THE INTEGRATION OF A POLICE MOBILE RADIO INTO THE PROJECT54 SYSTEM

BY

BORISLAV BATINIC

B.S. University of Novi Sad, 2000

THESIS

Submitted to the University of New Hampshire

in Partial Fulfillment of

the Requirements for the Degree of

Master of Science

In

Electrical Engineering

September 2002

This thesis has been examined and approved by

Thesis director, Dr Andrew L. Kun
Assistant Professor of Electrical Engineering

Dr. W. Thomas Miller III
Professor of Electrical Engineering

Dr. William Lenharth
Research Associate Professor of Electrical Engineering

Date

ACKNOWLEDGEMENTS

I would like to thank Dr. Andrew L. Kun for his constant support, guidance, and encouragement throughout my research and the realization of my thesis. I appreciate his availability every time I needed his help and advice.

I would also like to thank Dr. Andrew L.Kun and Dr. Neda Pekaric-Nadj for providing me the opportunity to perform graduate studies at UNH.

I would like to thank Dr. W. Thomas Miller III and Dr. William Lenharth for serving on my thesis committee and allocating their time to review my thesis.

I would also like to thank all members of the CATLab for their suggestions and interest in my work.

Finally, I wish to thank my family and friends for their help and support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

CAN	Controller Area Network
COM	Component Object Modeling
GPS	Global Positioning System
GUI	Graphical User Interface
IDB	Intelligent Transportation System Data Bus
NHSP	New Hampshire State Police
TTL	Transistor-Transistor Logic
CRC	Cyclic Redundancy Code
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
DCB	Device Control Block
DSP	Digital Signal Processor
EEPROM	Electrically Erasable Programmable Read Only Memory
IF	Intermediate Frequency
LSN	Least Significant Nibble
MFC	Microsoft Foundation Classes
MSN	Most Significant Nibble
PA	Power Amplifier
PC	Personal Computer
PIC	Programmable Interrupt Controller
RF	Radio Frequency
RSS	Radio Service Software

SAPI4	Speech Application Programming Interface 4
SBEP	Serial Bus Expanded Protocol
VCO	Voltage-Controlled Oscillator
VF	Vacuum Fluorescent

ABSTRACT

THE INTEGRATION OF A POLICE MOBILE RADIO INTO THE PROJECT54 SYSTEM

By

Borislav Batinic

University of New Hampshire, September, 2002

The Consolidated Advanced Technologies for Law Enforcement Program (Project54) is an effort to integrate embedded and wireless mobile technologies into police cruisers of New Hampshire State Police (NHSP).The main goal of the thesis was to integrate the VHF radio, used by the NHSP, into the Project54 system. First, the radio electronics was examined and a suitable way to interface the radio to the IDB was found. The Radio Interface Hardware was designed in order to connect the radio's external bus, which is electrically an RS-485 bus, and the Common IDB Interface, which provides the IDB signals in RS-232 or TTL form. The Radio Control Application, as the part of the Project54 system software, was designed to control the radio from the embedded PC. The Radio Control Application was implemented in Project54 as a Component Object Model (COM) component. The software provides a speech-based user interface as well as a graphical user interface (GUI) which can be easily expanded to support computer control of any of the functions provided by the standard radio control head. The system was successfully tested in laboratory conditions.

CHAPTER I

INTRODUCTION

The Consolidated Advanced Technologies for Law Enforcement program (Project54) is a collaborative effort between the Consolidated Advanced Technologies Laboratory (CATLab) at the University of New Hampshire and the New Hampshire Department of Safety (NHDS).

The Project54 program addresses problems in the integration of electronic devices within police mobile units. The integration of electronic devices implies the integration of the hardware, software and user interfaces of the equipment that an average police cruiser has. Today's police patrol cars are equipped with a large number of electronic devices, for example radar equipment, video equipment, sirens, emergency lights, global positioning systems (GPS), barcode scanners and police car radios. The integration of these devices into one common system, with a single user interface, would help police officers to keep their attention on the road and enable them to be focused on the problems they solve with fewer disturbances coming from buttons, flashing screens and displays, conversations with the dispatchers, etc. The goal of Project54 is to implement a system that would require no more than voice commands for controlling the integrated equipment.

In the Project54 system hardware integration is accomplished by connecting the devices using the Intelligent Transportation System Data Bus (IDB). Figure 1.1 shows some of the devices that need to be integrated into the Project54 system. At the center of the system is an embedded PC. The PC controls all the integrated devices. The PC uses the IDB to communicate with the devices. Most of the devices in the police cruiser were not designed to connect to the IDB. In order to allow the connection of the devices and the embedded PC to the IDB, the Project54 Common IDB Interface (CI in Figure 1.1) was developed [1]. The Common IDB Interface provides two options for connecting the devices to the IDB: using a serial RS232 port, and using TTL parallel lines.

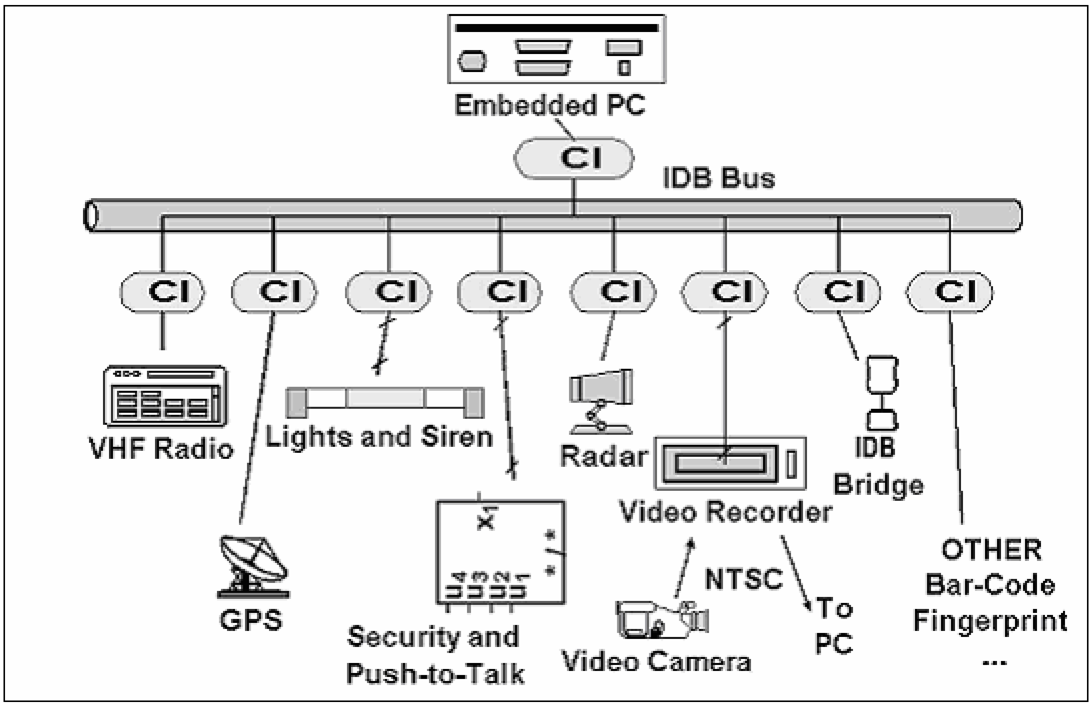


Figure 1.1 Project54 IDB interface connectivity

One of the electronic devices that has to be integrated into the Project54 system is the VHF radio. The radio is one of the fundamental tools that police officers use in performing their everyday tasks because it provides a communication link with dispatcher centers and with other officers. The radio's channels provide both voice and data communication ability.

While several devices shown in Figure 1.1 were successfully integrated into the Project54 system, the integration of the VHF radio used by New Hampshire State Police (NHSP), presented two problems that were not previously encountered. The first problem was that it is not possible to electrically connect this radio to the IDB using RS232 serial lines or the parallel TTL lines provided by the Common IDB Interface. The second problem was that the protocol used for the radio's internal signaling was not known. This internal signaling can be used to control radio operations and get feedback about the radio's status.

The main goal of the thesis was to integrate the VHF radio, used by the NHSP, into the Project54 system. The main goal comprised three partial goals. The first partial goal of this thesis was to create a hardware interface to connect the radio electronics to the IDB. The second partial goal was to create a software application to control the radio within the Project54 system. The third partial goal was to allow voice control of the radio.

We proposed that the first step to take in order accomplish the above goals was to examine the radio electronics and find a suitable way to interface it to the IDB. This was expected to provide requirements for the hardware interface and the signaling sequences for the control software. Our next proposed step was to design and implement a hardware interface between the radio and the IDB, based on the results of the first step. The third proposed step was to create a Project54 software module to control the radio and to implement a speech user interface to the radio. Finally, we proposed to test the new hardware, software and user interface in a laboratory settings.

The thesis is organized in eight chapters. The first chapter introduces the problems this thesis tackled.

The second chapter offers background information about the Motorola police radio used by the NHSP. It describes the operation and the structure of the radio system. It starts with a short description of basic radio operations and short explanations of the seven major assemblies that the Motorola police radio consists of. Further, it describes the control head assembly in more detail. The end of the second chapter offers an overview of several other research efforts related to the integration of electronic devices in cars, and to controlling in car devices with speech.

The third chapter describes how the radio electronics was examined. The chapter presents the information gained from this examination that was relevant to the design of the new hardware interface and control software.

Based on the results of the above examination new interface hardware, connecting the radio and the Common IDB Interface was designed, and it is presented in Chapter IV. The schematic design of the interface hardware and the description of its operation can be found in this chapter.

The fifth chapter illustrates the achieved results in software development. First, a brief overview of the Project54 system software is given. After that, the implemented algorithm is described in detail along with its advantages and disadvantages.

The final step of the research was to test the radio control system. The testing configuration, the testing procedure and test results are described in chapter six.

Chapter seventh summarizes and evaluates the achieved results. Chapter eight gives suggestions for future improvements.

At the end of the thesis, appendices offer more information about the designed hardware and the protocol used on the external serial bus between the control head assembly of the radio and the radio main housing. The IDB hardware schematic design, as well as a bill of materials and component datasheets, the board layout and a picture of the interface hardware are presented in Appendix A. A description of the serial bus protocol is of crucial importance for this research. This description can be found in Appendix B.

In order to carry out this thesis research, it was necessary to utilize information that is held as proprietary by Motorola,Inc. This information was obtained in part in documents provided by Motorola under a nondisclosure agreement, and in part by experimentally observing control and feedback messages within Motorola radios. All information about Motorola ASTRO radios found in the body of this thesis document can be found in publicly available documents from Motorola, such as the radio installation and operation manuals. All proprietary information needed has been assembled in Appendix B which is not included in normal copies at the thesis document.

CHAPTER II

BACKGROUND

2.1 Introduction

For better understanding of the radio operation and the problems this research aimed to tackle, this chapter will offer a general overview of the radio. The main assemblies of the radio will be listed and the modes of operation will be described in short. Since the operation of the radio's control head assembly is very important for this research, this assembly will be described in more detail.

Other research efforts, including those by several Project54 researchers, have encountered problems similar to those faced in this research. A general overview of several relevant research results will be described at the end of this chapter.

2.2 General overview of the ASTRO Digital Spectra radio

The New Hampshire State Police (NHSP) uses the Motorola ASTRO Digital Spectra radio in its mobile units. The ASTRO Digital Spectra radios are deployed with one of four different control head models (models W4, W5, W7 and W9). Figure 2.1 shows the radio installation diagram using the W7 control head model.

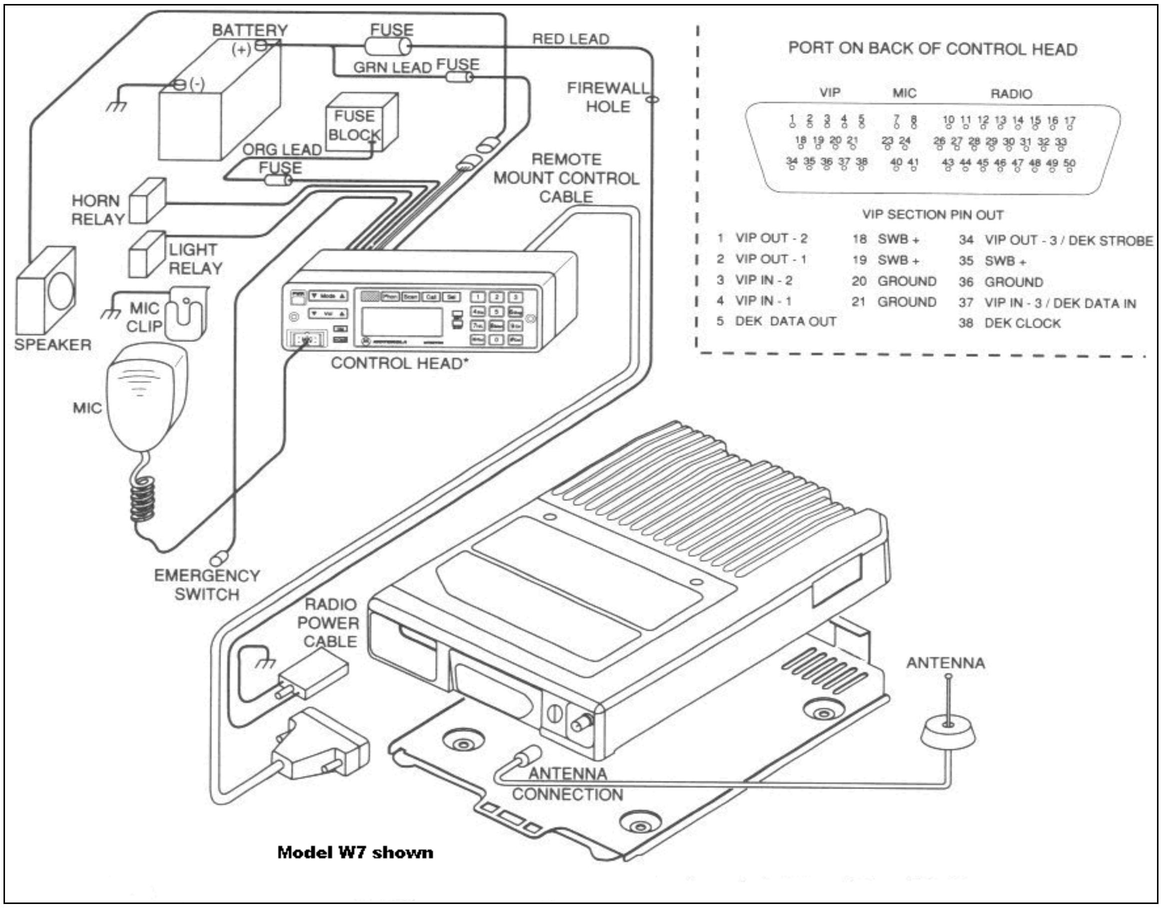


Figure 2.1 Radio installation using remotely mounted W7 control head

The ASTRO Digital Spectra radio is a microcontroller–based transceiver, incorporating a Digital Signal Processor (DSP). The microcontroller handles the general radio control, monitors status, and processes command input from the keypad or other user controls (e.g. the microphone on/off key). The DSP processes typical analog signals and generates signaling digitally to provide compatibility with existing analog systems [2].

The ASTRO Digital Spectra radio consists of the following seven major assemblies, six of which are located in the main radio housing:

- Control head assembly,

- Power amplifier,
- Front-end receiver assembly,
- Radio Frequency (RF) board,
- VCO/Buffer/Divider board,
- Command board,
- VOCON (VOcoder CONtroller) board.

The block diagram of the radio interconnection is shown in Figure 2.2.

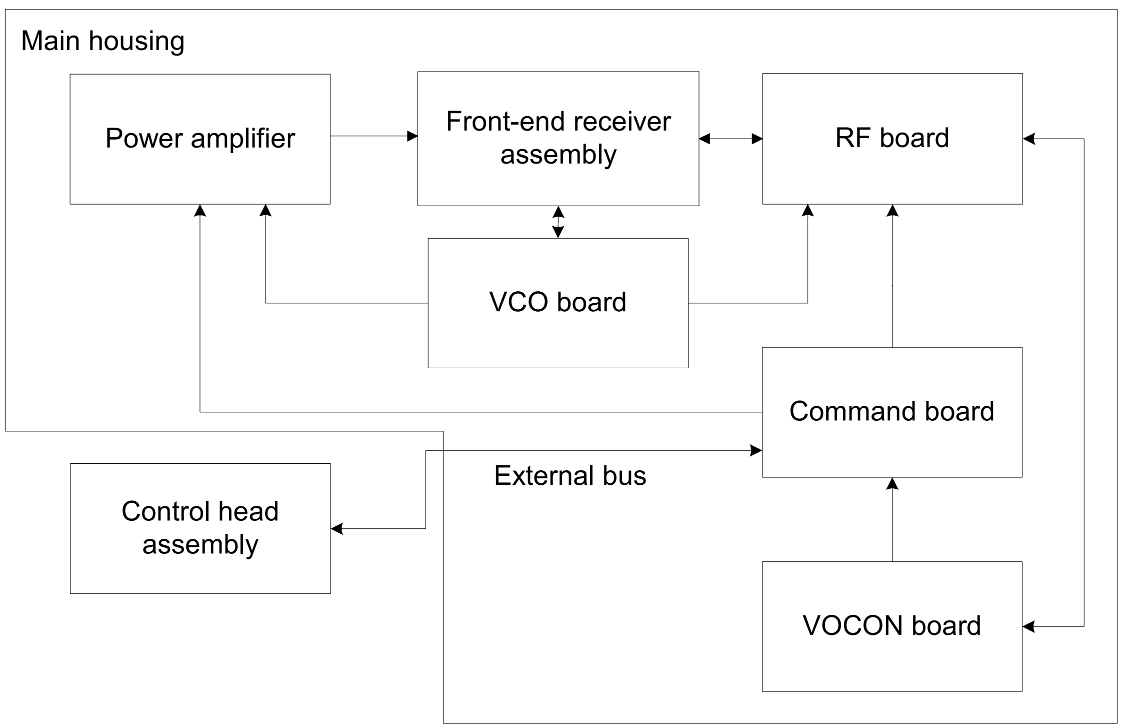


Figure 2.2 Radio interconnection

The control head assembly is connected, directly or remotely (direct-mount control heads attach directly to the radio housing, while remote-mount control heads attach over a cable), to the command board over the external bus. This assembly contains a vacuum fluorescent (VF) display, a VF driver, a microprocessor and a serial bus interface.

The power amplifier (PA) is a multi-stage, discrete-transistor RF amplifier consisting of the following: antenna switch, directional coupler/detector, harmonic filter, drivers, low-level power controlling stage and final amplifier.

The front-end receiver assembly consists of a preselector, a mixer circuit, and an injection filter.

The RF board contains a synthesizer, a dual intermediate frequency (IF) receiver and demodulation circuits.

The VCO/Buffer/Divider board contains the voltage-controlled oscillator (VCO), which generates variable frequency output signals controlled by two steering lines, a frequency divider, receive and transmit buffers.

The command board contains the power control/regulator, a digital-to-analog (D/A) and analog-to digital (A/D) converter, the serial bus interface, and the audio power amplifier (PA).

The VOCON (VOcoder CONtroller) board contains a microcomputer unit (MCU), its associated memory and memory management integrated circuit (IC), the Digital Signal Processor (DSP) and its associated memories and its support integrated circuit (IC) module.

The next two sections will explore how the assemblies listed above operate.

2.3 Analog mode of operation

When the radio is receiving, a signal comes from the antenna through the power amplifier board to the front-end receiver assembly. In the front-end receiver assembly the signal is filtered, amplified, and mixed with the first local oscillator generated by the voltage-controlled oscillator (VCO). After mixing, the signal is fed to the intermediate frequency (IF) circuitry on the RF board where it is filtered and amplified again. Then, the signal is mixed with the second local oscillator signal in the digital back-end IC on the RF board to create the second IF signal at 450kHz. The analog to digital (A/D) converter processes the analog IF signal, converts it to a digital bit stream and divides it down to the baseband signal. These samples are converted to current signals and sent to the DSP support IC. The DSP support IC digitally filters and discriminates the signal and passes it to the DSP. The DSP decodes the information in the signal and identifies the appropriate destination for it. The DSP routes digital voice to the DSP support IC for conversion to an analog signal which is presented to the audio power amplifier on the command board that drives the speaker.

When the radio transmits a signal, the microphone audio signal is first passed to the command board limiter and then to the DSP support IC for signal digitalization. The digital data is passed to the DSP which pre-emphasizes and returns the signal through a low-pass filter back to the DSP support IC. The DSP support IC reconverts the signal into an analog signal and scales it for application to the voltage-controlled oscillator as a modulation signal. Modulation information is passed to the synthesizer along the modulation line. A modulated carrier is passed to the power amplifier board that transmits the signal under dynamic power control.

2.4 Digital mode of operation

Motorola calls the digital mode of operation the ASTRO mode of operation. In this mode of operation the receiver handles an ASTRO mode signal identically to an analog mode signal up to the point where the DSP decodes the received data. In the ASTRO receive mode the DSP uses a proprietary algorithm to recover information. In the transmit mode, a microphone audio signal is processed as in the analog mode except that the DSP uses a different algorithm for information encoding.

2.5 Programming the radio

A variety of features can be programmed and customized into the ASTRO radio. For example, the functions of the control head buttons can be customized, transmit and receive frequencies can be programmed. Thus, Motorola has developed the Radio Service Software (RSS) [3] which allows the radio to be programmed. Using the RSS it is possible to personalize a radio with a unique set of features for each police officer. Since all NHSP officers use the same features, all NHSP radios are programmed identically. Hence, an NHSP officer can easily use any NHSP cruiser's radio.

2.6 Control head assembly and basic theory of operation

The control head assembly is the interface between the radio and the police officer or anyone else operating the radio. It contains switches and a keypad (buttons) for input and visual indicators that indicate the present settings of the radio. The control head assembly processes all the button inputs and visual indicators through a microprocessor.

The features of the ASTRO Digital Spectra radio differ among various control head models. There are four different control head models: W4, W5, W7 and W9. Two models were examined in this research, model W7 and model W9. Figure 2.3 and Figure 2.4 show the front panel of these two control head models.

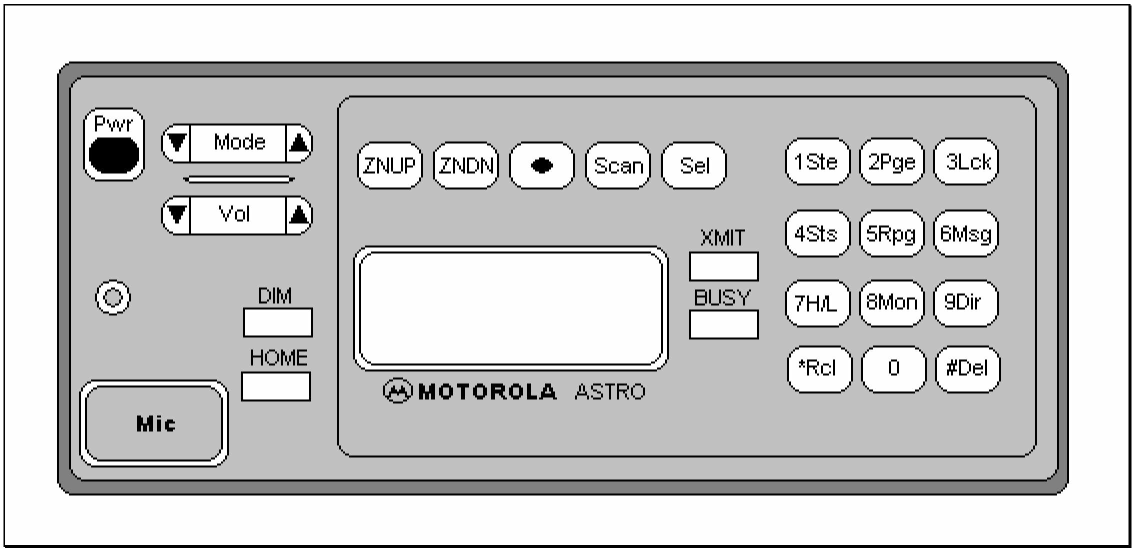


Figure 2.3 Control head model W7

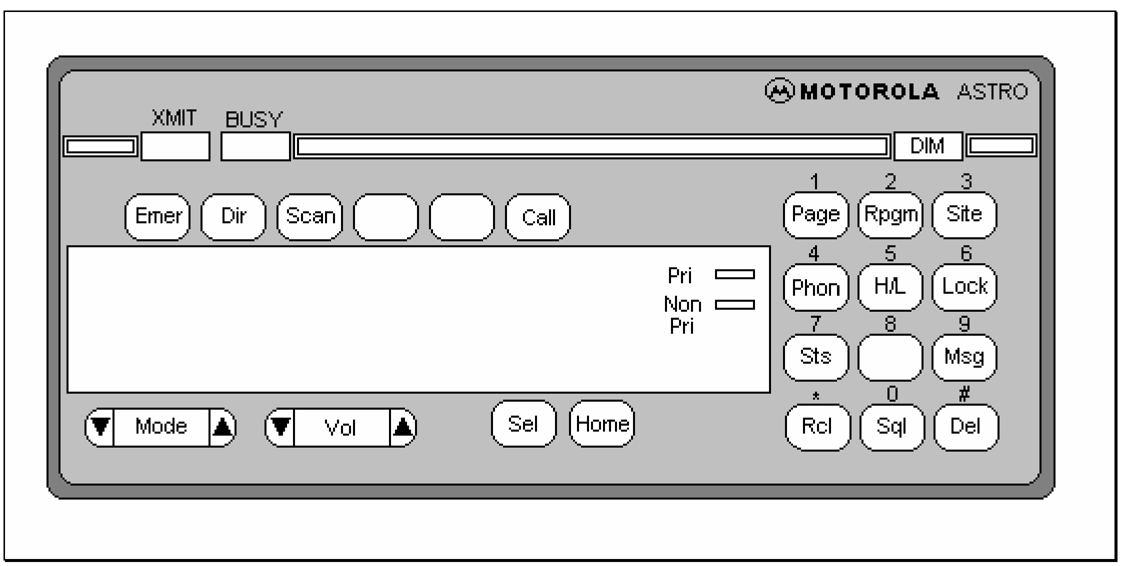


Figure 2.4 Control head model W9

Model W9 can store 255 channels, has an 11-character display and electronic knobs for volume and mode selection. The 12-button keypad allows the user to place phone calls or send pre-programmed status messages to the dispatcher. An optional feature also lets users control a compatible police siren directly from the keypad. This model is available in remote-mount configuration only, that is, the control head assembly can only be installed remotely from the main radio housing.

Model W7 can store 255 channels, has an 8-character display and electronic knobs for volume and mode selection. This model also includes a 12-button keypad for initiating telephone calls or sending pre-programmed status messages to the dispatcher. W7 is available in dash-mount (direct-mount) and remote-mount configurations. In the dash-mount configuration the control head is connected directly to the radio.

The basic control head operations examined in this research are changing the zones, channels and volume. Zones are logical groups of channels. Motorola introduced zones to simplify searching for a given channel. When an officer switches to a certain zone (defined by his or her agency), pressing the Mode button (see Figure 2.3 and Figure 2.4) will cycle through the channels defined for this zone only. The NHSP is organized into six troops, and in the radio configuration each troop is allocated several zones.

Both control head models, W7 and W9, have two rocker switches for changing channels and volume. The W7 model has two buttons for changing zones whereas the W9 model does not have specific buttons for this operation. However, any of the buttons on the control head can be programmed to change zones.

When a button is pressed, the button command is processed by the microprocessor in the control head. The control head sends a differential signal on the external bus to the microprocessor in the main radio housing. The information related to which button is pressed is processed in the main housing of the radio. In response to user input and some other events, the control head display is refreshed to show the current settings, the current channel, the kind of communication that is taking place on the radio, and other information.

2.7 Other efforts in integrating in-car devices and developing in car speech interfaces

In order to allow the integration of various electronic devices into the Project54 system Kun et al. [4] implemented a modular software system. The software system incorporates control modules that are responsible for the operation of individual electronic devices. The software system also allows for a centralized speech-based user interface. Martin et al. created an interface to be used to connect the IDB and electronic devices in the cruiser [1]. The interface, called the Common IDB Interface, provides two options for connecting devices to the IDB. The first option is to use an RS-232 serial line; the second is to use parallel TTL lines. Kozomora worked on the integration of radar equipment into the Project54 system [5]. Working with two radar devices, she implemented a software application within the Project54 system to control these radar devices.

Researchers at Texas A&M worked on improving the functionality of police cruisers using advanced electronics through the ALERT project [7, 8]. The project addressed the need to integrate various electronic devices in police cruisers. However the project did not result in the acceptance of a standard for integration; rather it was a proofof-concept effort. Wahab et al. worked on creating a dashboard system for intelligent vehicles that would be able to handle data and speech communication [9]. They paid special attention to combating noise in the car that interferes with voice communication. Muthusamy et al. created a prototype system for information retrieval in cars [10]. The system has a voice interface (input and output), and it can be used for voice dialing, Internet operations, and help with navigation.

CHAPTER III

RADIO SYSTEM EXAMINATION AND HARDWARE AND SOFTWARE TEST DESIGNS

3.1 Introduction

As explained in Chapter I (Figure 1.1) all electronic devices in the Project54 system are connected to the IDB using the Common IDB Interface hardware. The goal of this research was to incorporate the Motorola police radio into the Project54 system. However, the Common IDB Interface hardware was not designed to interface directly to the radio. In Chapter I we proposed that the first step of this research should be to examine the radio electronics and find a suitable way to connect it to the IDB. We needed information on how the radio hardware can be interfaced to the IDB. We also needed information on the signaling sequences that need to be used to control the radio.

3.2 Hardware requirements

We first examined the radio hardware to determine what type of interface needs to be designed to connect the radio to the IDB. The main tool in this effort was a Hewlett and Packard's Infinium digital oscilloscope. The attention was focused on the examination of the radio's external bus. The external bus is a communication bus that connects the control head and the microcontroller unit (MCU) inside the main radio housing. It is a combination of a serial data bus, audio, power and control bus. The audio bus carries voice data between the control head and the radio main housing. The control bus carries the signals required for data flow control. The power lines carry DC voltage to the control head. For the purposes of this research the attention is focused on the serial data interface part of the external bus (data bus, busy line and digital ground line) and these lines are what we will refer to as the external bus. RS-485 is the hardware standard used on the external bus of the radio. The RS-485 standard allows half-duplex communication between multiple devices, using a single pair of wires. Only one device can drive the line at any time, so any RS-485 line-drivers connected to the bus must be put into a high-impedance mode when they are not in use. All the connected devices are monitoring the bus while data is being transmitted.

Figure 3.1 shows the block diagram of the system with the external bus connecting the control head and the radio main housing. The figure also shows that the Project54 embedded PC can be connected to the bus in parallel with the radio devices.

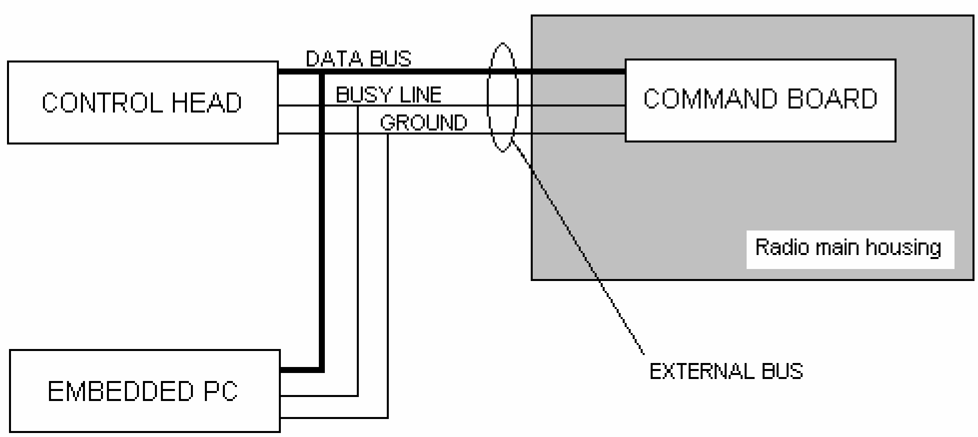


Figure 3.1 Block diagram of the system connected with the external bus

Serial communication on the external bus uses the bi-directional data bus lines, BUS+ and BUS-, and the bi-directional handshake BUSY line. The BUS+ and BUS- lines are differentially driven. Differential signals are less affected by noise, which can be quite overwhelming in an environment such as a motor vehicle. The idle state for the BUS+ line is logic high and for the BUS- line logic low. Data is transmitted synchronously at a 9600 baud rate.

To start a transmission a device (e.g. the microcontroller within the control head assembly) first examines the BUSY line. If the BUSY line is in the idle state (low voltage), the device puts it in the active state (high voltage). At the end of the transmission the device reverts the BUSY line to the idle state. This fact will have important implications on the hardware interface design, which is described later in the text.

When the radio's microcontroller unit (MCU) sends the data to the bus, the MCU also monitors the transmitted data as a collision detection measure. If a collision is detected, the MCU will stop transmission and resume it later. So, the MCU uses a technique similar to CSMA/CD (carrier sense multiple access with collision detection) for accessing the bus.

3.3 Software requirements

Having acquired the information described in section 3.2 we could turn our attention to the signaling sequences needed to control the radio. Therefore, the next step was to build a test hardware interface that would interface the radio's external bus and the computer. On the computer we created test software that utilized a test hardware interface. Building the test hardware interface and software we could determine the required signaling on the external bus.

3.3.1 Test hardware design

The designed test hardware interface provides a communication link between a laptop and the radio’s external bus.

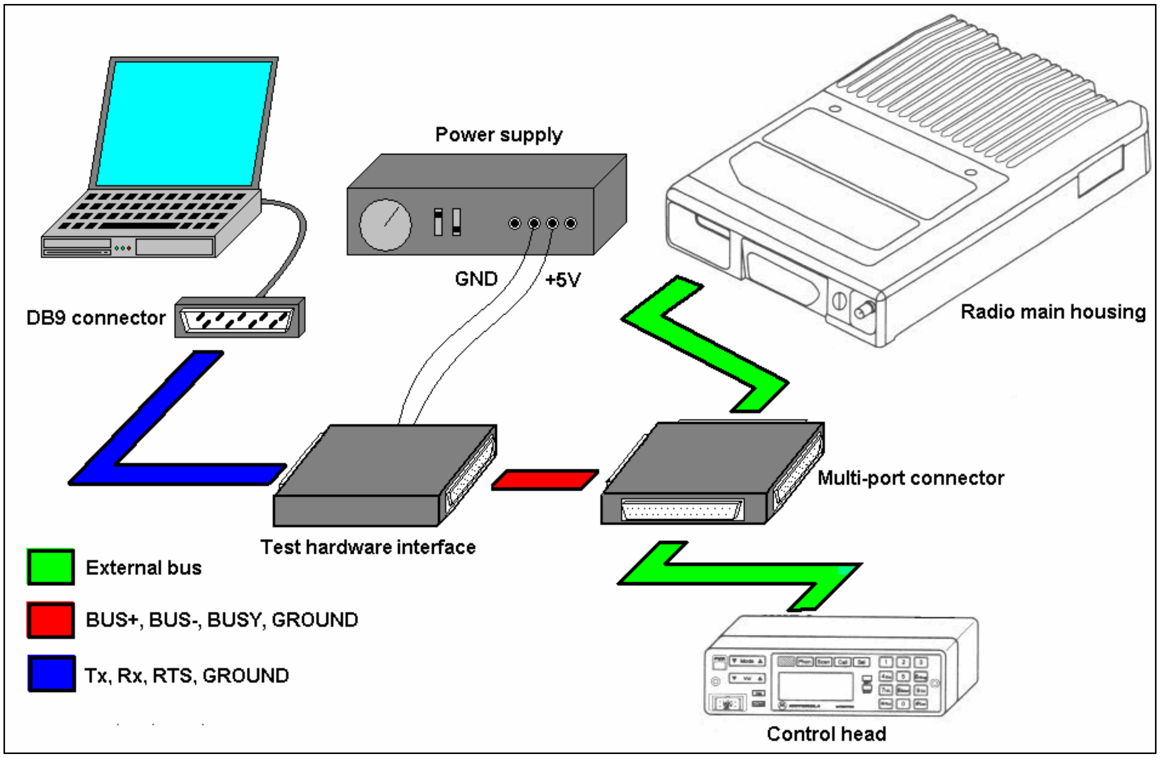


Figure 3.2 Connection diagram of the test system

Figure 3.2 shows the test system connection diagram. The test hardware interface is connected to the radio’s external bus and to the DB9 serial port of a laptop computer. As mentioned, RS-485 is the standard serial communication protocol used on the radio’s external bus while the computer uses the RS-232 protocol on the DB9 port. The test hardware interface provides signal conversion between the two protocols.

Figure 3.3 shows the test hardware interface schematic design.

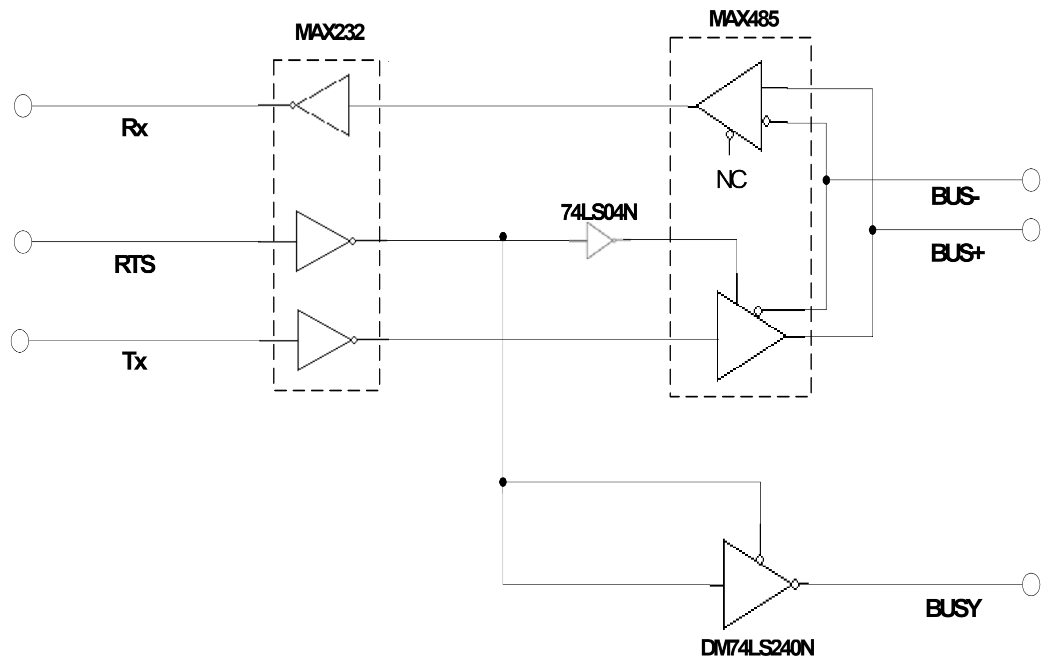


Figure 3.3 Test hardware interface – schematic design

Many RS232-to-RS485 converters are available on the market, but they are very expensive. We designed a converter using RS-232 and RS-485 drivers. The RS-232 driver (MAX232) provides the conversion between RS-232 and TTL signals. The RS-485 driver (MAX485) provides the conversion between TTL and RS-485 signals. One inverter and one buffer are used in order to control the RS-485 driver and the BUSY line on the radio external bus.

As mentioned before, in order to transmit data on the radio's serial bus the BUSY line has to be examined. In case it is in the idle state (logic low), the device that wants to transmit pulls this line high and transmits. Our test hardware does not examine the BUSY line before sending data. While this may be a source of collisions on the bus we did not run into problems during testing. The ready-to-send line (RTS) RS-232 signal provides a convenient way to manipulate the radio's BUSY line and handle the RS-485 driver from software running on the computer. The transmit line (Tx) is used for sending data from the computer to the radio. When the computer starts transmitting data, the data comes from the Tx line on the DB9 serial connector, and enters the MAX232 driver that converts the input signal from RS232 logic to TTL form. At the same time the RTS line provides the BUSY signal on the output of the hardware interface. While the RTS signal is high the data on the transmission line is valid. The TTL data enters the RS-485 driver and the data is converted to an RS-485 signal. The driver transmits the signal over the BUS+ and BUS- lines. When the computer stops transmitting data, the RTS is set to low which means the BUSY line goes into the high impedance state. After that, other devices are allowed to use the external bus. The BUS+ and the BUS- lines need to be in the idle state after transmission, too.

The receive line (Rx) always monitors the bus and allows the embedded PC to receive data at any time. If the radio transmits data over the external bus, it sets the BUSY line high, and data lines (BUS+ and BUS-) carry the traffic. The data goes through RS-485 driver, which converts it into a TTL signal. The MAX232 driver converts the TTL signal into an RS-232 signal.

The chip layout of the circuit is shown on the Figure 3.4.

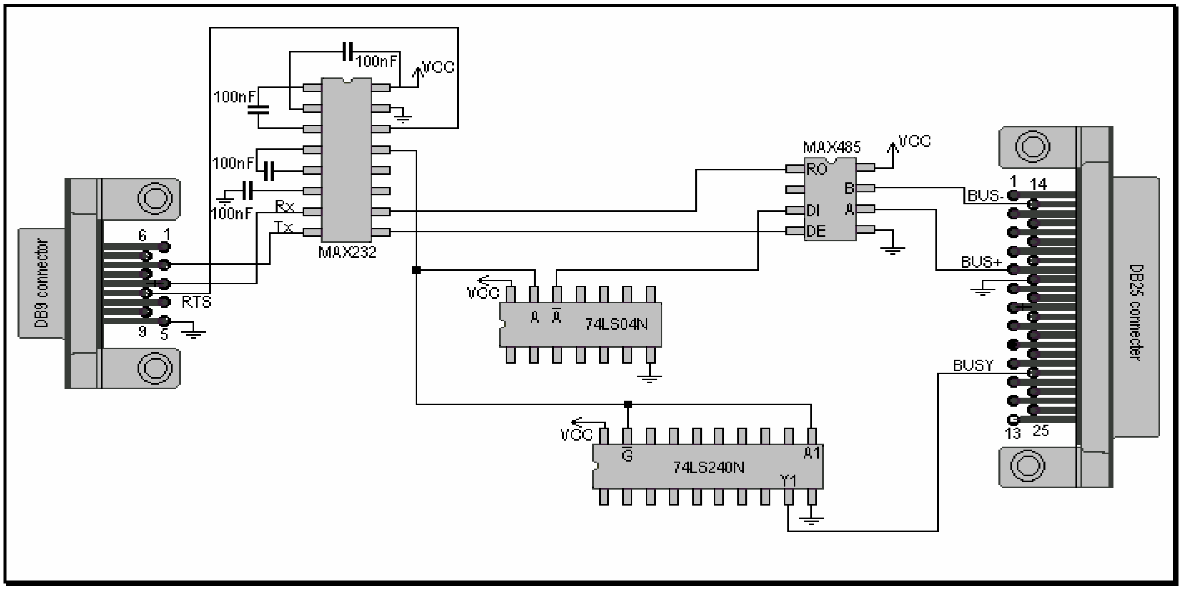


Figure 3.4 Radio Interface Hardware – test version

<u>3.3.2 Test software</u>

This section describes details of the software portion created for test purposes. The software is designed to accomplish two tasks. First, we needed an application that would acquire signal sequences from the radio's external bus and second, we needed an application capable of testing proposed ideas on what signal sequences mean. The software is written in the C++ programming language and it is created as a single dialog application using the Microsoft Foundation Classes (MFC). This application uses an outer shell based on the W9 radio control head model for a graphical user interface (see Figure 3.5).

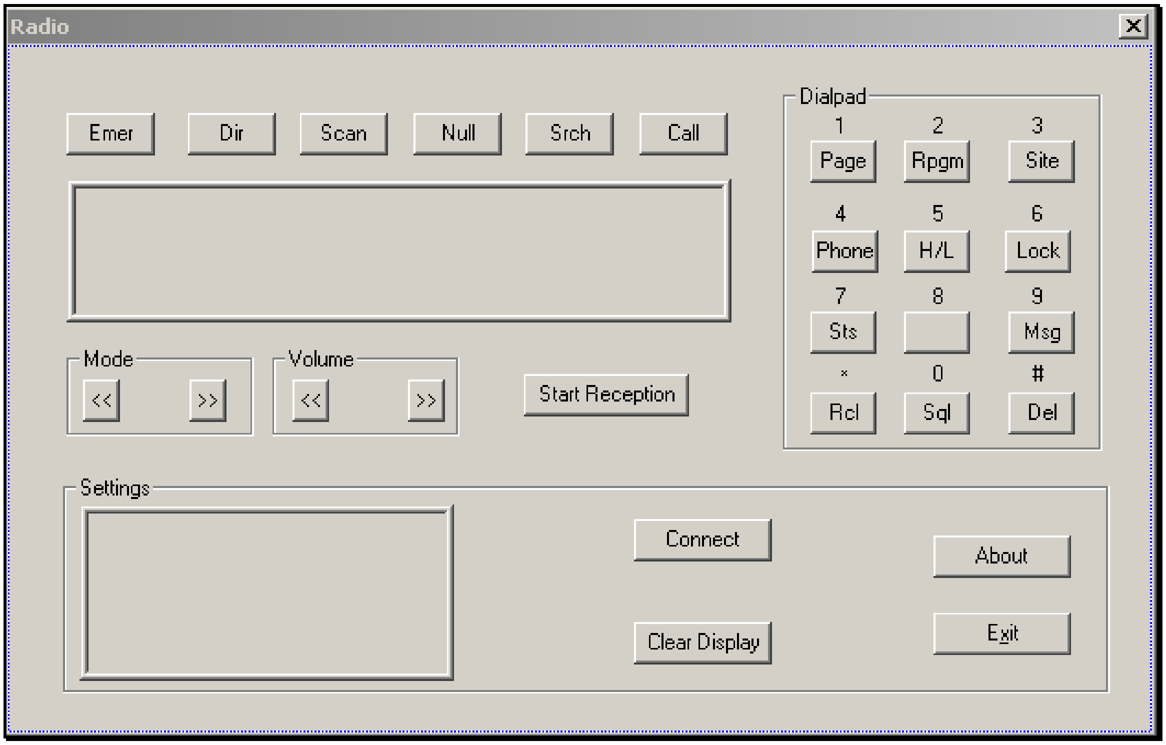


Figure 3.5 Radio Screen application user interface

The GUI window is separated into two sections. The upper section of the window is a reflection of the W9 control head model. The user interaction with it is the same as with the radio's control head. Not all buttons are allotted a certain function. For the purposes of this research only the buttons for changing the channels, zones and for adjusting the volume are enabled. A single mouse click on buttons initiates the appropriate data sequence to be sent to the radio.

On the other hand, in order to receive data from the radio's external bus, the "Start Reception" button has to be pressed. When this button is pressed the application creates a text file, and the received data is written in it in hexadecimal notation. A data sequence coming from the radio's external bus is arranged in the file in order to facilitate its analysis. Appendix B describes how such data can be interpreted. Figure 3.6 shows a sample file created when the button for changing a channel on the control head, was pressed.

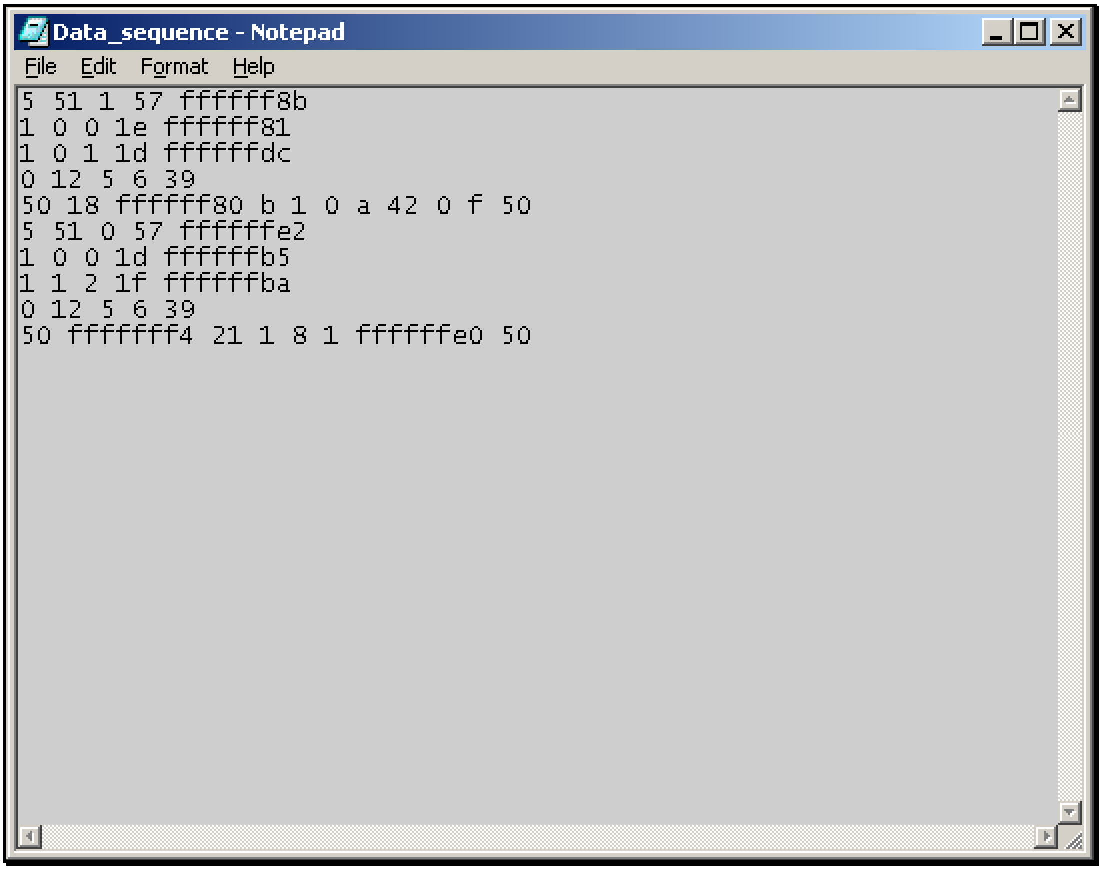


Figure 3.6 Text file created by the application

The bottom section of the GUI window, labeled “Settings”, contains the application’s command buttons and a text field used for displaying information about communication settings. The text field shows the baud rate, the DCB (Device Control Block) settings such as number of stop and byte bits, and whether the serial port used for communication was successfully initialized (see Figure 3.7).

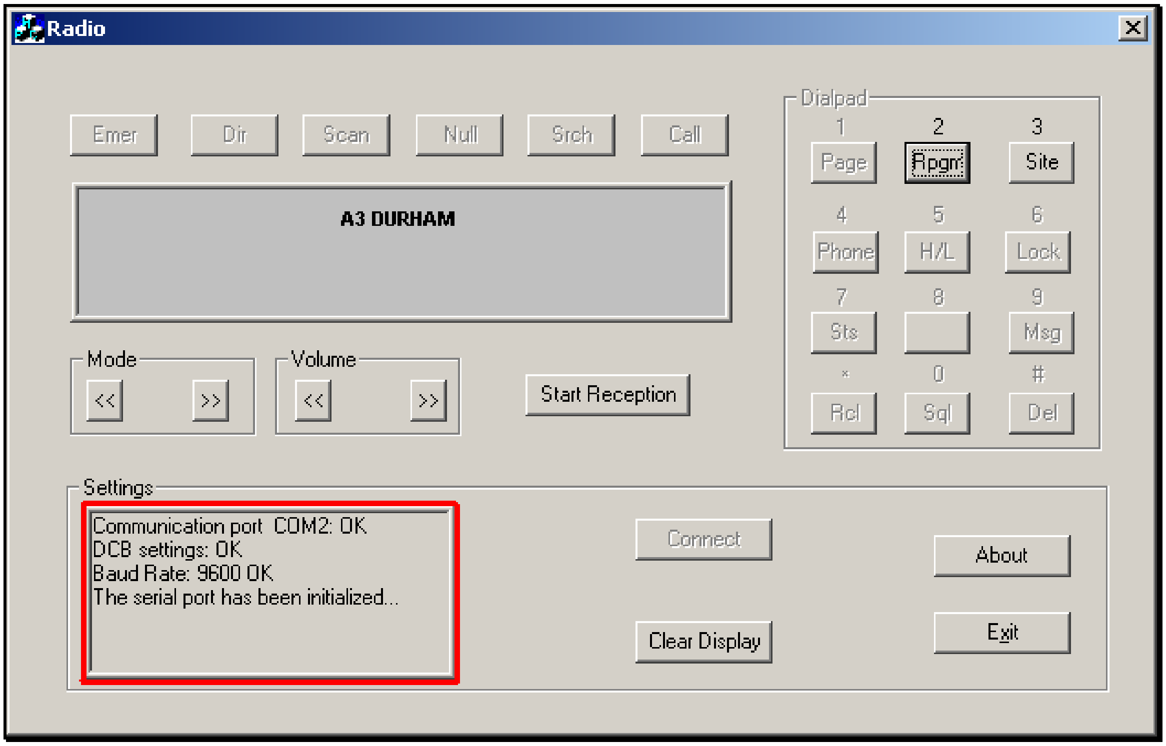


Figure 3.7 Settings

The settings section of the GUI window contains four command buttons. The "Connect" button is used to initialize the serial port of the computer and to establish communication. The "Clear Display" is used to erase the contents of the display field. The "About" button gives general information about the application and the "Exit" button is used for closing the application.

3.3.3 The results of the examination

After the test hardware and software were designed, the examination on the test system resulted in understanding the serial protocol on the radio's external bus. A description of the bus protocol can be obtained in Appendix B. The test system provided us the information needed to control the radio from the computer. The details of the serial protocol are propriety to Motorola, Inc. and are thus not included in the body of this thesis.

CHAPTER IV

HARDWARE DESIGN

4.1. Introduction

After the examination of the radio electronics and after the test hardware and software were designed, the next step, proposed at the beginning of this research, was to design and implement a hardware interface between the radio and the IDB. As we saw in Chapter III the radio's external bus is electrically an RS-485 bus. Therefore, in order to integrate the radio into the Project54 system we needed hardware that would interface the RS-485 bus to the IDB. However, the Common IDB Interface provides the IDB signals in RS-232 or TTL form. Since it is easier to convert signals from RS-232/TTL form to RS-485, than IDB signals to RS-485 form we use the Common IDB Interface as part of the interface hardware. Therefore, for the purpose of this research it is important to know certain details about the design of the Common IDB Interface hardware.

4.2 The Common IDB Interface

The Common IDB Interface hardware [1] is identical for all devices in the Project54 system. The control and data flow on-board the Common IDB Interface are carried out by a microcontroller. Currently two versions of microcontroller code are used for the Common IDB Interface. One version is used for the Common IDB Interface hardware that is connected to the embedded PC. This common IDB Interface is referred to as the computer side interface. All other Common IDB Interfaces on the IDB, referred to as the device side interfaces, use the second version of the microcontroller code. The main difference in the two versions of code is that the computer side Common IDB Interface accepts all data from the bus, but device side Common IDB Interfaces only accept packets that have addresses matching their hardware address.

The complete schematic design of the Common IDB Interface is enclosed in Appendix A. Figure 4.1 shows a block diagram of the Common IDB Interface hardware design. The block diagram will be sufficient to describe important facts for designing the hardware interface between the Common IDB Interface and the radio's external bus.

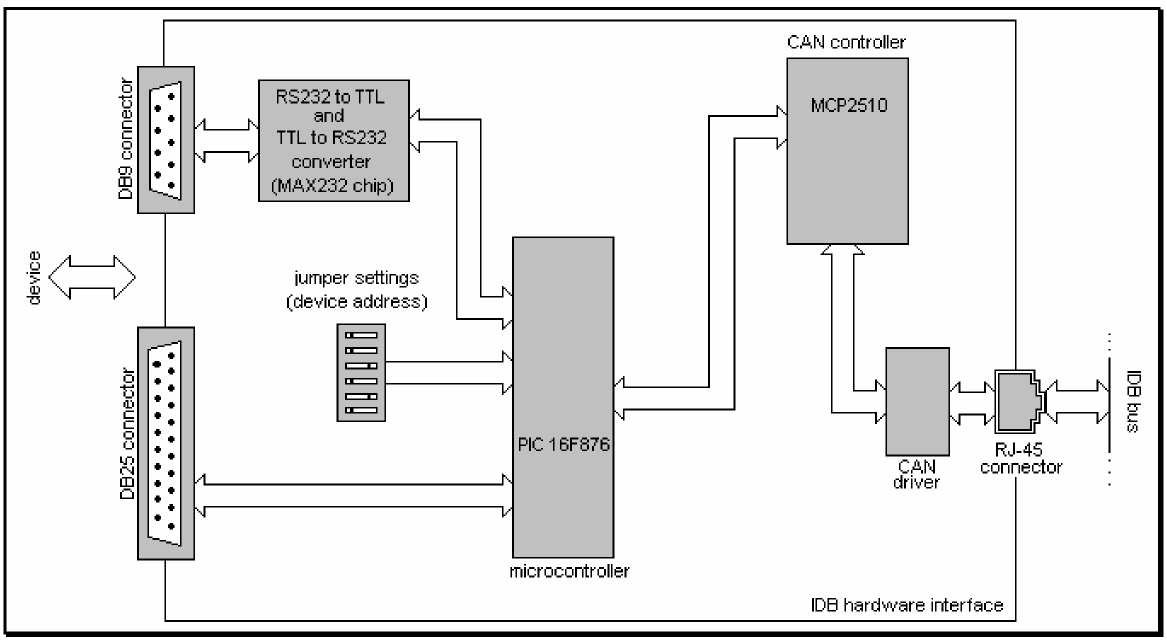


Figure 4.1 Block diagram of the common IDB interface hardware

When a data packet is transmitted over the IDB bus, every device linked to the system receives it. The IDB uses the Control Area Network (CAN) protocol as the method of data transmission. At startup, the code in the microcontroller sets an appropriate mask in the CAN controller according to the device address, so that the packet placed in the CAN controller is either processed through the microprocessor or rejected. The microcontroller reads the device address from the jumper settings. If an incoming data packet contains the address of the particular device, the microprocessor processes this packet. Some electronic devices in the Project54 system, such as the GPS, have a serial I/O (input/output) interface and use RS-232 or TTL level signals. For such devices the Common IDB Interface buffers the incoming packet and when the device is ready for the reception, it sends the data to the device. Data is sent using RS232 signals as well as using TTL signals. A TTL to RS232 converter (MAX232) converts the TTL logic of the output signal from the microcontroller to the RS232 signal logic and submits data packets to the device through the DB9 serial connector. At the same time, data is sent from the microprocessor to the DB25 serial connector as TTL signals.

If a serial I/O device wants to send data, the signal comes through the DB9 serial connector to the RS232 to TTL converter (MAX232) and goes into the microcontroller. Another option is to use the DB25 serial connector and receive TTL signals from the device. The microcontroller passes the data packet to the CAN controller. The CAN controller performs a kind of CSMA protocol. This means that the CAN controller transmits data packets as soon as it detects that the bus is free for sending data.

The operation of the Common IDB Interface with serial I/O devices is very important to us because the Motorola ASTRO radio is a serial device. In the next section we will see how the Common IDB Interface can be used to connect the radio to the IDB.

4.3 The Radio Interface Hardware

The major goal of this thesis was to connect the radio electronics to the IDB. To complete this new interface hardware, called the Radio Interface Hardware, was designed. The Radio Interface Hardware provides an electrical interface between the Common IDB Interface and the radio's external bus. The diagram shown in Figure 4.2 illustrates the placement of the designed Radio Interface Hardware. This figure shows that the Radio Interface Hardware is placed between the Common IDB Interface and the radio's external bus.

In order to connect lines coming from the Radio Interface Hardware to the external bus, a multi-port connector is used. The multi-port connector has four DB-25 ports. Inside the connector the lines of four 25-line ports are connected in parallel, so the connector can be used for monitoring and connecting four devices in parallel. The multi-port connector allows connecting the output of the Radio Interface Hardware on the radio's external bus in parallel with the control head output.

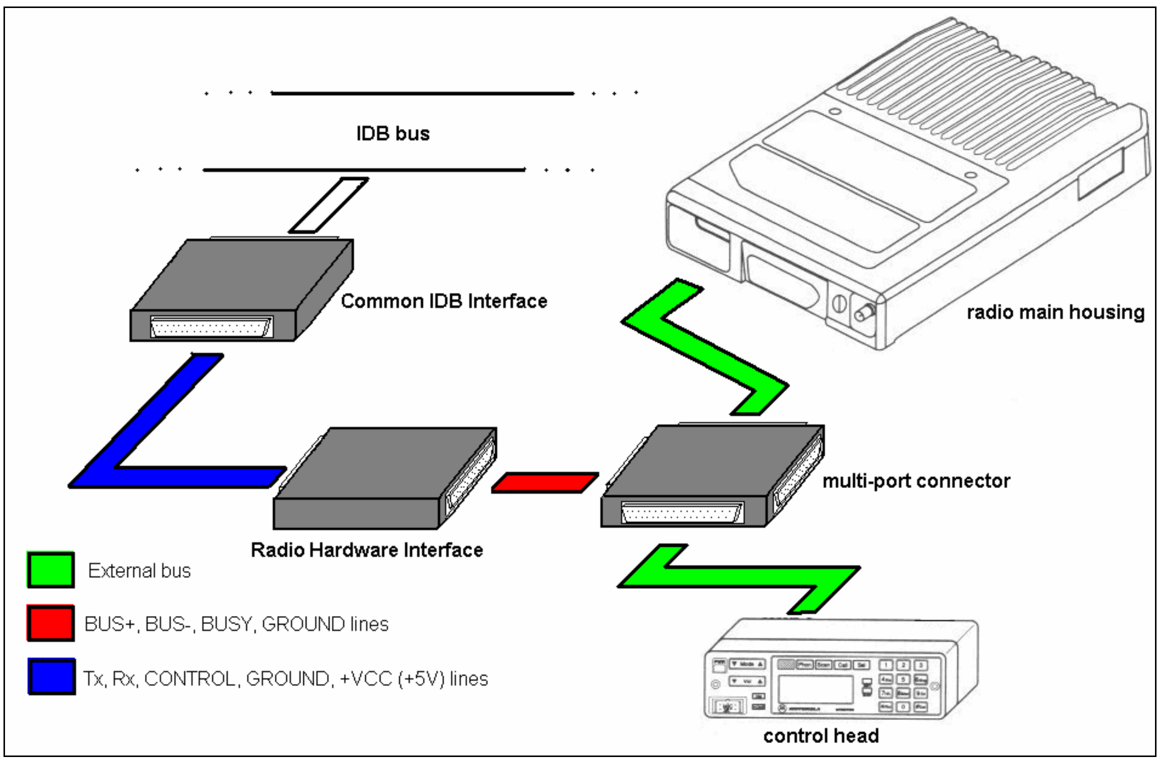


Figure 4.2 Radio device connectivity

The Radio Interface Hardware is connected to the Common IDB Interface. . The Common IDB Interface is set up to operate with a serial I/O device. It does not do any processing; the data is simply passed between the IDB and the output ports.

As mentioned before, there are two connectors at the output of the Common IDB Interface, the DB9 serial connector and the DB25 parallel connector. In this research, the DB25 connector was used. All of the signals available on the DB9 connector are also available on the DB25 connector. However, the voltage levels of the signals on the DB9 connector are levels defined by the RS232 protocol and the voltage levels on the DB25 connector are defined as TTL logic levels. Of course, the DB25 connector offers additional lines that can be used as control lines or can have some other purpose.

Since we use TTL signals at the output of the Common IDB Interface and the radio uses the RS-485 protocol on its external bus the Radio Interface Hardware has to provide the conversion between the TTL and RS-485 signals.

The Radio Interface Hardware schematic diagram is shown in Figure 4.3. The signals between the Common IDB Interface and the Radio Interface Hardware are the transmit line (Tx), the receive line (Rx), the control line (CTRL) and the handshake line that is included for future use (HSHK). The Radio Interface Hardware also receives the +5V power (VCC) and digital ground (GND) from the Common IDB Interface. The transmit line (Tx) is used for sending data from the embedded PC over the IDB bus to the radio. The receive line (Rx) is used for receiving data from the external bus of the radio. The control line (CTRL) is used to manipulate the radio's BUSY line and to handle the RS-485 driver (MAX485).

The signals connected to the radio's external bus are the bi-directional data bus lines, BUS+ and BUS-, and the bi-directional handshake BUSY line.

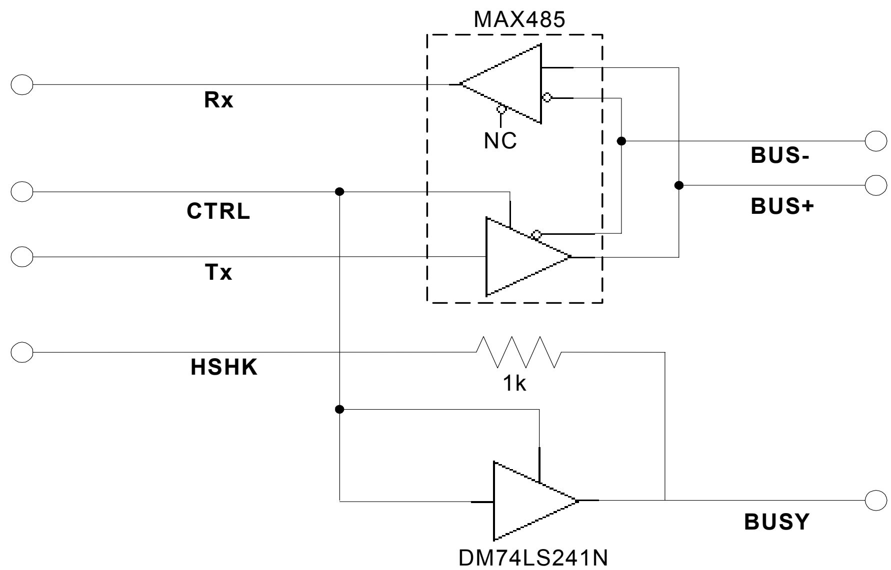


Figure 4.3 The Radio Interface Hardware - schematic design

The timing diagram, shown in Figure 4.4, will provide a better understanding of the circuit operation. If the embedded PC starts transmitting a data sequence at time T1, the data comes from the Tx line on the DB25 connector connected to the Common IDB Interface, and enters the MAX485 driver that converts the input signal from TTL logic to RS485 form. The driver transmits the signal over the BUS+ and BUS- lines.

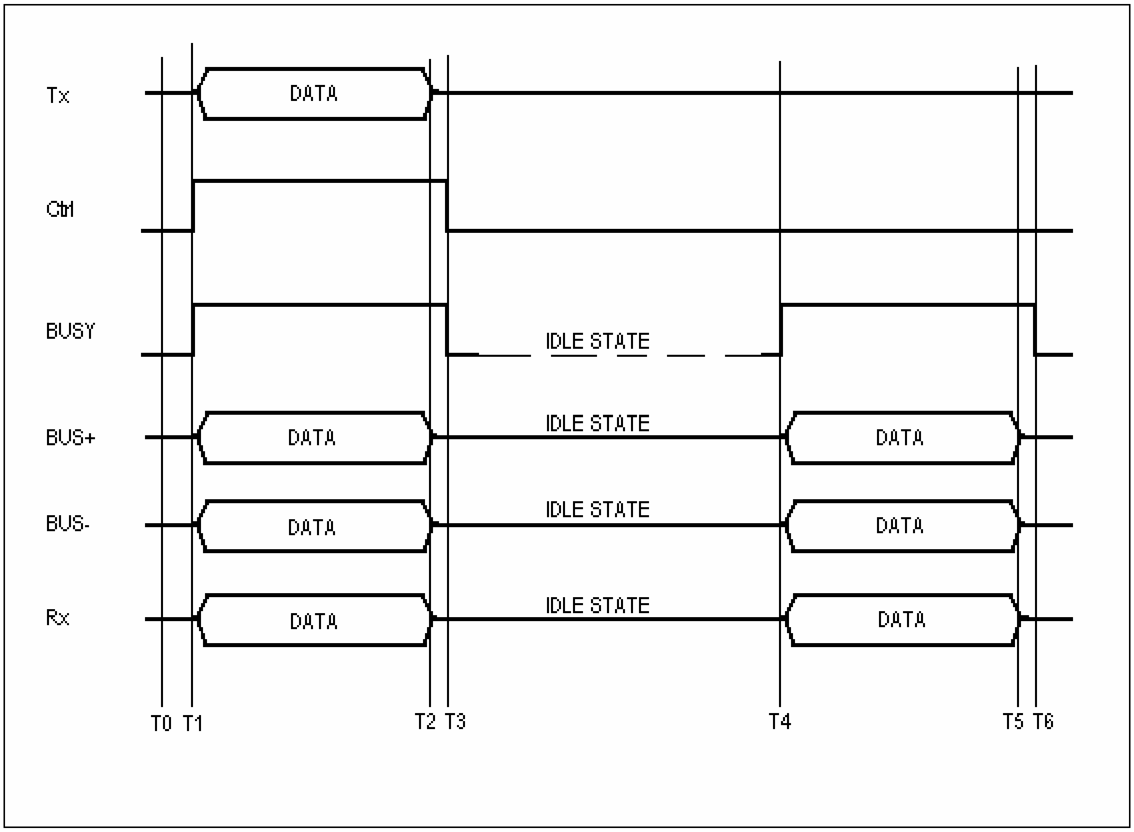


Figure 4.4 Timing diagram

At the same time the CTRL line, which was low, is set to high while transmitting. In that way we get the BUSY signal on the output of the Radio Interface Hardware. While the control signal is high the data on the transmission line is valid. When the embedded PC stops transmitting data at time T2, the control signal stays high for a few more milliseconds (between 4 and 6ms) and then it is set to low which means the BUSY line goes into the high impedance state. Consequently, other devices are allowed to use the external bus.

The reception of data is even more straightforward. The receive line Rx always monitors the bus and allows the embedded PC to receive data at any time. If we assume that the control head assembly transmits data over the external bus, it sets the BUSY line high, and data lines (BUS+ and BUS-) carry the traffic. The data on the data lines is converted to a TTL signal and sent over the Rx line to the IDB. Since the receive line is always open, even if the embedded PC transmits data packets the Rx line receives them.

It is important to note that in the current design, the Radio Interface Hardware may cause collisions because it does not check the BUSY line before data transmission. However, in order to enable future software versions to support collision avoidance, the external bus BUSY line was fed back through a resistor to a TTL sense pin on the IDB interface.

The resistor was included because testing showed that without it the output of a non-powered Common IDB Interface loads down the BUSY line. If the radio's Common IDB Interface does not have power the radio cannot be controlled through the embedded PC. However, if the BUSY line is stuck low, no other device on the radio external bus that wants to transmit will be able to pull the BUSY line high and transmit. This means that the officer could not use the radio's control head to operate the radio either. With the resistor in place the radio can be operated through the control head in case the Common IDB Interface accidentally loses power.

The chip layout in Figure 4.5 illustrates the implementation of the Radio Interface Hardware.

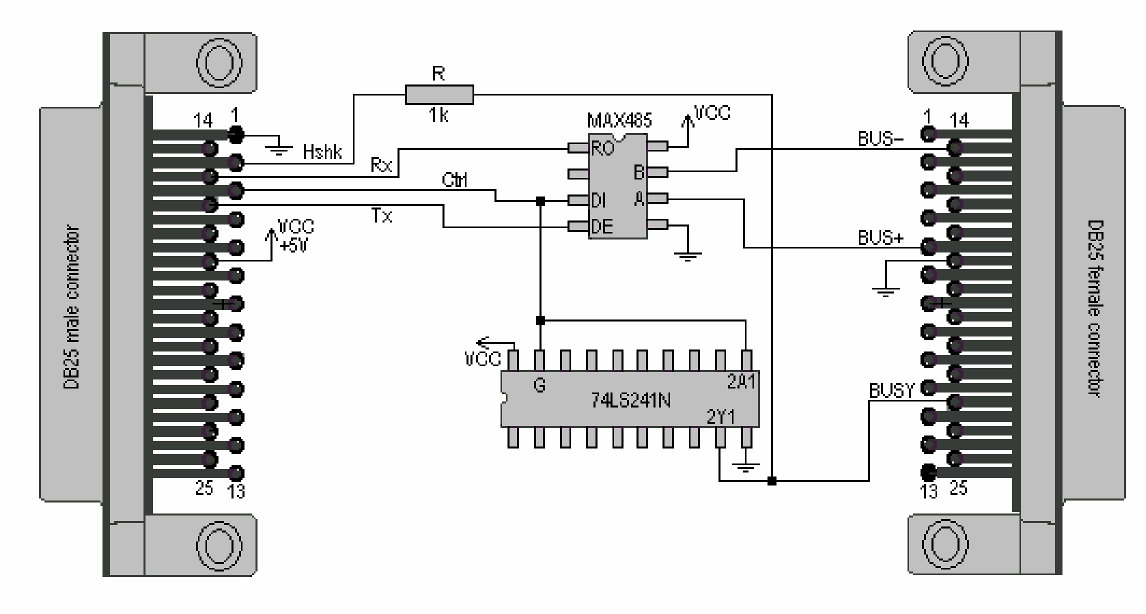


Figure 4.5 The Radio Interface Hardware – chip layout

Figure 4.5 shows that the design is very simple. It has only three components. Two of them are integrated circuits and one is a resistor. Therefore, no active discrete components are needed which makes the circuit easy to manufacture, robust, cheap and easy to debug. Datasheets for the components used in the Radio Interface Hardware design are given in Appendix A.

A schematic design of the Radio Interface Hardware is similar to the design of the test hardware shown in Figure 3.3. The difference is that the test hardware needed to perform a conversion between the RS-232 signal of the computer's output and the RS-485 signal of the Radio Interface Hardware. This required an RS-232 driver to be used in order to convert the RS-232 signal to a TTL signal. Since TTL signals are available from the Common IDB Interface, the Radio Interface Hardware does not need an RS-232 driver.

A picture of the Radio Interface Hardware is shown in Figure 4.6. The figure shows that the Radio Interface Hardware is designed to fit in a shell of a typical DB25 gender changer so no special case is required. This way the Radio Interface Hardware is easy to build, compact and rugged.

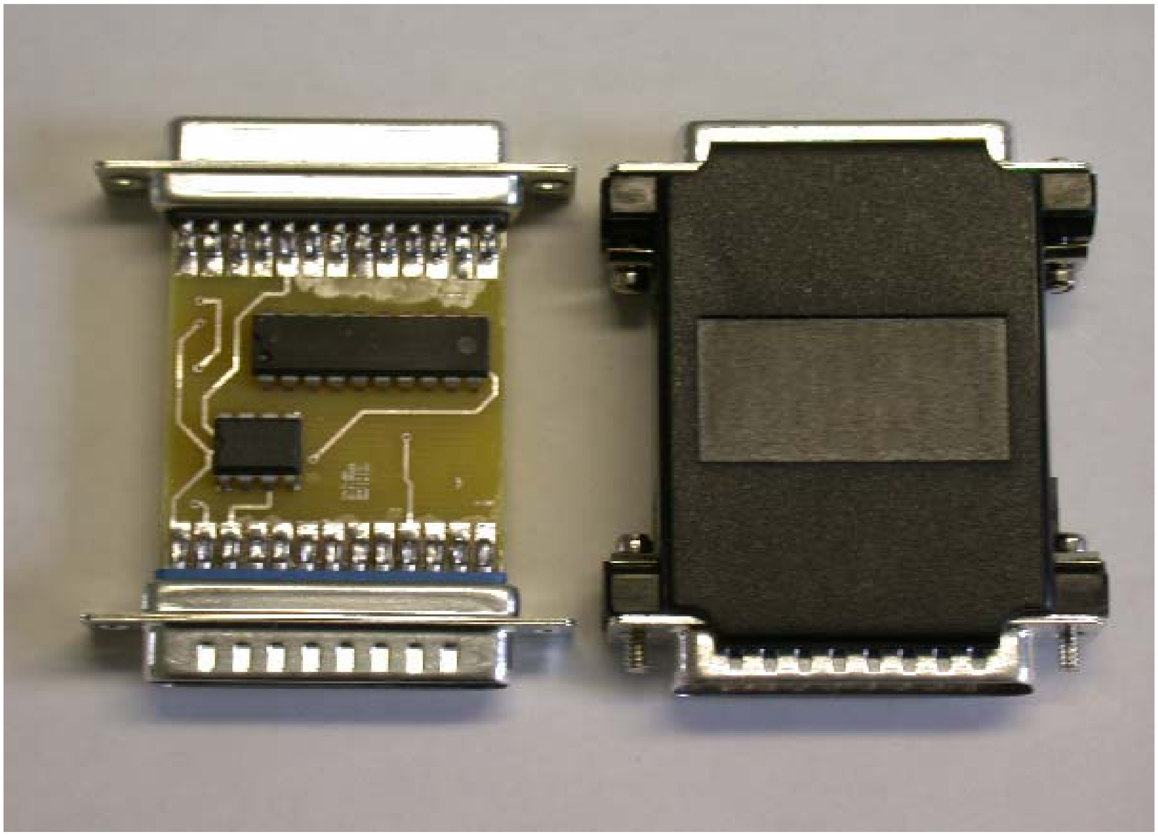


Figure 4.6 The Radio Interface Hardware

CHAPTER V

SOFTWARE DEVELOPMENT

5.1 Introduction

The third proposed step of this research was to create a Project54 software module to control the radio and to implement a speech user interface to the radio. The Project54 system software provides a way to integrate software modules that control individual electronic devices in the police cruiser. The block diagram of the Project54 system software is shown in Figure 5.1.

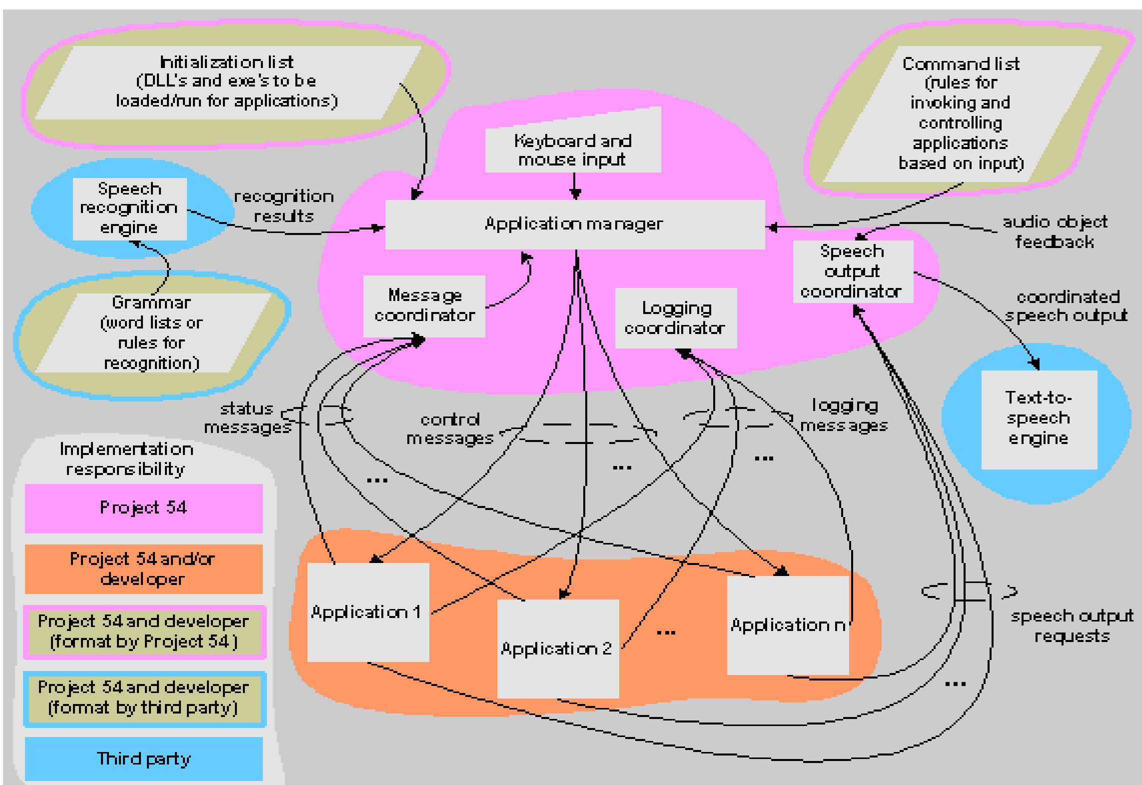


Figure 5.1 Project54 system software block-diagram

Every electronic device included into the Project54 system has its control application within the system software. The execution of the applications is managed by a central application called the application manager. Based on the speech input, mouse and keyboard input and data about the system, the manager creates, calls and sends messages to the control applications. The manager also receives feedback from the applications. The status messages from the applications are forwarded to the application manager through the message coordinator. The message coordinator is one of three modules that control applications talk to. The second module is the logging coordinator, which provides a centralized means for all applications to log events and errors. Finally, the third module is the speech output coordinator that is responsible for an orderly output of speech messages to the user.

The system software is developed so that individual control applications can be developed and changed easily. For this reason, the application manager is the only part of the system that interacts with a speech recognition engine. As a result, individual control applications do not need to know how to do this. Developers only need to be concerned with the grammar files used by the speech recognition engines. Grammar files list a set of rules that all valid utterances have to follow. An industry standard, called Speech Application Programming Interface 4 (SAPI 4), describes the standard format for grammar files.

The application manager contains three COM (Component Object Model [11]) objects that receive messages from individual control applications. These COM objects are the logging coordinator object, the message coordinator object and the speech output coordinator object. Every control application has to implement a message handler object that is responsible for receiving messages from the application manager. Figure 5.2 shows the COM objects relevant for application developers.

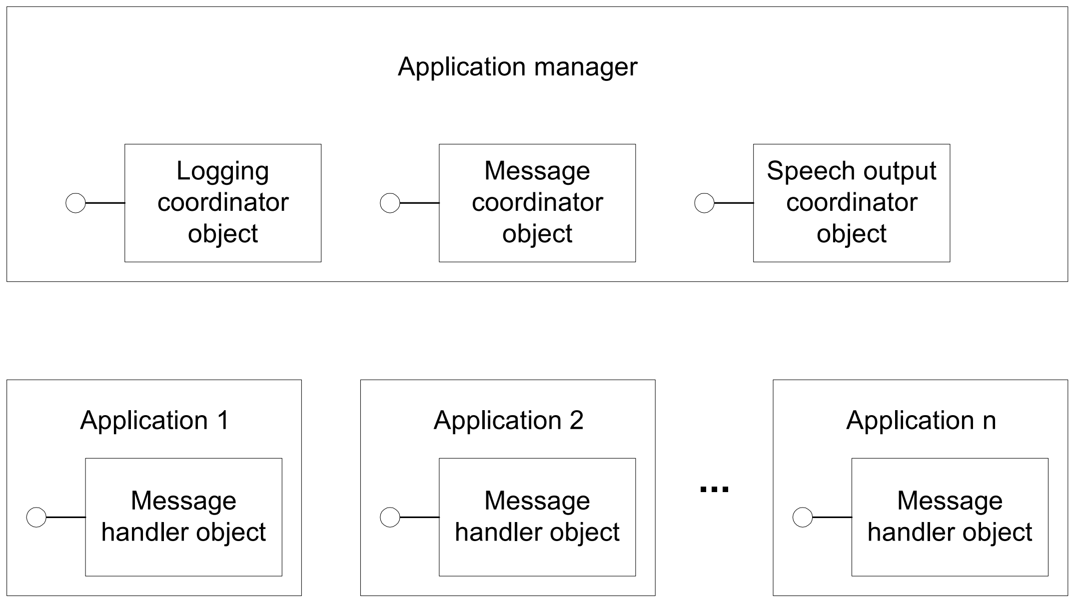


Figure 5.2 COM objects relevant for developers in the Project54 system software

The control application implemented for this research is called the Radio Control Application. The Radio Control Application is implemented in Project54 as a Component Object Model (COM) component.

5.2 Radio Control Application

In order to control the radio using the embedded PC, the Radio Control Application was developed. It uses a message handler COM object to talk with the application manager.

5.2.1 Channels, zones, troops and grammar files

For the description of the Radio Control Application it is important to be familiar with terms like channel, zone, troop and grammar file, and how they are related to each other in the Radio Control Application.

The NHSP is organized into six troops, Troop A through Troop F. Each troop covers about one sixth of the area of New Hampshire. Within each of these geographical areas there are a large number of local law enforcement and emergency agencies. Most of these agencies have radio networks with assigned channels. Channels are radio frequencies used for communication. The officers of the NHSP monitor these channels. The ASTRO radios are therefore programmed to allow communication on these channels. Of course, the ASTRO radios also provide a police officer a communication link with NHSP dispatcher centers and with other NHSP officers. This communication happens on NHSP channels.

Motorola introduced zones to simplify searching for a given channel. Zones are logical groups of channels. On the NHSP ASTRO radios the channels within each troop are divided into several zones. There are approximately 15 to 20 radio channels within each zone.

As mentioned before, grammar files list a set of rules that all valid utterances have to follow. The format of grammar files used in Project54 is described by the SAPI 4 industry standard. Since the Radio Control Application implements a speech user interface, all the voice commands the police officer can utter in order to select a desired channel, or issue a radio control command, are separated into seven grammar files. This is done in order to improve the performance of the voice recognition engine. All of the grammar files contain radio command utterances. The main grammar file contains voice commands for selecting the desired police troop. When the police officer selects a troop the Radio Control Application loads one of the other six grammars. On top of the radio control commands these six grammars contain commands related to switching radio channels. Six grammar files were created to organize the utterances necessary to switch between channels within each of the six troops. All the channels from all the zones within a given troop are included in that troop's grammar file. This means that, when using the Radio Control Application, the police officer does not have to be concerned with zones.

We also created a grammar file that includes all the utterances necessary to issue radio control commands and to select any radio channel within any police troop. This large grammar file may be useful if the performance of the speech recognition engine, used in Project54, does not need to be improved by restricting the number of valid utterances.

5.2.2. The graphical and speech user interface

In this section the graphical and speech user interface of the Radio Control Application will be described. When the Project54 application manager is started and the main screen of the Project54 application is shown, the user chooses the "Radio Controls" button to access the Radio Control Application. Figure 5.3 shows the Project54 main screen. The Radio Controls button is circled with a yellow line.

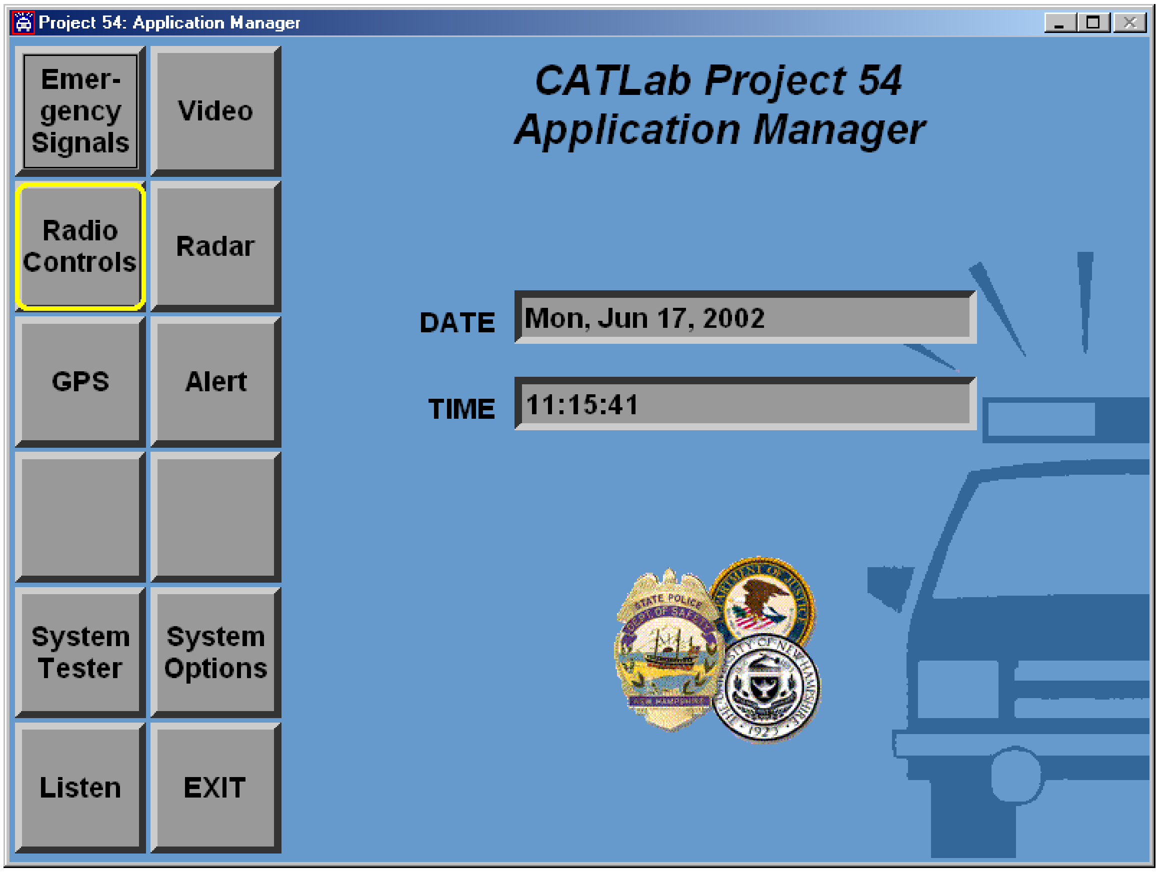


Figure 5.3 Project54 – Main Screen

When the Radio Controls button is pressed the Radio Control Application front panel is shown (see Figure 5.4). This window shows the main menu of the application user interface. From here, the police officer selects a police troop to communicate with or issues a radio command.

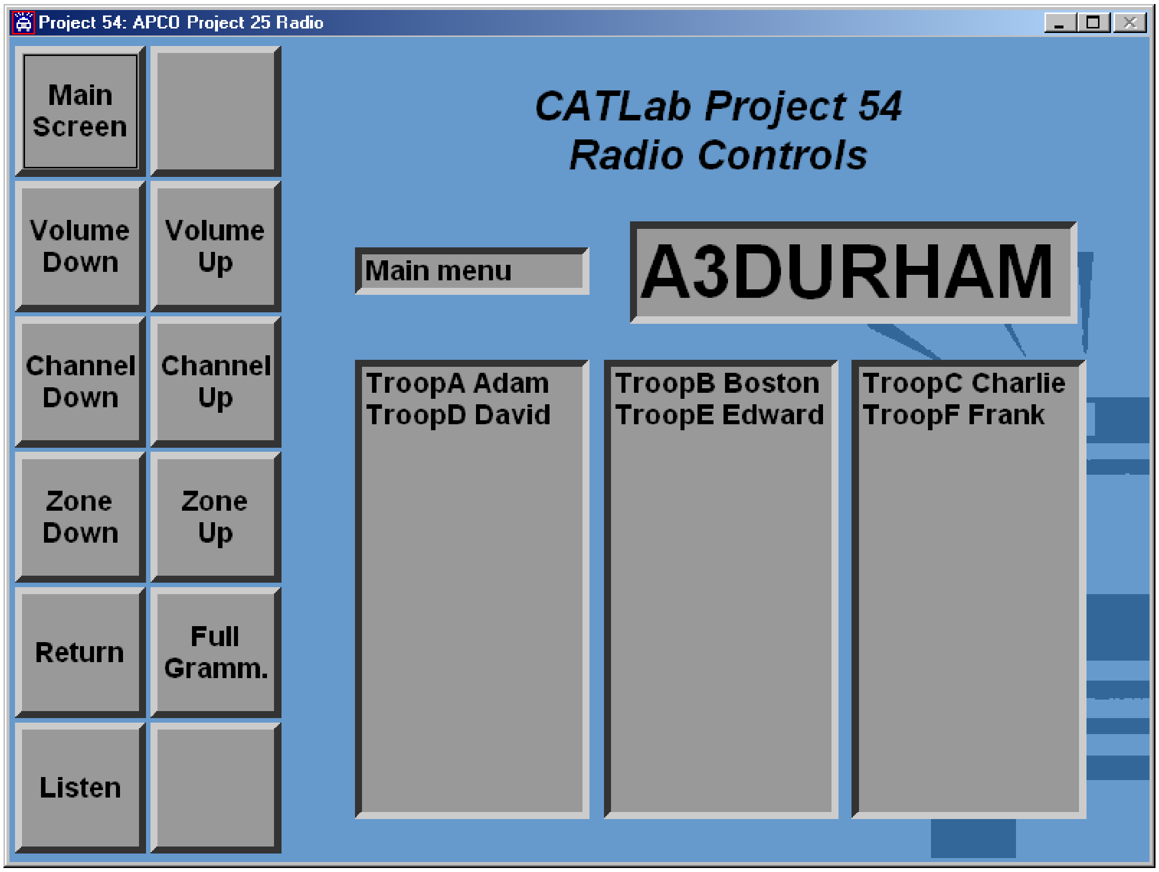


Figure 5.4 Radio Control Application window

The command buttons are placed on the left side of the application window (circled with a yellow line in Figure 5.5). Those command buttons are:

- **Volume Up, Volume Down** – volume adjustment;
- **Channel Up, Channel Down** – channel selection;
- **Zone Up, Zone Down** – zone selection;
- **Full grammar** - full grammar selection (this gives the ability to choose any radio channel without specifying the troop name);

- **Return** - button for enabling the return from one of the loaded grammar files to the main grammar file;
- **Listen** - speech input button;
- **Main Screen** - return to the main screen.

The buttons for volume, channel and zone manipulation operate exactly the same as the buttons on the radio control head assembly.

In Figure 5.5 the text field, outlined with a red line, corresponds to the display on the radio control head. The content of this display is always the same as the content of the radio control head display.

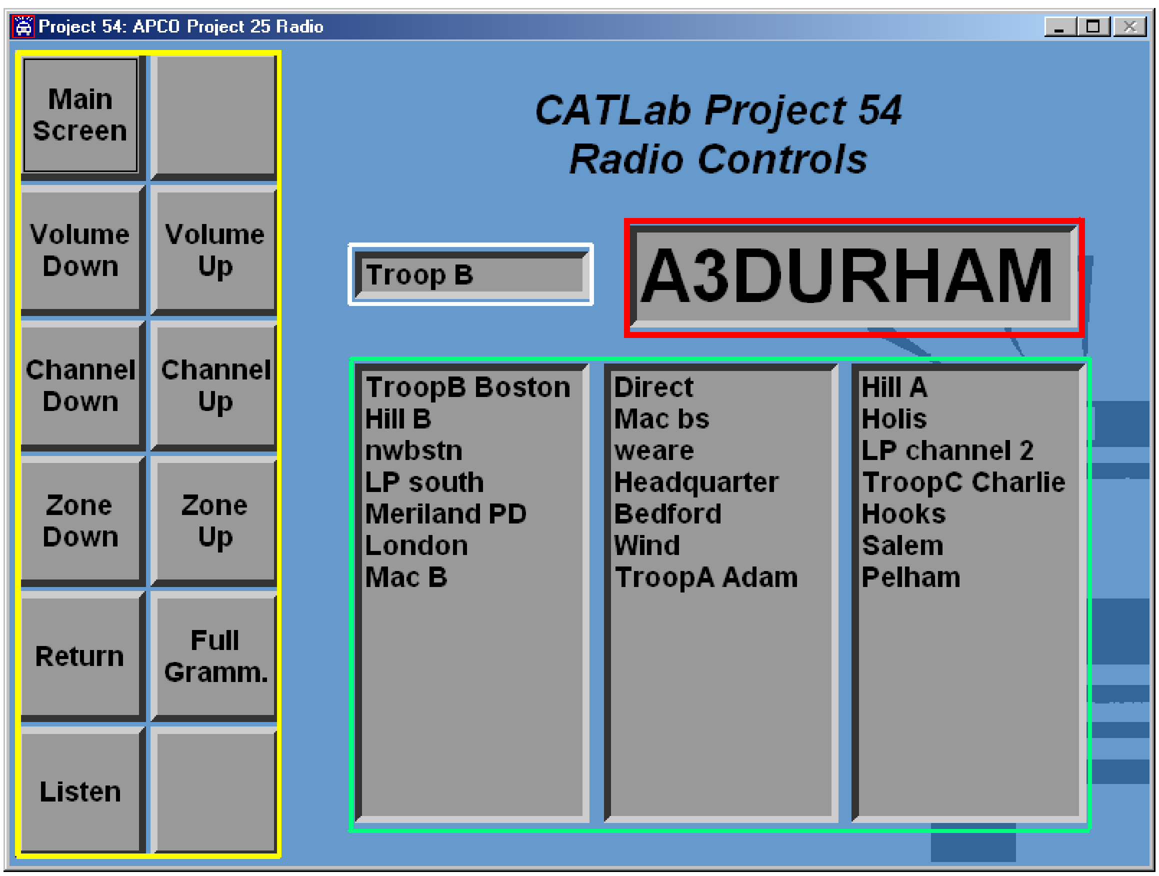


Figure 5.5 Radio Control Application window – window sections

In Figure 5.5, the text fields outlined with a green line contains voice command names that a police officer should utter in order to select the desired channel. The software accepts voice commands which are defined by grammar files.

The text field outlined with a white line in Figure 5.5 helps the police officer know which grammar file is loaded at that moment. In other words, this text field shows the troop name from which the police officer can choose the radio channels outlined in green. As we saw in section 5.2.1 every troop has a corresponding grammar file.

5.2.3 The registry

In order to simplify possible changes of the command names that police officer should utter, the command names are listed in the registry (Figure 5.6). If there is a need to change the command names, the user should only change the grammar file and the registry contents without recompiling the Radio Control Application. The NHSP is organized into six troops, Troop A through Troop F. The registry contains six keys corresponding to the six troops. The key for the troop named Troop B, for example, is //HKEY_LOCAL_MACHINE/Software/Catlab/Project54/Radio/TrpB. The command names for channels in Troop B are listed within that key. When a police officer selects Troop B, the Radio Control Application loads the list of the command names from the "TrpB" key (Figure 5.6) (the Radio Control Application also indicates the loading of the Troop B grammar file into the speech recognition engine). Only the list of the command names in the column named "Data" is used. Names in the column named "Name" have no meaning.

The GrammDisplay key (//HKEY_LOCAL_MACHINE/Software/Catlab/Project54/Radio/GrammDisplay) lists the names of the six troops (Figure 5.4). This represents the list of the main menu's command names. Again, only the list of the command names in the column named "Data" is used while the names in the column called "Name" have no meaning.

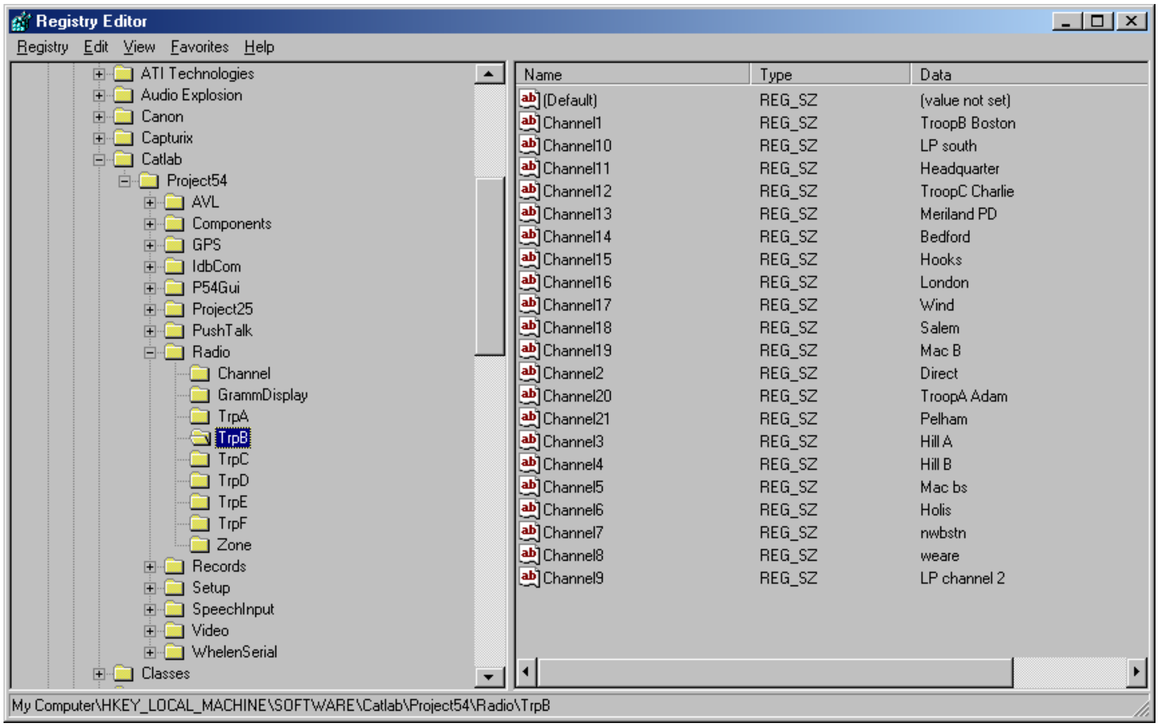


Figure 5.6 Registry contents – channel names

5.2.4. Radio Control Application algorithm

The project directory of the Radio Control Application consists of the following files:

- Source Files
  - **AppCode.cpp**
  - **AppID.cpp**
  - Lheap.cpp
  - P54AppBase.cpp
  - P54AppBase.def

- P54IDS.cpp
- **RadioGUI.cpp**
- REGISTRY.cpp

- Header Files
  - IdbComlib.h
  - Lheap.h
  - P54Guilib.h
  - P54Iface.h
  - REGISTRY.h
  - RegComlib.h
  - **RadioGUI.h**
- Resource Files
  - P54Guilib.lib
  - IdbComlib.lib
  - RegComlib.lib

The files shown in boldface were designed specifically for this application. The other files are identical to the files with the same name used in other Project54 applications. Application-specific code is implemented in the following routines:

- void AppHandleSpeech(LPWSTR speech_input) {};
- void AppHandleAlert(LPWSTR alert_message) {};
- void AppMain(void) {}.

These routines can be found in the file AppCode.cpp. These three routines run on independent threads. The RadioGUI.cpp file contains the RadioGUI thread for firing up and controlling the Radio Control Application graphical user interface (GUI). This file also contains most of the function definitions required for performing the algorithm of the Radio Control Application.

5.2.5 AppHandleSpeech procedure

As we saw in Section 5.1, every Project54 control application has to implement a message handler object. The AppHandle speech procedure is the primary procedure implementing the Radio Control Application's message handler object. It contains numerous "if" statements which are used to find the match for the input speech command. Figure 5.7 shows the block algorithm of the procedure.

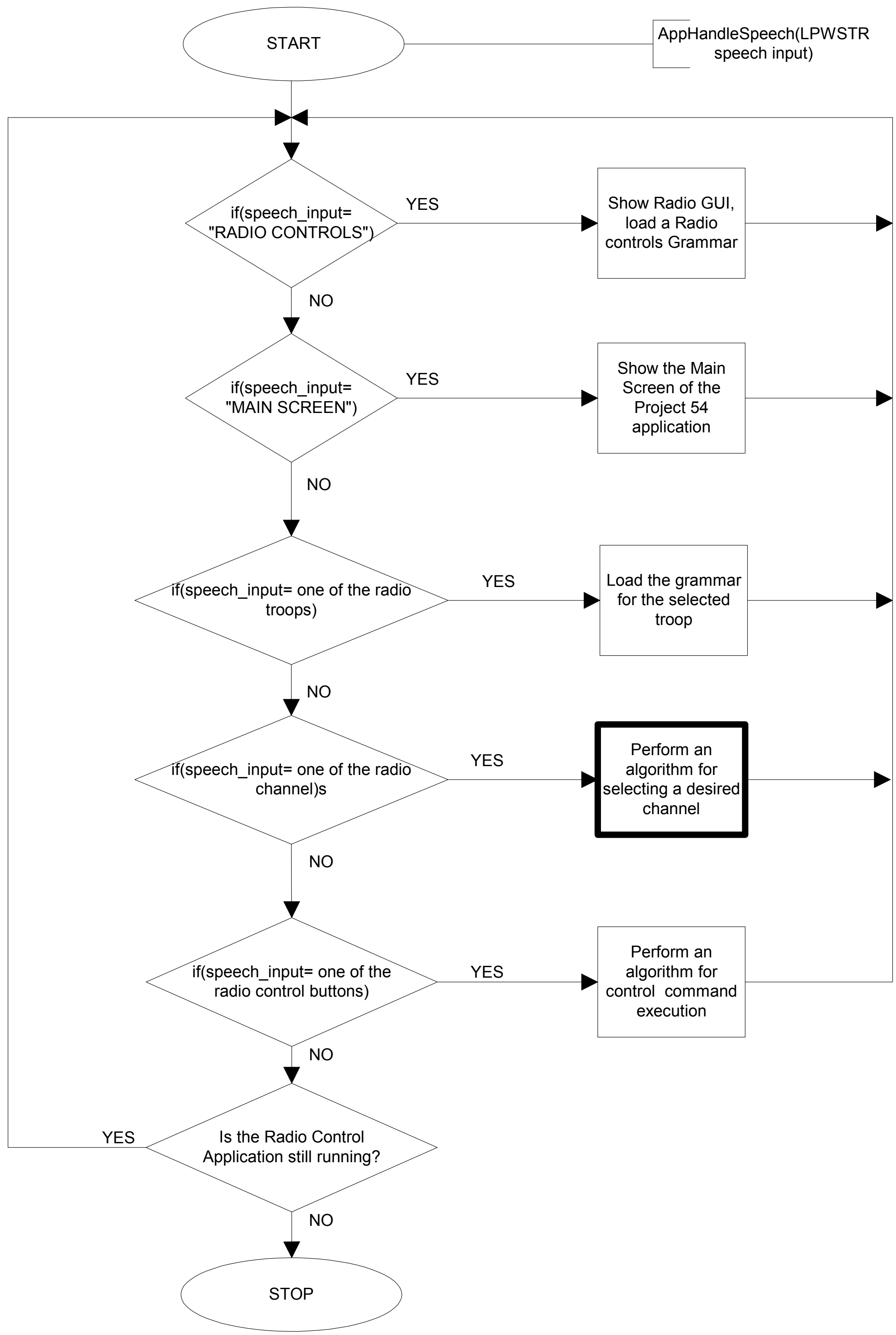


Figure 5.7 Block diagram of the AppHandleSpeech procedure

The “RADIO CONTROLS” and “MAIN SCREEN” speech inputs simply results in displaying the Radio Control Application and the Project54 main menu window, respectively. A command for any of the radio control buttons calls the appropriate function that sends a data sequence to the radio through the IDB. The data sequence initiates the execution of the desired radio control command (e.g. changing the volume). If the user utters a troop name, a grammar file for the selected troop is loaded. If the speech input is one of the channels, the embedded PC performs the algorithm described in the block diagram shown in the Figure 5.8 (this algorithm is represented by the bold box in Figure 5.7).

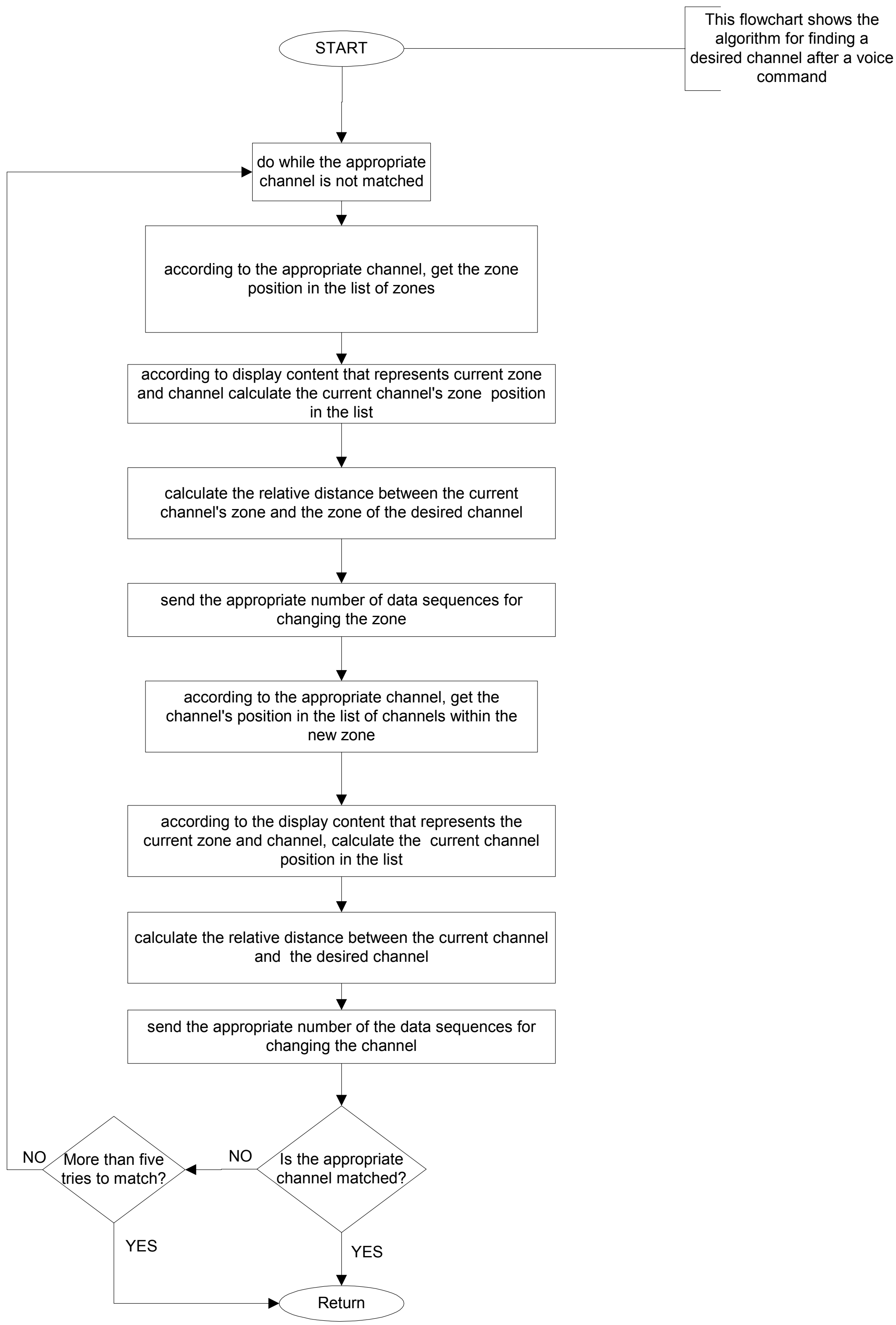


Figure 5.8 The algorithm for matching the appropriate channel to the voice command

The radio uses a large number of channels which are grouped into zones. Every NHSP troop is divided into several zones. When the police officer operates the control head of the radio and tries to reach the desired channel, he or she first has to find the zone where the desired channel is. Once the police officer reaches the correct zone, he or she searches for the desired channel within the selected zone. The officer changes zones and channels by pressing the appropriate buttons on the control head. In response to every press of a button the control head sends a data sequence to the microcontroller unit (MCU) over the radio serial bus. These data sequences change the zones and the channels in increments of one. The basic idea of the algorithm of Figure 5.8 is to calculate the number of data sequences that should be sent in order to reach the desired channel.

The registry stores a list of the zones and a list of the channels. The list of the zones is under //HKEY_LOCAL_MACHINE/ Software/Catlab/Project54/Radio/Zone. Figure 5.9 shows part of the list in the registry that lists zone names and their distances from a reference zone. We declared the zone called "Zone 1" as the reference zone. The distance of a zone from the reference zone is the number of times the officer has to press the "Zone Up" key on the control head in order to reach that zone from the reference zone. Using this logic the number assigned to the reference zone is zero.

The list of the channels is under //HKEY_LOCAL_MACHINE/ Software/ Catlab/ Project54/ Radio/ Channel. Figure 5.10 shows part of the list in the registry that lists channel names and their distances from a reference channel. One channel in each zone is declared as the reference. We declared the first channel in every zone as the reference channel for that zone. The distance of a channel from the reference channel is the number of times the officer has to press the "Channel Up" key on the control head in order to reach that channel from the reference channel. Using this logic the number assigned to the reference channel is zero.

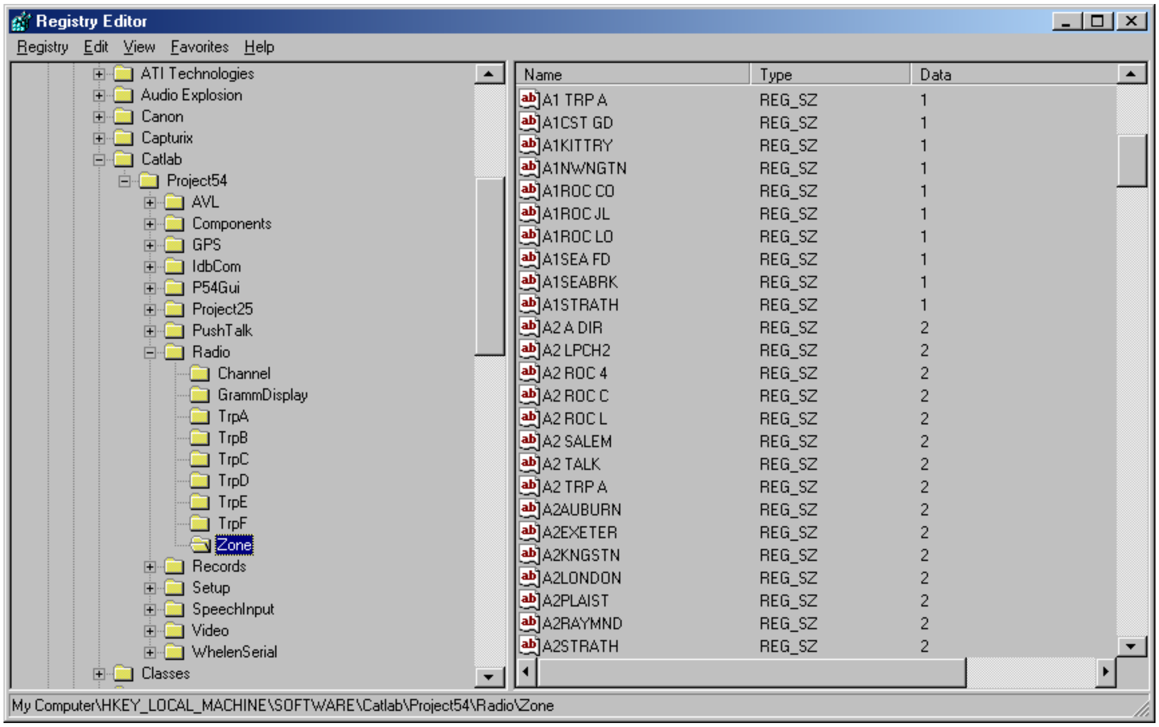


Figure 5.9 Registry contents – zone directory

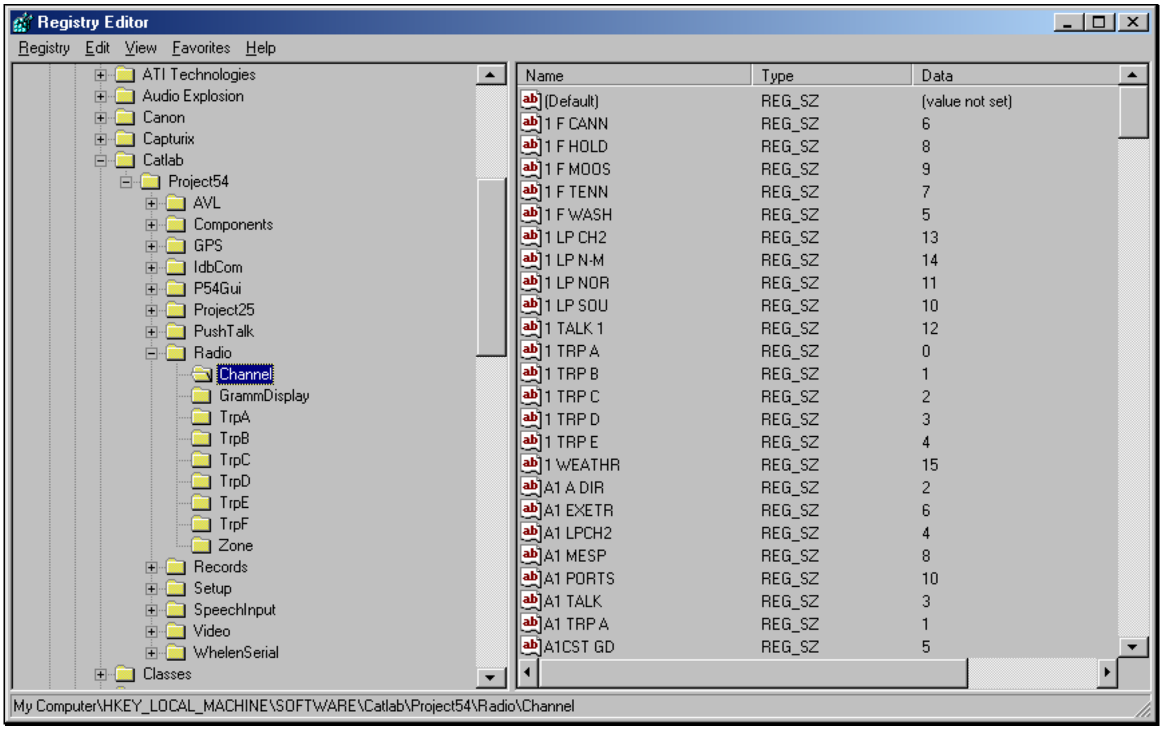


Figure 5.10 Registry contents – channel directory

The algorithm starts by obtaining the distance of the desired zone from the reference zone. The next step is to obtain the distance of the zone currently selected on the radio. The two distances are subtracted from one another and an additional global variable is introduced to save the information about the sign of the subtraction. The calculated number represents how many data sequences have to be sent in order to reach the zone where the desired channel is located. The introduced global variable is used for the direction of reaching the zone. In other words, it is used to decide if the commands to be sent to the radio are Zone Up or Zone Down.

Once in the appropriate zone, the algorithm performs the same steps only this time with the channels. The algorithm starts by obtaining the distance of the desired channel from the reference channel. The next step is to obtain the distance of the channel currently selected on the radio. The two distances are subtracted from one another and an additional global variable is introduced to save the information about the sign of the subtraction. The calculated number represents how many data sequences have to be sent in order to reach the desired channel. The introduced global variable is used for the direction of reaching the channel. In other words, it is used to decide if the commands to be sent to the radio are Channel Up or Channel Down.

5.2.5.1 Zone and channel calculation functions

A collection of functions provides supporting operations for controlling the radio. These functions and their short descriptions are shown in Table 5.1.

Function	Description
jumpCalc	Get distances from the registry assigned to a given zone or channel
relativeCalc	Calculate the number of steps in order to switch to the desired zone or channel
initialZonecalc	Calculate the distance of the currently selected zone from the reference zone
initialChanncalc	Calculate the distance of the currently selected channel from the reference channel
multipleZonechange	Send multiple data sequences for changing radio zone
multipleChannelchange	Send multiple data sequences for changing radio channel

Table 5.1 Zone and channel calculation functions

- **void jumpCalc(wchar_t *exact_channel,wchar_t *reg_fold, int &aps_zone_jump, int &aps_chann_jump)** – this function reads the registry and obtains the distances assigned to a zone or a channel. Depending on the selected registry folder, either the abs_zone_jump or the abs_chann_jump variable is used. The other variable will be assigned the value of zero.

Variable	Description
exact_channel	Wide character pointer to the unique zone or channel name string e.g.: L”A3DURHAM”
reg_fold	Wide character pointer to the unique folder name in the registry e.g. : L”Zone” or L”Channel”
abs_zone_jump	Reference for the zone integer value (absolute distance from reference zone) read from the registry
abs_chann_jump	Reference for the channel integer value (absolute distance from reference channel) read from the registry

Table 5.2 The jumpCalc function

- **void relativeCalc(int absolute, int initial_num)** – this function calculates the relative distance between the desired and currently selected zone or channel on the radio. This difference represents the number of data sequences that should be sent to the radio in order to change the zone or channel. This function also updates a global variable named “sign” which is used for determining the direction for reaching the zone or channel.

Variable	Description
absolute	Integer containing the distance of the desired zone or channel from the reference zone or channel
initial_num	Integer containing the position of the currently selected channel on the radio

Table 5.3 The relativeCalc function

- **void initialZonecalc(void) –** this function calculates the distance of the currently selected zone from the reference zone.
- **void initialChanncalc(void)** - this function calculates the distance of the currently selected channel from the reference channel.
- **void multipleZonechange(void)** – this functions contains an algorithm for sending the data sequences for changing the zones for a previously calculated number of times. A global variable called “times” is introduced in order to control

how many times the function for changing zones is called. This function uses variable “sign” for determining the direction for reaching the desired zone.

- **void multipleChannelchange(void)** - this functions contains an algorithm for sending the data sequences for changing the channels for a previously calculated number of times A global variable called “times” is introduced in order to control how many times the function for changing channels is called. This function uses variable “sign” for determining the direction for reaching the desired channel.

5.2.5.2 Radio command functions

Radio command functions include all functions which execute an algorithm for sending data on the IDB bus. These functions are listed in Table 5.4.

Functions	Description
setVolumeUp	Increase volume
setVoumeDown	Decrease volume
setChannUp	Switch a radio channel forward
setChannDown	Switch a radio channel backward
setZoneUp	Switch a radio zone forward
setZoneDown	Switch a radio zone backward

Table 5.4 Radio command functions

These functions execute almost the same algorithm. The only difference between them is which data sequences are sent on the IDB by a particular function. A more detailed description of these functions is given in section B.4.1 of Appendix B.

5.2.6 AppHandleAlert procedure

The AppHandleAlert procedure is a small routine for starting the Radio Control Application window. Within this procedure the Radio Control Application GUI is initialized.

5.2.7. AppMain procedure

The AppMain procedure is called when the Radio Control Application is loaded. The AppMain procedure contains an infinite loop for capturing data from the IDB. Several functions within the loop execute in order to buffer and process the captured data. The function that interprets control head display contents, called “processRadiomessage”, performs an algorithm based on information pertaining to Motorola confidential documentation which describes the serial bus protocol. A description of the serial bus protocol is given in Appendix B. Section B.4.2 of Appendix B explains the operation of the “processRadiomessage” function.

5.2.8 RadioGUI procedure

This procedure contains setup code for the radio graphical user interface (GUI) and contains an infinite loop that waits for user action with the application graphical user interface. Every time a GUI button is pressed, the proper function is called and, if necessary, a data sequence is sent on the IDB.

CHAPTER VI

SYSTEM TESTING

6.1 Introduction

The final step of the research was to test the radio control system. The system testing included various testing on the Radio Interface Hardware, the Radio Control Software and the user interface. The test s were performed in a laboratory settings.

6.2 Testing configuration

A middle section of a police cruiser, called "the LabCar", was used as the testing system. The "LabCar" is shown in Figure 6.1. The testing system within the "LabCar" is equipped with an IDB network and various integrated electronic devices. This testing system represents the full Project54 system. The electronic devices integrated in the "LabCar" were lights, siren, video, GPS and the embedded PC as the main control unit. In order to test the radio control system, the radio was integrated. The Common IDB Interface for the radio was connected to the IDB. After the Common IDB Interface, the radio with the Radio Interface Hardware was connected as shown in Figure 4.2. The embedded PC in the "LabCar" ran the full Project54 system software. In order to integrate the Radio Control Application in the Project54 system, the application was registered into the system. All eight grammar files pertaining to the Radio Control

Application were included into the Project54 system files. The eight grammar files each contained approximately a total of twenty words.

Figure 6.1 “LabCar” with IDB network and control units assembled behind the car

6.3 Testing procedure and test results

After integrating the radio control system in the Project54 system in the “LabCar”, a qualitative test was performed. After booting the Project54 software, the user interface for the Radio Control Application was examined.

After we made sure that the Radio Control Application was initiated properly, the tests started with the examination of the graphical user interface. All buttons created for the application were examined. The tests included adjusting the volume, changing channels and zones, and comparing the two displays on the radio control head and on the application window. The functionality of every button was tested and we concluded that the buttons performed theirs functions robustly and reliably.

The speech user interface was tested next. Pressing the push-to-talk button enabled the user to control the device by voice. Therefore, by using the push-to-talk button, all voice commands assigned to the Radio Control Application were uttered in order to test the speech recognition engine. The engine showed a high percentage in matching the voice commands. The radio completed all actions expected to be performed.

6.4 Remaining tests

Even though the "LabCar" represented a suitable environment for various tests, the radio control system is left to be tested in the environment of a patrol car. One of the "LabCar's" drawbacks is that all test in this system were taken without any background noise. The background noise has a large influence on the speech recognition engine and therefore on the accuracy of the actions performed by applications after exploring the voice commands.

Regarding the Radio Control Application particularly, tests on the collisions on the radio's external bus should be performed. In laboratory settings, no collisions were found.

CHAPTER VII

CONCLUSION

The main goal of the thesis was to integrate the VHF radio, used by the NHSP, into the Project54 system. The main goal comprised three partial goals. The first partial goal of this thesis was to create a hardware interface to connect the radio electronics to the IDB. The second partial goal was to create a software application to control the radio within the Project54 system. The third partial goal was to allow voice control of the radio.

The first proposed step of this research was to examine the radio electronics and find a suitable way to interface it to the IDB. We examined the hardware of the radio and found that it can be interfaced to the IDB through the external bus connecting the control head to the main housing. We found that electrically the external bus is an RS-485 bus. We also determined what the protocol used on this bus is. Having completed this step we determined the hardware requirements for the new hardware interface and the necessary signaling sequences for controlling the radio.

The second proposed step was to design and implement a hardware interface between the radio and the IDB, based on the results of the first step. Since all electronic devices in the Project54 are connected to the IDB using the Common IDB Interface hardware, the outputs of the Common IDB Interface were examined. It was found that the most suitable output is the output from the DB25 parallel connector. That output provides TTL signals that are the most suitable for the new hardware interface, called the Radio

Interface Hardware design. After designing the interface hardware it was implemented and connected to the IDB.

The third step was to create a Project54 software module to control the radio and to implement the speech user interface to the radio. Every electronic device included into the Project54 system has its control application within the system software. The control application for the radio is called the Radio Control Application. Based on the results of the examinations on the radio's external bus protocol, the Radio Control Application was created and voice control of the radio was allowed.

The final step was to test the radio control system. The system testing was performed in the laboratory settings. The radio control system emerged as a robust and reliable system. In conclusion we accomplished the three partial goals set at the beginning of the research and thus we have also accomplished our main goal of integrating the VHF radio into the Project54 system.

CHAPTER VIII

SUGGESTIONS FOR FUTURE DEVELOPMENTS

This chapter offers several suggestions for further enhancements of the designed system.

The Radio Interface Hardware sends messages from the IDB to the radio's external bus. Traffic on this bus is controlled using the BUSY line: when a device transmits on the bus it pulls this line high. Other devices wanting to transmit check the BUSY line and while it is high they cannon transmit. The current version of the Radio Interface Hardware does not examine the BUSY line before transmitting. Since this can be a cause of possible collisions, a future enhancement may provide a hardware handshake in order to prevent this.

The current version of the Radio Interface Hardware is designed to work with the Motorola ASTRO radio deployed with the W7 and W9 control head models. In the future, there might be a need for the integration of other control head models or radios from other manufacturers.

Since Motorola police radios, integrated into the Project54 system, can be programmed, many new features can be added. In that way the Radio Control Application can spread its field of activities and make use of the radio more efficiently. For example, the control of the channel encryption and the ability to enable and disable the scan radio mode could be added. The encryption would make it hard for unauthorized people to understand the radio communications. In the scan mode the radio scans the list of channels and seeks for communications on these channels.

The graphical user interface (GUI) of the Radio Control Application will be accommodated according to demands of the police officers after being tested in the field. The utilization of the radio control system in the real environment may also result in demands for new buttons to be added to the GUI.

The speech user interface can also be improved after utilizing the system in the field. There might be a way to change the grammar files and theirs contents in order to achieve better performance of the speech user interface.

LIST OF REFERENCES

[1]. Martin, Michael E., “The Project54 common interface for the intelligent transportation systems data bus version 4,” Technical report ECE.P54.2001.3, Electrical and Computer Engineering Department, University of New Hampshire, 2001

[2] “ASTRO Digital Spectra, Mobile radios, Detailed Service Manual,” Motorola Inc., 1995

[3] “ASTRO Portable/Mobile Radios, Radio Service Software, User’s Guide,” Motorola, 1998-1999

[4] Kun, Andrew L, Miller, W. Thomas, and Lenharth, William H. “Project54: Introducing advanced technologies in the police cruiser,” IEEE Spring VTC2002, Birmingham, AL, May 6-9, 2002

[5] Kozomora, Nevenka, “Radar Application,” Technical report ECE.P54.2002.5, Electrical and Computer Engineering Department, University of New Hampshire, 2002

[6] “SB9600 Serial Bus Protocol Manual,” version 05.13, Motorola Inc. August 28, 1995.

[7] http://alert.tamu.edu/ - Texas transportation institute - Advanced Law Enforcement & Response Technology (ALERT®) web page

[8] Ochoa, J. A., Witz J., Eshani M., and Morgan J., “Advanced law enforcement vehicle electronics and associated power,” Proceedings of the 18th Digital Avionics Systems Conference, St. Louis, MO, October 24-29, 1999

[9] Wahab A. and Chong T. E., “Intelligent dashboard with speech enhancement,” Proceedings of the International Conference on Information, Communications and Signal Processing, pp. 993-997, Singapore, September9-12, 1997

[10] Muthusamy Y., Agarwal R., Gong Y., and Viswanathan V., “Speech-enabled information retrieval in the automobile environment,” Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, Phoenix, AZ, May 15-19, 1999

[11] Rogerson, Dale, “Inside COM Microsoft’s Component Object Model,” Microsoft Press, 1997

APPENDIX A

A.1 Common IDB Interface

The schematic design of the Project54 Common IDB interface is shown in Figure A.1.

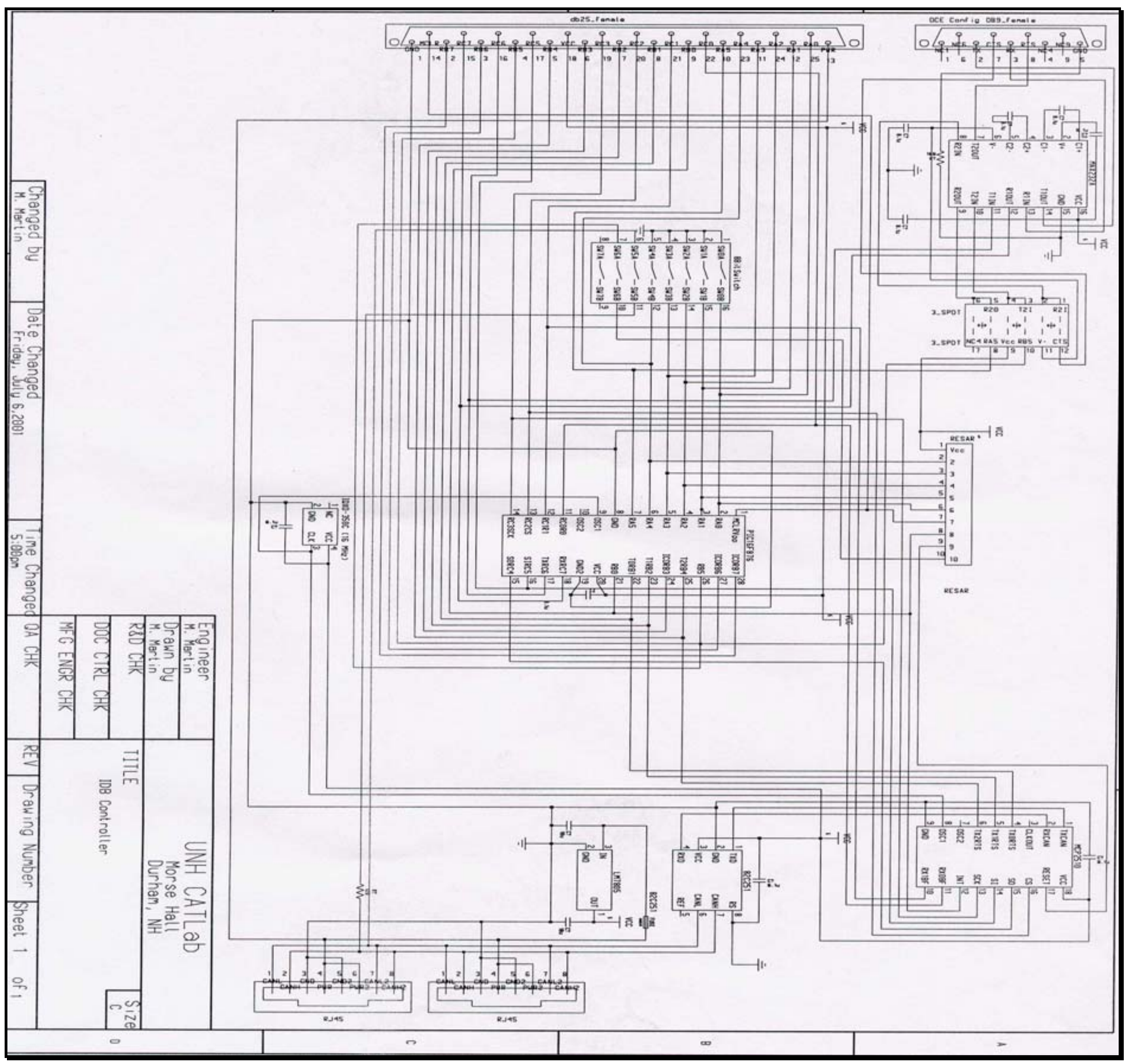


Figure A.1 Scheme of the IDB hardware interface

A.2 Bill of materials

The following two tables provide the bill of materials for the Radio Interface Hardware (Table A.1) and for the interface hardware designed for testing purposes (Table A.2).

Description	Item#	Manufacturer	Package / Case	Quantity
Half Cover Pair, DB25, Black Plastic	DHP25B	L-com		1
Solder Cup D-Sub Connector, DB25 Male	SD25P	L-com		1
Solder Cup D-Sub Connector, DB25 Male	SD25P	L-com		1
IC TXRX RS485/RS422 LOWPWR 8-DIP	MAX485CPA	Maxim	8-DIP	1
(N) TRI-STATE OCTAL BUFFER	DM74LS241N	Fairchild Semiconductor	20-DIP	1
RESISTOR	R = 1kOhm	-	-	1

Table A.1 Bill of materials – Radio Interface Hardware

Description	Part Number	Manufacturer	Package / Case	Quantity
IC TXRX RS485/RS422 LOWPWR 8-DIP	MAX485CPA	Maxim	8-DIP	1
IC 2DVR/2RCVR RS232 5V 16-DIP	MAX232ACPE	Maxim	16-DIP	1
(N) TRI-STATE OCTAL BUFFER	DM74LS240N	Fairchild Semiconductor	20-DIP	1
RESISTOR	R = 1kOhm	-	-	1
CAPACITOR	C = 0.0001mF	-	-	5

Table A.2 Bill of materials – test interface hardware

A.2 Datasheets

This section provides datasheets for the components used in the Radio Interface Hardware design and datasheets of the components used in test interface hardware design.

19-0122; Rev 6; 10/01

MAXIM

Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers

MAX481/MAX483/MAX485/MAX487–MAX491/MAX1487

General Description

The MAX481, MAX483, MAX485, MAX487–MAX491, and MAX1487 are low-power transceivers for RS-485 and RS-422 communication. Each part contains one driver and one receiver. The MAX483, MAX487, MAX488, and MAX489 feature reduced slew-rate drivers that minimize EMI and reduce reflections caused by improperly terminated cables, thus allowing error-free data transmission up to 250kbps. The driver slew rates of the MAX481, MAX485, MAX490, MAX491, and MAX1487 are not limited, allowing them to transmit up to 2.5Mbps.

These transceivers draw between 120µA and 500µA of supply current when unloaded or fully loaded with disabled drivers. Additionally, the MAX481, MAX483, and MAX487 have a low-current shutdown mode in which they consume only 0.1µA. All parts operate from a single 5V supply.

Drivers are short-circuit current limited and are protected against excessive power dissipation by thermal shutdown circuitry that places the driver outputs into a high-impedance state. The receiver input has a fail-safe feature that guarantees a logic-high output if the input is open circuit.

The MAX487 and MAX1487 feature quarter-unit-load receiver input impedance, allowing up to 128 MAX487/MAX1487 transceivers on the bus. Full-duplex communications are obtained using the MAX488–MAX491, while the MAX481, MAX483, MAX485, MAX487, and MAX1487 are designed for half-duplex applications.

Applications

Low-Power RS-485 Transceivers
Low-Power RS-422 Transceivers
Level Translators
Transceivers for EMI-Sensitive Applications
Industrial-Control Local Area Networks

Features

- In µMAX Package: Smallest 8-Pin SO
- Slew-Rate Limited for Error-Free Data Transmission (MAX483/487/488/489)
- 0.1µA Low-Current Shutdown Mode (MAX481/483/487)
- Low Quiescent Current:
  120µA (MAX483/487/488/489)
  230µA (MAX1487)
  300µA (MAX481/485/490/491)
- -7V to +12V Common-Mode Input Voltage Range
- Three-State Outputs
- 30ns Propagation Delays, 5ns Skew (MAX481/485/490/491/1487)
- Full-Duplex and Half-Duplex Versions Available
- Operate from a Single 5V Supply
- Allows up to 128 Transceivers on the Bus (MAX487/MAX1487)
- Current-Limiting and Thermal Shutdown for Driver Overload Protection

Ordering Information

PART	TEMP. RANGE	PIN-PACKAGE
MAX481CPA	0°C to +70°C	8 Plastic DIP
MAX481CSA	0°C to +70°C	8 SO
MAX481CUA	0°C to +70°C	8 µMAX
MAX481C/D	0°C to +70°C	Dice*

Ordering Information continued at end of data sheet.
**Contact factory for dice specifications.*

Selection Table

PART NUMBER	HALF/FULL DUPLEX	DATA RATE (Mbps)	SLEW-RATE LIMITED	LOW-POWER SHUTDOWN	RECEIVER/ DRIVER ENABLE	QUIESCENT CURRENT (µA)	NUMBER OF TRANSMITTERS ON BUS	PIN COUNT
MAX481	Half	2.5	No	Yes	Yes	300	32	8
MAX483	Half	0.25	Yes	Yes	Yes	120	32	8
MAX485	Half	2.5	No	No	Yes	300	32	8
MAX487	Half	0.25	Yes	Yes	Yes	120	128	8
MAX488	Full	0.25	Yes	No	No	120	32	8
MAX489	Full	0.25	Yes	No	Yes	120	32	14
MAX490	Full	2.5	No	No	No	300	32	8
MAX491	Full	2.5	No	No	Yes	300	32	14
MAX1487	Half	2.5	No	No	Yes	230	128	8

MAXIM — *Maxim Integrated Products* 1

Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers

ABSOLUTE MAXIMUM RATINGS

Supply Voltage (V_{CC}) 12V
Control Input Voltage ($\overline{RE}$, DE)..........-0.5V to (V_{CC} + 0.5V)
Driver Input Voltage (DI)..........-0.5V to (V_{CC} + 0.5V)
Driver Output Voltage (A, B)..........-8V to +12.5V
Receiver Input Voltage (A, B)..........-8V to +12.5V
Receiver Output Voltage (RO)..........-0.5V to (V_{CC} +0.5V)
Continuous Power Dissipation (T_A = +70°C)
8-Pin Plastic DIP (derate 9.09mW/°C above +70°C) 727mW
14-Pin Plastic DIP (derate 10.00mW/°C above +70°C) ..800mW
8-Pin SO (derate 5.88mW/°C above +70°C)..........471mW
14-Pin SO (derate 8.33mW/°C above +70°C)..........667mW
8-Pin μMAX (derate 4.1mW/°C above +70°C)..........830mW
8-Pin CERDIP (derate 8.00mW/°C above +70°C)..........640mW
14-Pin CERDIP (derate 9.09mW/°C above +70°C)..........727mW
Operating Temperature Ranges
MAX4_ _C_ _/MAX1487C_ A ....0°C to +70°C
MAX4_ _E_ _/MAX1487E_ A..........-40°C to +85°C
MAX4_ _MJ_/MAX1487MJA ....-55°C to +125°C
Storage Temperature Range..........-65°C to +160°C
Lead Temperature (soldering, 10sec)..........+300°C

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC ELECTRICAL CHARACTERISTICS

(V_{CC} = 5V ±5%, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
Differential Driver Output (no load)	V_{OD1}					5	V
Differential Driver Output (with load)	V_{OD2}	R = 50Ω (RS-422)		2			V
		R = 27Ω (RS-485), Figure 4		1.5		5	
Change in Magnitude of Driver Differential Output Voltage for Complementary Output States	ΔV_{OD}	R = 27Ω or 50Ω, Figure 4				0.2	V
Driver Common-Mode Output Voltage	V_{OC}	R = 27Ω or 50Ω, Figure 4				3	V
Change in Magnitude of Driver Common-Mode Output Voltage for Complementary Output States	ΔV_{OD}	R = 27Ω or 50Ω, Figure 4				0.2	V
Input High Voltage	V_{IH}	DE, DI, $\overline{RE}$		2.0			V
Input Low Voltage	V_{IL}	DE, DI, $\overline{RE}$				0.8	V
Input Current	I_{IN1}	DE, DI, $\overline{RE}$				±2	μA
Input Current (A, B)	I_{IN2}	DE = 0V; V_{CC} = 0V or 5.25V, all devices except MAX487/MAX1487	V_{IN} = 12V			1.0	mA
			V_{IN} = -7V			-0.8	
		MAX487/MAX1487, DE = 0V, V_{CC} = 0V or 5.25V	V_{IN} = 12V			0.25	mA
			V_{IN} = -7V			-0.2	
Receiver Differential Threshold Voltage	V_{TH}	-7V ≤ V_{CM} ≤ 12V		-0.2		0.2	V
Receiver Input Hysteresis	ΔV_{TH}	V_{CM} = 0V			70		mV
Receiver Output High Voltage	V_{OH}	I_O = -4mA, V_{ID} = 200mV		3.5			V
Receiver Output Low Voltage	V_{OL}	I_O = 4mA, V_{ID} = -200mV				0.4	V
Three-State (high impedance) Output Current at Receiver	I_{OZR}	0.4V ≤ V_O ≤ 2.4V				±1	μA
Receiver Input Resistance	R_{IN}	-7V ≤ V_{CM} ≤ 12V, all devices except MAX487/MAX1487		12			kΩ
		-7V ≤ V_{CM} ≤ 12V, MAX487/MAX1487		48			kΩ

SWITCHING CHARACTERISTICS—MAX481/MAX485, MAX490/MAX491, MAX1487 (continued)

(V_{CC} = 5V ±5%, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Driver Enable from Shutdown to Output High (MAX481)	$t_{ZH(SHDN)}$	Figures 7 and 9, C_L = 100pF, S2 closed		40	100	ns
Driver Enable from Shutdown to Output Low (MAX481)	$t_{ZL(SHDN)}$	Figures 7 and 9, C_L = 100pF, S1 closed		40	100	ns
Receiver Enable from Shutdown to Output High (MAX481)	$t_{ZH(SHDN)}$	Figures 5 and 11, C_L = 15pF, S2 closed, A - B = 2V		300	1000	ns
Receiver Enable from Shutdown to Output Low (MAX481)	$t_{ZL(SHDN)}$	Figures 5 and 11, C_L = 15pF, S1 closed, B - A = 2V		300	1000	ns

SWITCHING CHARACTERISTICS—MAX483, MAX487/MAX488/MAX489

(V_{CC} = 5V ±5%, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Driver Input to Output	t_{PLH}	Figures 6 and 8, R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF	250	800	2000	ns
	t_{PHL}		250	800	2000	
Driver Output Skew to Output	t_{SKEW}	Figures 6 and 8, R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF		100	800	ns
Driver Rise or Fall Time	t_R, t_F	Figures 6 and 8, R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF	250		2000	ns
Driver Enable to Output High	t_{ZH}	Figures 7 and 9, C_L = 100pF, S2 closed	250		2000	ns
Driver Enable to Output Low	t_{ZL}	Figures 7 and 9, C_L = 100pF, S1 closed	250		2000	ns
Driver Disable Time from Low	t_{LZ}	Figures 7 and 9, C_L = 15pF, S1 closed	300		3000	ns
Driver Disable Time from High	t_{HZ}	Figures 7 and 9, C_L = 15pF, S2 closed	300		3000	ns
Receiver Input to Output	t_{PLH}	Figures 6 and 10, R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF	250		2000	ns
	t_{PHL}		250		2000	
$\| t_{PLH} - t_{PHL} \|$ Differential Receiver Skew	t_{SKD}	Figures 6 and 10, R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF		100		ns
Receiver Enable to Output Low	t_{ZL}	Figures 5 and 11, C_{RL} = 15pF, S1 closed		20	50	ns
Receiver Enable to Output High	t_{ZH}	Figures 5 and 11, C_{RL} = 15pF, S2 closed		20	50	ns
Receiver Disable Time from Low	t_{LZ}	Figures 5 and 11, C_{RL} = 15pF, S1 closed		20	50	ns
Receiver Disable Time from High	t_{HZ}	Figures 5 and 11, C_{RL} = 15pF, S2 closed		20	50	ns
Maximum Data Rate	f_{MAX}	t_{PLH}, t_{PHL} < 50% of data period	250			kbps
Time to Shutdown	t_{SHDN}	MAX483/MAX487 (Note 5)	50	200	600	ns
Driver Enable from Shutdown to Output High	$t_{ZH(SHDN)}$	MAX483/MAX487, Figures 7 and 9, C_L = 100pF, S2 closed			2000	ns
Driver Enable from Shutdown to Output Low	$t_{ZL(SHDN)}$	MAX483/MAX487, Figures 7 and 9, C_L = 100pF, S1 closed			2000	ns
Receiver Enable from Shutdown to Output High	$t_{ZH(SHDN)}$	MAX483/MAX487, Figures 5 and 11, C_L = 15pF, S2 closed			2500	ns
Receiver Enable from Shutdown to Output Low	$t_{ZL(SHDN)}$	MAX483/MAX487, Figures 5 and 11, C_L = 15pF, S1 closed			2500	ns

FAIRCHILD
SEMICONDUCTOR™

August 1986
Revised March 2000

DM74LS240 • DM74LS241
Octal 3-STATE Buffer/Line Driver/Line Receiver

General Description

These buffers/line drivers are designed to improve both the performance and PC board density of 3-STATE buffers/drivers employed as memory-address drivers, clock drivers, and bus-oriented transmitters/receivers. Featuring 400 mV of hysteresis at each low current PNP data line input, they provide improved noise rejection and high fanout outputs and can be used to drive terminated lines down to 133Ω.

Features

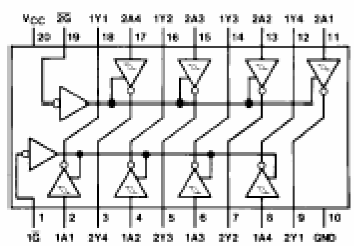
- 3-STATE outputs drive bus lines directly
- PNP inputs reduce DC loading on bus lines
- Hysteresis at data inputs improves noise margins
- Typical I_{OL} (sink current)
  24 mA
- Typical I_{OH} (source current)
  −15 mA
- Typical propagation delay times
  Inverting 10.5 ns
  Noninverting 12 ns
- Typical enable/disable time 18 ns
- Typical power dissipation (enabled)
  Inverting 130 mW
  Noninverting 135 mW

Ordering Code:

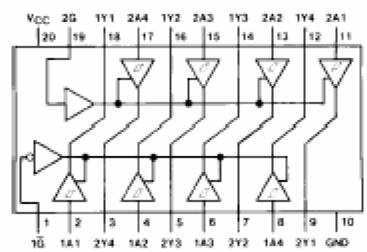
Order Number	Package Number	Package Description
DM74LS240WM	M20B	20-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-013, 0.300 Wide
DM74LS240SJ	M20D	20-Lead Small Outline Package (SOP), EIAJ TYPE II, 5.3mm Wide
DM74LS240N	N20A	20-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide
DM74LS241WM	M20B	20-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-013, 0.300 Wide
DM74LS241N	N20A	20-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide

Devices also available in Tape and Reel. Specify by appending the suffix letter "X" to the ordering code.

Connection Diagrams



DM74LS240



DM74LS241

SN5404, SN54LS04, SN54S04, SN7404, SN74LS04, SN74S04 HEX INVERTERS

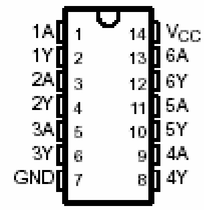
SDLS029B – DECEMBER 1983 – REVISED FEBRUARY 2002

- Dependable Texas Instruments Quality and Reliability

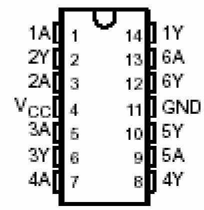
description

These devices contain six independent inverters.

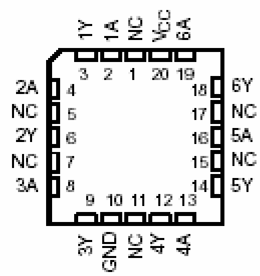
SN5404 . . . J PACKAGE
SN54LS04, SN54S04 . . . J OR W PACKAGE
SN7404 . . . D, N, OR NS PACKAGE
SN74LS04 . . . D, DB, N, OR NS PACKAGE
SN74S04 . . . D OR N PACKAGE
(TOP VIEW)



SN5404 . . . W PACKAGE
(TOP VIEW)



SN54LS04, SN54S04 . . . FK PACKAGE
(TOP VIEW)



NC – No internal connection

19-4323; Rev 10; 8/01

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220–MAX249

General Description

The MAX220–MAX249 family of line drivers/receivers is intended for all EIA/TIA-232E and V.28/V.24 communications interfaces, particularly applications where ±12V is not available.

These parts are especially useful in battery-powered systems, since their low-power shutdown mode reduces power dissipation to less than 5µW. The MAX225, MAX233, MAX235, and MAX245/MAX246/MAX247 use no external components and are recommended for applications where printed circuit board space is critical.

Applications

Portable Computers

Low-Power Modems

Interface Translation

Battery-Powered RS-232 Systems

Multidrop RS-232 Networks

Features

Superior to Bipolar

- Operate from Single +5V Power Supply (+5V and +12V—MAX231/MAX239)
- Low-Power Receive Mode in Shutdown (MAX223/MAX242)
- Meet All EIA/TIA-232E and V.28 Specifications
- Multiple Drivers and Receivers
- 3-State Driver and Receiver Outputs
- Open-Line Detection (MAX243)

Ordering Information

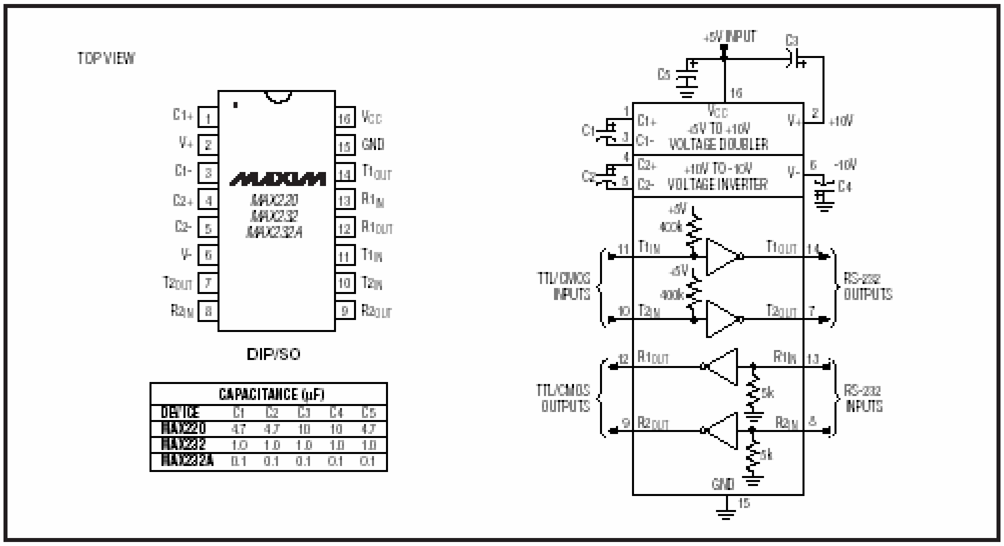
PART	TEMP. RANGE	PIN-PACKAGE
MAX220CPE	0°C to +70°C	16 Plastic DIP
MAX220CSE	0°C to +70°C	16 Narrow SO
MAX220CWE	0°C to +70°C	16 Wide SO
MAX220C/D	0°C to +70°C	Dice*
MAX220EPE	-40°C to +85°C	16 Plastic DIP
MAX220ESE	-40°C to +85°C	16 Narrow SO
MAX220EWE	-40°C to +85°C	16 Wide SO
MAX220EJE	-40°C to +85°C	16 CERDIP
MAX220MJE	-55°C to +125°C	16 CERDIP

Ordering Information continued at end of data sheet.

*Contact factory for dice specifications.

Selection Table

Part Number	Power Supply (V)	No. of RS-232 Drivers/Rx	No. of Ext. Caps	Nominal Cap. Value (µF)	SHDN & Three-State	Rx Active in SHDN	Data Rate (kbps)	Features
MAX220	+5	2/2	4	0.1	No	—	120	Ultra-low-power, industry-standard pinout
MAX222	+5	2/2	4	0.1	Yes	—	200	Low-power shutdown
MAX223 (MAX213)	+5	4/5	4	1.0 (0.1)	Yes	✔	120	MAX241 and receivers active in shutdown
MAX225	+5	5/5	0	—	Yes	✔	120	Available in SO
MAX230 (MAX200)	+5	5/0	4	1.0 (0.1)	Yes	—	120	5 drivers with shutdown
MAX231 (MAX201)	+5 and +7.5 to +13.2	2/2	2	1.0 (0.1)	No	—	120	Standard +5/+12V or battery supplies; same functions as MAX232
MAX232 (MAX202)	+5	2/2	4	1.0 (0.1)	No	—	120 (64)	Industry standard
MAX232A	+5	2/2	4	0.1	No	—	200	Higher slew rate, small caps
MAX233 (MAX203)	+5	2/2	0	—	No	—	120	No external caps
MAX233A	+5	2/2	0	—	No	—	200	No external caps, high slew rate
MAX234 (MAX204)	+5	4/0	4	1.0 (0.1)	No	—	120	Replaces 1488
MAX235 (MAX205)	+5	5/5	0	—	Yes	—	120	No external caps
MAX236 (MAX206)	+5	4/3	4	1.0 (0.1)	Yes	—	120	Shutdown, three state
MAX237 (MAX207)	+5	5/3	4	1.0 (0.1)	No	—	120	Complements IBM PC serial port
MAX238 (MAX208)	+5	4/4	4	1.0 (0.1)	No	—	120	Replaces 1488 and 1489
MAX239 (MAX209)	+5 and +7.5 to +13.2	3/5	2	1.0 (0.1)	No	—	120	Standard +5/+12V or battery supplies; single-package solution for IBM PC serial port
MAX240	+5	5/5	4	1.0	Yes	—	120	DIP or flatpack package
MAX241 (MAX211)	+5	4/5	4	1.0 (0.1)	Yes	—	120	Complete IBM PC serial port
MAX242	+5	2/2	4	0.1	Yes	✔	200	Separate shutdown and enable
MAX243	+5	2/2	4	0.1	No	—	200	Open-line detection simplifies cabling
MAX244	+5	8/10	4	1.0	No	—	120	High slew rate
MAX245	+5	8/10	0	—	Yes	✔	120	High slew rate, int. caps, two shutdown modes
MAX246	+5	8/10	0	—	Yes	✔	120	High slew rate, int. caps, three shutdown modes
MAX247	+5	8/9	0	—	Yes	✔	120	High slew rate, int. caps, nine operating modes
MAX248	+5	8/8	4	1.0	Yes	✔	120	High slew rate, selective half-chip enables
MAX249	+5	6/10	4	1.0	Yes	✔	120	Available in quad flatpack package



CAPACITANCE (μF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	4.7	4.7	10	10	4.7
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

Figure 5. MAX220/MAX232/MAX232A Pin Configuration and Typical Operating Circuit

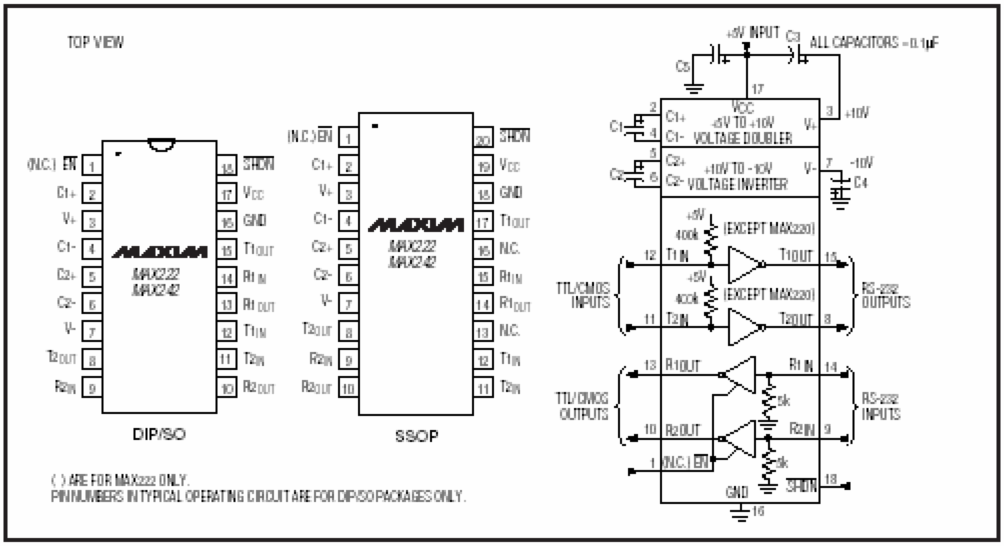


Figure 6. MAX222/MAX242 Pin Configurations and Typical Operating Circuit

A.3 Board layout and picture of the Radio Interface Hardware

Figure A.2 shows the board layout of the Radio Interface Hardware. This is a two-layer board and this figure shows both layers. The wires on the same side are marked with the same color.

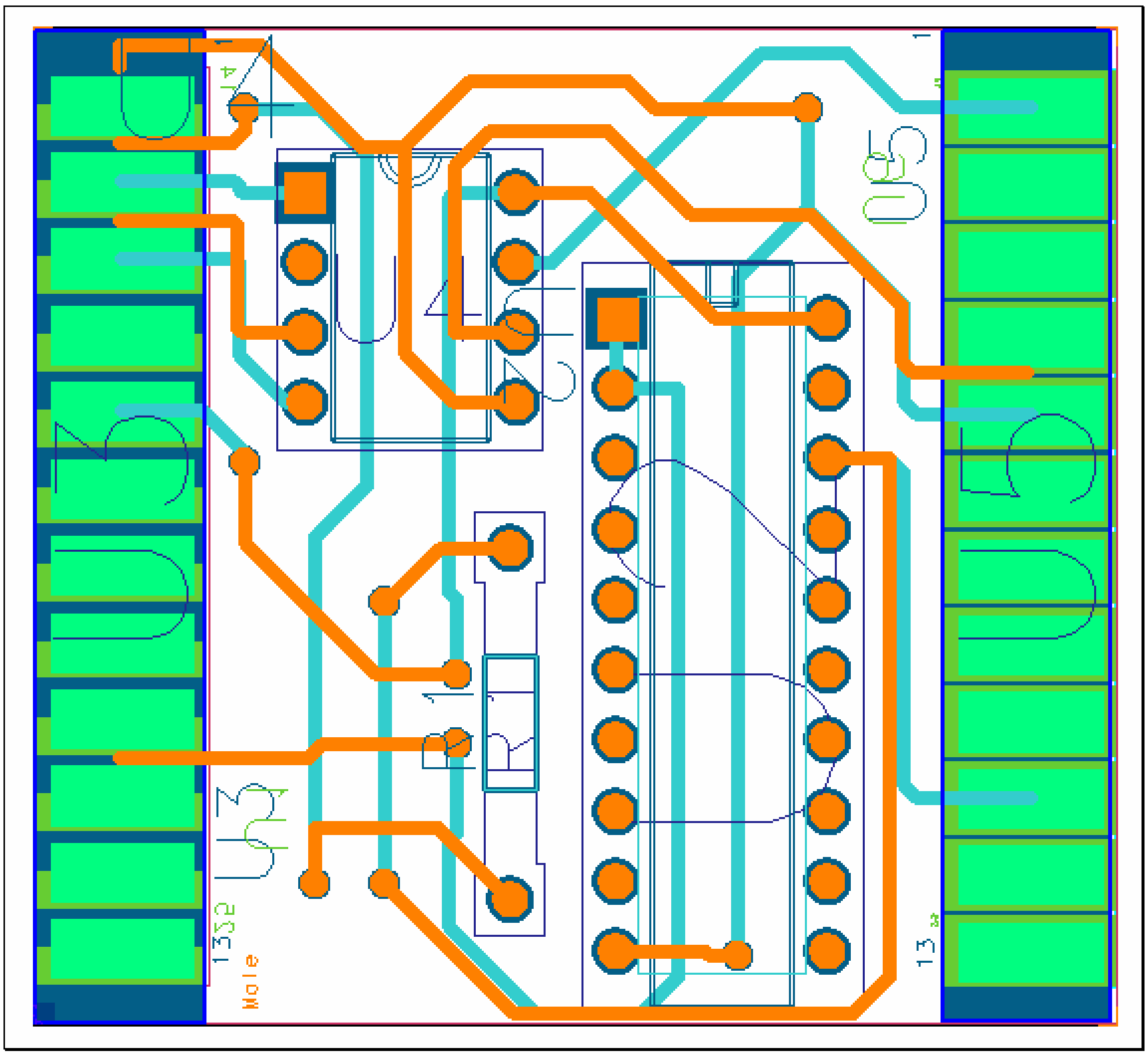


Figure A.2 Board layout of the Radio Interface Hardware

Figure A.3 shows the Radio Interface Hardware board and cover.

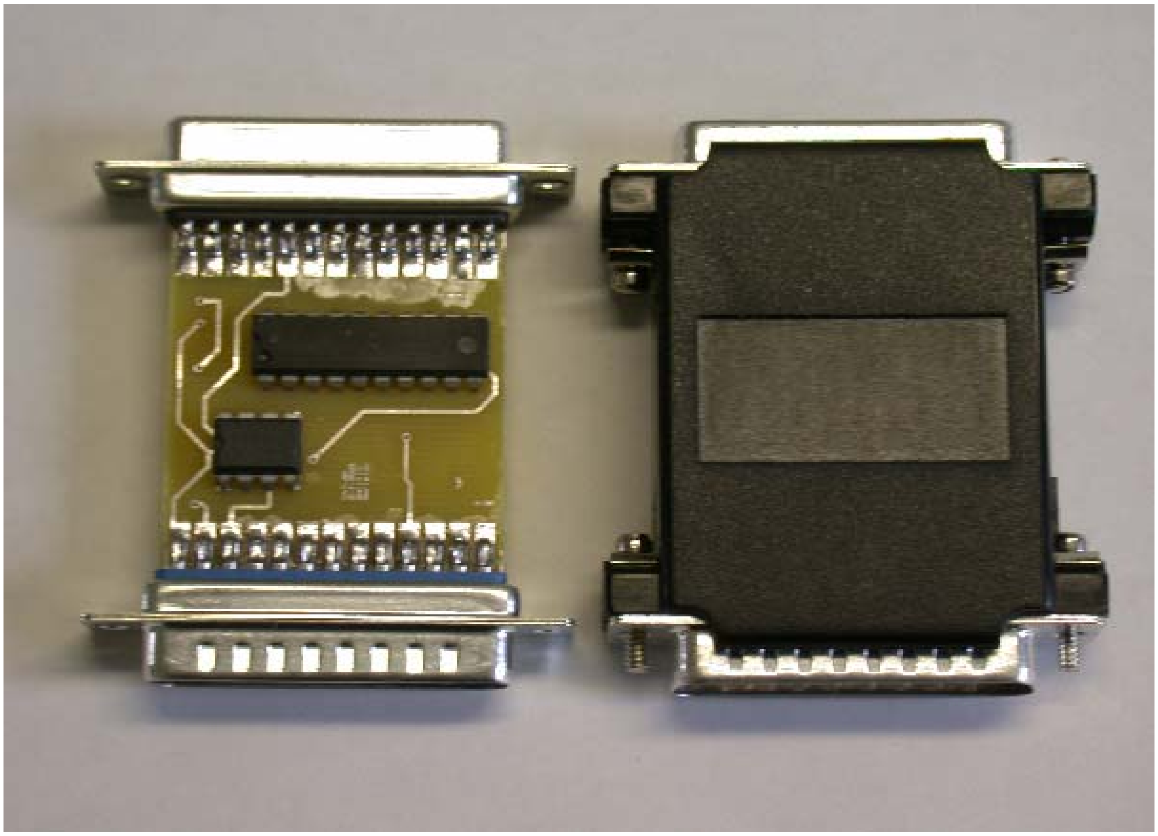


Figure A.3 The Radio Interface Hardware

APPENDIX B

SB9600 SERIAL BUS PROTOCOL

This appendix intends to describe the operation of the serial bus between the control head and the main radio housing. Since the referenced document is Motorola confidential property [6], this part of the thesis is available only with the permission of authorized Catlab personnel.