SVF File

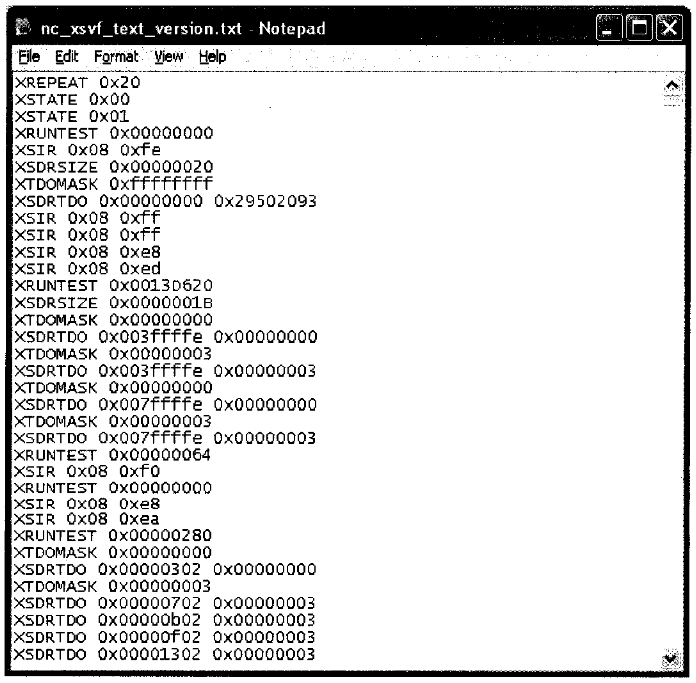


```
▯ ▯ ▯▯▯     ▯▯þ▯     ▯ÿÿÿÿ               )P "▯▯ÿ▯▯ÿ▯▯è▯▯í▯ ▯Ö ▯
▯▯        ?ÿþ     ▯     ▯     ?ÿþ    ▯▯        ▯ÿþ    ▯    ▯     ▯ÿþ
 ▯▯     d▯▯Ŏ▯     ▯▯è▯▯ê▯  ▯€▯          ▯▯     ▯    ▯     ▯▯    ▯
  ▯▯     ▯           ▯▯    ▯           ▯▯     ▯           #▯    ▯
  '▯     ▯           +▯    ▯           /▯     ▯           Ò▯    ▯
  @"     ▯           D▯    ▯           H▯     ▯           L▯    ▯
  P▯     ▯           ƒ▯    ▯           ‡▯     ▯           <▯    ▯
  ▯▯     ▯           "▯    ▯           £▯     ▯           §▯    ▯
  «▯     ▯           ¯▯    ▯         | °▯     ▯           À▯    ▯
  Ä▯     ▯           È▯    ▯           Ì▯     ▯           Ð▯    ▯
 ▯▯▯     ▯          ▯▯▯    ▯          ▯▯▯     ▯          ▯▯▯    ▯
 ▯▯▯     ▯          ▯#▯    ▯          ▯'▯     ▯          ▯+▯    ▯
 ▯/▯     ▯          ▯O▯    ▯          ▯@▯     ▯          ▯D▯    ▯
 ▯H▯     ▯          ▯L▯    ▯          ▯P▯     ▯          ▯ƒ▯    ▯
 ▯‡▯     ▯          ▯<"    ▯          ▯▯▯     ▯          ▯"▯    ▯
 ▯£▯     ▯          ▯§▯    ▯          ▯«"     ▯          ▯¯▯    ▯
 ▯°▯     ▯          ▯À▯    ▯          ▯Ä▯     ▯          ▯È2    ▯
 ▯Ì▯     ▯          ▯Ð▯    ▯          ▯▯▯     ▯          ▯▯▯    ▯
 ▯▯▯     ▯          ▯▯▯    ▯          ▯▯▯     ▯          ▯#▯    ▯
 ▯'▯     ▯          ▯+▯    ▯          ▯/▯     ▯          ▯O▯    ▯
 ▯@▯     ▯          ▯D▯    ▯          ▯H▯     ▯          ▯L▯    ▯
 ▯P▯     ▯          ▯ƒ▯    ▯          ▯‡▯     ▯          ▯<▯    ▯
 ▯▯▯     ▯          ▯"▯    ▯          ▯£▯     ▯          ▯§▯    ▯
 ▯«"     ▯          ▯¯▯    ▯          ▯°▯     ▯          ▯À▯    ▯
 ▯Ä▯     ▯          ▯È▯    ▯          ▯Ì,     ▯          ▯Ð▯    ▯
 ▯▯▯     ▯          ▯▯▯    ▯          ▯▯▯     ▯          ▯▯▯    ▯
 ▯▯▯     ▯          ▯ ▯    ▯          ▯&▯     ▯          ▯+▯    ▯
 ▯/▯     ▯          ▯Q▯    ▯          ▯@▯     ▯          ▯D▯    ▯
 ▯H▯     ▯          ▯L▯    ▯          ▯P▯     ▯          ▯,▯    ▯
 ▯„▯     ▯          ▯Š▯    ▯          ▯▯▯     ▯          ▯'▯    ▯
 ▯¢▯     ▯          ▯¤"    ▯          ▯¨▯     ▯          ▯¯,    ▯
 ▯°▯     ▯          ▯À▯    ▯          ▯Ä▯     ▯          ▯È▯    ▯
 ▯Ì▯     ▯          ▯Ð▯    ▯          ▯▯▯     ▯          ▯▯▯    ▯
 ▯▯▯     ▯          ▯▯▯    ▯          ▯▯▯     ▯          ▯#"    ▯
 ▯'▯     ▯          ▯+▯    ▯          ▯/▯     ▯          ▯O▯    ▯
```

Figure 6.6 – Sample Portion of an XSVF File

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The other less obvious difference is that the instructions contained in the SVF file are typically describing at a higher level the IEEE 1149.1 (JTAG) while those at the XSVF level typically leverage more direct control of the hardware. For example, reference Figure 6.7 which is the XSVF file placed in an ASCII readable format for presentation purposes (called here an ASCII Readable or text XSVF file). Note that commands such as XREPEAT and XSTATE are actually just single bytes in the original XSVF code but are represent here by their common command name (which can be constant value defined as equal to the byte in simple assembly level code). Also, each byte that follows a command (each of which are placed on a new, separate line) is translated from binary to its hexadecimal, ASCII equivalent. For example the first line XREPEAT 0x20 is really only two bytes long in the XSVF file; one byte for XREPEAT and one for the 0x20 (or 20 in hexadecimal). (Note that the ASCII Readable XSVF file is now 78,924 bytes in size, still less than the SVF format.)


nc_xsvf_text_version.txt - Notepad
File Edit Format View Help

```
XREPEAT 0x20
XSTATE 0x00
XSTATE 0x01
XRUNTEST 0x00000000
XSIR 0x08 0xfe
XSDRSIZE 0x00000020
XTDOMASK 0xffffffff
XSDRTDO 0x00000000 0x29502093
XSIR 0x08 0xff
XSIR 0x08 0xff
XSIR 0x08 0xe8
XSIR 0x08 0xed
XRUNTEST 0x0013D620
XSDRSIZE 0x0000001B
XTDOMASK 0x00000000
XSDRTDO 0x003ffffe 0x00000000
XTDOMASK 0x00000003
XSDRTDO 0x003ffffe 0x00000003
XTDOMASK 0x00000000
XSDRTDO 0x007ffffe 0x00000000
XTDOMASK 0x00000003
XSDRTDO 0x007ffffe 0x00000003
XRUNTEST 0x00000064
XSIR 0x08 0xf0
XRUNTEST 0x00000000
XSIR 0x08 0xe8
XSIR 0x08 0xea
XRUNTEST 0x00000280
XTDOMASK 0x00000000
XSDRTDO 0x00000302 0x00000000
XTDOMASK 0x00000003
XSDRTDO 0x00000702 0x00000003
XSDRTDO 0x00000b02 0x00000003
XSDRTDO 0x00000f02 0x00000003
XSDRTDO 0x00001302 0x00000003
```

Figure 6.7 – Sample Portion of an ASCII Readable XSVF File

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

While the use of a lower level instruction set for the XSVF file might be expected to result in some additional size (without byte compression) this approach is done for two reasons. First, in general the size still is less due to the fact that the SVF file is rather verbose in its description of operations, in an attempt to make it more readable. Also, some of the instructions are not necessary but can be redundant, a characteristic that is stripped away in XSVF. The second advantage to using the lower level language contained within XSVF is that it will inherently require less parsing by the programming microcontroller and will result in greatly simplified assembly or embedded microcode. By doing this, most of the work is done in the original compilation at the PC level (a task that must only be done once) versus at the much slower and more involved PCB level (a task that must be done for each device).

Transition between the SVF and XSVF files is done using the Xilinx SVF-to-XSVF Writer utility which is discussed further in Appendix D.

Microcontroller Based JTAG Programming

The 80C51 microcontroller is used to program the CPLD with the XSVF code file received over the serial port. This is done by receiving the XSVF data and interpreting the information to then generate the programming instructions, data, and control signals for the CPLD. The physical, hardware connection necessary at the CPLD end are typical to a TAP interface and are shown in Figure 6.8. Also depicted is the necessary serial port connection to a PC (note that the TTL<->RS232 conversion hardware needed for the serial port is not shown). In both interfaces the data flow direction for each line, as discussed in the TAP overview, is shown.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

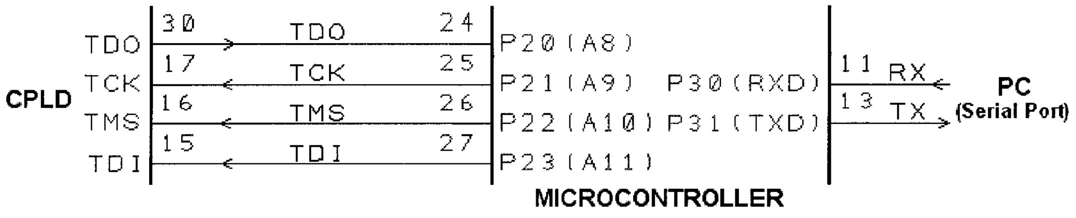


Figure 6.8 – Required Hardware Interfaces

Programming of the CPLD is instigated from the main MRCM firmware menu by transmission of the 7 character command string "*RPCPLD" (for ReProgram CPLD; the '*' character helps to insure that the command is not inadvertently broadcast). At this time the microcontroller enters the XSVF Program Loop, shown in Figure 6.9, that is contained as a sub-program within its firmware.

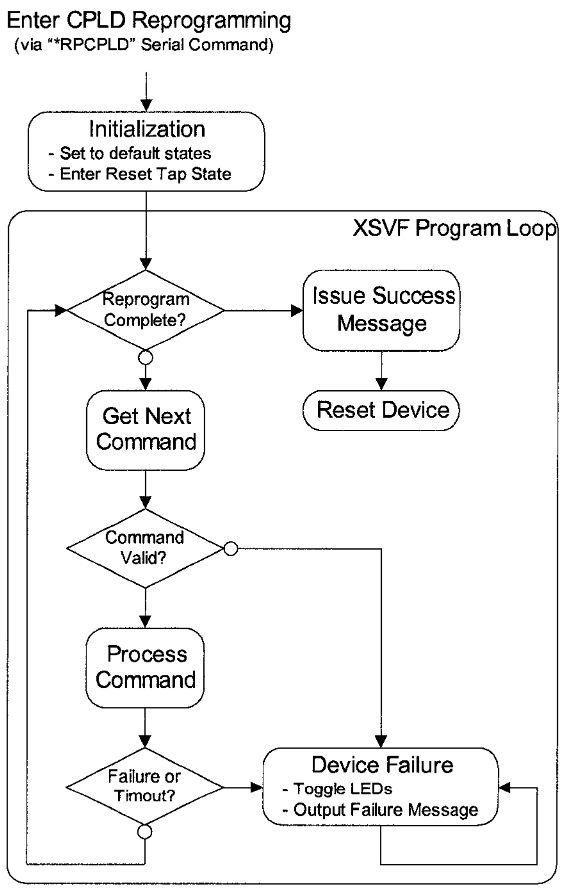


Figure 6.9 – Block Diagram of Microcontroller JTAG Interface Firmware

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Note that there are only two ways this loop can be exited. The first is with a successful reprogramming of the CPLD after which the device will automatically be reset via a standard software soft reset. The other case is where the reprogram fails. This can be due to a time-out while waiting for the next command byte from the connected PC, due to the reception of an invalid command byte, or due to an actual hardware failure of the CPLD (which would incorrectly respond to a command check by the microcontroller). In any event, when a failure occurs the system enters a cursive loop that involves toggling the main system status LED and outputting a failure message via the serial port. The only method for exiting the loop is with a hardware reset (which can be accomplished by powering the device OFF and then back ON).

The XSVF instruction bytes are sent over the serial port one byte at a time. The microcontroller will query the PC for the next byte by transmitting a '\$' when it is ready. This prevents the microcontroller's internal serial port buffer from becoming overloaded with incoming bytes, particularly at portions in the code execution, such as chip erase, were a large delay in processing is required.

The main XSVF program loop is based upon a Xilinx sample C-code program, designed for the 8051. The program is available from Xilinx as part of a web package for JTAG programming of Xilinx devices [30]. Written in C, while the rest of the 8051 firmware is written in straight assembly, it would have still been possible to use the code in the design. However, some obvious flaws were observed in the code and it would, in fact, not even compile. Thus, the Xilinx code was not able to be used directly, although it was used as a procedural reference/guideline for the assembly code implementation used.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The XSVF file type was originally created by Xilinx to afford better data compression and as such is ensured to meet all the needs associated with their series of FPGAs, CPLDs, and configuration PROMs. However, some command and/or data types were not necessary for this CPLD only application. Therefore some instructions have been excluded and others have only limited functionality as it would apply to CPLDs. The list of XSVF commands is shown in Table 6.3 along with their equivalent binary code and whether they are supported or not.

The XSETSDRMASKS and XSDRINC commands shown in the table both deal with XSVF compression which can be further used to reduce the XSVF file size. However, they are not supported by the programming algorithm developed in this thesis and therefore neither is the Compressed XSVF file type. Note that, in the command line used to generate the XSVF file that is used in programming, this is what the –nc stands for, no compression during translation (reference Appendix D for additional information on the generation of the XSVF file). For the most part it was observed that with the device used compression had little to no effect and therefore was not worth the additional firmware overhead to support it.

The XSIR2 command is only partially supported in that if it can be reduced to a simple XSIR command, then that operation will be done. If however, it is not, but is a command that contains more than the maximum length of four bytes (which is the real application of this command), then a failure will result and the device will enter the failure mode. Again, this command is more common for FPGAs and was not observed as ever occurring in the XSVF files generated from the VHDL code used in the Xilinx CPLD.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

XSVF Instruction	Binary Encoding (Hex)	Support
XCOMPLETE	0x00	Supported
XTDOMASK	0x01	Supported
XSIR	0x02	Supported
XSDR	0x03	Supported
XRUNTEST	0x04	Supported
XREPEAT	0x07	Supported
XSDRSIZE	0x08	Supported
XSDRTDO	0x09	Supported
XSETSDRMASKS	0x0a	Not Supported
XSDRINC	0x0b	Not Supported
XSDRB	0x0c	Supported
XSDRC	0x0d	Supported
XSDRE	0x0e	Supported
XSDRTDOB	0x0f	Supported
XSDRTDOC	0x10	Supported
XSDRTDOE	0x11	Supported
XSTATE	0x12	Supported
XENDIR	0x13	Supported
XENDDR	0x14	Supported
XSIR2	0x15	Partial Support
XCOMMENT	0x16	Supported
XWAIT	0x17	Supported

Table 6.3 – XSVF Command Set and Design Support

A much more detailed breakdown of the basic algorithms followed for the JTAG programming, as well as a description of the XSVF commands, is contained in [28].

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Other sources also exist that provide more detailed examinations of the more general SVF file type [31, 32]. Additional details on the MCS51 assembly firmware sub-program used by the microcontroller for reprogramming the CPLD via the JTAG are provided in Appendix C. Also, Appendix D contains the procedure used to reprogram the CPLD via a serial interface between the microcontroller and a PC running MATLab.

Control Signal Buffering

The control signals for the switching matrix are generated by the microcontroller. However, the fan-out of these control lines is fairly large, each going to two switch matrix input and the CPLD as well as being used to drive an LED for the user interface. The LEDs are set to operate with a draw current of around 10 mA using a 249 Ω feed resistor. The CPU port pins, when in high mode (the case when the LEDs and switch enables are active), are typically only able to source 10-50 μA, therefore some method of buffering is required. The solution for the eight main Channel-Meeting control lines (CTRL1A, CTRL1B, CTRL2A, ..., CTRL4B) is to use a 74HCT574 Latch composed of D-type Flip-Flops. This can then be written to as any external data memory location using the circuit configuration shown in Figure 6.10. The figure also shows the LEDs used to inform the user of connection status.

Note that the $\overline{WR}$ (write) signal is supplied by the CPU during write operations and is an active low pulse. The $\overline{ME}$ (output chip enable) signal serves the critical purpose of insuring that, particularly at startup, the latch does not hold an invalid connection byte, as the latch will only output values so long as this signal is low. On reset, the CPU defaults with this pin high, allowing the port to be written before it is

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

active. Note that when the latch is in the inactive state the LEDs will pull the control lines low so that the switch enable inputs do not inadvertently float high.

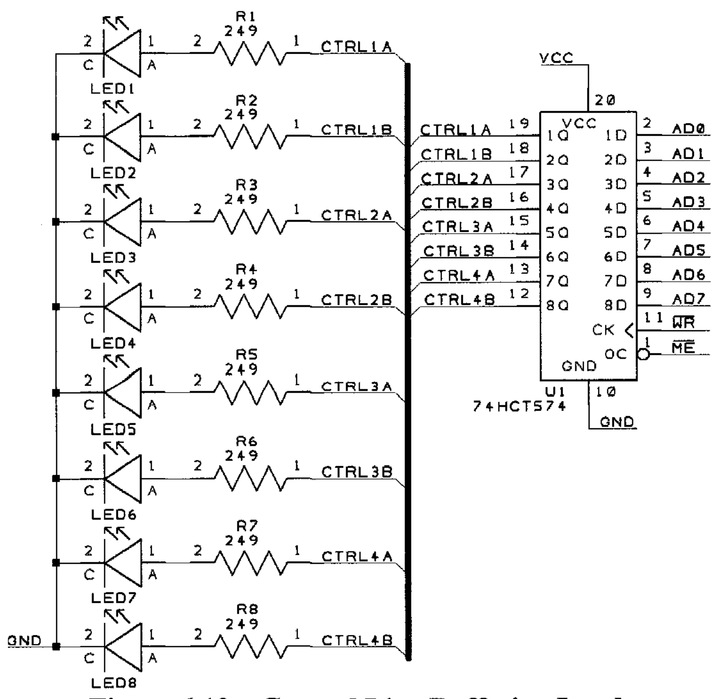


Figure 6.10 – Control Line Buffering Latch

The active high CTRLMA and CTRLMB Master Microphone switch control lines also needed some form of buffering in order to drive their LEDs. Since there are only two of them, requiring less CPU pins, they were given direct connection to the microcontroller instead of expending resources on a latch buffer. Therefore, the LEDs are driven by simple NPN transistor switches that will source the 10 mA currents needed. The circuit diagram is shown in Figure 6.11.

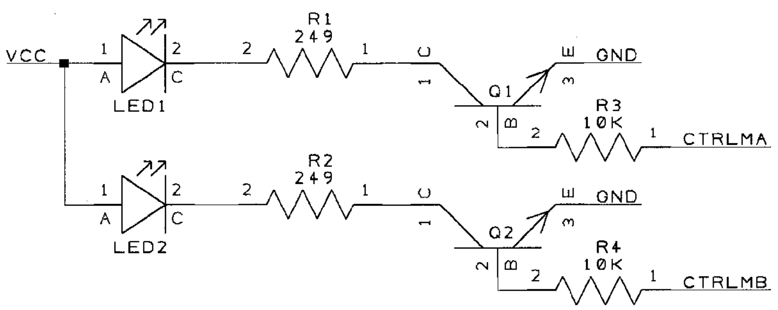


Figure 6.11 – Master Microphone Control Line Buffering Transistors

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Note that CTRLMA and CTRLMB are supplied directly by the microcontroller and therefore will default to high (active) on startup. However, these control lines only affect the audio switch input ends so having them both high does not result in any conflict or erroneous connections being made upon reset of the CPU. Also, the first group of initialization routines turns these OFF so that by the time the channel connections are established the Master Microphone is no longer necessarily connected to either one or both of the meetings.

Based on the feed resistor, the transistor draw on the microcontroller lines will be an absolute maximum of 0.5 mA. However, with only 10 mA being used by the LED, and assuming, based on the datasheet, an h_{FE} for the transistor of around 200, this will most likely be closer to only 50 μA, which should produce acceptable operation. This was tested in the circuit and proved to be correct. (The specific NPN transistors used are the MMBT4401.)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER VII

MIXED-SIGNAL PCB DESIGN CONSIDERATIONS

Prototype PCB Layout

It was early on decided that the prototype design would be implemented on a PCB. The main reason for this is that bread boarding or using a perforated circuit board would be too difficult for the level of complexity involved. Also, there were too many issues relating to reliability that wouldn't be met with loose wire connections. In order to simplify the testing and debugging however the use of a two layer PCB was determined. With a two layer PCB, top and bottom copper plates sandwich a single dielectric layer. The copper is then etched with the appropriate circuit patterns. This allows for all of the contacts and etches to be fully accessible to probing which makes testing easier. Also, in the event that temporary jumpers are needed, having all metal layers external can be helpful in locating solder points

The use of a 2-layer PCB means that typical power planes (ground and V_{CC}) cannot be used. Instead options include ground and power fingers or enclosed ground and power loops. The ground and power fingers are designed by having etches for each extend from opposite ends of the board. These extensions however do not typically merge back together on the other side. The result is that noise can sometimes be present on one or more fingers and cross-talk can result.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The power and ground loops help to solve this problem by connecting the fingers at the other edge of the circuit board. This is the approach taken here as the noise on the power lines, a major concern in this project, is typically much less. The key task in laying out the ground (and power) loops is to make them, when taken as a whole, act as close as possible to a true ground plane. This must also be done while still allowing the signal etches enough room to travel on the back side of the circuit board when necessary. Figure 7.1 shows the ground loops while Figures 7.2 and 7.3 show the V_{CC} and V_{SS} power loops respectively (viewed transparently from the top). Note that the V_{SS} -5 V power supply rails only extend into the analog section as they are not used by the digital.

In order to help facilitate the routing process (which was done manually) an organized approach to etch layout can be quite beneficial. The approach taken was to attempt as best as possible to adhere to the following rule: all horizontally traveling etches should be placed on the bottom layer while vertical etches should be placed in the top layer of copper. Obviously with some components, i.e. surface mounts, this is not always possible but the general rule can help insure that the majority of locations on the PCB can be reached without crossing over another etch on the same layer. (By way of illustrating this layout rule, reference Figures E.3 and E.4 in Appendix E which show the top and bottom coppers respectively for the Revision C circuit board. Note that the majority of etches in the top copper are vertical while those in the bottom copper are mostly horizontal.)

Note that the ground and power loops are kept in the bottom layer. As such they are horizontal. However, the ends of the ground plane remain in the bottom copper even though they are vertical. This is permitted since they should never be crossed but act as a

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

shield around the edge of the PCB. Note Figures 7.1, 7.2, and 7.3 for the Ground, V_{CC}, and V_{SS} loops respectively show this. These images are viewed from the top side.

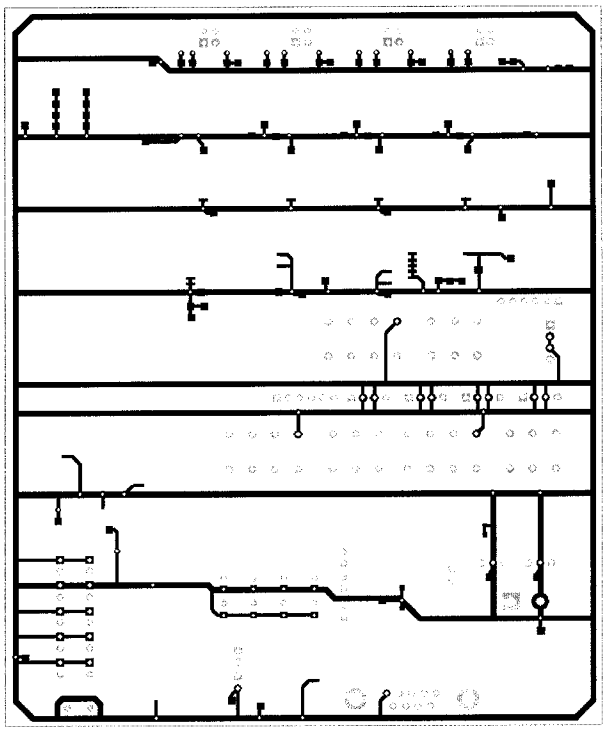


Figure 7.1 – Revision C PCB Ground Loops

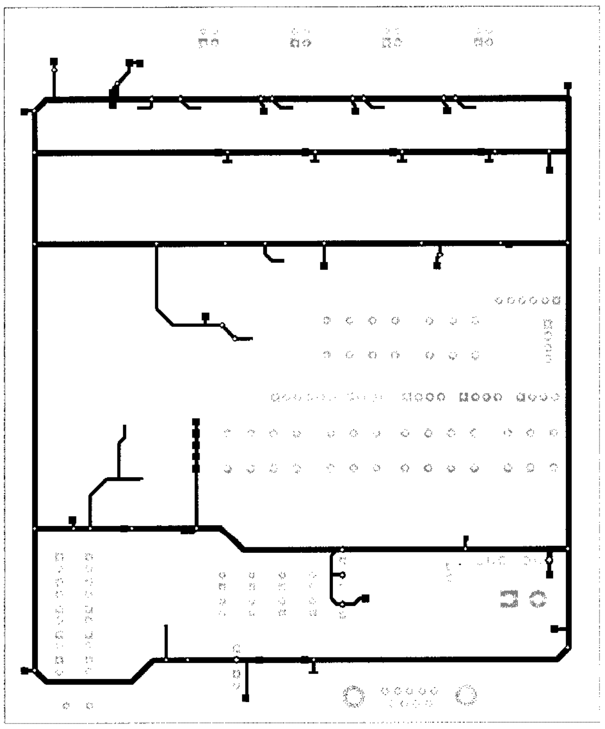


Figure 7.2 – Revision C PCB V_{CC} +5V Power Loops

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

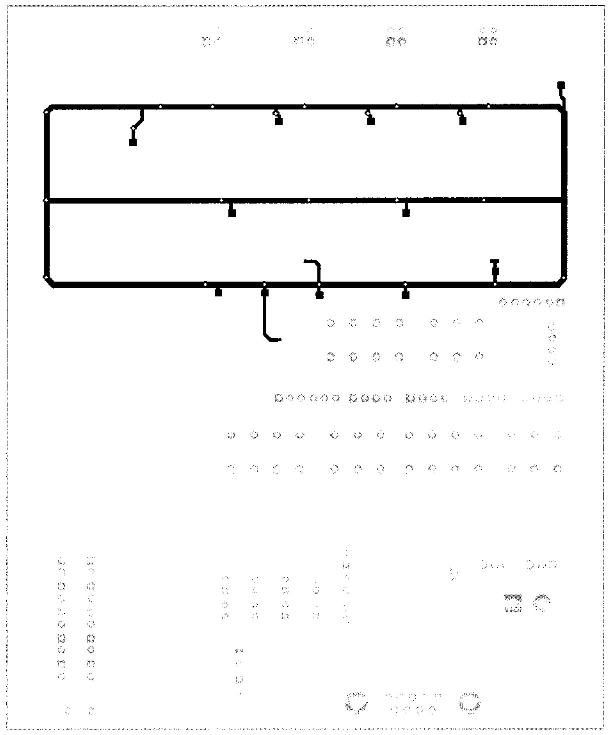


Figure 7.3 – Revision C PCB V_{SS} -5V Power Loops

The last issue for discussion with the layout of the PCB prototype design is the major package technology that will be used. The decision was to go with SMT (Surface Mount Technology) over THD (Through Hole Device). The main reason for this is that the SMT components are typically smaller and will consume less space. Although not really an issue for the prototype it is still a concern for later, possible release editions that will be based on the prototype, since for a mixed-signal PCB layout is very important and changing the technology might have undesirable performance effects.

Another reason for selecting the SMT is that it keeps the components on the top side of the board and reduces the number of drill holes required (which can translate into extra cost during circuit board fabrication). From a manufacturing perspective, having top-mounted components means that they can be placed with an automated pick-and-place machine, as opposed to THDs which often must be hand placed and then wave

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

soldered. Note that since the prototype model was being hand soldered the specific SMT size technology employed was a typical 50 mil (milli-inch) spacing on the component pads. Also, etches maintained a minimum spacing of 13 mils with typical signals running at a trace width of 12 mils. The Power and Ground traces are 50 mils wide for the major loops and 25 mils wide for device connections. These increased widths are to help with the handling of current and to minimize noise.

Note that the actual Revision C PCB design is shown in greater detail, including locations of major components and connectors, in the Performance Summary & Specifications section of *Chapter IX - Prototype Design Analysis*. *Appendix E - Revision C PCB Design Information* contains more detailed specifications on the PCB design including full schematics, top and bottom copper plots, complete part lists and component layouts, and the CAD files.

Revision A Power Supply Noise and Heat Dissipation

Early in the design testing some form of feedback or noise runaway was observed on the Revision A prototype resulting in excessive noise in the audio outputs. The noise also was not present at start-up but rather took a period of time, on the order of minutes, to accrue to a threshold level after which it quickly saturated drowning out any existing test audio on the channels and corrupting the signals.

In testing to determine the source of this noise it was observed that the same runaway noise also became present on the power supply lines (V_{CC}, V_{SS}, and Ground). (Note that these had initially been tested for basic voltage levels with a DC voltage meter and found to be operating correctly at +5 V, -5 V, and 0 V DC. However, for the noise

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

testing, observation was conducted using an oscilloscope.) At this point there were two possibilities. Either some component on the board was creating the noise, and a subsequent current draw that was pulling the supply voltages, or else there was a problem with the power supply circuit. Whatever the source of the noise it would need to be reduced or ideally eliminated altogether in order for the interoperability device to meet the required performance levels.

To test this the input voltage (13.8 V DC) to the power supply circuit was examined and found to also have the noise at the circuit board side input connector. However, the noise was not present directly at the output of the 13.8 V DC power source. Also, the output of the power source never exceeded 400 mA. Note that the power supply input voltage is delivered to the circuit board connector via to 20 AWG wires that are approximately six inches in length. Considering that the resulting noise voltage drop across the wire was quite large, on the order of volts, this would imply a significant amount of current. From this it was deduced that the circuit board's power supply must be failing in some way.

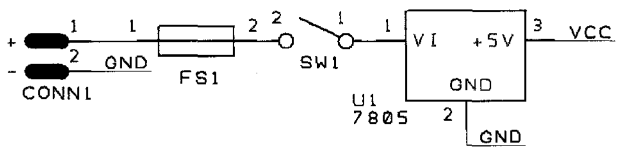


Figure 7.4 – Original Revision A Power Supply Circuit

The original Revision A Power Supply circuit for the +5 voltage rail is shown in Figure 7.4. Note that it used only a single regulator to drop the voltage from 13.8 V to 5.0 V DC. This however was determined to be acceptable based on the specifications given for the Micro Commercial Co. MC7805CT 5 volt regulator selected for the design. Its maximum input voltage is specified as 30 V and the maximum current is given as >1.0

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

A (with a peak of 2.2 A). More importantly the power dissipation was given as 15 W [33]. Based on the 13.8 voltage input with a 5 volt output at 370 mA the power that would need to be dissipated in the regulator would be:

$$
\begin{aligned}
P_D &= I_{\max}\left(V_{in} - V_{out}\right) \\
&= 0.37A \times \left(13.8V - 5V\right) \\
&= 3.256\,W
\end{aligned}
$$

This is well below the maximum of 15 W and therefore the device should be able to handle it. It was noted however that some other regulators used P_D curves that varied with the ambient temperature and based on the heat sink used. For example, at room temperature with no heat sink a similar device by ON Semiconductor (that also had a maximum P_D of 15 W) reported a P_D of 2 W. However, with a moderate heat sink this was changed to over 6 W and therefore the design was still considered acceptable as a heat sink was able to be employed in the application [34].

However, this power dissipation appears to be the problem with the Revision A power supply design as running at a lower system input voltage (i.e. 10 V instead of 13.8 V) the problem was no longer observed. The heat sink and regulator were also both observed to become quite warm at the higher car battery voltage level. Therefore, it has been deduced that the internal thermal overload protection circuit was activating and turning the regulator off, momentarily, resulting in severe spikes on the output voltage and the connected ground. This also appears to have impacted the input side voltage as well. Therefore, it must be assumed that either the heat sink is insufficient or the specifications for the Micro Commercial component relating to a P_D curve (which was not provided in the datasheet) are not as good as the ON Semiconductor device. In either

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

case it was decided that the power supply was operating too close to its limitations for the in-car application environment (where temperature could easily reach well above room temperature). Thus, a second voltage regulator was added in the Revision B design that would drop the input source voltage from 13.8 V down to 9 V, thereby splitting the dissipated power between the two. Note that the 5 V regulator must now dissipate 1.48 W while the 9 V regulator would need to handle 1.776 W. Based on the ON Semiconductor worst case power dissipation curves this would allow for in specification operation, without a heat sink, all the way up to around 122° F (50° C). The worse case P_D curve supplied by On Semiconductor is shown in Figure 7.5.

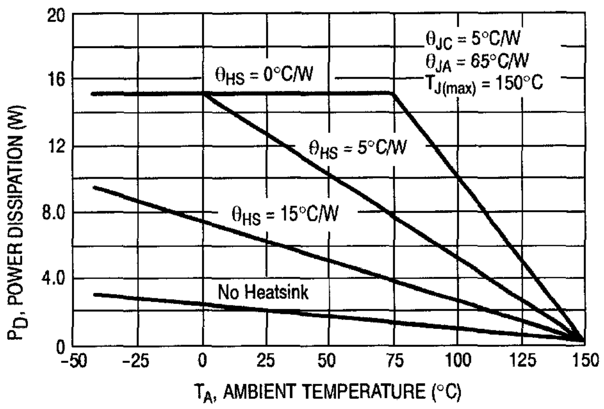


Figure 7.5 – ON Semiconductor Worse Case Power Dissipation Curves

Testing of the Revision B and Revision C boards show that this problem has been eliminated. It is recommended however that a double heat sink still be used for the two regulators to give better reliability to the system. For this purpose the regulators have been laid out with the correct spacing from one another in Revision C.

Revision B Digital Interference and Crosstalk with the Analog Audio

The original Revision A PCB was laid out with reasonable separation of the analog and digital portions of the circuit board. The digital output portion, dominated by

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

the CPLD, and the analog output portion, composed of OPAMP buffer circuits, were close together but the etches came into the four output screw terminals from opposite sides as shown in Figure 7.6. (Note that the four output connectors are in the region between the Analog Output, the CPLD, and the Main Digital Section.) Also, the PTT digital etches from the CPLD were routed on the top side of the circuit board as they were vertical in orientation while the analog audio signals were in the bottom copper being horizontal in nature. The result is that there was no noticeable crosstalk between the analog and digital output signals as reasonable isolation was maintained and coupling was minimized.

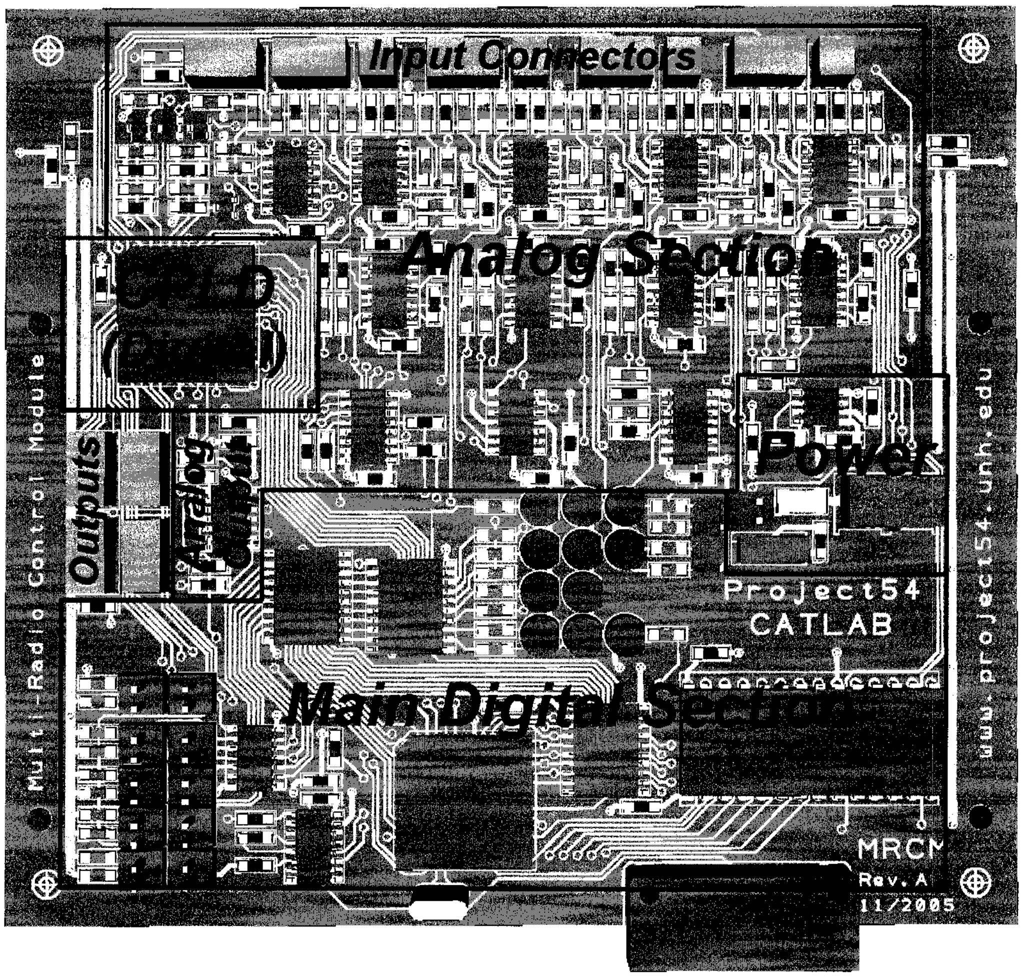


Figure 7.6 – Revision A PCB Layout Image

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

However, in Revision B the automatic circuit isolation relays were added in order to prevent queuing the radios during a crash. This required, not only the addition of components, but that those relay components also be able to access both the input and output ends of the circuit board. This then meant that the output and input connectors had to be located reasonable close to one another and the new relays. To minimize effort in the redesign the attempt was made to keep as much of the Revision A design intact and simply route connects to the new locations. (Note that with the transition to the new microcontroller a significant portion of the main digital circuit section was able to be removed.) The resulting Revision B circuit board is shown in Figure 7.7.

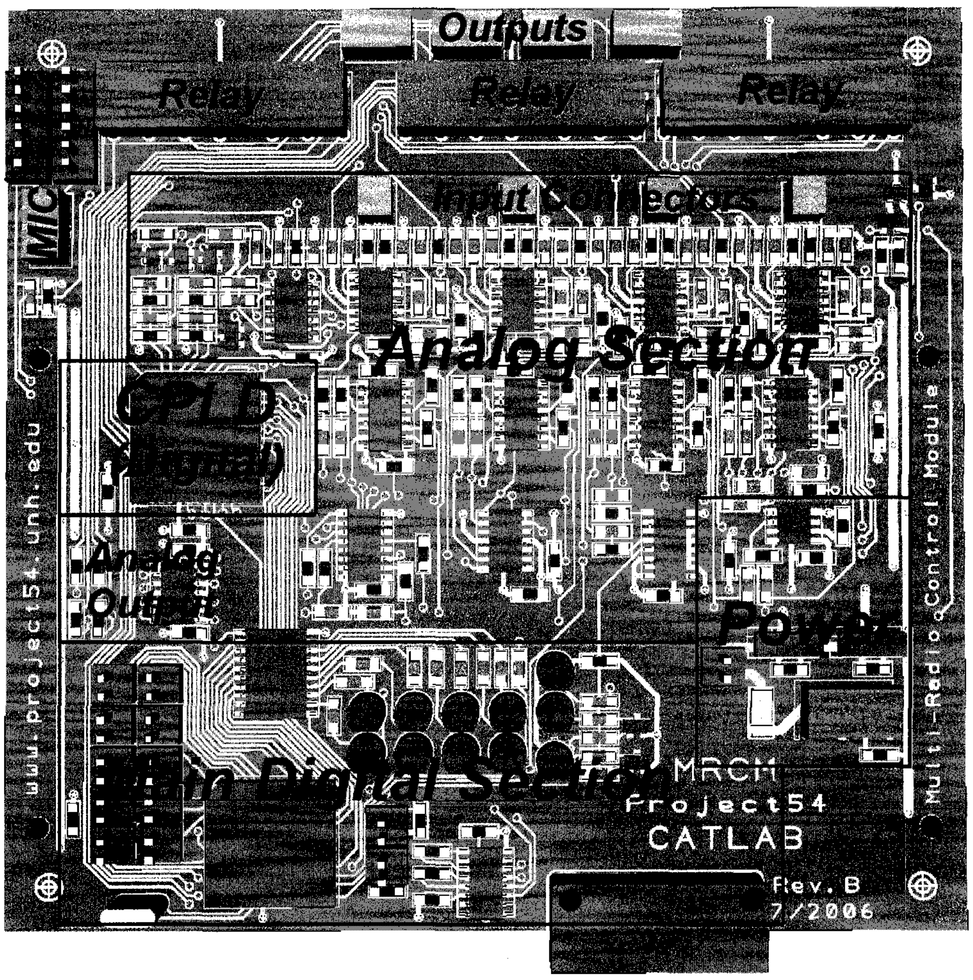


Figure 7.7 – Revision B PCB Layout Image

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

However, in testing of the Revision B circuit board, significant noise was found to be present on the audio outputs whenever the PTT signals were in transition. The effect was found across all four of the audio channels as well (not just the one corresponding to the transitioning PTT signal. Using an oscilloscope it was determined that the PTT introduction onto the audio lines was being caused by a small amount of ground bounce on the last audio OPAMP and a significant amount of cross-talk between the PTT and audio output etches. Figure 7.8 shows how closely the PTT and audio output etches are routed next to one another. Note that the four audio output lines are routed in parallel with and right next to the high-speed digital PTT etches for a distance of about 1½", on average. The result is a strong coupling that needed to be eliminated.

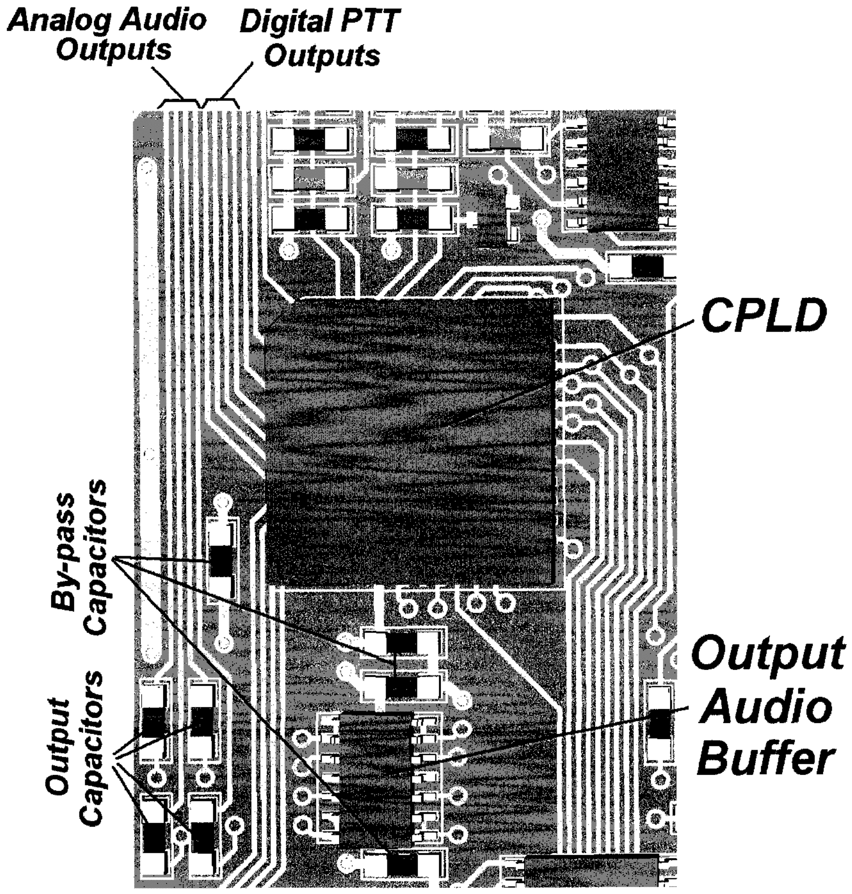


Figure 7.8 – Zoomed in View of the Revision B PCB Layout Image

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The first method attempted to reduce the digital interference on the analog signals was to modify the by-pass capacitors. Specifically, it was examined whether increasing them would help to reduce ground and power supply problems on the analog OPAMP. However, this attempt was unsuccessful, largely because the main problem was being caused by the close proximity of the digital and analog etches. There was no noticeable effect and therefore another approach needed to be taken.

Another possible means of reducing the cross-talk is to change the loading on the signal lines [35, 36]. This included reducing the output capacitor values and even removing them altogether so that the analog line was more strongly driven [37]. Also, since the output capacitors are still needed for AC isolation the effect of locating them closer to the output connectors was tried. The use of resistive loading was tested by placing 10 KΩ resistors directly at the outputs in an attempt to bleed away the cross-talk noise. In each test scenario the PTT switching noise was still observed on the audio output lines. Thus, it was determined that weighting the lines differently, including directly connecting them to the OPAMP, has little to no impact as the AC coupling is too strong.

The final modification that was attempted, short of implementing a new circuit board revision, was to examine the impact that reducing the transition switching speed of the Xilinx CPLD would have. By allowing for a longer PTT rise time, which would have little impact on system performance, the frequencies generated by the digital PTT signal would be lower. Then, in theory, the lower frequencies should not propagate quite as significantly over to the neighboring audio etches [38]. Also, with slower slew rates in the CPLD, the switch effects on the power supply lines should also be reduced resulting

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

in less noise. The Xilinx devices have the ability for certain connections to be defined with lower slew rates for this purpose [39, 40]. Note that this is a setting handled within the Xilinx ISE software used for writing the VHDL code for compilation to SVF format. This was implemented in the CPLD in a test version of code along with the grounding of all the unused pins (via jump wires and some VHDL code modification). However, the effect on the audio noise problem appeared negligible.

Based on the tests conducted on the Revision B PCB it was determined that a new circuit board revision, Revision C, would be needed in order to eliminate the PTT noise on the audio output signals. Prior to doing the revision, research was however conducted into standard techniques for reducing interference noise on mixed-signal PCBs.

A common technique that is used in very noisy environments is differential signaling. The main idea is that noise that is coupled onto the (+) signal will also be coupled on the (-) signal in an approximately equal amount. Then, the difference between the two will still remain the same and the noise will be inconsequential. However, the additional circuitry overhead would be extensive, as would the need for additional circuit board space for routing of the etches. Concerns also arise in that when in close proximity to the digital signals the cancellation would not be perfect and the noise not completely eliminated, thus other crosstalk reduction techniques would still need to be employed [41]. For these reason the differential signaling option was not implemented.

Another approach that can help reduce crosstalk between neighboring signals is running them orthogonally to one another. Two different signal types should rarely be run in parallel with each other and then only for very short distances. By doing this it

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

reduces the coupling between the lines, especially as compared with the Revision B circuits where they originally ran in parallel over a long stretch [38, 42]. Also note that based on the PCB layout technique employed this means that they will also be on different board layers altogether, further reducing the interference and simplifying the operation. This was pursued in the redesign wherever practical.

The simplest, and most common, approach to the problem is to simply increase the isolation between digital and analog sections of the circuit board wherever possible. By widening the distance between the aggressor and victim the crosstalk falls off sharply. There is also significant discussion of what can be done to the power and ground planes (or etches) in terms of the isolation.

Some current research recommends splitting both the ground and power planes into two separate sections - analog and digital. Then, the analog and digital circuits can be kept to their respective sections. In the event of dual type circuitry, such as Analog to Digital Converters (ADC), they must be placed as bridges across the gap. Of additional importance is that the planes themselves from the divergent sections do not overlap on different layers [43, 44]. In this way the circuits are highly segregated with only ADC control lines and ground/power bridges at the power supply.

While the complexities involved in developing split ground and power planes are typically not very difficult, when only using loops there is a little more concern that the ground paths in particular will be insufficient to bear certain loads. Also, for a truly mixed-signal design there is the concern that signals, such as the switch matrix control lines, that cross over from one section to another, will not have a reasonable mirror

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

current return path. Some additional research on the design of mixed-signal PCBs discusses a simple way that this can be avoided.

In regards to this, many researches have actually said that the ground plane should not be split, particularly in more complicated systems, as routing over the separation of the planes worsens the problems instead of helping to reduce them. As an alternative, the recommendation is to use a single unified ground plane while still maintaining physical separation of the components into analog and digital regions. The reason why a common ground plane doesn't hurt performance but can actually help it is that the ground plane currents will naturally try to mirror the currents in the signal plane(s). Thus, so long as the analog and digital traces are kept to their respective regions the ground currents should as well (thus there is no negative impact). Also, by having a unified plane, etches that intentionally transition between sections, for example the switch matrix control lines, have a minimal length current return path in the continuous ground plane [45]. The main thing that must be strictly insured if taking this approach is that different signal types are kept to their respective regions. This is the approach taken for the design, although without a ground plane. Instead, the loops for each section are freely connected together.

Figure 7.9 shows the new PCB implementation of these design changes in Revision C. Note that the CPLD has been moved down into the digital section to remove it completely from the analog region. Also, a large gap has been placed between the two sections where the output connectors are now located. Thus, as in Revision A, the analog audio and digital PTT signals are able to come from completely different areas and therefore do not run parallel to one another for significant distance and generate cross-talk.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

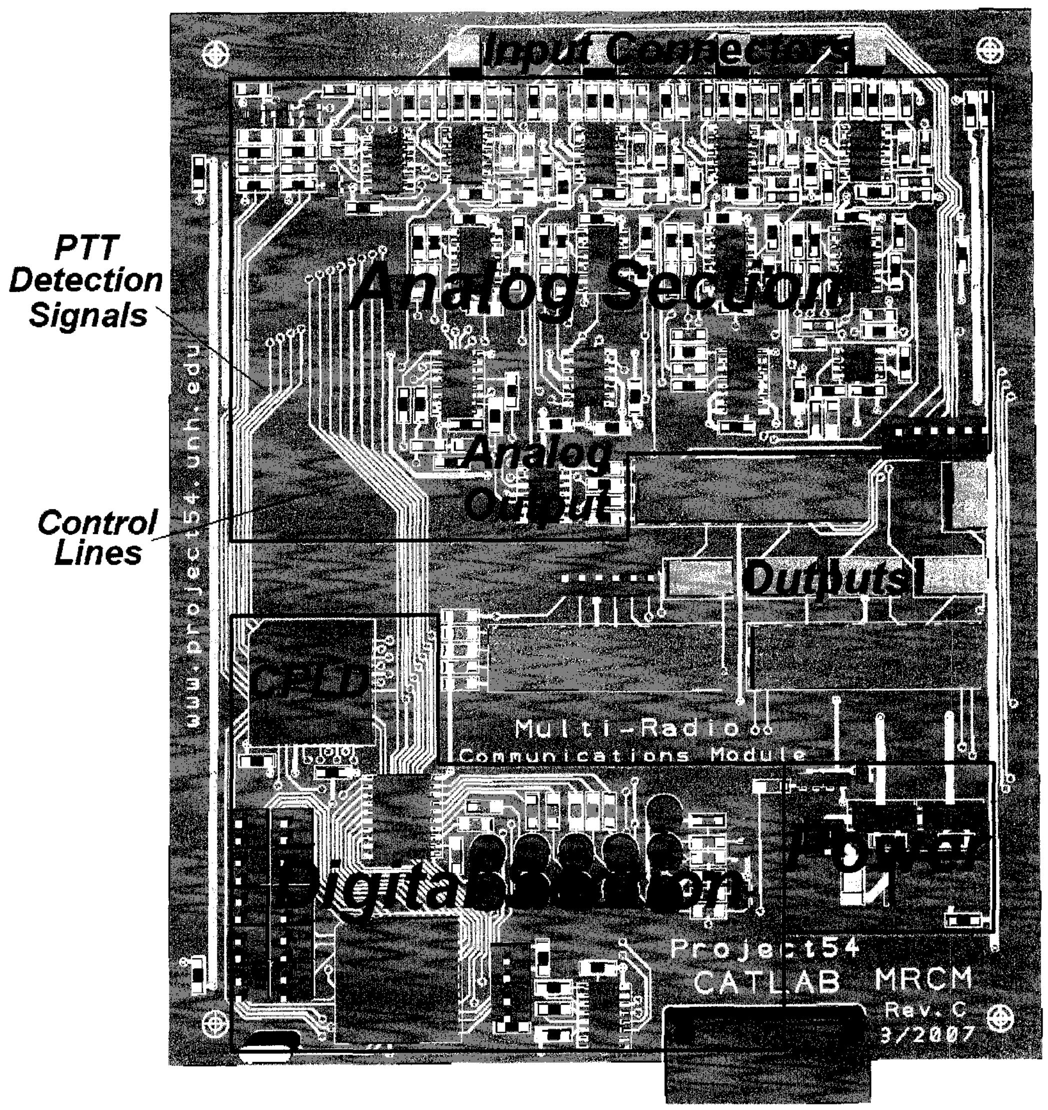


Figure 7.9 – Revision C PCB Layout Image

Also, in Figure 7.10 the back of the circuit board can be seen. Note that in the center division area there are now two strong ground lines with the top line having all analog connections above it and the bottom line having all digital ground connections below it. In this way the ground etches will act as if there is a unified plane that currents can flow through without crossing over into the other region's ground loops. Note that any mirror currents in the digital section will most likely not cross over into the analog section's ground but rather go through the lower-center ground etch and visa versa with

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

the analog mirror currents and the upper-center ground. The only things appearing between these two etches are the output radio connectors and the main microphone PTT input switch connector.

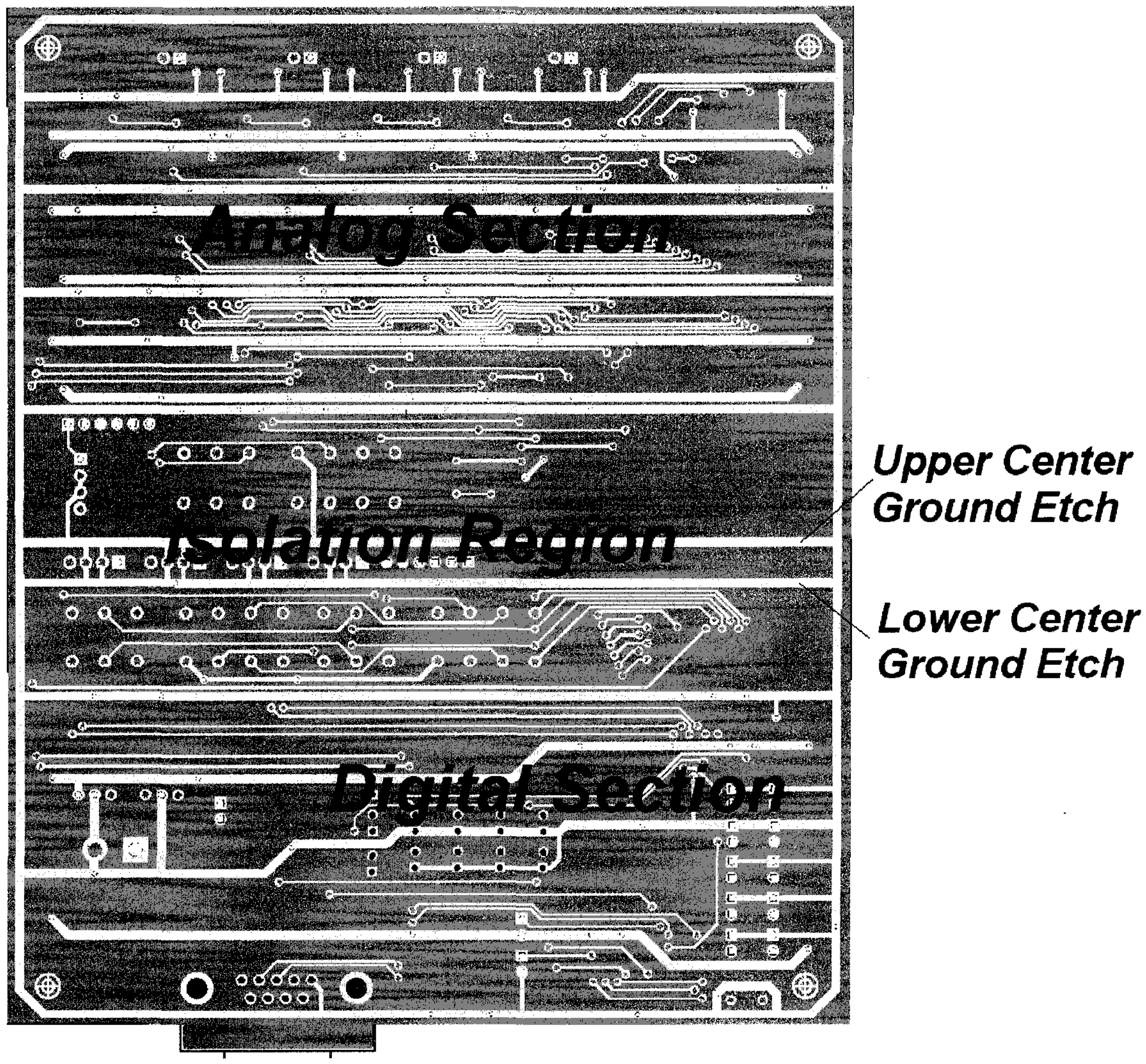


Figure 7.10 – Revision C Bottom Side Ground Etch Division

Note that an additional technique of raising the Signal-to-Noise Ratio (SNR) was also implemented by changing to a higher base audio signal amplitude. This however did not require any board redesign but instead simply changing resistor values. It is covered further in discussions on signal levels in *Chapter IX - Prototype Design Analysis*.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Field Programmable Analog Arrays

An interesting area of research that might aid in the development of a higher performance mixed-signal circuit board is the Field Programmable Analog Array (FPAA). Based along the lines of the more established Field Programmable Gate Array (FPGA), it contains internal analog circuitry instead of digital.

The main problem is that the development is still somewhat early and the technology not well established. (The first paper to appear on this topic was in 1999 at the MAPLD 1999 Conference [46].) There are also concerns that are common to any digitally based design which include increased power consumption. When considered for use in-vehicle the power demands of a single FPAA can however be assumed negligible [47].

Another issue is that the system is not fully configurable but is instead limited to Configurable Analog Blocks (CABs). These can be programmed to perform limited functions that involve a standard set of components; for example an OPAMP, programmable capacitor arrays (PCAs), programmable resistor arrays, and/or configurable switches [47]. For the most part the circuits designed in this solution are relatively straight forward however. Therefore, a moderately well equipped FPAA might supply most of the needed circuitry, allowing only for some basic buffering.

The advantage is that the design can be readily adjusted (increased flexibility) by a simple reprogramming. This might also include some form of signal level calibration that can be used to correct for lower level audio signals depending on the radio types. Then, mounted in combination with a reprogrammable, digitally based CPLD or FPGA, the whole circuit footprint could be greatly reduced and many of the mixed-signal

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

concerns from running etches too near each other can be avoided altogether. The whole is also much more flexible and can involve on-the-fly changes if needed [47]. This is something that in the future might prove to be a viable option as the current prototype models mature, as has occurred with FPGAs. There is also the promise of using such a system for prototype development. This would allow the testing of alternative circuits without requiring commitment to a specific hardware layout. Some work on such a prototype design circuit board is being pursued by the Nano-interconnects, Devices and Circuit Simulation Laboratory (NDCSL) at the University of Arizona although it appears to have thus far focused only on applications at the prototype level [48, 49].

Final Version PCB Layout Improvements

While the Revision C PCB meets the necessary performance results intended for the prototype design, there are some changes that can be made. Note that these were not previously implemented as they may have been impractical for a prototype (as with a four layer board) or simply because they are not necessary for in-lab and preliminary field testing. They are however recommended prior to any large scale production release, especially for non-developmental usage.

First, the design needs to be compressed down as much as possible in order to reduce the system footprint within the command vehicle. This will necessarily include some rerouting of the PCB etches but examination should also be made into alternate circuits and device packaging for space improvements. It is critical however that in performing any of these changes the original specifications for the devices by met or exceeded. Also, in re-laying out the PCB, care must be taken to maintain analog and

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

digital isolation (as discussed in the Revision B Digital Interference and Crosstalk with the Analog Audio section above). Considerable space was used to yield an excessively large isolation area for the prototype testing and some of this can be reclaimed. However the existence of two distinct circuit board sections must be continued.

One design change that could result in significant area reduction is the transition to a 4-layer PCB complete with ground and V_{CC} planes. (More planes could also be added although they would probably not be necessary or cost effective.) The main reason for not doing this on the prototype is that it limited the ability to make change to the design from the workbench. However, at this stage the basic designs have been proven and should be fairly static. Transitioning to a circuit board with planes will not only have the advantage of saving space by allowing the Power/Ground etches to be mostly removed but it should also result in a noise level improvement. This is because the enclosed current loops are then nearly minimal as the current in the planes can largely mirror that in the etches on the external layers.

In developing a final release version the examination of EMI effects should also be conducted. This is especially important for in-car applications and as it is designed for use with radios. Note that the addition of the power supply planes that result from going to a multi-layer board should help to reduce any stray radiation. Furthermore, the circuit board is in general a low frequency design with the highest oscillator frequency being the internal microcontroller clock at 22.1184 MHz. Some emissions may also be caused by the high speed digital logic although the propagation distances should be small.

As a final release design step the PCB must be fitted to some form of housing. Currently, the Revision C design does not include any mounting holes as it is not planned

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

for direct, large-scale field deployment. Therefore, any Revision D design will need to add these mounting holes at the correct locations according to the positions determined by the selected case. The locations of the external connectors such as LEDs, buttons, the power switch, and the serial port on the case will also need to be worked out. All of the interface connections (excluding the serial port) are via headers or screw terminals. This means that there is some flexibility as to where their final case locations are. However, the buttons and LEDs should be laid out in a logical pattern as described in the Hardware User Interface section of *Chapter VIII - Device Operation & User Interfaces*. Note that the case eventually used could also be metallic, as with the latest version of the Project54 IDB Interface box, to further reduce EMI. However, plastics might also perform adequately. This will be something that can be examined in the context of the EMI work.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER VIII

DEVICE OPERATION & USER INTERFACES

General Operation Overview

Any type of user control interface should adhere to human factor design standards. In the interest of minimizing system complexity, therefore, the controls for the device have been made very simple. Individual channels can be added to or removed from meetings. When connected to a meeting, a channel receives whatever comes in from all of the other channels that are also connected to that meeting. The system is configured as two main meeting groups that form the backbone of the system. Also, a channel cannot be connected to more than one meeting. Thus, steps are taken in the firmware to correctly remove a channel from a meeting if it is being added to the other. The user does not have to manage this limitation.

Commands to make or break the connections can be made using the hardware-based manual control head (consisting of buttons for commands and LEDs for feedback) or via commands received through the serial port (possibly from a GUI based program, such as a Project54 Application, running on a PC). The two methods, as originally required in the specifications, work in parallel with each other and can be used in any order and at any time. Synchronization is maintained by echoing of serial commands and notifications sent out the serial port when the manual control head is activated. (Note that this does however require that the serial cable be connected, although if it is disconnected

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

means are provided for the PC to resynchronize by requesting the current states.) Figure 8.1 depicts of simplified block diagram of the available user interfaces. Each are detailed in the following sections.

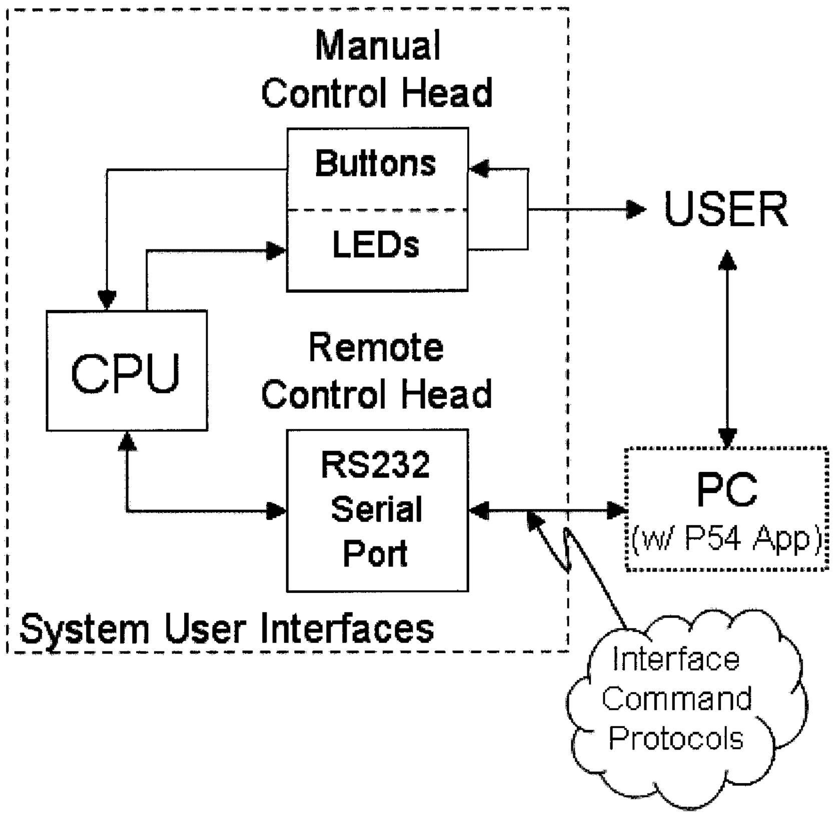


Figure 8.1 – Block Diagram of the User Interfaces

<u>Manual Control Head User Interface</u>

The hardware based, manual control head is very straight forward. Input to the system is provided through 10 buttons (two for each channel with the Master Microphone also treated as a channel for input purposes) and a main power switch. An output display to the user is accomplished through the means of 10 connection state LEDs and 1 status LED. This user interface was largely determined in the initial specifications for the system.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The 8 channel buttons interface directly to the Keyboard Interrupt Interface port on the microcontroller. The two Master Microphone buttons get devoted interrupt port pins, Int1 and Int0 for Meetings A and B respectively. The whole circuit is shown in Figure 8.2. The buttons are laid out in a grid of two columns and five rows. The columns correspond to the two meetings, A and B. The rows correspond, from top to bottom, with the channels in the following order – Master Microphone, Channel 1, Channel 2, Channel 3, and Channel 4. Pressing a button will toggle the connection state (connect/disconnect) of the channel associated with that button with the meeting associated with that button. (Note that the buttons themselves are not toggle buttons but are momentary contact buttons. Otherwise a software interface issued command would have no means of reversing a hardware button press.) Figure 8.3 depicts a sample screen for the front panel of the device that includes the buttons and LEDs.

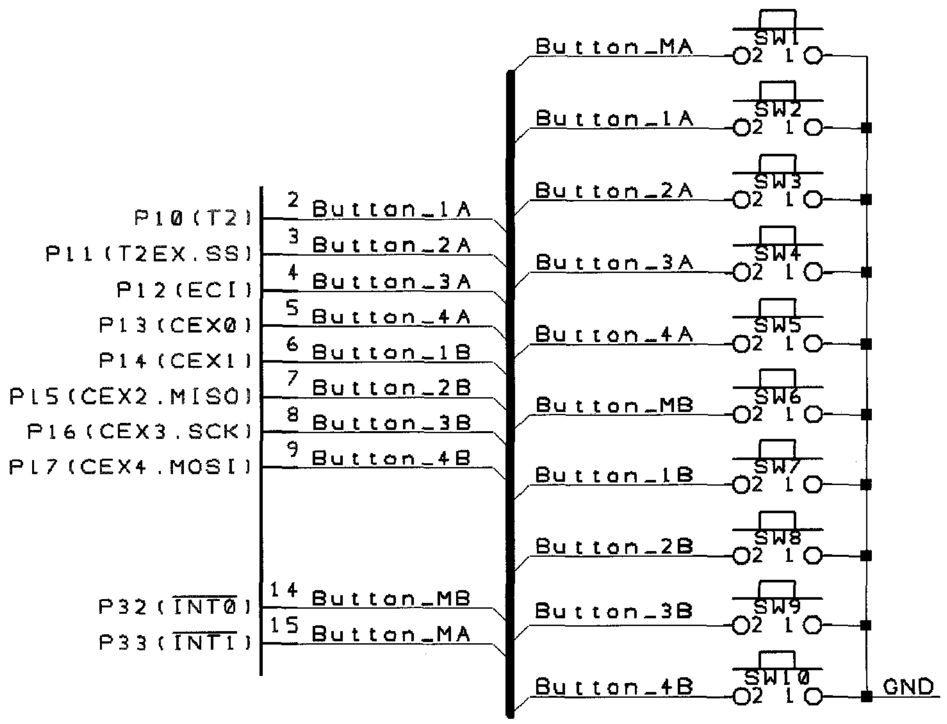


Figure 8.2 – Button Interface Circuit

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

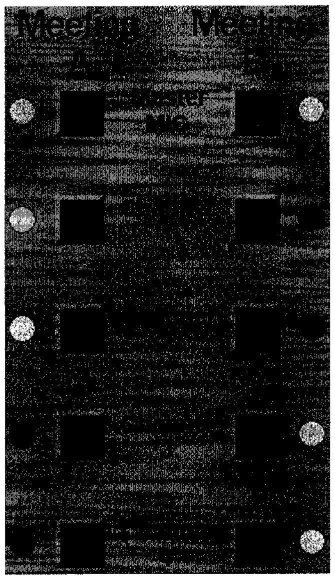


Figure 8.3 – Sample Button & LED Grid Layout

The buttons operate by grounding the microcontroller pins which are weakly pulled high through internal pull-ups. When this occurs an external interrupt is generated and the button interrupt firmware routine is executed (note that the Master Microphone interrupts actually have their own separate, processing routines). The routine performs a debounce on the pin to insure that button was actually pressed and there wasn't simply a brief glitch on the line. The code then determines which button specifically was pressed and the Channel-Meeting connection associated with it. Table 8.1 provides a listing of the buttons based on their microcontroller port pin and the associated Channel-Meeting mapping. (Note that Channel M is used to denote the Master Microphone.) Also observe that the way the Port 1 Keyboard Interrupt Interface port is set up, once the count of the pressed button is pressed (given by the Port 1 pin number) the third bit will be 0 for Meeting A and 1 if it was Meeting B. The channel can then be determined by clearing the third bit and then adding one. This simplifies the assembly code parsing of which button was pressed.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Port Pin	Channel	Meeting
P1.0	1	A
P1.1	2	A
P1.2	3	A
P1.3	4	A
P1.4	1	B
P1.5	2	B
P1.6	3	B
P1.7	4	B
P3.2	M	B
P3.3	M	A

Table 8.1 - Button Port Pin Channel-Meeting Assignments

After determining the correct Channel-Meeting pairing corresponding to the button the Interrupt Service Routine (ISR) executes the command specified by the button press and waits for the button to be released. The execution involves the following steps. First, the determination is made if the command is to connect or disconnect the channel from the meeting (if it is a connection change then the other meeting, if it is connected to that channel, will automatically be disconnected). Next, the microcontroller's internal status and control registers that store the current connections are updated. The actual output port can then be modified by writing to the latch buffer. This modifies the switch matrix control lines and the actual connection. Lastly, notification of the connection change is broadcast out the serial port to synchronize any attached PC software program to the manual control head connection change.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The LED display is laid out in a similar pattern with an active LED depicting that a particular channel is connected to the associated meeting. The LEDs are driven via the 8-bit latch buffer (in the case of the 8 channel LEDs) and simple transistor buffers (for the Master Microphone LEDs). Details of this are provided in the Additional Digital Hardware section of *Chapter VI – Digital Control Hardware & Firmware*. Figure 8.3 shows a sample screen that could be used for the layout of the LEDs and buttons. Note that in the example image Meeting A contains Channels 3 and 4 while Meeting B contains Channels 1 and 2. The Master Microphone is not connected to anything.

Not shown in the figure is the system status LED. This Status LED, which is green, flashes when the device is turned on (via the main power switch) and then goes solid once the initializations are complete. (It will also flash during a failed CPLD reprogramming.) Its main purpose is simply to notify the user of connection of power and basic operation. If at power-up the LED does not flash then either the hardware has been damaged or the microcontroller's program memory has been corrupted.

Remote/Software User Interface

RS232 Serial Communications Port

The main form of remote or software control is provided through the RS232 serial port. This communication was selected for two, closely related, reasons. First, serial ports are very common to PCs. Secondly, the IDB network used by Project54 is designed for interfacing and control of devices via a serial port. As a large part of this design involves making it compatible with, and therefore able to tap into the benefits of, the Project54 system an RS232 serial communications port was the only reasonable option.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Figure 8.4 shows a circuit diagram of the TTL-to-RS232 conversion circuit that was employed and the connection to the DB9 Female serial port connector. The heart of the conversion circuit is the MAX232. It uses a capacitor switching circuit to generate the voltage levels required for RS232 operation. In then acts as a converter to translate the RS232 signals into TTL for the microprocessor and visa versa.

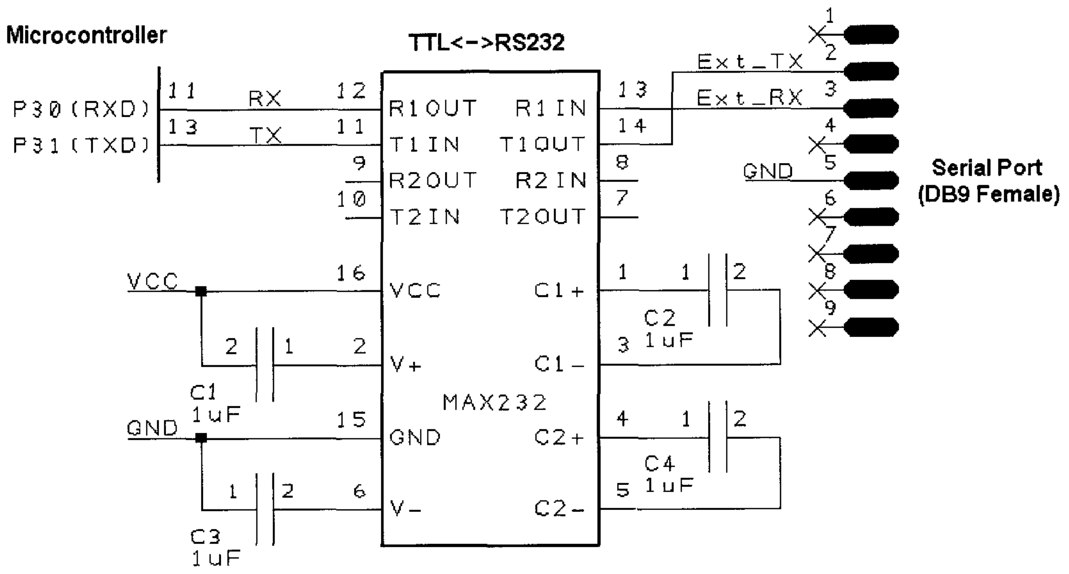


Figure 8.4 – TTL to RS232 Conversion and Serial Port Circuit

Note that the device will transmit on line 2 of the serial cable and receive on line 3. As such it is configured as a Data Circuit Terminating Equipment (DCE) unit. This means that it can connect directly to a PC with a straight cable (instead of a null modem cable which swaps pins 2 and 3). The main reason for doing this is that it will also allow the current revision of the Project54 IDB Interface box, which is configured as a Data Terminal Equipment (DTE), to be connected directly up to the device (without needing a cable if that is so desired in the installation process). In any event, it is important to insure that the correct cabling is used as otherwise the device will fail to properly

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

communicate and might in fact cause damage to its own or the other device's serial port driver hardware.

The PC end serial communications port must also be configured correctly in order to insure correct operation with the microcontroller's firmware. The basic settings are shown in Table 8.2. Note that flow control is used in order to ensure that the serial port circular buffer used in the microcontroller does not overflow and the XON/XOFF flow control is really only a precaution. The serial port buffer (queue) is a circular buffer that self manages and will transmit XOFF when it is half full. Received bytes are deposited at the head (or high address location) and bytes are removed from the tail (or low address) for processing. Note that the buffer size is set to 40 bytes which can be adjusted in the RAM_DEFS.txt assembly file.

Interface Type:	Serial
Standard:	RS232
Baud:	9600
Protocol:	8 Data bits 1 Stop bit No Parity
Flow Control:	XON/XOFF
PCB Connector:	DB9 - Female

Table 8.2 - Communications Port Settings

The serial port baud can also be easily adjusted by making some simple changes to the microcontroller firmware. This and other additional configuration details are contained in the assembly code file Serial.txt that is provided in Appendix C.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Serial Commands & Responses

The device is designed to respond to commands and/or requests received via the serial port. The commands are designed to be relatively short in order to minimize traffic on the IDB network while at the same time being mnemonically simple enough that they could be entered manually by a user operating a text terminal such as Microsoft HyperTerminal. In the more advanced case the serial commands could also be issued by a software program, such as an application running within the main Project54 software, which might provide the user with some form of a graphical user interface (GUI). Table 8.3 shows the serial port commands that are available and the special characters that are valid for insertion into the command strings.

Command	Command String	Special Character(s)
Reprogram CPLD	*RPCPLD	- None -
Request Version Number	CV	- None -
Request Long Version Number	CL	- None -
Request Connection States	CS	- None -
Change Internal Master PTT State	CPs	s = {'0', '1', '?'}

Modify Connection	CMcms	c = {'0', '1', '2', '3', '4'}
		m = {'A', 'B'}
		s = {'0', '1'}

Table 8.3 – Available Serial Port Commands

All general operation commands to the device start with a 'C'. Note that the capital letters are all fixed but the lower case characters need to be replaced with an appropriate value. For the Change Internal Master PTT State the 's', standing for state, can be '0' or '1' for OFF and ON respectively, or it can be '?' meaning that a state

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

request is being made for the current Internal Master PTT state. The ‘s’ in the Modify Connection command can however only be ‘0’ or ‘1’ for OFF and ON respectively since a state request is handled by the Request Connection States command.

Also in the Modify Connection command, the ‘c’, which stands for channel, can take on the numerical value of a channel: ‘0’, ‘1’, ‘2’, ‘3’, or ‘4’. Note that ‘0’ is the special channel number that is given to the Master Microphone in the firmware. The ‘m’ character should be replaced with the desired meeting involved in the connection modification and can take on values of ‘A’ or ‘B’. By way of example, the command string “CM0A1” will connect the Master Microphone to Meeting A. Similarly, “CM2B0” will remove Channel 2 from Meeting B.

In all cases the microcontroller will respond to a command/request received via the serial port by broadcasting back a message. Reference Table 8.4 for a listing of possible responses from the microcontroller to the PC. (Note that the command to reprogram the CPLD is a special command and as such is covered in the JTAG Interface Programming of the CPLD section of *Chapter VI – Digital Control Hardware & Firmware.* There will be no immediate response to this command short of the prompt character, ‘$’, used for requesting the next XSVF file byte.)

All serial port responses from the microcontroller, except for the Long Version String, start with the character ‘R’ and end with a carriage return, represented by ‘\r’. The Version Number string is transmitted in response to the Request Version Number command and similarly the Long Version Number String is transmitted after receiving the Request Long Version Number command. These are most often used to determine the firmware of the microcontroller to check if it is up-to-date.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Command	Command String	Special Character(s)
Version Number	RVsvs\r	svs = Short Version String
Long Version Number	lvs	lvs = Long Version String
Connection States	RS$s_M s_1 s_2 s_3 s_4$\r	s_M = {'A', 'B', '0'} s_1 = {'A', 'B', '0'} s_2 = {'A', 'B', '0'} s_3 = {'A', 'B', '0'} s_4 = {'A', 'B', '0'}
Modify PTT State	RPs\r	s = {'0', '1'}

Modify Connection	RMcs\r	c = {'0', '1', '2', '3', '4'}
		s = {'A', 'B', '0'}

Error - Reboot	RE\r	- None -

Table 8.4 – Serial Port Responses

The Connection States response transmits the connections states for the Master Microphone and all four channels. Note that only state identifiers are transmitted, not channels. The channel that corresponds to each state is determined from the state's location relative to the preceding RS. Thus, the response can also be written as R S {State of Master Microphone} {State of Channel 1} {State of Channel 2} {State of Channel 3} {State of Channel 4}. Note that each state can be either 'A', 'B', or '0' for whether the channel is connected to Meeting A, Meeting B, or neither. This response is typically done following reception of the Request Connection States command but is also done on initial power-up. This is done to notify any connected PC software that the system is either just starting up or has been reset. The effect is that synchronization between the hardware and any mirror software will be maintained.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The Modify PTT State response is used to echo back the current/new state for the Master Microphone's Internal PTT signal. It will be the same whether the original command was a change order or a state request. Note that this response is also used on power-up in order to notify and synchronize the PC software to the hardware.

After the Modify Connection command is received the microcontroller will transmit the Modify Connection command as an echo to notify the PC that the connection has been changed and the connection can now be updated in the software mirror as well. The software should not change its perceived connection status until it receives this echo. The Modify Connection response can also be initiated by a hardware interface action (or button press) as discussed in the next section.

Note that the Error – Reboot response will be given if an error is detected in the switch matrix control line output buffer. Typically this would occur as a result of a channel being on both meetings. This is prevented in the firmware and if it is observed would reveal either a firmware glitch or a hardware failure. Following the broadcast of this message the microcontroller will automatically reboot through a software reset.

Hardware Command Feedback

One of the critical aspects of the design was that the device be able to operate completely on its own or through control by a PC over some type of interface, preferably a serial port. Since commands received from the PC will update the microcontroller's registers, and thus the hardware connections, there is little concern of the hardware control head not keeping synchronization with a changed ordered via the software. However, the other direction, a changing being made via the manual, hardware control

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

head, may not have the same results. Therefore, it is important that any time a button on the manual control head is pressed and a state change made, notification also be sent out the serial port. This is accomplished through the dual use of the Modify Connection response which is both an echo for a command and also serves to notify of hardware control head action. This response is unique in this regard in that it does not necessarily respond to stimuli from the serial port. It must be stressed that this operation is critical to insure that the PC's software doesn't lose synchronization with the hardware. The format of the Modify Connections response is discussed further in the previous Serial Commands & Responses subsection.

Interaction with a P54 Control Application

The main Project54 system is contained within a PC that is meant to operate within a police cruiser or other public safety personnel vehicle. Therefore, even though this public safety radio interoperability solution is designed with being self-sufficient in mind, it is also beneficial if it can be connected to the larger Project54 computer-based structure. This is one of the main reasons for establishing the parallel control of the device via a serial port. This then allows it to connect up to the Project54 IDB network and be controlled like any other piece of hardware that is currently on the market.

One of the greatest advantages to operating this device with the Project54 software deals with the fact that up to four different radios may be in operation within the vehicle at any given time. Current development of the Project54 software includes a multi-radio control application(s). This will allow even further ease of use for the public safety personnel in charge of the device as they can easily adjust settings on each of the

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

four radios and control the interoperability hardware all from a simple touch-screen display and voice commands.

A simple, P54 Application was created to test the basic serial port interface of the system. The GUI menu screen is shown in Figure 8.5. The program is designed to scan in the firmware version number of a connected MRCM device. This is done so that the operator can easily determine if the latest revision of firmware is active in the microcontroller and take appropriate update measures if it is not. For this example screen shot however no device was connected over the IDB and therefore the system has returned the --- **MRCM Not Connected** --- message to inform the operator that no interoperability device was accessible. (Note that the CH, Control Head, display button will also turn from red to green when an operational interoperability device is connected.)

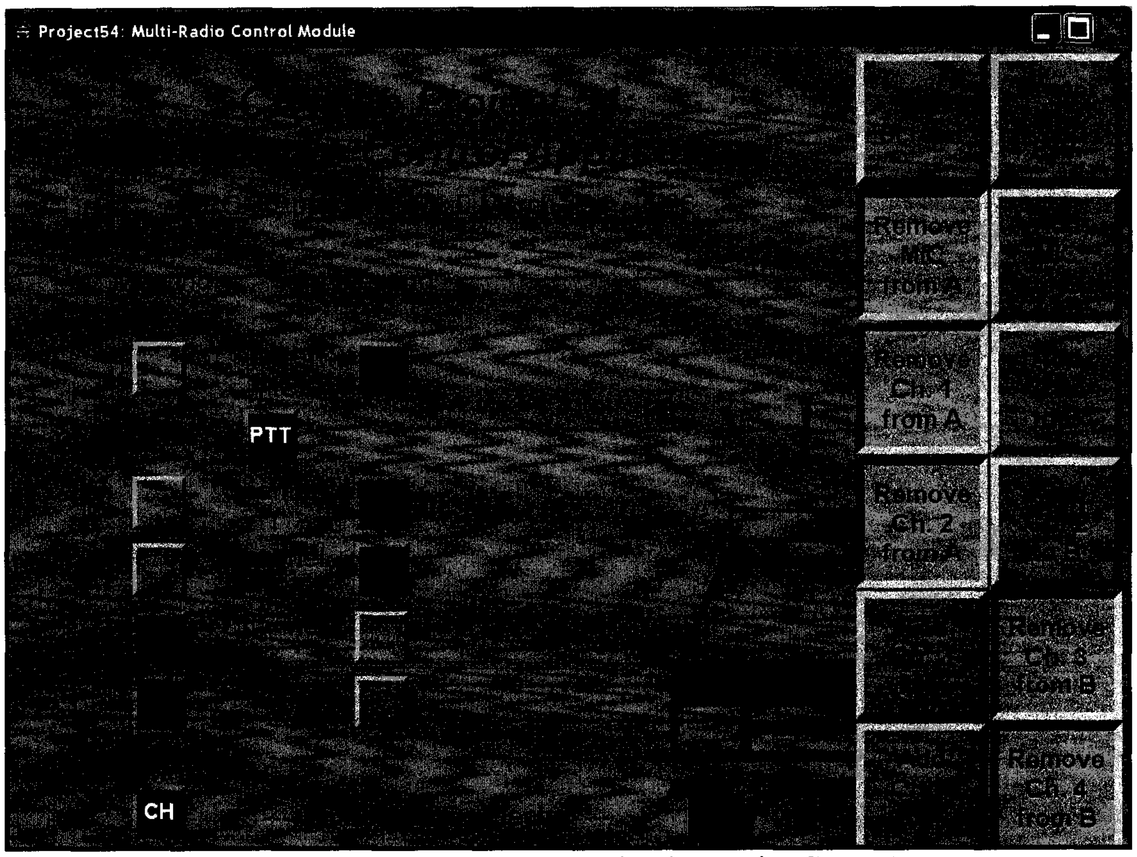


Figure 8.5 – Test P54 Application for Device Control

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The P54 Application as shown is operating in Demonstration Mode which allows it to act as though a device is connected. The two main display button columns denote the radio Meeting groups and whether each radio is connected to those meetings or not. When a GUI button becomes depressed the corresponding Channel-Meeting display button will change from red to green indicating that there is now a connection between that radio and any other radios that are in that meeting. For the example shown, Radios 3 and 4 are able to communicate with each other while Radios 1 and 2, as well as the in-car Master Microphone, are also connected with one another. Note that the two meetings, or conversations on each of the meetings, do not interfere with each other. Thus, audio transmitted from the radio on Channel 3 to the radio on Channel 4 will not appear on either Channels 1 or 2.

Note that the Master MIC PTT button can be used to manually trigger the Master Microphone's PTT signal in the hardware. Thus, any audio signal present on the Master Microphone will be transmitted out to any radios that share the same meeting, in this case Channels 1 and 2. This allows the PC to actually transmit its own audio signals out of a sound card or other port if desired. Note that when activated by pressing the Master MIC PTT button the PTT display button will turn from red to green. Also note that the button is a momentary button only and will not toggle and lock in the down state.

The main purpose of this button is currently for demonstration purpose as the operator could just as easily manually queue the in-car microphone. However, more importantly it demonstrates that the software could automatically queue the Master Microphone's PTT (without having to involve the button or the user). An important application of this is that the Project54 system microphone, already used for PC

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

interaction through voice commands, could also be used to now talk over the radio by routing the signal out a sound port to the interoperability device and simultaneously queuing the Master Microphone. Again this would help to consolidate the user interface for the public safety personnel in command of the vehicle.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER IX

PROTOTYPE DESIGN ANALYSIS

Performance Summary & Specifications

Power Supply Specifications

A critical design features is that the developed circuit board be able to operate off of the in-car power supply in order to meet the design goal of mobility with an in-vehicle command application. An automobile battery typically outputs a voltage of 13.8 volts. Current consumption should also be reduced as much as possible in order to avoid draining the battery. Table 9.1 shows some of the power supply voltage specifications for the completed design. Note that some values are actually estimations based on the manufacturer datasheets although some application specific metrics have been calculated from laboratory testing. All voltage are given in DC values.

In the table the recommended supply voltage has a minimum of 9.0 V to insure proper operation. The maximum recommended voltage is computed for the design both without ($V_{IN\text{-}max}$) and with ($V_{IN\text{-}max\text{-}HS}$) a heat sink, following the curves supplied by ON Semiconductor [34], for a temperature of approximately 77° F (25° C) and a current of 375 mA, with a small safety margin. It also is selected to insure that the fuse remains within specifications. Note that the heat sink used for these calculated values is specified to have a thermal resistivity coefficient of θ_{HS} = 15° C/W.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Recommend Supply Voltage	$V_{IN\text{-}min}$	9.0 V
	$V_{IN\text{-}max}$	14.0 V
	$V_{IN\text{-}max\text{-}HS}$	24.0 V
Absolute Maximum Supply Voltage	$V_{IN\text{-}AMAX}$	30 V
Target Operating PCB Voltages	V_{CC}	+ 5.0 V (± 0.1)
	V_{SS}	- 5.0 V (± 0.1)
V_{CC} Out of Specifications	$V_{IN\text{-}AMIN}$	8.3 V
	$V_{CC\text{-}AMIN}$	< 4.9 V
Relay Shutdown Voltage	$V_{IN\text{-}R}$	5.0 V
	$V_{CC\text{-}R}$	2.0 V
Microcontroller Shutdown Voltage	$V_{CC\text{-}SD}$	< 2.0 V

Table 9.1 – Power Supply Voltage Specifications

The board will not function at peak performance if the main power signals, V_{CC} and V_{SS}, are out of specification (when the supply voltage goes below 8.3 V DC). However, the microcontroller will still be operational as will all other parts on the circuit board (although analog voltage ranges will be greatly reduced). Once the input supply voltage reaches 5.0 V DC the PCB V_{CC} will be at 2.0 V and the relays will no longer be held closed. Thus the board will automatically isolate. Note that even at this level however the microcontroller will still remain operational and respond to manual control head commands. (The serial port may no longer be operational if the connected device is more sensitive to out of specification voltages.)

The current requirements of the Revision C PCB are shown in Table 9.2. Note that the original estimate is also shown for comparison. The $I_{CC\text{-}LED6}$ is the current with the maximum number of LEDs (six) ON while $I_{CC\text{-}LED1}$ is the necessary current with only

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

one LED ON (the status LED), which implies no radio connections. It should be pointed out that the $I_{CC\text{-}LED6}$ is not in actuality a true maximum as it is taken with the microcontroller in the idle, lower power consumption, state. However, the time spent outside of the idle state is typically very brief and can be viewed mainly as a disturbance to the typical current draw. Also, the current will increase slightly during high audio activity; although this should also be negligible, unless the outputs are driving some very low impedance loads. Taken as a whole the analog circuits might increase current consumption by 20 to 40 additional mA. This margin is built into the consideration of the recommended current supply, I_{CC}. Note all measurements are taken with the input supply voltage at 13.80 V DC.

Recommended Supply Current	I_{CC}	400 mA
Supply Current - 1 LED ON	$I_{CC\text{-}LED1}$	250 mA
Supply Current - 6 LEDs ON	$I_{CC\text{-}LED6}$	310 mA
Original Current Estimate	$I_{CC\text{-}est}$	370 mA

Table 9.2 – Power Supply Current Specifications

Device Connection Specifications

In order for the designed device to be used correctly it must also be properly connected to a power supply, the necessary radios, the hardware of the manual control head, and the optional PC. Figure 9.1 shows an image of the Revision C PCB with the crucial components marked off. The following section discusses some of these components and specifically the pin-outs of the connectors and the proper hardware that can be used for the interfaces.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

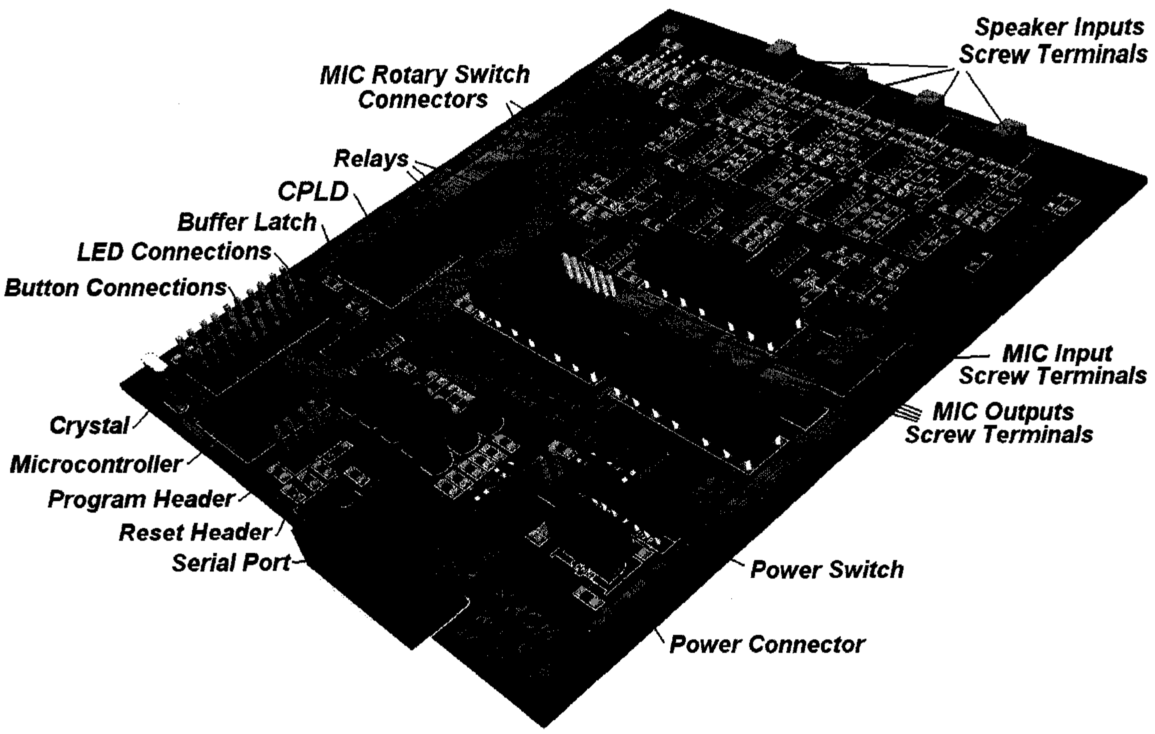


Figure 9.1 – Crucial Elements of the Revision C PCB

<u>Channel Speaker Input Screw Terminals</u> - The inputs from each channel speaker are shown in Figure 9.2. Note that the connectors used are 2-pin screw terminals to allow for easy connection. The inputs are differential and as such have positive and negative ends although the distinction is unimportant for this application. The resistive input impedances on the Revision C design are approximately 1 MΩ.

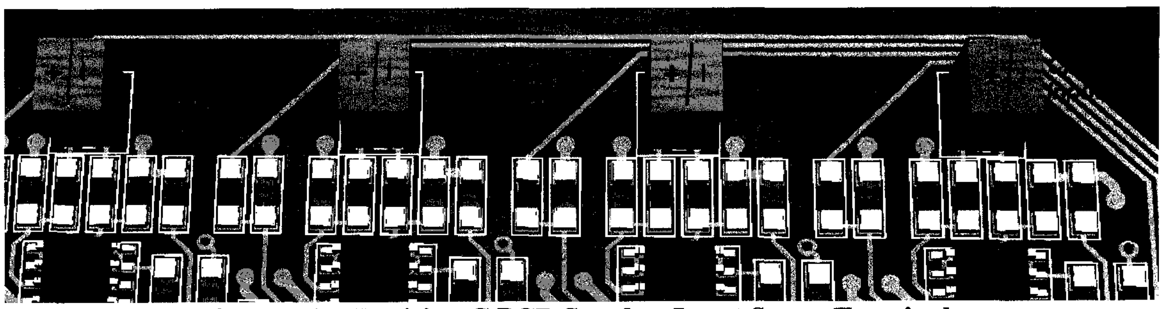


Figure 9.2 – Revision C PCB Speaker Input Screw Terminals

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Main System Microphone Screw Terminal - The main microphone used by the system is also connected via a screw terminal. The four pin connections are shown in Figure 9.3. Note that the input impedances for the audio of the Master Microphone and each channel microphone are 1 MΩ. The PTT signals also are connected to pull-up resistors which are tied to V_{CC} (+5 V). The resistors used for this purpose are 100 KΩ. These are connected only one at a time, when the rotary switch (see the next sub-section) selects the specific channel associated with that audio input impedance / PTT pull-up.

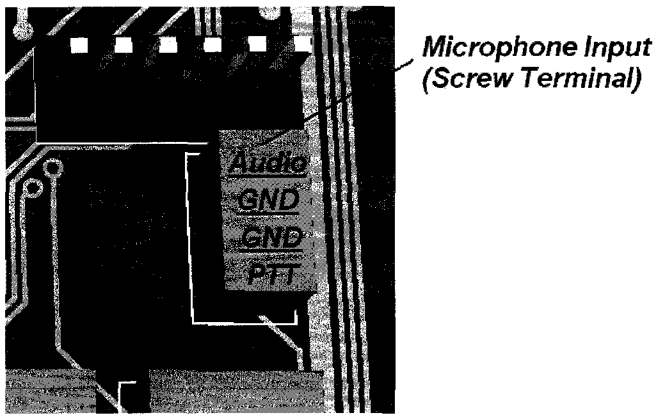


Figure 9.3 – Revision C PCB System Microphone Connection

Main System Microphone Rotary Switch Connectors - The system uses a single microphone to operate as the Master Microphone and as a microphone for each of the four radios. To accomplish this, a Double-Pole, 5-Throw (DP5T) switch is used. The first pole is for the microphone audio and the second is for the microphone's PTT signal. The five throw positions are for the four channels and the Master. The specific connections are as labeled in Figure 9.4. Note that the 6-pin headers used are uni-directional and have a .100" pitch.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

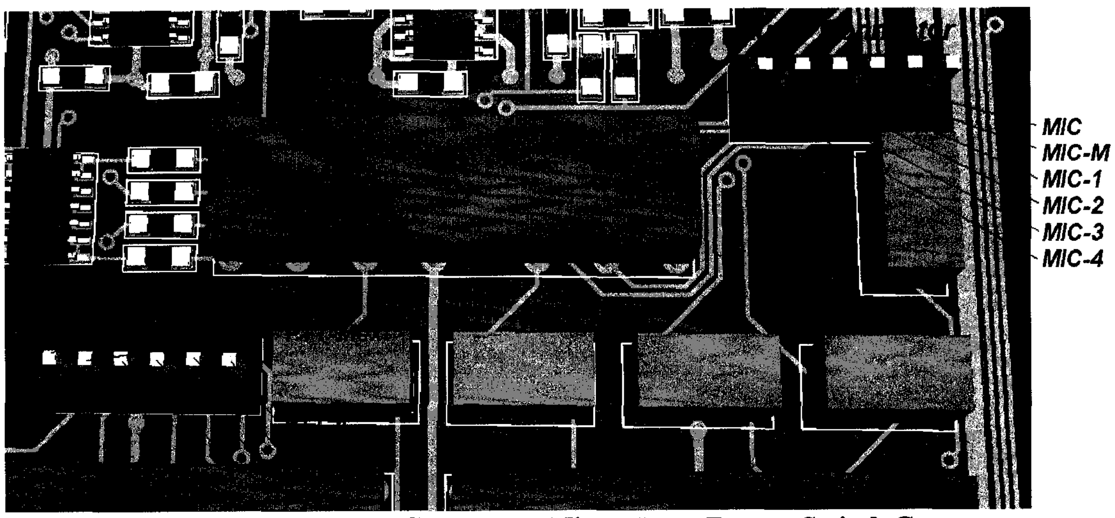


Figure 9.4 – Revision C PCB System Microphone Rotary Switch Connectors

Channel Output Screw Terminals - Each channel's output is available from the screw terminals shown in Figure 9.5. Note that the outputs are designed to be connected, if necessary, directly to a radio's microphone input (with proper signal level adjustment). The audio outputs are directly supplied from a buffering OPAMP (LM324) and as such have relatively low output impedance. The PTT signals are supplied directly by the CPLD. Its output pins should be able to source in the range of 24 mA [25].

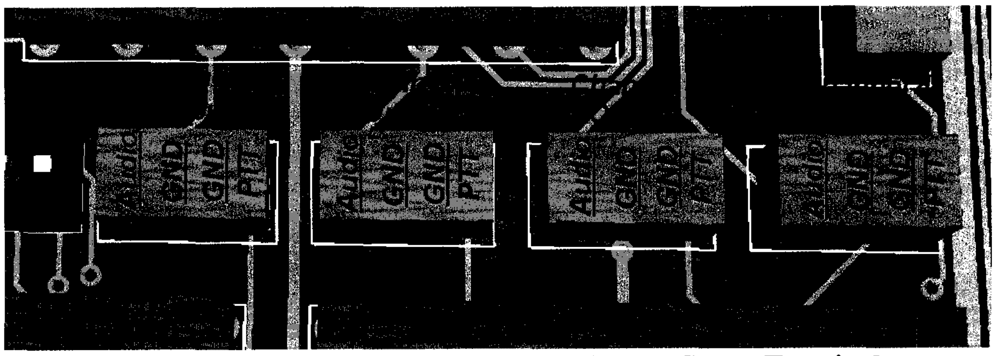


Figure 9.5 – Revision C PCB Channel Output Screw Terminals

Power Connecters - There are two power connectors on the circuit board. The first is a two pin, .100" pitch header that is used to connect the main Power Switch to the

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

PCB (note that its orientation is unimportant). The second is the Power Connector which provides the main supply voltage to the circuit board. It is done with a .250" pitch, 2-pin directional AMP header connector. The pin connections are as shown in Figure 9.6.

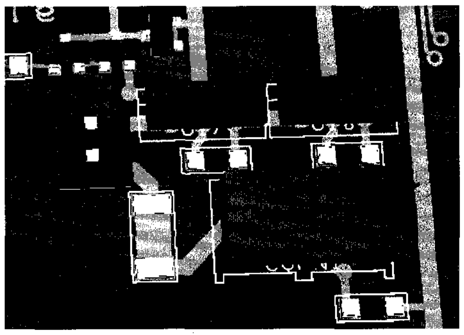


Figure 9.6 – Revision C PCB Power Connections

Button Connections - The buttons are designed to be connected via two pin headers to the PCB. Note that this allows them to be panel-mounted for greater flexibility in the case layout and construction. The headers used are .100" pitch and are designed to be single-directional and locking. Figure 9.7 shows the specific PCB button layout. Note that since the buttons are simple SPST momentary switches their orientation is really insignificant.

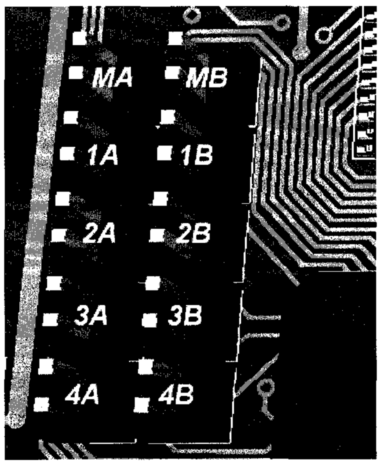


Figure 9.7 – Revision C PCB Button Connection Details

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

LED Connections - The LEDs are also designed to be connected via two pin headers to the PCB. Note that this enables them to be panel-mounted for greater flexibility in the case layout and construction. The headers used are .100" pitch and are designed to be single-directional and locking, which is important for the LEDs. Figure 9.8 shows the LEDs and labels important aspects including what each LED is associated with (based on its channel and meeting) and the pin-outs. All the LEDs are oriented the same direction with the Cathode having the square pad.

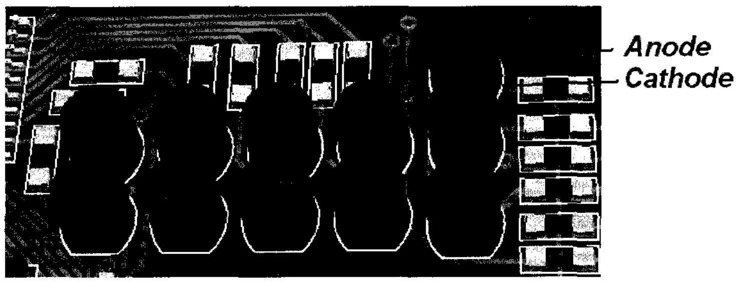


Figure 9.8 – Revision C PCB LED Connection Details

Reset and Programming Headers - These two .100" pitch headers are designed for use during the microcontroller reprogramming (ISP) operation. Figure 9.9 shows the specific pin names. The upper header is referred to as the Reset Header and the lower is the Program Header.

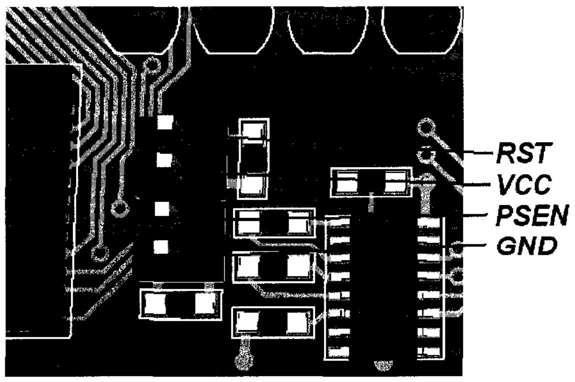


Figure 9.9 – Revision C PCB Reset and Program Headers

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Audio Signal & PTT Generation Latency

The Latencies & Their Effects

One of the critical design issues addressed in this research has been designing a solution that minimizes system latency in order to improve performance while also minimizing cost. The main solution to this has been using analog front-end hardware that eliminates the need for slower digital PC-based processing and the associated buffer. Also involved has been the use of high speed digital logic, such as the CPLD, for the routing of the PTT signals (which becomes important at the beginning of a conversational phrase when the PTT signal first transitions to active low).

However, the problem observed by Mock with the PTT detection circuit used by the VoIP Interoperability solution in [22], is that depending on the input audio signal's voltage level there was also an inherent response time to the generation of the PTT signal. For example, with a 900 mV RMS input the response was 18.82 ms. However, for a 200 mV RMS input the PTT signal was delayed by 84.55 ms! The net result is that clipping of an operator's words was observed to occur at the beginning of transmissions.

A possible solution proposed, but not implemented, was software based audio stream buffering. The ideas is that a buffer of the audio stream could be continuously maintained. Then, when the PTT signal was tripped, the buffer, which would reach back some specified amount of time, 100 ms for example, could be used at the start of the transmitted data stream [22]. Thus, even if the PTT signal was 85 ms later then the start of the voice audio, nothing would be clipped.

This option however is unacceptable as first it assumes that the system being used already involves digital sampling of some form. This then means that there is already

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

going to be some latency in the system due to the normally required buffering of the input and output data streams as well as delays relating to the performance of the PC being used. Then, by using the delayed buffer technique, even more latency will be potentially added to the system, for this example the length of the buffer or 100 ms, resulting in a net delay that is far greater than what is acceptable.

Previous studies have been conducted into the effect of transmission latency of voice and/or data on a conversation [50]. These have found that when the total, one way latency (from mouth-to-ear) is less than 150 ms the users for the most part "experience essentially transparent activity", although for some applications there might be some modest affect. Also, once the latency is increase to more than 350 ms the majority of users become dissatisfied with the level of performance and there is a noticeable impact on the conversation. Even further limiting the range of acceptable system delays it should be noted that for "highly interactive tasks" delays of less than 100 ms can even become disruptive.

While these overall system delays have been briefly dealt with in the previous, digitally based interoperability solutions, another consideration has been left out of the process. The research into the effect of latencies is typically made when referring to the maximum network delays assuming that the VOIP or other solution encompasses the whole system. This is not the case however when dealing with the public safety radio interoperability problem from a system gateway approach. Instead consideration must also be made of the latencies that are already inherent to the system - repeater delays, wave propagation delay times, existing radio equipment latencies, etc. And, observance

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

should be made of the fact that these delays are doubled as transmissions must go through both radio systems.

Varying the PTT Generation Parameters

The current hardware based solution is obviously a vast improvement over the alternative of increasing the total system latency, in the accordance with the original design criterion of developing a high performance, low latency system. Therefore, some other means must be taken in order to minimize the clipping resulting from the PTT detection circuit as much as possible without sacrificing system latency. By examination of the results from [22] two methods for reducing the delay for the PTT generation circuit have been determined. First, having a larger input voltage was observed to reduce the PTT generation time. Therefore, noting the basic circuit similarities, the use of a higher base audio signal voltage of approximately 2 volts (peak-to-peak) as done for this work should keep the delay time on the level of 10 to 20 ms. Also, and perhaps more importantly, the circuit design calls for the use of a comparator instead of digital logic as a threshold detect of the amplified signal. This then allows the threshold voltage level to be adjusted by changing the $R_2/(R_1 + R_2)$ ratio of Figure 9.10 (a re-showing of Figure 4.8) as discussed in Chapter IV. This has the further result of also decreasing the rise time if the performance is insufficient. Thus, the design still is able to maintain very limited clipping while maintaining the low latency associated with the hardware based approach. (Also note that the comparator pull-up resistor R_3 can also be decreased to reduce the rise time.)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

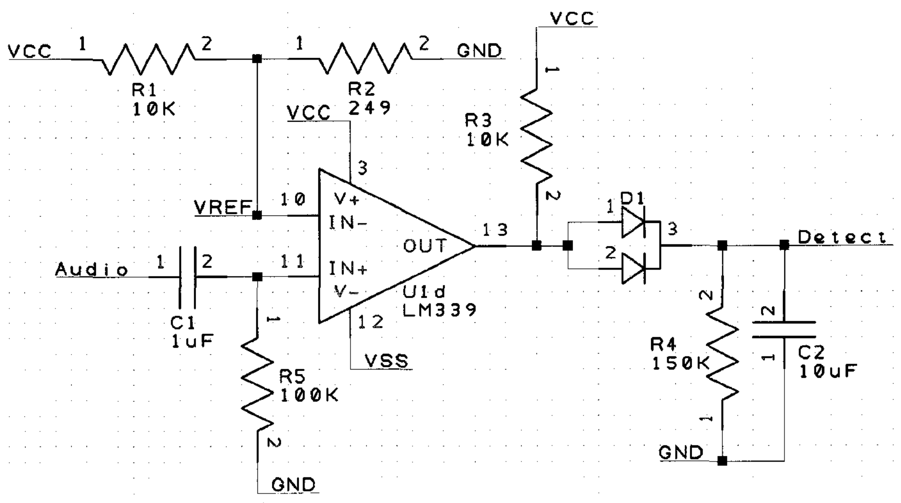


Figure 9.10 – PTT Signal Generation Circuit

Note that in the figure the original design values for the resistors are shown. However, these could be varied to improve performance. This was first done in a simulation in MATLab that was created to approximate the circuit using various manufacturing parameters. The MATLab script used for this purposes is shown in Appendix F. Note that a test signal of 8 KHz was used for the audio stimulus which was on for a duration of 2 seconds. The parameter values used are as shown in Table 9.3.

Parameter	Value
R_1	10 KΩ
R_2	100 Ω
R_3	2 KΩ
R_4	250 KΩ
C_2	10 μF
V_{diode}	0.3 V
V_{CPLD} (threshold)	2.0 V

Table 9.3 – MATLab Simulation Circuit Parameters

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Figure 9.11 shows the raw PTT signal that is generated by the PTT Detection Circuit modeled by the MATLab script. Note the distinctive rise time while the input audio is active. The circuit then holds throughout the input but then begins to decay away once the signal is removed as shown by the decay curve on the right. The rise, or response, time is 23.0 ms and the hold time is 2.0951 seconds using the indicated values.

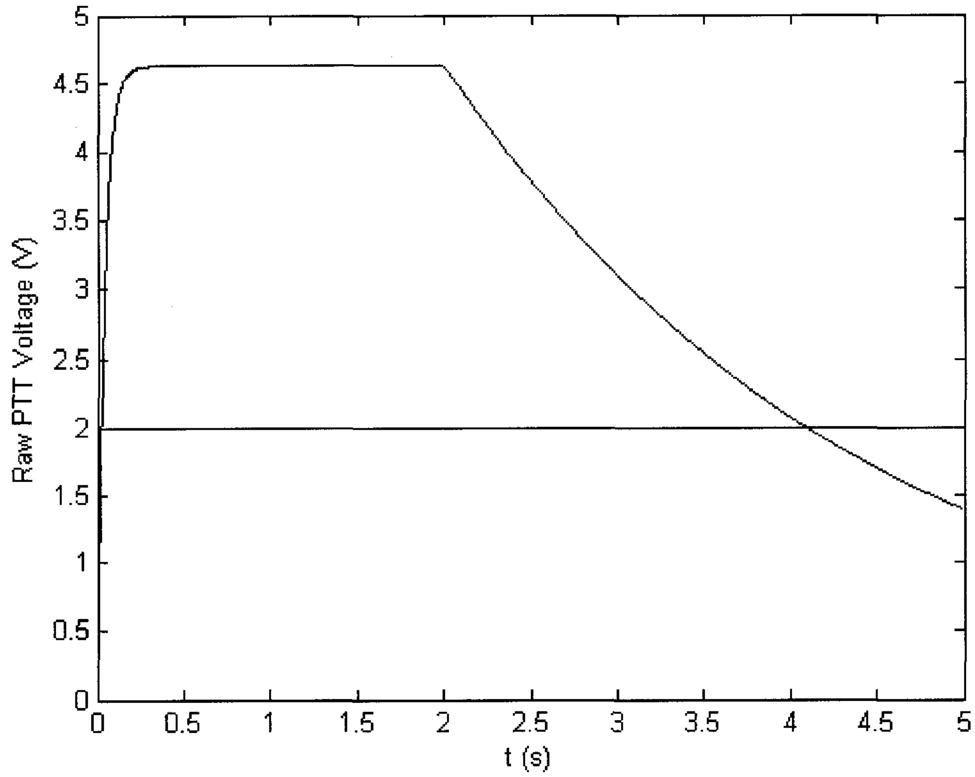


Figure 9.11 – Raw PTT Signal Generated

Figure 9.12 shows the input waveform, the clipped output waveform, and the digital PTT signal. The PTT signal digitization uses a threshold $V_{CPLD} = 2.0$ V (shown as a horizontal line in Figure 9.11) which was consistent with the Xilinx CPLD. It is obtained directly from the raw PTT signal of Figure 9.11. Observe that only a very small sliver, 23 ms long, is clipped off on the output audio.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

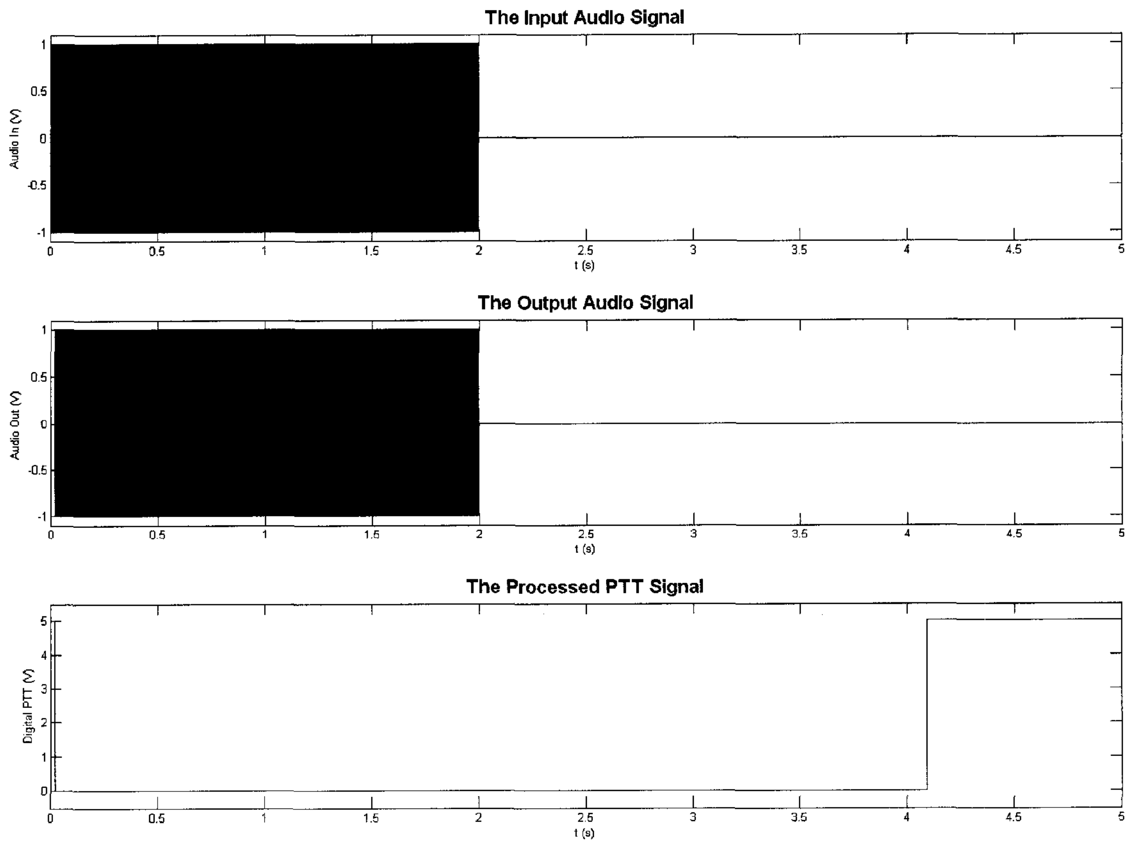


Figure 9.12 – Input and Output Audio with the PTT Signal

To check the effect of changing the R_3 pull-up resistor, it was swept over a range of values. The values used were 100 Ω, 250 Ω, 500 Ω, 1 KΩ, 2 KΩ, 5 KΩ, and 10 KΩ. The resulting rise and hold times for the PTT detection circuit are shown in Figures 9.13 and 9.14 respectively. These showed that in general a lower R_3 value would improve performance by reducing rise time (PTT signal latency) and the voice clipping. The reduction in hold time was minimal and could easily be re-increased by raising the value of R_4 (the RC decay resistor). The value of 2 KΩ was decided upon however in order to reduce the current draw required by the comparator. As configured it is now around 5 mA per channel. (Further reduction of this to 2.5 mA might also be achieved by connecting the lower power rail of the comparator to ground instead of -5 volts. This could easily be accomplished in another board revision.)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

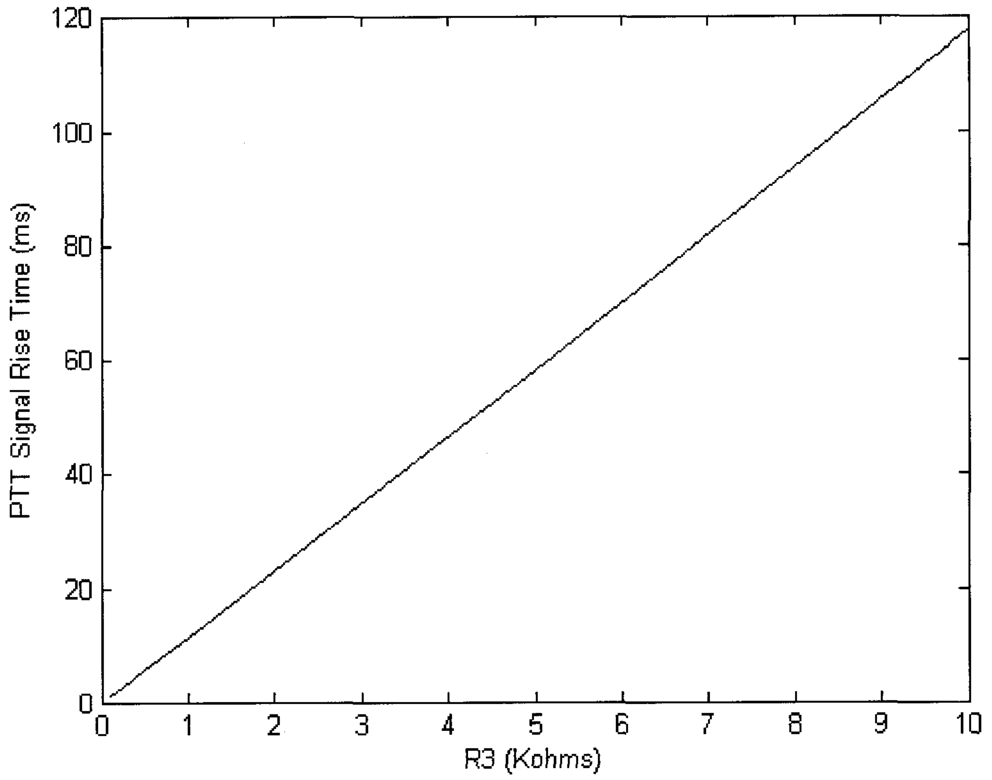


Figure 9.13 – PTT Signal Rise Time for Swept Pull-up Resistor

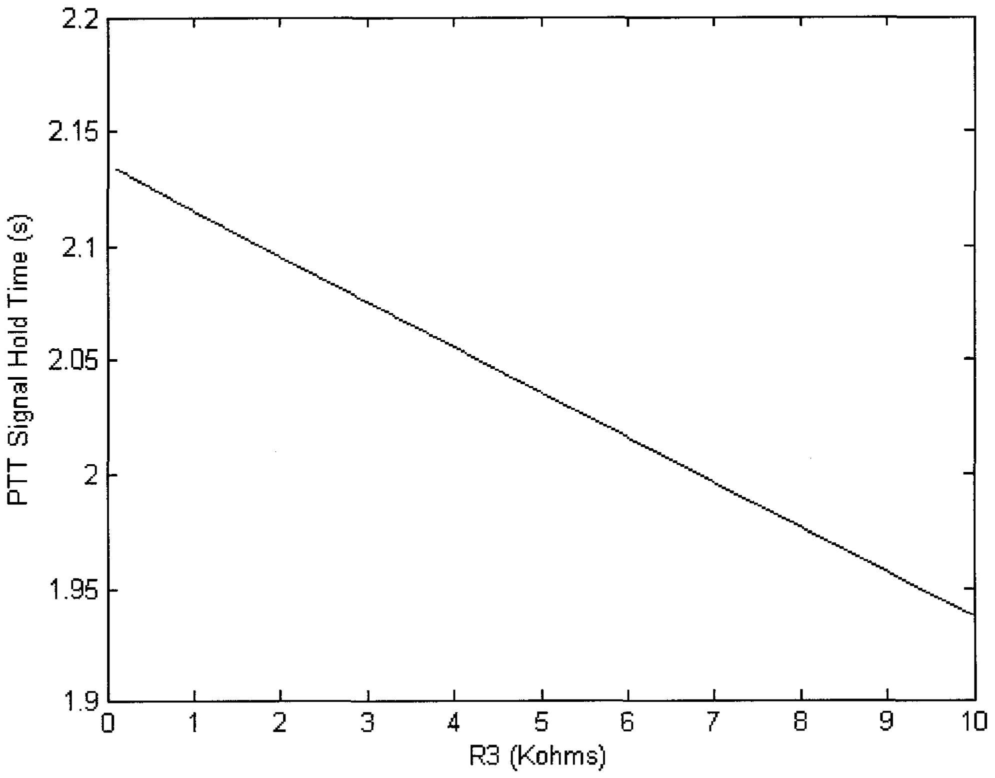


Figure 9.14 – PTT Signal Hold Time for Swept Pull-up Resistor

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

PTT Generation Circuit Test

The physical circuit board was modified to reflect the results of the PTT simulations. Testing was then done to determine the actual PTT generation rise and hold times of the circuit. Initially a test tone of 8 KHz was input (via another MATLab script in Appendix F). The input audio is shown in Figure 9.15 along with the output PTT signal as captured on an oscilloscope. Note the hold time is approximately 3 seconds long (more than predicted by the simulations).

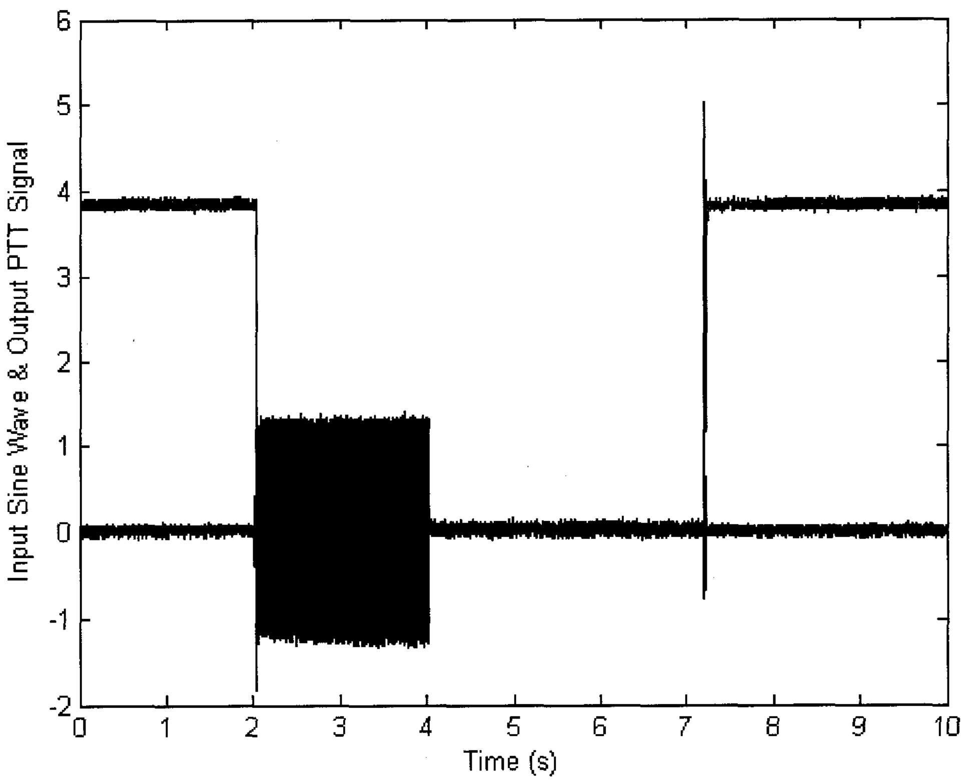


Figure 9.15 – Input Audio Tone and Output PTT Signal

Figure 9.16 shows a zoomed in image of the input audio and output PTT signals. (Note that the time is not referenced to the same point as in Figure 9.15.) From the figure it can be observed that the rise time delay (actually fall time since the PTT signal is

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

active-low) is around 14 ms. This is actually a slight improvement over what was determined in the simulation.

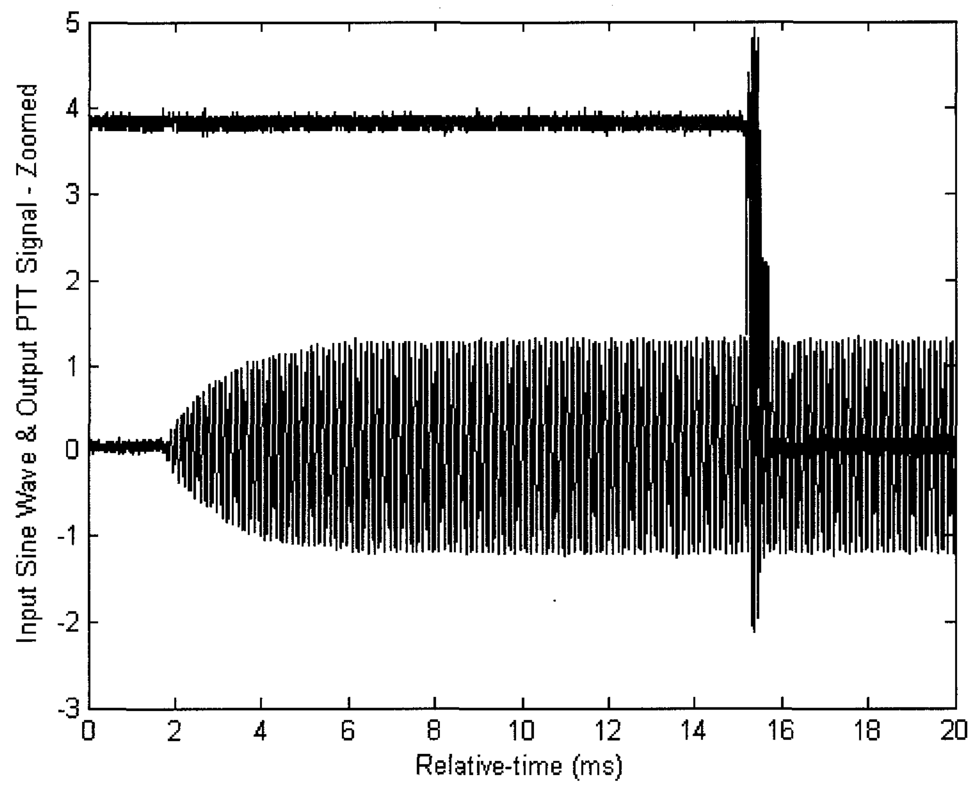


Figure 9.16 – Input Audio Tone and Output PTT Signal - Zoomed

Next, the PTT response was tested by using an actual test phrase. The phrase used was "testing". Note that this phrase contains two distinct syllables that can be seen. The same plots as for the testing tone are shown in Figures 9.17 and 9.18 for the overall and zoomed in oscilloscope data captures respectively. Note from Figure 9.17 that the hold time exceeds the window size. More importantly note that any pause in an officer's speech will not cause an automatic loss of the PTT signal. From Figure 9.18 it can be seen that the rise (response) time is around 15 ms as observed for the sine wave. Note that the slight increase may be because there is less energy in the initial portion of the speech as opposed to the sine wave. (Due to play-back and oscilloscope resolution limitations the zoom-in for the testing phrase was not as far as used for the tone.)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

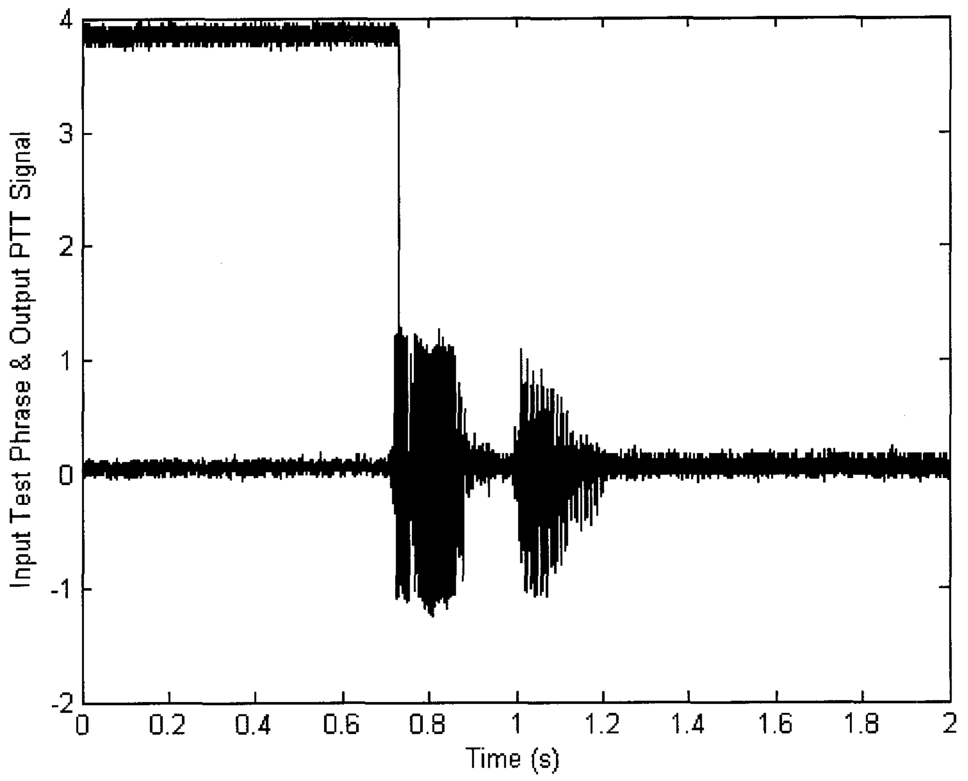


Figure 9.17 – Input Audio Test Phrase and Output PTT Signal

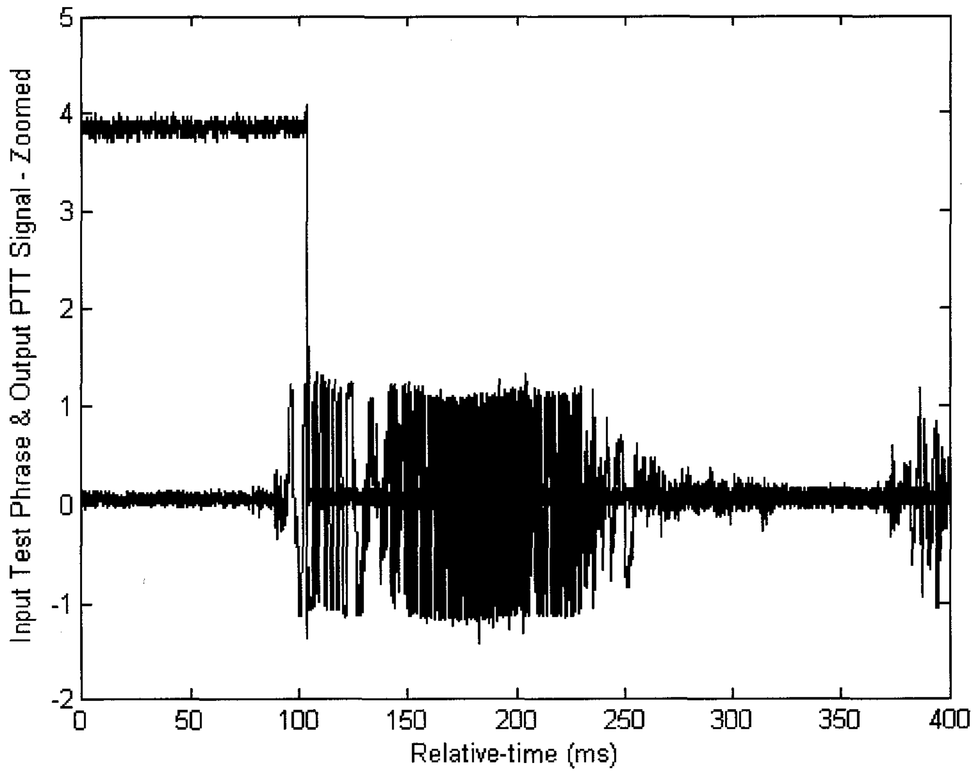


Figure 9.18 – Input Audio Test Phrase and Output PTT Signal - Zoomed

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

As can be clearly seen in Figures 9.17 and 9.18 the loss due to clipping in the phrase will be minimal and should be barely noticeable if at all. Based on these analyses it can be concluded that the PTT detection circuit performs quite well and that the impact to public safety communication will be minimal. (It is important to note that these tests were conducted for unloaded PTT output signals. The effect of this becomes important in the system analysis.)

Connection and Signal Level Issues

One of the first design considerations that needed to be made was the base signal level for which the circuits would be designed. Initially, in the Revision A PCB design, the signal level of the in-car microphone was selected as providing the base signal level that would be used throughout the system. This created some significant problems however as the microphone selected, for it simplicity, was a passive microphone. The result was that an audio signal typically had a peak-to-peak voltage of only 200-300 mV. Therefore, at the speaker audio inputs the voltage levels had to be dropped down from the range of 2 voltages to around 200 mV in order to make the signals comparable. This also then meant that the majority of the audio signals running on the circuit board were extremely low voltage and therefore more susceptible to noise interference, especially when signal amplification might later be needed on the output

As a result of these studies it was determined that the design idea of using passive signaling on the circuit boards would inherently lower the SNR between the desire audio signals and digital or external noises. It was then decided that this was unacceptable as it needlessly reduced the performance of the system. The next question was then what,

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

higher, voltage level should be used as the basis for the analog circuits in the design. The analog circuits have an available voltage swing of -5 to +5 volts. However, typical OPAMPs sometimes exhibit clipping of signals, with a +/- 5 volt supply, as low as +/- 3.5 V [51]. Also, some modest safety margins should be included in the system to allow for loud speaking and minor gain variations - either at the radio volume level or even within the system itself. As the EFJohnson radio system used a voltage on its auxiliary port of around 2 volts (peak-to-peak) in a moderately quiet conversation, it was decided that this would be the level used as the base reference.

It also worked well for the passive microphone which would now need an initial gain of approximately 10 in order to be brought up to the same level as the main speaker audios. (Note that the use of a passive microphone was determined to be essential even at the cost of signal levels. This is because using a passive microphone insured that the microphone connected to the interoperability board wouldn't become inoperative, due to a loss of power, if the circuit board were to fail. Recall that the microphone can still be used to access each of the four radios directly, via the by-pass relays, in the event the system does crash.

The other main thing that must be dealt with in the circuit design is signal level normalization, in other words getting all the input and outputs converted to this base voltage level so that they are comparable to one another. This is especially important on the input side although some scaling may also be needed at the outputs. Currently the prototype design voltage scaling is handled by adjusting the resistor values. However, this was only done for proof of concept purposes and to check the levels needed. To assume that different resistors could be soldered in is impractical and would make the

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

system extremely inflexible, again not a desirable option. Therefore, while the implementation of circuit board signal level normalization has been held off for some future revision (Revision D, etc.), some possibilities are discussed here as well as a temporary solution that might be employed with the prototype model.

The current design is able to use hardware signal adapter circuit boards. Essentially composed of a simple amplifier circuit, its main application is on the output side for adjustment of the signal to be fitted to go to a radio microphone input, although it can also be used at the input side to adjust the signal level coming from a speaker. The circuit gain is variable from 0 to 10. The reason for not simply being a 1 to 10 gain circuit is that in this manner it can also be used as an active signal attenuator. The gain variation is accomplished through the use of a potentiometer (variable resistor) that can be adjusted with a small screw driver. Note that power must be supplied from an external source, either the interoperability circuit board or one or more of the radios.

A final solution could involve moving some of the same circuits onto the interoperability board. The solution to signal scaling would then be accomplished by manually adjusting the gain potentiometers until the appropriate voltage levels were reached. However, while having external adapter circuit boards, as with the prototype, means that additional cases and connectors will be needed, this solution may in fact be more desirable. The reason for this is that pre-tuned external adapters could then be supplied for each specific radio type/audio signal level. This would improve the simplicity of the entire connection process which is critical for emergency situations.

Another approach to the signal level problem that was examined was the use of automatic gain control (AGC) circuits to be used for each audio channel input. The

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

advantage to this would be a significant increase in the flexibility of the system as, no matter what the voltage signal levels of a specific radio type, the audio would be scaled to the circuit board base constant level. There are some problems that arise and some of them are examined here for possible development in future work.

Specific to most hardware based automatic gain control operations is the need for a great deal of additional hardware to implement the necessary control loop. A number of basic gain control circuits were examined, however most became fairly involved requiring a variable gain amplifier, some form of detector, a low-pass filter, and finally some form of difference amplifier. This complexity exceeds the introductory scope of this prototype interoperability solution. Of an especially large concern is the idea of gain runaway that could result from an unstable circuit (due to the feedback). As there is always some noise on the audio channels the AGC must have some threshold signal level, which applies across all radio signal levels, after which it will commence normalization. (Also, note that in order to properly implement the system at least four AGC circuits would need to be added to the PCB. This will consume additional space and will also add significantly to the overall system cost.)

The complex circuit designs encompass numerous details including the normalization attack time (response time), hold time, a maximum gains. These issues are also common to software automatic gain control or signal normalization and as such have been encountered in previous software gateway approaches for their signal normalization routines prior to mixing [24].

However, in most of these cases the assumption is made that the AC audio signal should be maintained at a constant output level. This however is not always the case and

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

attention should be given to intentional signal level deviations, such as silence and shouting, and how they are dealt with. Also, if the attack is too steep the audio can be severely impacted. This would then require a more complicated analysis of how the signal as speech propagates through the system. Note for example the case of a small noise of 100 mV appearing on a speaker audio input. This may then be tracked and amplified by the AGC until it is large enough to trip the PTT and be broadcast on all the output channels. The only solution would be to have an additional detection circuit that would block the AGC whenever the signal was below a threshold, for example 200 mV or a Voice Activity Detection circuit (VAD) [52]. Also, for another sample case, consider that during a brief pause in user speech, the AGC circuit would start to amplify the background noise. Then, when speech was resumed the gain would be too high (unless the attack was very strong which then means the noise would also have been amplified too much). The net result of these discussions is that use of a hardware AGC is very sophisticated and involves numerous aspects that could be pursued as a separate problem in itself. The main concern is that during speech the gain should be held constant (implying a slow attack). However, too slow an attack and the first words of a low-voltage audio signal would always be missed. Therefore, while some advanced (typically software based) solutions do exist [53], a fixed gain amplifier, that can be adjusted as with a potentiometer really appears to be the best solution for this low-cost, low-complexity interoperability approach.

A key component of signal level research that must still be conducted is determining the specific audio connection aspects of some of the more common radio types. This is mostly concerned with determining the correct voltage levels that need to

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

be used with each radio type – both in terms of what needs to be presented to the audio inputs and what can be expected at the audio speaker outputs. Note that some systems, such as the EFJohnson radio system used in many of the prototype test, provide an auxiliary port. These can be greatly beneficially in terms of simplifying the connections however the voltage output levels may vary between brands. Analysis should also be made as to whether the I/O impedances become important to consider for some radios types, especially if using the speaker output ports. The operations should largely be conducted on an individual radio basis as public safety demand for each type is created. The end result should be some form of connection harness that mates with the appropriate radio system connectors as well as some voltage gain settings for the hardware. (Note that the current prototype design uses screw down terminal blocks that allow for easy connection of wires to the system I/O contacts. This avoids the difficulties associated with the fact that there currently is no universal MIC/Radio connection for all the brands. It is however strongly advised that detailed connection diagrams, for supported radios types, be provided in some form of a user's manual. It might also be beneficially to transition to some form of quick-connect terminal instead of a screw down.)

Future Design Ideas and Enhancements

A major focus of the approach taken in this research has been to provide a low-cost, low-complexity solution that also exhibits performance result that are comparable to high end systems. Having observed some of the successes of the overall design, consideration should also be made as to how it can be expanded to a larger scale, possibly for deployment with larger more complex departments/municipalities. Future work

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

should examine the possibility of creating a large MRCM PCB that would provide the opportunity of incorporating more meetings and more radios. A possible approach that would also add some desirable flexibility would be the use of motherboard type designs that would contain the meetings. Daughter circuit boards for individual radios could then be plugged into it making this system customizable depending on the size of the interoperability event. If done properly this would greatly simplify the layout process and allow for easy expandability. Some ideas relating to the scalability of any interoperability design are discussed in [19].

One potential difficulty that might be associated with this is the mapping of the PTT signals as the simple CPLD would be insufficient. To implement the same design with a larger group of radios would significantly increase the overhead and might raise some issues with regards to propagation delays and signal fan-out. Options such as open collector circuits that might allow for shared driving of the meeting PTT signals and reduce the needed logic also should be examined.

Consideration must also be made as to how the current design can fit into the larger interoperability picture. Examination can be made into the creation of, or technical specification of, special access ports providing connections to the Meeting PTT and Audio lines. These might then be used for connecting PC/VoIP based solutions for expansion in order to also leverage some of their benefits [22, 10].

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER X

SYSTEM MOCKUP TESTING

The Setup of the Test

To test the basic functionality and performance of the system, an in-laboratory mockup was created that would simulate a scenario were two radio systems and PC based-communications system needed to communicate with one another but, with the current hardware, were unable. The radio systems used were constructed of the EFJohnson 5300 series and the Motorola Astro. Note that two of each type of radio were used in the testing; one for the mock control vehicle and the other for the alleged remote site. The PC provided an audio feed from its sound card and its speaker also served as a channel output destination. (The fourth device channel was connected only to a speaker for outputting of audio.) A block diagram of the test system is shown in Figure 10.1.

The EFJohnson 5300 radio used at the "command vehicle" has a Control Head Auxiliary Connector on the back that allows for direct tapping into some of the audio and control lines of the radio. The connection is via a DB-15 connector with the pin-outs as shown in Table 10.1. Note that the I/O references are from the perspective of the radio. No signal level scaling was necessary for the EFJohnson as it was used as the reference voltage level for the entire circuit design.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

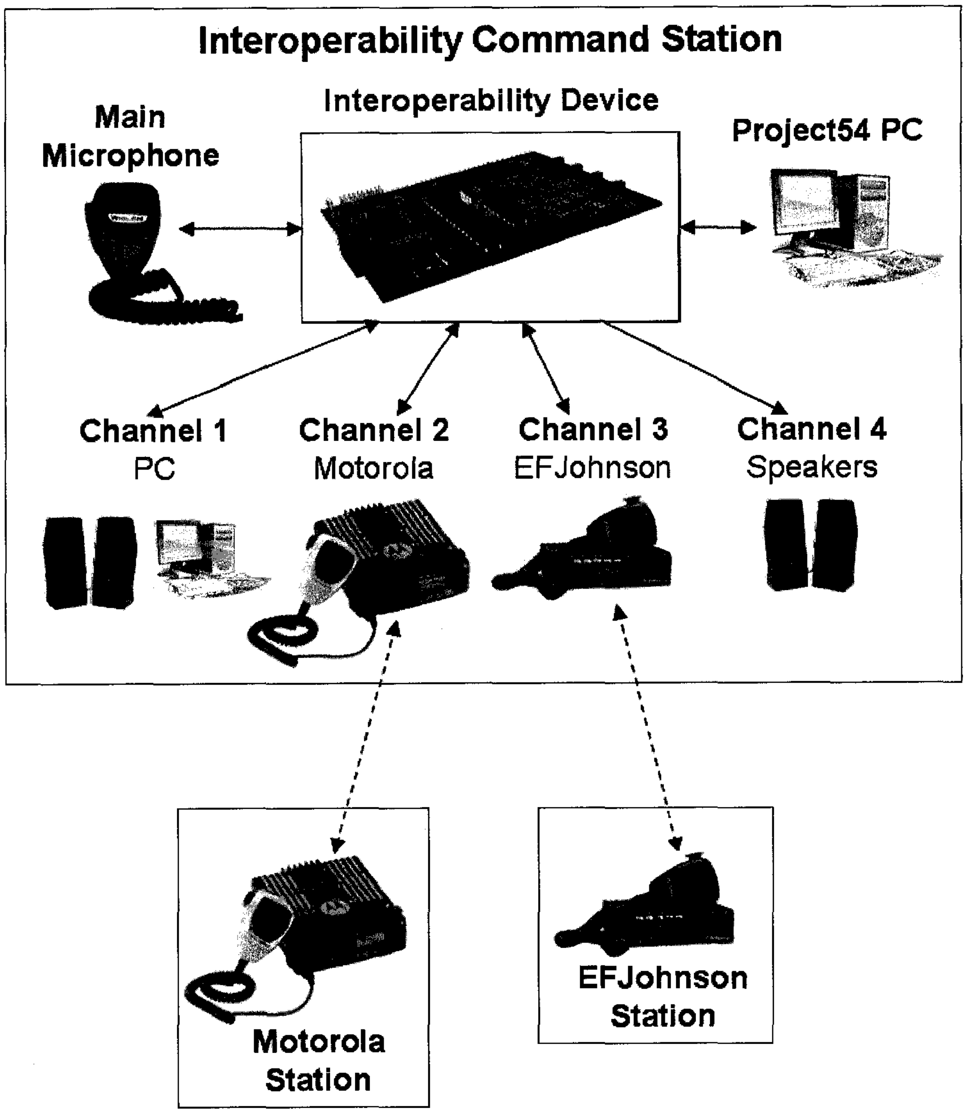


Figure 10.1 – Block Diagram of the Mockup Testing System

The harness wire colors can be used to identify the correct lines on the harness created for connecting the EFJohnson radio to the interoperability circuit board. Table 10.2 lists the same wire colors but with the named connections from the perspective of the interoperability circuit board and using the notation followed throughout this work. Note also that the PWR (Power) and GND (Ground) connections are not used by the PCB but are made available in the event an audio adapter board must be used and powered off the radio.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Contact Name	**DB-15 Pin #**	**Harness Wire Color**
Speaker +	5	Orange
Speaker -	6	Orange/White
Audio In +	1	Blue
Audio In -	2	Blue/White
PTT In	3	Brown
PTT GND	14	Brown/White
GND	14	Green
PWR	11	Green/White

Table 10.1 – EFJohnson DB15 Harness Connections

Contact Name	**Harness Wire Color**
Speaker + Input	Orange
Speaker - Input	Orange/White
Audio Out +	Blue
Audio Out -	Blue/White
PTT Out	Brown
PTT GND	Brown/White
N/A	Green
N/A	Green/White

Table 10.2 – Interoperability Circuit Board Harness Connections

The Motorola Astro radio also has a Remote Mount Accessory Connector located in the rear. The connection is made via a DB-25, the pin-out of which is provided in

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Table 10.3. Note that the same harness wire coloration scheme as shown in Table 10.2 is used here.

Contact Name	**DB-15 Pin #**	**Harness Wire Color**
Speaker +	25	Orange
Speaker -	24	Orange/White
Audio In +	8	Blue
Audio In -	10	Blue/White
PTT In	1	Brown
PTT GND	10	Brown/White
GND	10	Green
PWR	20	Green/White

Table 10.3 – Motorola DB25 Harness Connections

Since the Astro is not necessarily at the same signal level as the EFJohnson it was possible that some signal scaling would be required. However, the Astro has a number of different inputs and outputs that can be selected from and they were chosen so that the signal levels are the same, or nearly the same, as those for the EFJohnson. Any differences are negligible and not noticeable. Note that for the output from the radio the differential speaker outputs were used (versus the Received Filtered Audio Out connections). This is the same as was performed for the EFJohnson and as they both are designed to drive typical speaker loads their signal levels were comparable. For the return to the radio the Auxiliary Audio Input was used. This input is tolerant of the EFJohnson signal level and appeared to work well without any additional scaling. Other options for the input could include a microphone input that has a much lower voltage

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

range or another, comparable audio input line. However, this one appears to perform the best without any additional signal scaling and therefore is employed.

Test Scenarios

Table 10.4 presents some of the testing scenarios employed and the audio outputs that resulted for the specified configurations. Note that the connection table specifies which meeting the channel is connected to. Also, a '-' means that there is no connection or audio present. X and Y are used to denote some arbitrary audio streams that enter via the channel inputs. These should then be passed along to the connected outputs as well. (Note that Channel 4 is an output only channel because it is just connected to a speaker.)

Test Number	Channel Connections					Channel Input Audio						Channel Output Audio			
	M	1	2	3	4	M	1	2	3	4		1	2	3	4
1	A	A	A	A	A	X	-	-	-	-		X	X	X	X
2	A	A	A	B	B	X	-	-	-	-		X	X	-	-
3	B	A	A	B	B	X	-	-	-	-		-	-	X	X
4	-	A	A	A	A	-	X	-	-	-		X	X	X	X
5	-	A	A	B	B	-	-	X	Y	-		X	-	-	Y
6	-	A	A	A	-	-	-	X	-	-		X	-	X	-
7	B	-	B	B	-	X	Y	-	-	-		-	X	X	-
8	A	B	B	B	B	-	-	-	X	-		X	X	-	X
9	A	-	B	-	-	X	-	Y	-	-		-	-	-	-
10	B	-	A	A	B	X	-	-	Y	-		-	Y	-	X

Table 10.4 – Test Scenario Results

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The system performed as it was intended with no faulty connects or bleeding through between meetings observed. Note however, that a channel that is sourcing an audio stream will have that stream presented on that channel output of the interoperability board. With the case of a radio this is not a problem since the PTT signal is not activated. This is shown in the diagram for the two radios (Channels 2 and 3) by simply recording the Channel Output Audio at the output of another radio in the same system. However, this is not possible for the PC. Therefore, as in Test #4, the audio presented back on the output is the same as the original input. This might prove a problem if using a PC based system for interfacing as they typically do not look for a PTT signal but simply detect the presence of audio. A solution to this problem would be to implement another output analog switch that is controlled by that channel's output PTT signal. With four channels this would only require the addition of one quad-package IC.

Appendix G contains some images of the test setup and also includes a reference to an available video demonstration of the mockup test. Note that the testing was performed at three different stations where three operators are able to communicate together over the system. The perspective supplied is that from the view of the main personnel in charge of the interoperability device.

PTT Latency Problems

Over the course of testing the complete interoperability system with two radios attached, a problem was observed. The clipping at the beginning of phrases was noticeable, often cutting off the entire first syllable or word. Based on the PTT testing,

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

this should not have been occurred, and therefore additional testing of the complete system, involving all the radios, was conducted.

First note that the testing from Chapter IX measured the PTT latency of the arrival of an audio signal at the interoperability circuit board's Differential Audio Speaker Input to the PTT Outputs also on the interoperability circuit board. However, in the Mockup testing, the PTT signal is passed from one remote radio to another local radio. That radio's audio then enters the interoperability board and trips the PTT generation circuitry, which then queues the local radio from the other system for passage to the other system's remote radio. The concept is illustrated in Figure 10.2.

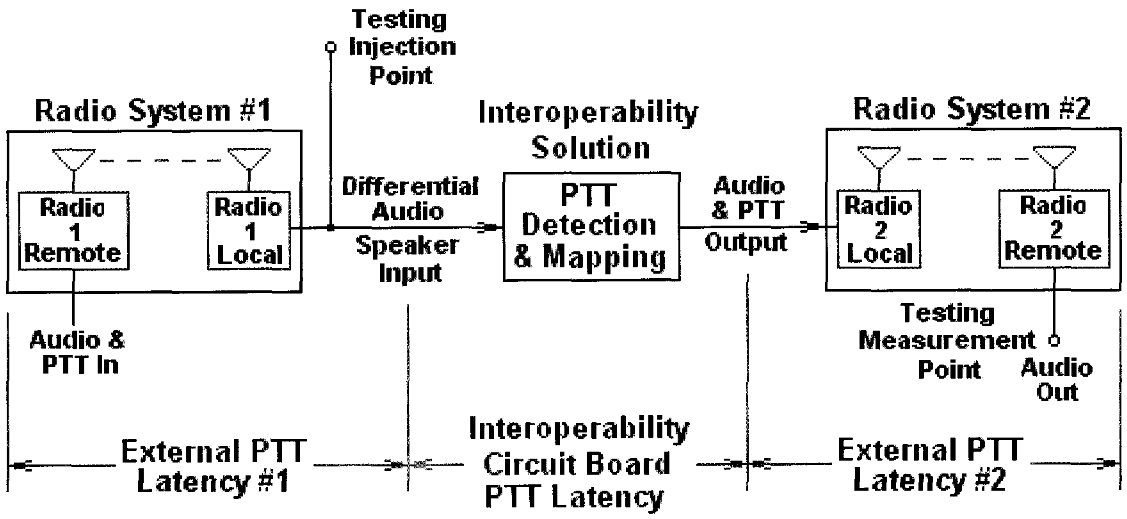


Figure 10.2 – Diagram of Potential PTT Latency Sources

Note that the Interoperability Circuit Boards PTT Latency is shown. Also, by listening to the audio coming out of the local radio to the first radio system and observing no clipping at the start of the phrase, External PTT Latency #1 can be assumed negligible. Therefore, the bulk of the PTT latency must be assumed to be coming from the External PTT Latency #2. (Note swapping the order of the radio systems does

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

nothing to correct this PTT latency problem, therefore the problem isn't unique to connecting up to a particular radio brand.)

First the observed, complete interoperability system PTT latency needed to be quantified. This would be difficult however as it involved three testing sites (one for each remote radio and one for the main base station) so instead only the Interoperability Circuit Board PTT Latency + the External PTT Latency #2 latency was examined (involving the main base station and one external radio). This quantification of the PTT latency was accomplished by sending ten separate variations of the word "testing" into one of the Differential Audio Speaker Inputs on the interoperability board and then recording the output audio at the remote radio. The injection and test measurement points are also shown in Figure 10.2.

The two sound files that were saved, both the original one transmitted and the received one, could then be examined to determine exactly how much of the audio of the word "testing" was clipped off. Note that the word "testing" was used as it is fairly long. Shorter words, like "one", were in many cases completely lost and comparison thus made impossible. However, the use of a full sentence may make it easier to synchronize up the two sound files as, with the radio channel noise, this can be difficult.

Table 10.5 presents the estimated delays associated with the ten different trials along with an average PTT latency. The Interoperability Circuit Board PTT Latency + External PTT Latency #2 ranged from 213.5 ms all the way up to 353.0 ms. Note that these estimates are fairly crude however as time synchronization was not available. Considering that the Interoperability Circuit Board (internal) PTT Latency was around 15 ms this means that the External PTT Latency #2 is around 200 to 350 ms with an average

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

of around 275 ms. This is very significant, especially when compared to the small PTT latency of the designed PTT detection circuit, which was originally presumed to be the main thing that this design had any control over when using the current PTT approach.

Trial	**PTT Latency**
Trial #1	300.2 ms
Trial #2	274.5 ms
Trial #3	353.0 ms
Trial #4	263.7 ms
Trial #5	342.1 ms
Trial #6	288.4 ms
Trial #7	347.0 ms
Trial #8	231.0 ms
Trial #9	305.0 ms
Trial #10	213.5 ms
Average	291.8 ms

Table 10.5 – Interoperability Circuit Board & External #2 PTT Latency

Figure 10.3 shows the input and output audio for the ninth trial. Note that the amplitude is largely unimportant since the recording will have some attenuation from the microphones and speakers. Also, the two waveforms are not time synchronized as they were recorded on different PCs. More importantly it should be note that the input contains two distinct syllables - "test" and "-ing". However, only a part of the "-ing" actually appears in the output. The entire "test" syllable has been clipped off. For this trial the estimate is that 305.0 ms in all were lost before the remote PTT was activated.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

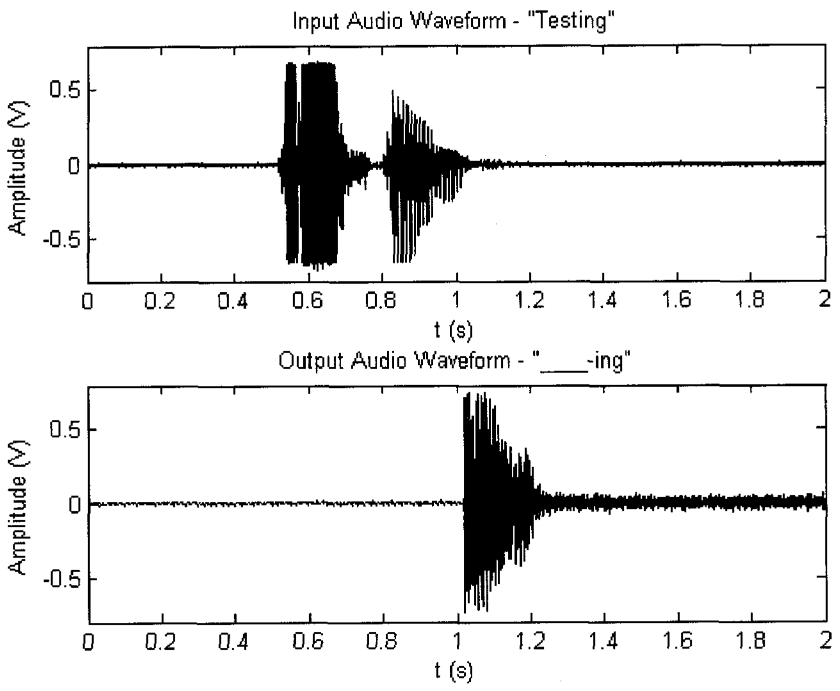


Figure 10.3 – Ninth Trial Input and Output Audio

Currently this problem, which appears to be an issue with the interoperability circuit board and the way it interfaces to the radio PTT inputs, has not been resolved. Also, while it does not make this solution unusable, it certainly impairs the performance. For this reason, it would be desirable for future designs to examine ways to work around this and determine why it is occurring.

One possible explanation is that the radios digitally sample the PTT input on the microphone port at some given interval (i.e. every 150 ms). Thus, if the PTT signal goes high during this time, the radio may not detect it until some brief time later. (This also might not be observed when using a manual microphone PTT as the 150 ms is quite small and officers are trained to press the button, pause briefly, and then commence speaking.) Based on the fact that all the delays are consistently around 250 ms however this seems unlikely as some should appear less then the sampling rate depending on when the queue occurs in the cycle. This has also been support by some testing on the radios themselves

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

which have found the radio PTT latencies to be around 115 ms for the Motorola and 134 ms for the EF Johnson. With an average latency of around 275 ms, 150 ms still remains unaccounted for in the current explanation. This performance concern will necessarily involve additional study.

Another explanation is in the applied signal levels. Note that for the measurement of the internal interoperability circuit board PTT delays the audio input signals were at an amplitude of around 2.5 V peak-to-peak (for both the tones and test phrases). However, when applied within the system testing they were at only around 1.5 V (peak-to-peak). This reduced amplitude may have caused the rise time to be longer in the PTT detection circuitry, although this should be relatively insignificant based on the design involving a selectable threshold, digital comparator. Some further examination of the operation of the design may be called for.

Another option is that the PTT outputs, which are currently direct driven by the CPLD, are insufficient for driving the radio PTT inputs as they cannot supply the necessary current levels. This might be tested by observing the PTT signal output of the loaded circuit board while running the system tests and comparing the PTT delays to those observed at the audio outputs. Potential fixes would necessarily involve a new board revision that incorporated some form of high-current buffering for the CPLD. It is essential however that this issue be resolved prior to any small-scale field application tests of the design in a public safety deployment.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

CHAPTER XI

CONCLUSION

As demonstrated in the testing scenarios, the designed mixed-signal hardware device performs very well as a System Gateway, Public Safety Radio Interoperability prototype solution. Good performance is observable on all of the channel outputs with virtually no audio latency, a result of which is coherent conversation between the involved parties. Furthermore, there appears to be no audibly significant and noticeable end-to-end distortion or noise present on the audio signals other than what is typically incurred over public safety radio transmissions. The main flaw, which must be corrected prior to full test deployment, is the PTT latency which must be further reduced to avoid clipping of initial syllables.

Perhaps more important in many regards, from the perspective of the intended small-department public safety market, the system is also very simply to operate and very inexpensive. Control operations are performable via the manual control head or through the Project54 software application connected through the serial port. This diversity of control options ensures that the system is flexible and can be used in a multitude of situations. Addressing the concern of limited funding the design also has a very small price tag of around $100 in low volumes, a significant improvement considering the level of performance still attained.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Lastly the system has been shown to be robust in its performance. This is critical for the intended applications. Any failure on the part of the system should not also bring down the larger public safety communication systems. Such an event could prove disastrous and has been shown to have been successfully avoided. However, the prototype has not yet been tested for in-car environments, an important factor considering the target use. This must also be the continued subject of future studies.

The need for public safety personnel from different departments to be able to communicate with each other when the need arises is absolutely essential, be it in the event of a natural disaster or some other man-made incident. While the System Gateway approach to enabling this communications has some drawbacks, as a stop-gap measure for immediate use it presents some of the greatest rewards for the imposed cost. Personnel can still employ the same equipment that they use on a daily basis yet speak with persons from other departments and agencies. By meeting the basic demand of public safety radio interoperability, this device can help, not only to make the job of the public safety personnel easier, but save lives as well.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

REFERENCES

[1] "P54 Overview – About Project54". Online. CATLab – Project54 Website. 2007. Available: http://www.project54.unh.edu. Consolidated Advanced Technologies for Law Enforcement Program (CATLab), University of New Hampshire. Durham, NH.

[2] Kun, A. L.; Miller, W. T.; Lenharth, W. H. "Modular System Architecture for Electronic Device Integration in Police Cruisers". *Proceedings of the 2002 IEEE Intelligent Vehicle Symposium.* Versailles, France. June 18-20, 2002.

[3] Rep.Jim Nussel (Iowa). "Title III - Digital Television Transition and Public Safety Act of 2005: Sec. 3006 - Public Safety Interoperable Communications". Congressional Record (No. 164 - Book 2). House Conference Report: 109-362. H12646 (Daily Edition). December 18, 2005. Government Printing Office.

[4] "Sago Mine Information Single Source Page". Online. U.S. Department of Labor: Mine Safety and Health Administration Website . February 3, 2006. Available: http://www.msha.gov/sagomine/sagomine.asp. USDL - MSHA.

[5] "Fury over US mine 'rescue' fiasco". Online. BBC News Website [UK]. January 4, 2006. Available: http://news.bbc.co.uk/1/hi/world/americas/4579754.stm. BBC.

[6] "Why Can't We Talk? Working Together To Bridge the Communications Gap To Save Lives". February, 2003. National Task Force on Interoperability.

[7] "Guide to Radio Communications Interoperability Strategies and Products". TE-02-02. April 1, 2003. Agile.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

[8] "Why Can't We Talk? Working Together To Bridge the Communications Gap To Save Lives: Supplemental Resources". February, 2003. National Task Force on Interoperability.

[9] Taglang, K. "The Digital Television Transition and Public Safety Act of 2005". January 6, 2006. Washington, DC. The Benton Foundation.

[10] Elhart, Ivan; Bond, Timothy E.; Kun, Andrew L.; Miller, W. Thomas III. "Prototype System for Radio Patching in a Mobile Environment". Technical Report ECE.P54.2007.6. March, 2007. Project54, CATLab, Electrical and Computer Engineering Department, University of New Hampshire. Durham, NH.

[11] Lund, D. A. "Learning To Talk: The Lessons of Non-Interoperability in Public Safety Communications Systems". April, 2002. Durham, New Hampshire. ATLAS Project, University of New Hampshire Justiceworks.

[12] Dwyer, J. "Before the Towers Fell, Fire Dept. Fought Chaos". *The New York Times*. January 30, 2002. New York City, New York. The New York Times Company.

[13] Kopytoff, V. "Communications severely tested, Calls overload telephone networks, some equipment destroyed". *The San Francisco Chronicle*, page 2. September 12, 2001. San Francisco, California. Hearst Communications Inc.

[14] "First Report and Order: Authorization and Use of Software Defined Radios". FCC 01-264. November 14, 2001. Federal Communications Commission.

[15] Uehara, K.; Araki, K.; and Umehira, M. "Trends in Research and Development of Software Defined Radio". *NTT Technical Review*, vol. 1, issue 4. July, 2003.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

[16] Glossner, J., Moudgill, M., Iancu, D., Nacer, G., Jintukar, S., Stanley, S., Samori, M., Raja, T., and Schulte, M. "The Sandbridge Sandblaster Convergence Platform". 2005. Sandbridge Technologies.

[17] Chapin, J., Bose V. "The Vanu Software Radio System". 2002 Software Defined Radio Technical Conference. February, 2002.

[18] Ryser, P. "Software Defined Radio with Reconfigurable Hardware and Software: A Framework for a TV Broadcast Receiver". Embedded Systems Conference 2005 - San Francisco. San Francisco, California. March 10, 2005.

[19] Cox, M. W. "Handbook of Patchwork Interoperability: a complete (non-technical) primer". June, 2004. Raytheon Corp., JPS Communications.

[20] "Products & Applications: Products & Applications Overview". Online. Raytheon, JPS Communications Website. March, 2006. Available: http://www.jps.com/index.asp?node=118. Raytheon Corp., JPS Communications.

[21] Careless, J. "Charlottesville's Multiple Interoperability Solutions". *Law and Order*. November, 2005. Hendon Publishing Co.

[22] Mock, John H. "A Voice Over IP Solution to the Problem of Mobile Radio Interoperability". Master of Science. University of New Hampshire. May, 2003.

[23] Shah, A.; Nimmer, J.; and Franklin, D. "A Prototype All-Software Public Safety Interoperability System". 2004 Software Defined Radio Technical Conference. Phoenix, Arizona. November, 2004.

[24] Bouin, A.; Lengrand, B. “Audio Switch”. Technical Report ECE.P54.2006.6. June, 2006. Project54, CATLab, Electrical and Computer Engineering Department, University of New Hampshire. Durham, NH

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

[25] "XC9536 In-System Programmable CPLD Product Specification". Publication DS064 (v6.3) . April, 2006. Xilinx, Inc.

[26] "AT89C51RB2/RC2 8-bit Microcontroller with 16K/32K Bytes Flash Product Specification". Revision 4180E-8051-10/06. October, 2006. Atmel Corporation.

[27] Bennetts, R.G. "Boundary Scan Tutorial, Version 2.1" . September, 2002. ASSET InterTech, Inc.

[28] "Xilinx In-System Programming Using an Embedded Microcontroller". Application Note: Xilinx Families XAPP058 (v3.1) . June, 2004. Xilinx, Inc.

[29] "Using Serial Vector Format Files to Program XC9500/XL/XV Devices In-System". Application Note: XC9500/XL/XV Family XAPP067 (v2.0). May, 2002. Xilinx, Inc.

[30] "XAPP058 Reference C Code v5.01". Online. Supplementary Webpack v5.01 to Application Note XAPP058. October, 2004. Available: ftp://ftp.xilinx.com/pub/swhelp/cpld/eisp_pc.zip. Xilinx, Inc.

[31] "Serial Vector Format Specification, Rev. E". Part Number: ASSET-SVF-DOC. March, 1999. Texas Instruments and ASSET InterTech, Inc.

[32] "IEEE Standard Test Access Port and Boundary-Scan Architecture". *IEEE Standard 1149.1-2001*. 2001. Institute of Electrical and Electronics Engineers, Inc.

[33] "MC7805CT Three-Terminal Positive Voltage Regulators, Revision 2". Product Specification. May, 2006. Micro Commercial Components.

[34] "MC7800, MC7800A, NCV7805 1.0 A Positive Voltage Regulators, Rev. 16". Product Specification. Publication Order Number: MC7800/D. April, 2007. ON Semiconductor, Semiconductor Components Industries, LLC.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

[35] Brooks, Douglas. "Brookspeak - Crosstalk, Part 1: The Conversation We Wish Would Stop!". *Printed Circuit Design.* November, 1997. Miller Freeman, Inc. and UltraCAD Design, Inc.

[36] Brooks, Douglas. "Brookspeak - Crosstalk, Part 2: How Loud Is Your Crosstalk?". *Printed Circuit Design.* December, 1997. Miller Freeman, Inc. and UltraCAD Design, Inc.

[37] Vittal, Ashok; Marek-Sadowska, Malgorzata. "Crosstalk Reduction for VLSI". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* Vol. 16, No. 3. March, 1997.

[38] "Basic Principles of Signal Integrity". White Paper WP-SGNLNTGRY-1.2, ver. 1.2. July, 2004. Altera Corporation.

[39] "8.1i CPLD, CPLDFit - How do I adjust the slew rate for outputs?". Online. Xilinx Answer Database, Answer Record 983. April, 2006. Available: http://www.xilinx.com/xlnx/xil_ans_display.jsp?getPagePath=983. Xilinx, Inc.

[40] "XC9500 In-System Programmable CPLD Family Product Specification". Publication DS063 (v5.4). April, 2006. Xilinx, Inc.

[41] Brooks, Douglas. "Crosstalk Coupling: Single-Ended vs. Differential". Technical Publication. September, 2005. Mentor Graphics Corporation and UltraCAD Design, Inc.

[42] Smith, Wade. "A Study of the Electromagnetic Compatibility of Integrated Circuits". 2000. Technical Report. School of Electrical Engineering & Computer Science, University of Central Florida.

[43] Rizvi, Syed; Khan, Imran. "In the Mix: Tips for Mixed-Signal Design". *Printed Circuit Design & Manufacture.* February, 2005. UP Media Group, Inc.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

[44] Sienski, Ken; Field, Calvin; Schreiner, Clint; Chivers, Mark. "Mixed Signal digital Sub-Band Tuner Multichip Module". *MCMC-96 Proceedings*. 1996 IEEE Multi-Chip Module Conference (MCMC '96). February, 1996.

[45] Ott, Henry W. "Partitioning and Layout of a Mixed-Signal PCB". *Printed Circuit Design*. June, 2001.

[46] Edwards, Timothy R.; Strohbehn, Kim; Jaskulek, Steven E.; Katz, Richard. "Analog Module Architecture for Space-Qualified Field-Programmable Mixed-Signal Arrays". MAPLD 1999 Conference. September, 1999

[47] Reiser, Cornel; Znamirowski, Lech; Palusinki, Olgierd A.; Vrudhula, Sarma B. K.; Rakhmatov, Daler. "Dynamically Reconfigurable Analog/Digital Hardware - Implementation Using FPGA and FPAA Technologies". October, 1998. *Journal of Circuits, Systems and Computers*. World Scientific Pub.

[48] "Nano-interconnects, Devices and Circuit Simulation Laboratory". Online. NDCSL website. 2007. Available: http://www.ece.arizona.edu/~cmsl/. University of Arizona.

[49] Warecki, Sylwester; Palusinki, Olgierd A.; Vrudhula, Sarma B. K.; Mensch, William D. "Mixed Signal Analog/Digital Board". MIXDES 2000. Gdynia, Poland. June, 2000.

[50] “One-way Transmission Time”. ITU-T Recommendation G.114. May, 2000. International Telecommunication Union.

[51] "LM324, LM324A, LM224, LM2902, LM2902V, NCV2902 Single Supply quad Operational Amplifiers, Rev. 20". Product Specification. Publication Order Number: LM324/D. October, 2006. ON Semiconductor, Semiconductor Components Industries, LLC.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

[52] "GIPS Automatic Gain Control: Consistent Audio Level with GIPS AGC". Product Specification. 2007. Global IP Solutions.

[53] "Automatic Gain Control - C54X". Datasheet. January, 2002. Adaptive Digital Technologies, Inc.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

APPENDICES

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

APPENDIX A

XILINX CPLD VHDL CODE

The code used for the Xilinx CPLD is in VHDL. It is done using the Xilinx ISE Project Navigator version 8.2i. While the ISE environment produces a number of sub-files during its building process the complete VHDL code created and used for the implementation is actually quite short and only embodies simple mapping equations for the PTT signals and some additional logic. The main code is in the PTT Map behavior file (PTT_map.vhd). This defines the overall mapping structure used for the PTT signals. The Meeting behavior and Channel behavior files contain the logic equations that govern the generic Meeting PTT signal and the generic meeting PTT derived output signal respectively. The constraints file, ptt_map.ucf, provides the pin constraints that fix the pin-out of the PLCC-44 package CPLD to match the design schematic. All four of these files are listed below. Note that the code must be compiled and applied for use by the Xilinx XC9536 CPLD. This is best accomplished using the Xilinx ISE software program and can be set to output in SVF or XSVF format, among other options.

'PTT_map' Behavioral Definition File - PTT_map.vhd

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity PTT_map is
   Port (
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
      -- Required Inputs
      -- Channel A Connection Controls, active high, inputs
      C1A, C2A, C3A, C4A                     : in std_logic;

      -- Channel B Connection Controls, active high, inputs
      C1B, C2B, C3B, C4B                     : in std_logic;

      -- Master MIC Connection Controls, active high, inputs
      CMB, CMA                               : in std_logic;

      -- Signal Level Detection (PTT), active high, inputs
      Det1, Det2, Det3, Det4                 : in std_logic;

      -- Internal/External Master MIC PTT, active low, inputs
      PTTMint, PTTMext                       : in std_logic;

      -- Channel PTT MIC, active low, inputs
      PTT1in, PTT2in, PTT3in, PTT4in         : in std_logic;

      -- Required Outputs
      -- Derived Master MIC PTT Signal, active low, output
      PTTMin                                 : inout std_logic;

      -- Inverted Channel PTT MIC Inputs, active high, outputs
      nPTT1in, nPTT2in, nPTT3in, nPTT4in     : out std_logic;

      -- Channel PTT Output, active low, outputs
      PTT1out, PTT2out, PTT3out, PTT4out     : out std_logic;

      -- Optional/Reference Outputs
      -- Meeting A PTT, active high, reference output
      PTTA                                   : inout std_logic;

      -- Meeting B PTT, active high, reference output
      PTTB                                   : inout std_logic
   );
end PTT_map;

architecture behavior of PTT_map is
   -- signal PTTA, PTTB                      : std_logic;

   -- Declaration of type 'meeting' component
   component meeting
      Port (
         Control, PTTM                       : in std_logic;
         C1, C2, C3, C4                      : in std_logic;
         Det1, Det2, Det3, Det4              : in std_logic;
         Meeting                             : out std_logic
      );
   end component;

   -- Declaration of type 'channel' component
   component channel
      Port (
         CA, CB                              : in std_logic;
         PTTA, PTTB                          : in std_logic;
         Det                                 : in std_logic;
         PTTin                               : in std_logic;
         Channel_PTT                         : out std_logic
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
      );
    end component;

    -- Declaration of signals used
    -- none --

begin
    -- Determination of Misc. Logic Outputs for General Use
    PTTMin <= PTTMext AND PTTMint;
    nPTT1in <= NOT( PTT1in );
    nPTT2in <= NOT( PTT2in );
    nPTT3in <= NOT( PTT3in );
    nPTT4in <= NOT( PTT4in );

    -- Determination of Meeting PTT Signals
    PTT_Meeting_A: meeting port map (
       Control => CMA, PTTM => PTTMin,
       C1 => C1A, C2 => C2A, C3 => C3A, C4 => C4A,
       Det1 => Det1, Det2 => Det2, Det3 => Det3, Det4 => Det4,
       Meeting => PTTA
    );
    PTT_Meeting_B: meeting port map (
       Control => CMB, PTTM => PTTMin,
       C1 => C1B, C2 => C2B, C3 => C3B, C4 => C4B,
       Det1 => Det1, Det2 => Det2, Det3 => Det3, Det4 => Det4,
       Meeting => PTTB
    );

    -- Determination of Channel PTT Outputs
    PTT_Channel_1: channel port map (
       CA => C1A, CB => C1B,
       PTTA => PTTA, PTTB => PTTB,
       Det => Det1,
       PTTin => PTT1in,
       Channel_PTT => PTT1out
    );
    PTT_Channel_2: channel port map (
       CA => C2A, CB => C2B,
       PTTA => PTTA, PTTB => PTTB,
       Det => Det2,
       PTTin => PTT2in,
       Channel_PTT => PTT2out
    );
    PTT_Channel_3: channel port map (
       CA => C3A, CB => C3B,
       PTTA => PTTA, PTTB => PTTB,
       Det => Det3,
       PTTin => PTT3in,
       Channel_PTT => PTT3out
    );
    PTT_Channel_4: channel port map (
       CA => C4A, CB => C4B,
       PTTA => PTTA, PTTB => PTTB,
       Det => Det4,
       PTTin => PTT4in,
       Channel_PTT => PTT4out
    );
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
end behavior;
```

'Meeting' Component Behavioral Definition File

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity meeting is
   Port (
      Control, PTTM               : in std_logic;
      C1, C2, C3, C4              : in std_logic;
      Det1, Det2, Det3, Det4      : in std_logic;
      Meeting                     : out std_logic
   );
end meeting;

architecture behavior of meeting is
begin
   -- Definition of Meeting's PTT behavior (active high)
   Meeting <= (C1 AND Det1) OR (C2 AND Det2) OR (C3 AND Det3) OR
              (C4 AND Det4) OR (Control AND NOT(PTTM));
end behavior;
```

'Channel' Component Behavioral Definition File

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity channel is
   Port (
      CA, CB        : in std_logic;
      PTTA, PTTB    : in std_logic;
      Det           : in std_logic;
      PTTin         : in std_logic;
      Channel_PTT   : out std_logic
   );
end channel;

architecture behavior of channel is
begin
   -- Definition of Channel's PTT behavior (active low)
   Channel_PTT <= ((NOT( (CA AND PTTA) OR (CB AND PTTB) )) OR
                  Det) AND PTTin;
end behavior;
```

Constraints File - ptt_map.ucf

```
NET "C1A"  LOC = "P40"  ;
NET "C1B"  LOC = "P39"  ;
NET "C2A"  LOC = "P38"  ;
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
NET "C2B"  LOC = "P37"  ;
NET "C3A"  LOC = "P36"  ;
NET "C3B"  LOC = "P35"  ;
NET "C4A"  LOC = "P34"  ;
NET "C4B"  LOC = "P33"  ;
NET "CMA"  LOC = "P20"  ;
NET "CMB"  LOC = "P19"  ;
NET "Det1"  LOC = "P13"  ;
NET "Det2"  LOC = "P14"  ;
NET "Det3"  LOC = "P11"  ;
NET "Det4"  LOC = "P12"  ;
NET "nPTT1in"  LOC = "P6"  ;
NET "nPTT2in"  LOC = "P7"  ;
NET "nPTT3in"  LOC = "P8"  ;
NET "nPTT4in"  LOC = "P9"  ;
NET "PTT1in"  LOC = "P1"  ;
NET "PTT1out"  LOC = "P25"  ;
NET "PTT2in"  LOC = "P2"  ;
NET "PTT2out"  LOC = "P28"  ;
NET "PTT3in"  LOC = "P3"  ;
NET "PTT3out"  LOC = "P27"  ;
NET "PTT4in"  LOC = "P4"  ;
NET "PTT4out"  LOC = "P26"  ;
NET "PTTA"  LOC = "P22"  ;
NET "PTTB"  LOC = "P24"  ;
NET "PTTMext"  LOC = "P5"  ;
NET "PTTMin"  LOC = "P44"  ;
NET "PTTMint"  LOC = "P29"  ;
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

APPENDIX B

MICROCONTROLLER FIRMWARE BOOTLOADER

In order to allow for easy system upgrades, the system was designed so that the microcontroller firmware, that performs the main control and interface operations for the circuit board, is fully upgradeable via the serial port. The following outlines the method for reprogramming the AT89C51RC2 microcontroller through the use of the ATMEL FLIP (**FL**exible **I**n-system **P**rogrammer) software running on a connected PC.

1.) Connect the device to the computer's serial port with a straight serial communications cable (male termination).

2.) Power the device up from an in-specification power supply.

3.) Launch the ATMEL FLIP software (Version 2.4.6 or later).

4.) From the menu select File>Load Hex File...> and then select the correct .hex file for programming that the firmware was saved to (for example MRCM.hex).

5.) From the menus select Device>Select...>AT89C51RC2
(The default buffer size of 32 Kbytes should be used)

6.) Select from the menus Settings>Communications>RS232
This will launch a new RS232 Communications window

7.) Enter the following communications settings:

Port:	COM1	(or the PC communications port used)
Baud:	9600	(or other similarly values - 4800, 19200, etc.)
Manual Sync:	Unchecked	

8.) Place the device in reset mode by connecting the RESET pin to V_{CC} (+5 V) and holding it there.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

9.) Connect the PSEN pin to GND (0 V) and hold it there.

10.) Release from reset by disconnecting RESET from V_{CC}. The microcontroller will now begin running the embedded bootloader program.

11.) After a brief pause release the PSEN pin from GND

12.) Press the Connect Button (in the software) to begin reprogramming. Notification will be given when the reprogramming is complete.

13.) Power down and then re-power the device to hard-reset it. Reprogramming has now been successfully completed!

Note that the FLIP program also comes with an AutoISP feature that will automatically handle the setting of the hardware conditions on the RESET and PSEN microcontroller pins so that they do not have to be done manually with the push buttons. To use this feature the RS232 DTR signal must be connected to the RESET pin on the microcontroller and the RS232 RTS signal must be connected to the PSEN pin. This was not implemented in the current circuit board design and therefore cannot be used. This feature may however be useful to consider for an eventual programming implementation over the IDB. Additional comments on the hardware and software for this process are provided in the Bootloader for In-System Programming Flexibility section of *Chapter VI - Digital Control Hardware & Firmware.*

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

APPENDIX C

MCS51 ASSEMBLY FIRMWARE

The MCS51 Assembly code files for Revision C of the circuit board design can be found in the *MCS51 Assembly MRCM Code* folder in the attached CD. The firmware code files are simply created using a text editor, such as notepad, and written in the MCS51 Assembly language. These are then assembled using an 8051 Cross Assembler. For this project the free Metalink 8051 Macro Assembler by Metalink Corporation (http://www.metaice.com) was used. Table C.1 outlines the contents of the main code folder. Details on the contents of each file are also included.

File Name	Description of File Contents
'Build.bat'	Batch file used to run the Metalink assembler to create the hex file from the other assembly files. (Note that it will need to be edited to target the ASM51.exe program.)
'CommExec.txt'	Assembly code file that deals with the execution of commands. Commands can be ordered from either the manual button interface or via the serial communications port. These functions manipulate the actual lines that control the switch matrix and the resulting audio connections.
'CommProc.txt'	Assembly code file that handles the processing of commands received via the serial port. It involves a large switch to parse the command and then calls the appropriate execution routine(s).

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

'Ext_Int.txt'	Assembly code file that contains the button external interrupt processing routines.
'Inits.txt'	Contains all the assembly initialization code for start-up including aspects for the SFRs, port pins, interrupts, and timers.
'MOD51RX2.txt'	Assembly code file listing the common names for the SFR bytes and bits and linking them to the appropriate addresses.
'MRCM.txt'	Main assembly code file. Contains the start-up routine, primary program loop, and includes of the other code files.
'RAM_DEFS.txt'	Assembly code file allocating 8051 memory space from RAM for use by the firmware. (Only involves assigning usable names to numerical register locations.)
'Serial.txt'	Assembly code file used for managing the 8051's serial communications port. Includes code for initialization, maintenance (byte reception interrupt handling), serial buffer maintenance, data querying, and data transmission.
'String.txt'	Assembly code file containing all of the (null terminated) strings used within the firmware. Note that in most cases the strings are for outputting via the serial communications port.
'Timers.txt'	Assembly code file dealing with the control and use of timer interrupts and delay functions of various duration used for timing.
'ERR_LIST.txt'	Complete, merged code file for the project containing all of the assembly code as well as the machine code translations and code line assignments. It also provides the main means of troubleshooting assembling errors in the code. If errors exist the MRCM.hex file will not be created.
'MRCM.hex'	Final machine code hex file. Note that it is provided in the Intel Hex format and is ready

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

	for loading via the bootloader as discussed in Appendix B. This is the file that must be targeted by that operation.
'Notes/Function Listings.txt'	Notes file that lists the functions that are contained in each assembly code text file. It also lists the super- and sub-functions for each base function.
'XSVF/'	Sub-folder containing the XSVF code files created to send the CPLD code to the 8051 so that it can load it into the CPLD via a JTAG interface. (For folder contents reference Table C.2 below.)

Table C.1 – MCS51 Assembly MRCM Code Folder File Contents

The XSVF/ subfolder contains the 8051 assembly code that deals with the use and control of the JTAG interface port between the 8051 CPU and the Xilinx CPLD. Its purpose is to process the XSVF commands received from the PC and executed them on the JTAG port. Table C.2 lists the code file contents of the folder and provides a brief summary of the purpose of each code file.

File Name	Description of File Contents
'Do_Comm.txt'	Assembly code file containing the specific functions designed to perform particular XSVF commands. Physical implementation is performed through the calling of the general low-level functions in the MiscFunc.txt file.
'MiscFunc.txt'	Assembly code file containing some low-level general functions that are used extensively throughout the XSVF programming process to physically interface the other functions to the JTAG port.
'MOD51FA.txt'	Assembly code file listing the common names for the SFR bytes and bits and linking them to the appropriate addresses.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

'ProcComm.txt'	Assembly code file that handles the processing of each XSVF command received via the serial port during the CPLD programming process. It involves a large switch to determine the XSVF Instruction Set command code and call the appropriate execution routine.
'XSVF_Str.txt'	Assembly code file containing all of the (null terminated) strings used within the XSVF firmware.
'XSVFDEFS.txt'	Assembly code file allocating 8051 memory space from RAM for use by the firmware. (Only involves assigning usable names to numerical register locations.) It also contains the equates for the XSVF Instruction Set command codes from text format to their binary equivalent.
'XSVFIO.txt'	Critical assembly file providing an interface between the XSVF assembly code and the main MRCM project code. It provides some critical generic input points for the main code to link to acting as a bridge between the two. Note that no changes should be needed in the rest of the XSVF code in order to interface to any other main assembly code program.
'XSVFMain.txt	Main XSVF assembly code file. Contains the programming entry point, initialization code, and the main programming loop.
'Notes/Function Listings.txt'	Notes file that lists the functions and sub-functions that are contained in each assembly code text file.

Table C.2 – XSVF Assembly Code Folder File Contents

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

APPENDIX D

CPLD JTAG PROGRAMMING

In order to allow for system flexibility and further facilitate changes that may arise in the prototype design the CPLD was selected so that it was also in-system reprogrammable. The reprogramming operation is handled by the 8051 microcontroller through a simple interface. The Xilinx XC9536 CPLD used for the digital PTT signal mapping, as described in Chapter V, is programmed via a JTAG port. For this design the programming interface is done with a four pin connection - TDI (Test Data In), TDO (Test Data Out), TCK (Test Clock), and TMS (Test Mode Select). The optional connection TRST (Test ReSeT) is not used. The VHDL code loaded onto the device is supplied in Appendix A while the 8051 firmware used to perform the loading of the CPLD is detailed in Appendix C. The following outlines the procedure used to order the CPU to program the CPLD with data received over the serial port. Note that currently a MATLAB script is employed which, while functional, is subject to certain faults that MATLAB has in dealing with the serial port. (For this reason this program should only be used in limited quantities for prototyping.) However it should serve as an excellent guide for the creation of any final CPLD Program Loading software.

1.) **Connect the device to the computer's serial port with a straight serial communications cable (male termination).**
2.) **Power the device up from an in-specification power supply.**

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

3.) Load and run the following MATLAB script on the PC:

```
% Open the CPLD Program File
program = fopen('Rev. C Program/nc_xsvf_program_file.xsvf');

% Start-up and open the COM port
com_port = serial('COM1',...
                  'baudrate', 9600,...
                  'databits', 8,...
                  'stopbits', 1,...
                  'ReadAsyncMode', 'continuous',...
                  'Timeout', 1000)
fopen(com_port);

% Transmit the command to Reprogram the CPLD
fwrite(com_port, '*RPCPLD')

Command_Counter = 0;
command = (fread(program, 1));
Control = 0;

while (Control < 3)
  if (com_port.BytesAvailable > 0)
    % Wait for the 'R'eady Echo
    rcvd = char ( fread(com_port, 1) );
    if (rcvd ~= '$')
      fprintf(rcvd);
      if (rcvd == '.' && feof(program))
        Control = Control + 1;
      end
    else
      fwrite( com_port, command, 'char' );
      Command_Counter = Command_Counter + 1;
      if (Command_Counter/1000 == round(Command_Counter/1000))
        fprintf('%d\n', Command_Counter);
      end

      % Get the next command or EOF
      command = (fread(program, 1));
    end
  end
end

% Close down the COM port
fclose(com_port);
delete(com_port);

% Close the CPLD Program File
fclose(program);
```

Note that the original fopen command will need to be adjusted to point to the relative position of the XSVF format file to be used in the CPLD programming. The file MUST be in the XSVF format or the programming will not be successful!

4.) The device will return a successful message if the programming passed; otherwise, if the program appears to stop, simply close

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

MATLAB, power the device down, and repeat steps 2 and 3 until successful.

The XSVF format file can be generated from the SVF format file using the svf2xsvf version 5.02 Xilinx SVF-to-XSVF Writer utility by Xilinx, Inc. This is most easily done by writing a batch file to perform the operation. The following depicts a sample batch file that will convert the file program.svf to a XSVF format file named program.xsvf, when the batch is located in and run from the same folder as the source file. Note that the location of the svf2xsvf.exe program will need to be varied according to its location on the local PC.

```
"C:\svf2xsvf502.exe" -d -i program.svf -o program.xsvf -nc
```

The ASCII Readable XSVF file (program_xsvf.txt) can also be generated using a batch file containing the commands:

```
copy program.xsvf temp.xsvf
"C:\svf2xsvf502.exe" -d -i program.svf -o program.xsvf -a
      program_xsvf.txt -nc
del nc_xsvf_program_file.xsvf
copy temp.xsvf nc_xsvf_program_file.xsvf
del temp.xsvf
```

Note that written this way the program.xsvf file will not be created (or modified). However, both the XSVF and Text XSVF can be created simultaneously by just using the command on the second line.

Additional configuration details for the Xilinx SVF-to-XSVF Writer utility are available by running the executable without specifying input/output files. (The .exe file is downloadable from Xilinx in a special web package at

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

ftp://ftp.xilinx.com/pub/swhelp/cpld/eisp_pc.zip and subfolder xapp058_v5.01\bin\nt\. Note that this is the same package that included the sample C code for compilation onto an 8051 microcontroller used for the base JTAG port interface design implemented in assembly.)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

APPENDIX E

REVISION C PCB DESIGN INFORMATION

Easy-PC Design Files

The design files for Revision C of the PCB Board Design can be found in the *Easy-PC Design Files* folder in the attached CD. The CAD files were created using the Easy-PC CAD software program by Number One Systems (http://numberone.com). Table E.1 outlines the contents of the included files. Figure E.1 is the Revision C schematic, the parts of which are listed in Table E.2 (a more complete listing of the parts and the schematic labels is in Table E.3). The PCB layout of the components is provided in Figure E.2 while Figures E.3 and E.4 show the top and bottom coppers respectively.

File Name	Description of File Contents
'Board Details.txt'	General Board Description File
'MRCM, RevC.sch'	Easy-PC Schematic File
'MRCM, RevC.pcb'	Easy-PC PCB Layout File
'MRCM, RevC.report'	Aperture and Drill Tool Report File
'MRCM, RevC - Top Copper.gerber'	Top Copper Gerber File
'MRCM, RevC - Bottom Copper.gerber'	Bottom Copper Gerber File
'MRCM, RevC - Drill Data.ncd'	NCD Drill Tool File

Table E.1 - Easy-PC Design File Contents

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Design Schematic, Revision C

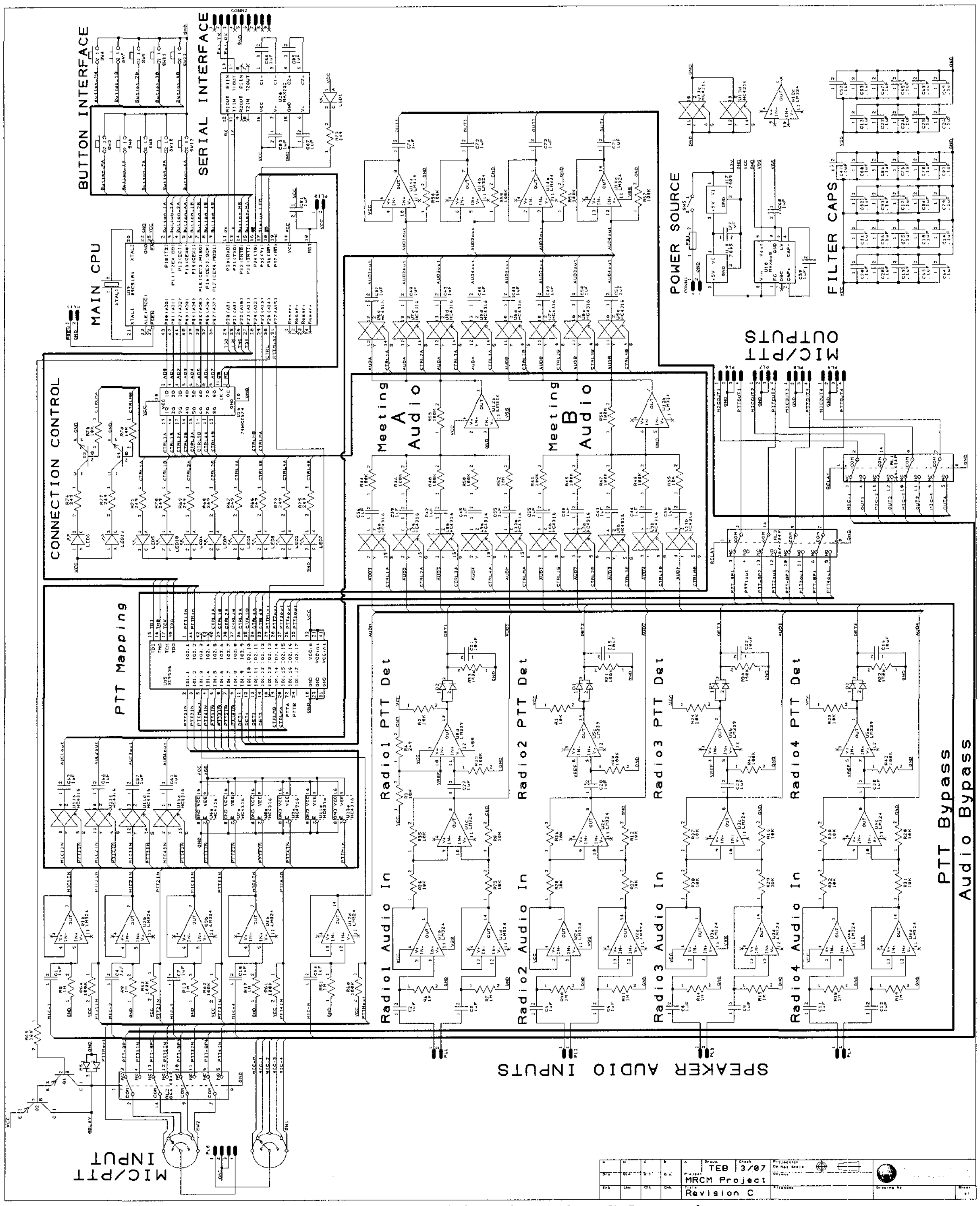


Figure E.1 – Revision C Design Schematic

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Parts List, Revision C

Qty	Description	Part Number
1	FAB, MRCM - Rev. C	Fab-MRCM-C, Custom
1	7805, Volt. Reg	MC7805CT-BP
1	7809, Volt. Reg	MC7809CT-BP
1	74HC574	MM74HC574WM
1	89C51RX2 (CPU)	AT89C51RC2-SLSIM
5	Diode, Dual Input	BAV70-TP
32	Cap, .1 uF, SM-1206	311-1179-1-ND
51	Cap, 1 uF, SM-1206	PCC1896CT-ND
4	Cap, 10 uF, SM-1206	PCC2414CT-ND
1	22.1184 Crystal	AB-22.1184MHZ-B2
1	DB9, Female	A23304-ND
1	Fuse, .75A/24V, Resettable	MF-MSMF075/24-2
3	Relay, 4PDT, 5V	G6A-434P-STUS-DC5
6	74HC4316	CD74HC4316M
24	Header, 2pin, .100 pitch	A1921-ND
10	LED, 3mm, Panel Mount, Red	SSI-LXH312ID-150
1	LED, 3mm, Panel Mount, Green	SSI-LXH312GD-150
6	OpAMP, SOIC-14	LM324DR2
1	LM339	LM339DR2
1	Header, 2pin, Male, .250 pitch	350428-1
1	MAX232	MAX232D
1	MAX660	MAX660M
2	Transistor, NPN, SOT23	MMBT4401-TP
2	Transistor, PNP, SOT23	MMBT4403-TP
10	Push Button, SPST-NO, Panel Mnt	GH1368-ND
12	Res, 249 Ohm, SM-1206	RHM249FCT-ND
26	Res, 10 KOhm, SM-1206	RHM10.0KFCT-ND
23	Res, 100 KOhm, SM-1206	RHM100KFCT-ND
4	Res, 150 KOhm, SM-1206	RHM150KFCT-ND
13	Res, 1 MOhm, SM-1206	RHM1.00MFCT-ND
1	Rotary Switch, 2P5T, Panel Mnt	GH5604-ND
2	Header, 6pin, .100 pitch	A1923-ND
1	Switch, SPDT, Panel Mount	EG2360-ND
4	Term Block, 2pin, .100 pitch	277-1273-ND
5	Term Block, 4pin, .100 pitch	277-1275-ND
1	XC9536, PLCC-44	XC9536-7PC44C
1	Enclosure	Unknown, Custom
24	Receptacle, 2pin, .100 pitch	22-01-3027
2	Receptacle, 6pin, .100 pitch	22-01-3067
60	Crimp Pin, Female, molex	08-50-0114
1	Plug, 2 pin, .250 pitch	1-480698-0
2	Socket, Female, 20-14 AWG	A1441
1/2	Heat Shrink Tubing, 1/8"x2'	RNF018K-ND
	Wire	

Table E.2 – Revision C Basic Parts List

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Detailed Parts Identification List, Revision C

Part	Qty	Value	Ref
7805	1		U18
7809	1		U17
74HCT574	1		U16
89C51RX	1		U19
BAV70	5		D1, D2, D3, D4, D5
Capacitor, 1206	32	.1uF	C13, C14, C17, C18, C19, C20, C23, C24, C25, C26, C31, C32, C33, C34, C47, C52, C53, C56, C63, C64, C65, C67, C68, C69, C70, C75, C76, C78, C79, C80, C82, C86
	51	1uF	C1, C2, C3, C4, C5, C6, C7, C8, C9, C10, C11, C12, C27, C28, C29, C30, C35, C36, C37, C38, C39, C40, C41, C42, C43, C44, C45, C46, C48, C49, C50, C51, C54, C55, C57, C58, C59, C60, C61, C62, C66, C71, C72, C73, C74, C77, C81, C83, C84, C85, C87
	4	10uF	C15, C16, C21, C22
Crystal Oscillator	1		XTAL1
DB990F	1		CONN2
Fuse	1		FS1
G6A-434P	3		RL1, RL2, RL3
HC4316	6		U6, U7, U8, U9, U11, U13
Header, 2P	2		PL10, PL11
LED	11		LED1, LED2, LED3, LED4, LED5, LED6, LED7, LED8, LED9, LED10, LED11
LM324	6		U1, U2, U3, U4, U12, U14
LM339	1		U5
Mate-N-Lok 2	1		CONN1
MAX232	1		U20
MAX660	1		U10
MMBT4401	2		Q3, Q4
MMBT4403LT1	2		Q1, Q2
Push Button NO	10		SW3, SW4, SW6, SW7, SW8, SW9, SW10, SW11, SW12, SW13
Resistor, 1206	12	249	R4, R66, R67, R68, R69, R70, R71, R72, R73, R74, R75, R77
	26	10K	R1, R2, R3, R8, R12, R16, R20, R23, R24, R25, R26, R27, R28, R29, R30, R31, R32, R35, R36, R37, R38, R52, R55, R65, R76, R78
	23	100K	R39, R40, R41, R42, R43, R44, R45, R46, R47, R48, R49, R50, R53, R54, R56, R57, R58, R59, R60, R61, R62, R63, R64
	4	150K	R21, R22, R33, R34
	13	1M	R5, R6, R7, R9, R10, R11, R13, R14, R15, R17, R18, R19, R51
Rotary Switch	2		SW1, SW2
SPST Switch	1		SW5
Terminal, 2P	4		PL1, PL2, PL3, PL4
Terminal, 4p	5		PL5, PL6, PL7, PL8, PL9
XC9536	1		U15

Table E.3 – Revision C Detailed Parts Identification List

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Component Layout, Revision C

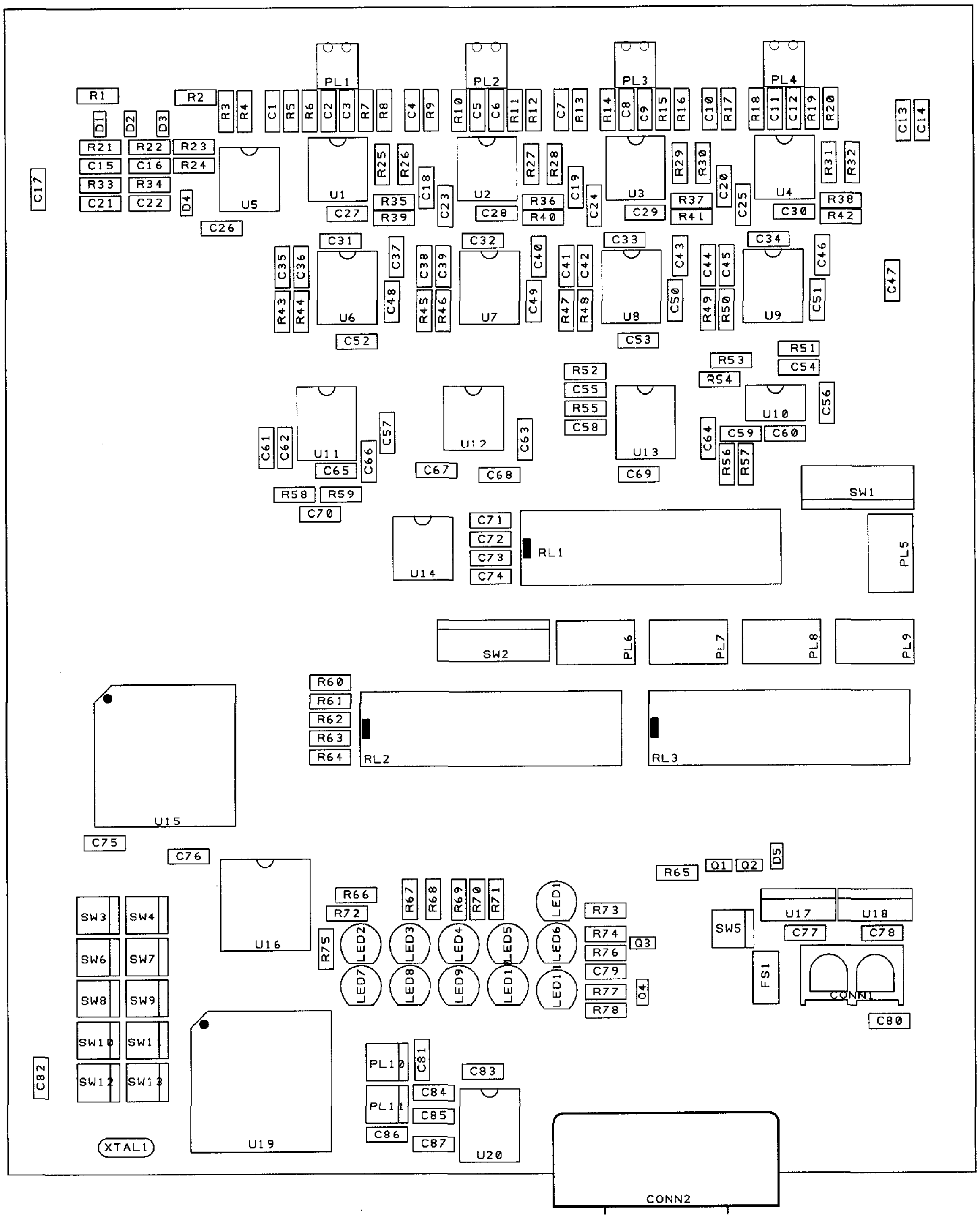


Figure E.2 – Revision C Component Layout Diagram

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Top Copper, Revision C

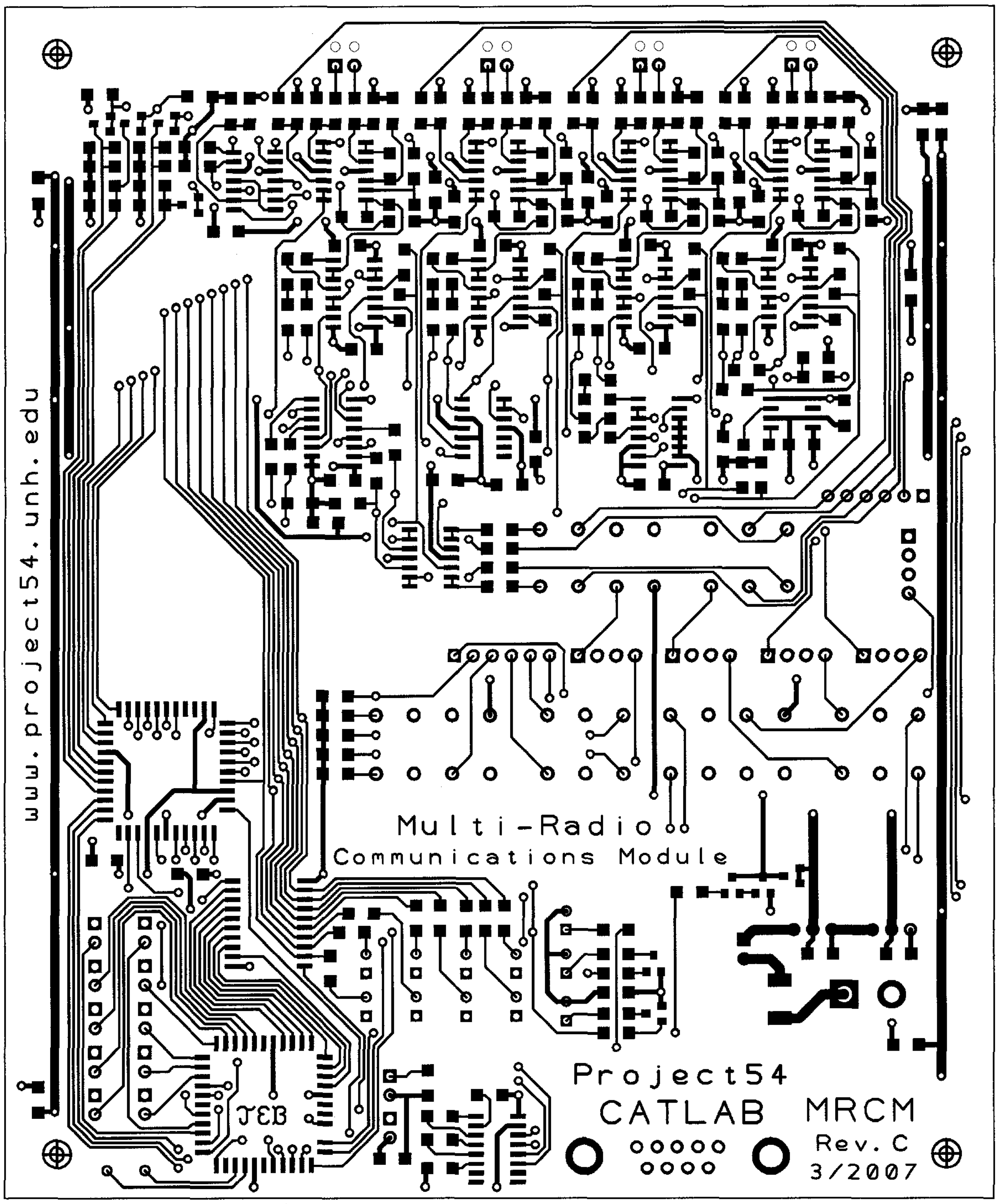


Figure E.3 – Revision C PCB Top Copper Physical Layout (viewed from top)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Bottom Copper, Revision C

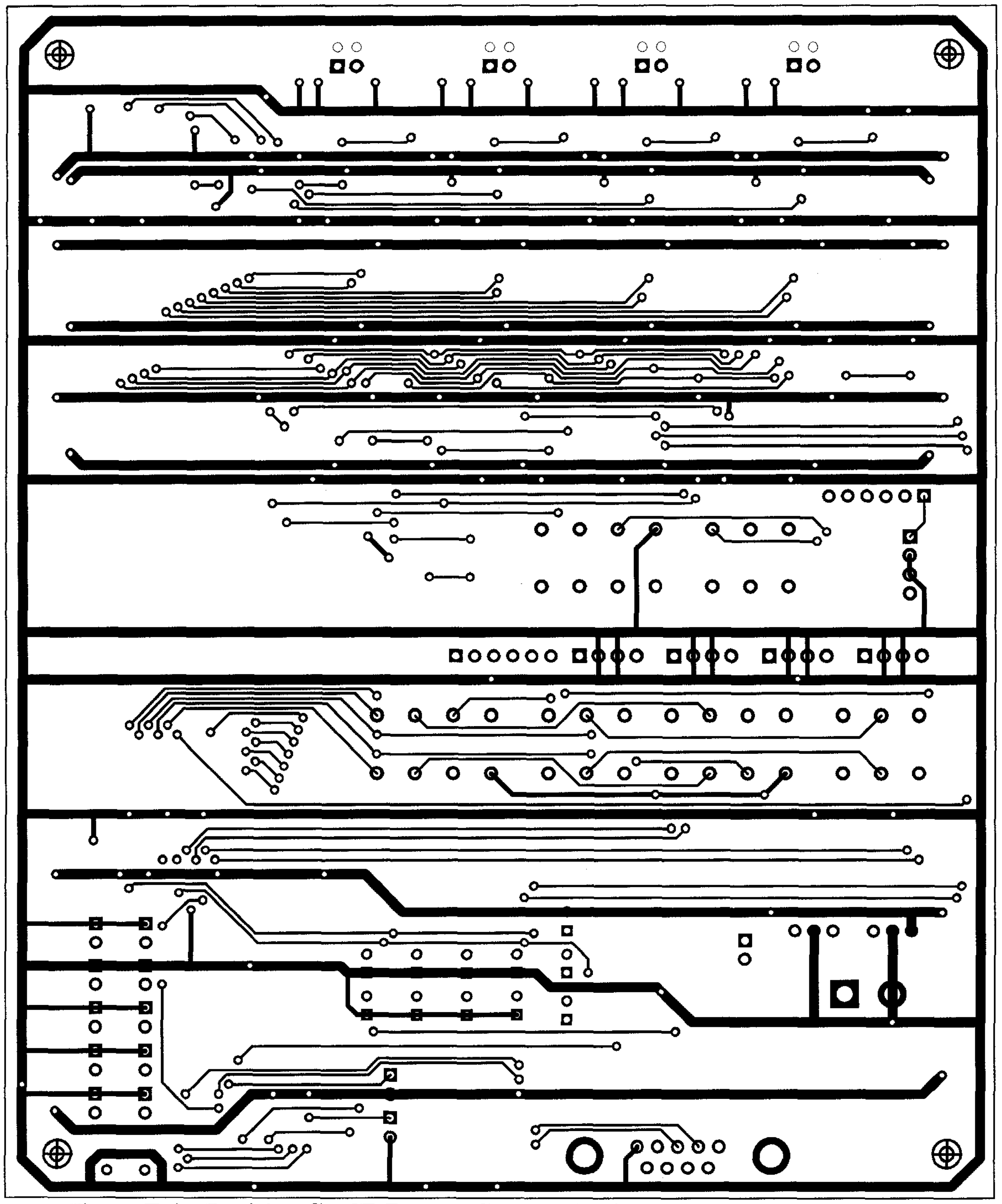


Figure E.4 – Revision C PCB Bottom Copper Physical Layout (viewed from top)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

APPENDIX F

PTT SIMULATION & TESTING MATLAB SCRIPTS

PTT Parameter Simulation

The following is the MATLab script used to simulate the PTT circuit in order to examine variations in parameter values. Some of the values were adjusted within the code for the purposes of making tests. Those shown reflect the final circuit used in the main testing. (Note that the R3 finally used was 2 KΩ.)

```
tmax = 5;
duration = 2;
f = 8000;
timestep = 1/f/64;

t = (0:timestep:tmax-timestep);
nmax = tmax/timestep;

R3n = [100 250 500 1000 2000 5000 10000];
used_R3 = 2e3;
R4 = 2.5e5;
C2 = 10e-6;
Dv = .3;
R2 = 100;
R1 = 1e4;
Vcc = 5;
VCPLD = 2.0;
w = 2*pi*f;

Vref = R2/(R1+R2)*Vcc;

for m = 1:length(R3n)
    Vin = zeros(1, nmax);
    Vptt = zeros(1, nmax);
    Vout = zeros(1, nmax);
    i = zeros(1, nmax);
    intic = 0;
    PTT = Vcc*eye(1, nmax);
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
R3 = R3n(m);

for n = 2:nmax
    if t(n) < duration
        Vin(n) = sin(w*t(n));
    else
        Vin(n) = 0;
    end

    if Vin(n) > Vref
        Vs = Vcc;
    else
        Vs = 0;
    end

    if Vs-Vptt(n-1)>Dv
        i(n) = (Vs-Vptt(n-1)-Dv)/R3;
    else
        i(n) = 0;
    end

    intic = intic+(i(n)-Vptt(n-1)/R4)*timestep;
    Vptt(n) = intic/C2;

    if Vptt(n) > VCPLD
        PTT(n) = 0;
        Vout(n) = Vin(n);
    else
        PTT(n) = Vcc;
        Vout(n) = 0;
    end

    if PTT(n) < PTT(n-1)
        PTT_Response_Timen(m) = t(n);
    elseif PTT(n) > PTT(n-1)
        PTT_Hold_Timen(m) = t(n)-duration;
    end
end

if R3 == used_R3
    PTT_Response_Time = PTT_Response_Timen(m)
    PTT_Hold_Time = PTT_Hold_Timen(m)

    figure(1);
    plot(t, Vptt);
    xlabel('t (s)');
    ylabel('Raw PTT Voltage (V)');
    hold on;
    plot([0 timestep*nmax], [VCPLD, VCPLD]);
    axis([0 nmax*timestep 0 5]);
    hold off;

    figure(2);
    subplot(3,1,1);
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
        plot(t, Vin);
        axis([0 nmax*timestep -1.1 1.1]);
        title({'\bf\fontsize{16}The Input Audio Signal'});
        xlabel('t (s)');
        ylabel('Audio In (V)');
        subplot(3,1,2);
        plot(t, Vout);
        axis([0 nmax*timestep -1.1 1.1]);
        title({'\bf\fontsize{16}The Output Audio Signal'});
        xlabel('t (s)');
        ylabel('Audio Out (V)');
        subplot(3,1,3);
        plot(t, PTT);
        axis([0 nmax*timestep -.5 5.5]);
        title({'\bf\fontsize{16}The Processed PTT Signal'});
        xlabel('t (s)');
        ylabel('Digital PTT (V)');
    end
end

figure(3);
plot(R3n/1000, PTT_Response_Timen*1000);
xlabel('R3 (Kohms)');
ylabel('PTT Signal Rise Time (ms)');

figure(4);
plot(R3n/1000, PTT_Hold_Timen);
xlabel('R3 (Kohms)');
ylabel('PTT Signal Hold Time (s)');
```

PTT Parameter Simulation & Testing Input Signal Generation

The following MATLab script was used to generate an 8 KHz tone of 2 seconds duration for input to the circuit board on one of the Differential Audio Speaker Inputs. The resulting audio and PTT signals were then capture via an oscilloscope.

```
Fs = 48000;
F = 8000;
Duration = 2;
t = 0:1/Fs:Duration;
x = zeros(1, length(t));

for i = 1:length(t)
    x(i) = sin(2*pi*F*t(i));
end
wavplay(x, Fs, 'sync');
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

APPENDIX G

DEMONSTRATION OF THE MOCKUP TESTING

The images shown in Figures F.1, F.2, and F.3 are of the three stations use in the mockup testing. Note that the interoperability circuit board is located at the interoperability device operator station. In this station the left-most device is the EFJohnson radio in the mock command vehicle. To its immediate right is the Main Microphone for the system that is currently configured in the Master Microphone mode. The PCB resting between the Motorola radio and its speaker is the prototype Revision C interoperability circuit board. Also shown on the far-upper right side is the IDB network interface box. (The PC, not shown, provides its audio signal directly to the circuit board via the attached cables. The speaker for the PC is shown in the top-center of the image.

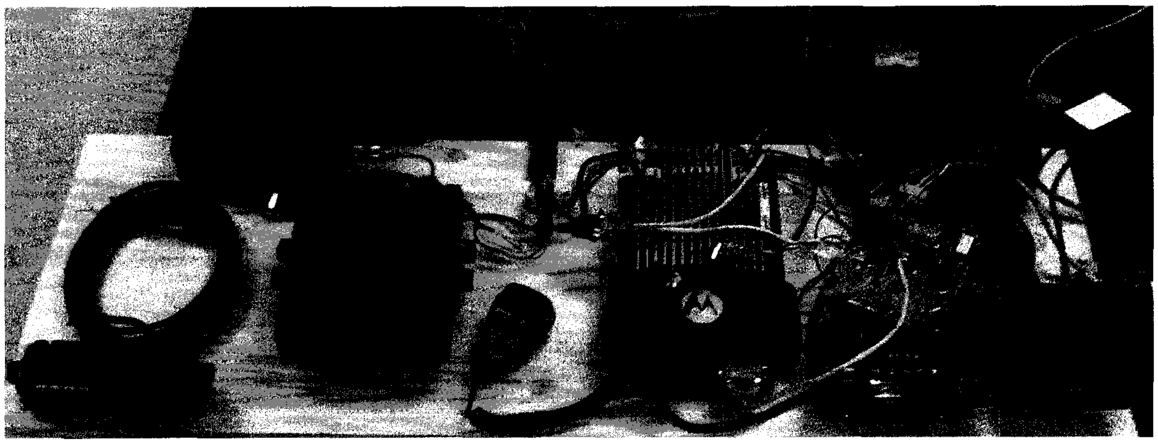


Figure F.1 – Image of the Interoperability Device Operator Station

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

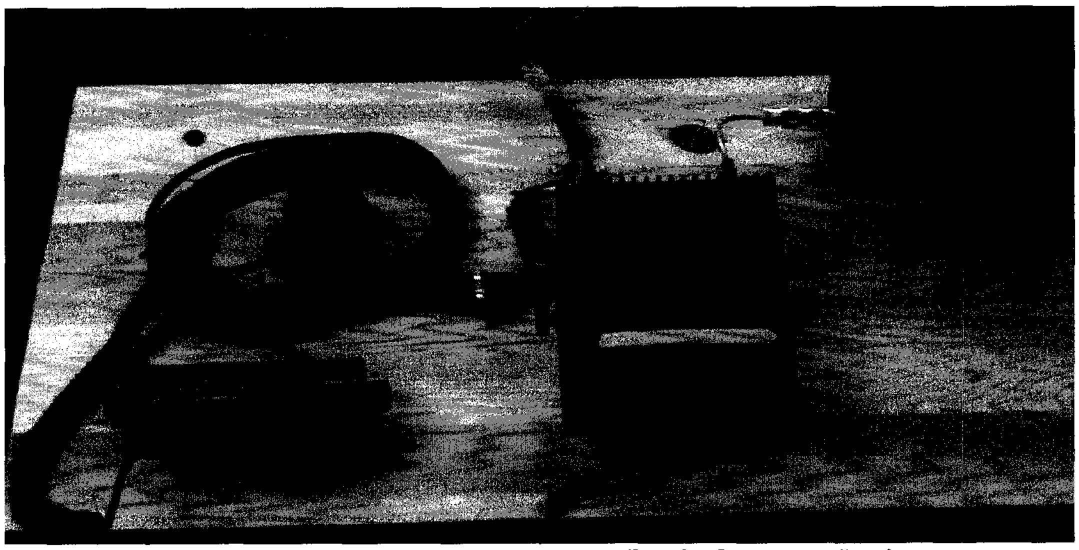


Figure F.2 – Image of the EFJohnson Radio Operator Station

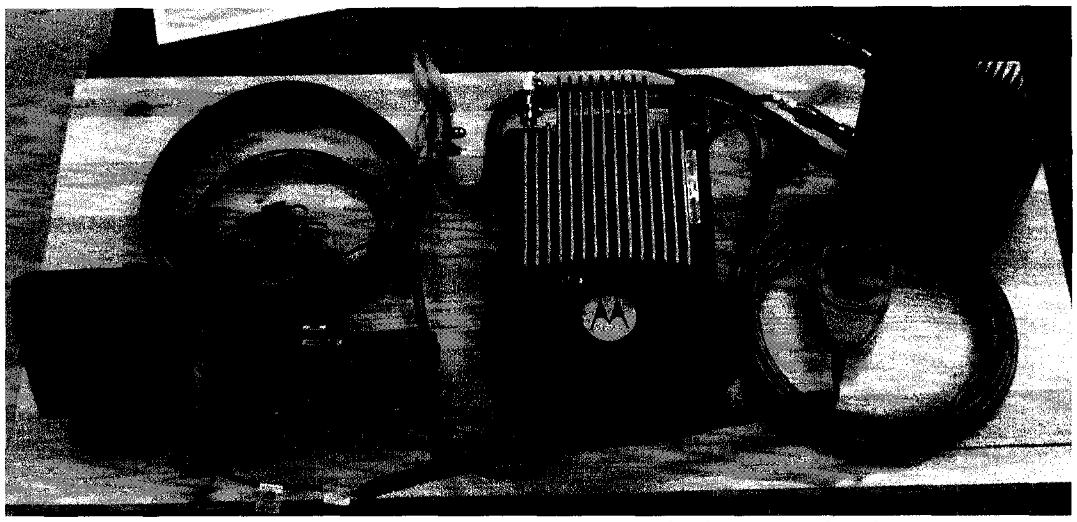


Figure F.3 – Image of the Motorola Radio Operator Station

The demonstration video for the mockup testing is included in the *Demo Video* folder in the attached CD. Note that the video is filmed from the perspective of the command vehicle operator (at the interoperability device). Table F.1 contains the testing script employed during the making of the video by the operators. It is supplied for reference purposes. (Note that the Command operator represents the operator that is in charge of the interoperability device and the person who necessarily controls the overall flow of the conversation(s).)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Command:	"Connecting to EFJohnson." "EFJohnson this is Command, can you hear me?"
EFJohnson:	"Command this is EFJohnson, I hear you."
Command:	"Connecting to Motorola." "Motorola this is Command, can you hear me?"
Motorola:	"Command this is Motorola, I hear you."
Command:	"Connecting EFJohnson and Motorola."
EFJohnson:	"Motorola this is EFJohnson, can you hear me?"
Motorola:	"EFJohnson this is Motorola, I hear you." "Can you hear me?"
EFJohnson:	"Motorola this is EFJohnson, I hear you

Table F.1 – Mockup Testing Script

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.