Technical report ECE.P54.2005.4
June 2, 2005

# Measuring New Hampshire State Police radio usage

Nathan Purmort

**Abstract:**

One of the many benefits of the Project54 system is the reduction in the amount of radio bandwidth a police officer uses, the magnitude of which has never been monitored. By writing a small module to snap into the Project54 software's framework, the amount of radio usage can be recorded and then analyzed to give an indication of how much less the officers use their radios. The data obtained from this research will be valuable in future endeavors of the project including sending Automatic Vehicle Location (AVL) data and remote-diagnostics data.

**Introduction:**

The primary objective of the Project54 system, developed in collaboration between the university and the State Police agency, is to make the job of a police officer both safer and easier through the integration of in-car electronics [1]. It allows all of the devices in a police cruiser to be controlled by an on-board computer running Windows XP, providing a single, speech-based user interface. One of the most frequently-used of these devices is the mobile radio; it is used to communicate with other patrol units, with the dispatcher, and to query remote vehicle and driver record databases.

Without the Project54 system, the latter is rather cumbersome as the officer must perform the records checks by radioing the dispatcher back at headquarters. The officer then relays the vehicle registration or driver's license information to the dispatcher who queries the database at the Department of Motor Vehicles (DMV) using a computer and keyboard. The record information is then read back to the officer over the radio. This consumes a large amount of radio talk time (tens of seconds) and makes the officer responsible for either remembering the information or finding a pencil and paper to write the record data down.

The Project54 system allows the officer to query the DMV server directly using the wireless data capabilities of his/her mobile radio. This should reduce the amount of radio traffic considerably as the data packets are much smaller and transmit much more quickly than the equivalent information being transmitted as voice traffic. This reduction seems logical, but we have no quantitative data as to the actual impact this has on the overall amount of radio traffic. Also, with the pending implementation of transmitting Automatic Vehicle Location (AVL) data over the radio [2], it has become necessary to be able to accurately measure the effects of the Project54 system on the amount of radio bandwidth used. AVL data will be sent from cruisers to headquarters at regular intervals. It is important to determine what these intervals should be so that AVL data would not overwhelm the radio network.

In order to provide quantitative data on the impact of the Project54 system on the State Police radio network usage, a piece of software was developed, called RadioTrafficAnalyzer, to analyze the usage of a single radio channel as a function of time. In order to understand how this software can be used to monitor radio traffic accurately, it is important to understand how the radio itself works.

**Basic Radio Functionality:**

The Motorola radios used by the State Police are Astro Spectra digital models with a variety of control heads. The radios transmit voice or data as Internet Protocol packets. They divide all of the programmed channels up into zones, which are really groups of channels. Each channel is assigned a common name (or personality) which is mapped to the proper communication frequency. Each zone is assigned a common name as well. In the State Police system, the zone is allowed 2 characters and the channel is allowed 6. When the term “base channel” is used, it refers to the channel currently being monitored – this name is composed of both the channel name and its zone assignment. For example, the base channel “A1TRP_A_” is in zone “A1” and is channel “TRP_A_”. Because a channel can be assigned to multiple zones, the channels “B1TRP_D_” and “1_TRP_D_” are the same as far as communication is concerned, but because of the different zone, are considered to be entirely different from the Project54 system’s perspective.

The display will always show the base channel, except when an officer is speaking. This feature is called Push to Talk ID (PTT-ID) and allows a unique identifier for each officer to display on the radio whenever he/she is speaking [3]. The State Police uses each car’s license plate number as the identifier and 1000 for the dispatchers. Figure 1 depicts the radio with no activity: the base channel “1_TRP_A_” is displayed; Figure 2 shows the radio when the dispatcher is talking.


**Figure 1- Motorola W7 Control Head**


**Figure 2 – Motorola W7 Control Head - Dispatcher Transmitting**

As shown in Figure 1, there are two lights on the front of the control head which are used to signify if the base channel is busy (another officer is talking or sending data) or if the radio is busy transmitting (you are speaking over the radio or sending data). For example, Figure 2 shows the busy light on when the dispatcher is speaking. The busy and transmit lights, in combination with the state of the display, can be used to determine whether a transmission is voice or data traffic, the start and end of a transmission (and thus its duration), as well as the source (officer who initiated the transmission) of a voice-type broadcast. These items are all calculated using the feedback messaging structure built into the Project54 system.

**Feedback Messaging Structure:**

Part of the Project54 radio interface is the software used to read and control the features of the device. This software is capable of notifying other Project54 applications of the radio's status through the use of a feedback structure [4]. Thus, the RadioTrafficAnalyzer application was created to receive and interpret these feedback messages and create a log of the results.

The first and most important part of a logging application is to identify what data needs to be recorded in order to create an informative log file. Since this program will determine the amount of radio traffic over time, it is important to record the duration of the broadcast as well as its type (voice or data). The broadcast channel, start time, and start date of the transmission are also logged so that additional data processing can be performed later.

The Radio Application is capable of sending messages to any other application signed up for feedback [5]. It can forward the status of every radio function, but only three of these are needed to log radio traffic: "STATUS CHANNEL" (signifying a channel change), "STATUS BUSY" (signifying the radio activity), and "STATUS BUSY OFF" (signifying the end of radio activity). The busy status and channel name need to be detected in order to differentiate between the two types of radio traffic. Additionally, a channel reset from the officer's PTT-ID to the base channel name does, in certain situations, signify the end of a broadcast.

After monitoring the feedback messages sent and received during radio broadcasts, it was determined that there are five possible orders in which they can be received. Only two arrangements should be expected (one for data, one for voice), but in certain cases, some of the messages are sent out of the anticipated order, creating the need for an intricate message processing routine. Each different order signifies a special type of radio broadcast and must be interpreted correctly for the software to be accurate. These are outlined in Table 1 and Appendix B shows the radio feedback messages with their respective P54RadioLog file entries indented underneath – this is to give a better idea of how these feedback messages are interpreted.

| | |
|---|---|
| Type 1:<br>STATUS BUSY<br>STATUS BUSY OFF | This is the order for a data broadcast. No channel change occurs as no officer is speaking. For this reason, it is not possible to log which officer is sending the data packets. |
| Type 2:<br>STATUS BUSY<br>STATUS CHANNEL XXX<br>STATUS BUSY OFF<br>STATUS CHANNEL A1 TRP A | This is the most widely seen message order and is a standard voice broadcast by officer XXX. The busy light is turned off and then the channel is reset almost immediately. |
| Type 3:<br>STATUS BUSY<br>STATUS CHANNEL XXX<br>STATUS CHANNEL A1 TRP A<br>STATUS BUSY OFF | This order is similar to type 2 except the channel is reset before the busy light is turned off. A channel reset to A1TRP A must mean the radio knows the broadcast has ended, so a new line is written to the log file on this message instead of on BUSY OFF. |
| Type 4:<br>STATUS BUSY<br>STATUS CHANNEL XXX<br>STATUS CHANNEL A1 TRP A<br>STATUS CHANNEL YYY<br>STATUS CHANNEL A1 TRP A<br>STATUS BUSY OFF | Here are two voice transmissions, one from officer XXX and the other from officer YYY. The end of the first transmission is assumed to be the channel reset to A1TRP A, and the beginning of the second starts on CHANNEL YYY and ends on the channel reset. The BUSY OFF is ignored as in type 3. This format is usually only seen during peak radio usage periods. |
| Type 5:<br>STATUS BUSY<br>STATUS CHANNEL XXX<br>STATUS BUSY OFF<br>STATUS BUSY<br>STATUS CHANNEL A1 TRP A<br>STATUS CHANNEL YYY<br>STATUS CHANNEL A1 TRP A<br>STATUS BUSY OFF | This is two overlapping broadcasts; the channel reset occurs after a new broadcast has started. If this message order were not handled, the second transmission would be recorded as voice activity on A1TRP A, which is not possible. For this reason, if a channel reset directly follows a STATUS BUSY, the channel reset is ignored. |

**Table 1 - Radio Feedback Messaging Structures**

**Software Design:**

The source code for the message-handling routine is included in Appendix A. In it, there are several key function calls, but those called at the start and end of a broadcast are the most important. The first is “`void SetTransStart(void)`”, a function which creates a timestamp at the detection of radio activity. It stores the current value of the system timer to be used later in calculating the elapsed broadcast time. The function then calls the “GetDateFormat” and “GetTimeFormat” Win32 functions to store the current date and time. The code for this function is shown in Figure 3:

```
void SetTransStart(void)
{
   // start radio activity timer
   startcount = GetTickCount();

   // record timestamp
   getDateAndTime(date, time);
}
```

**Figure 3 - SetTransStart() function**

The second important function call is to “void SetTransEnd(bool isdata)” (see Figure 4) when the end of a radio broadcast is detected. This function records the current value of the system timer and calculates the elapsed time between the two timer counts.

Next, based on the status of “isdata”, the transmission is either considered data or voice traffic, and this determination is done throughout the message routine, changing the state of this variable as necessary. The decision on traffic type is made based on the order in which messages are received – for example, if a “STATUS BUSY OFF” message directly follows a “STATUS BUSY” message, then, as seen in type 1 in Table , the transmission must be data (isdata = true).

```
// stop the timer and create the log file
void SetTransEnd(bool isdata)
{
   // stop radio activity timer
   stopcount = GetTickCount();

   // calculate new duration
   duration = (long)stopcount - (long)startcount;

   // determine new type of radio activity
   TrafficType(isdata, type);

   // Log the radio event
   wcitoa(duration, durationwc);
   LogString(wszRadioLogFile, type, channelname, date, time, durationwc);
}
```

**Figure 4 - SetTransEnd() function**

At this point, the logger has collected all of the necessary information to describe a radio broadcast, so the last step is to log the transmission to a file. To do this, all of the transmission-describing variables are sent to "`LogString()`" which concatenates them into one long string which is then written into a new line in the log file. The name and location of this file is specified in the Windows registry (\Paths\Logs\P54RadioLog) and is set to "c:\\Project54\\Logs\\P54RadioLog.txt" by default. The data in this file is in tab delimited format so that it can be imported into a database utility for later analysis. Appendix B shows several sample radio feedback messages with their respective P54RadioLog file entries indented underneath – this is to give a better idea of how these feedback messages are interpreted.

**Program Implementation:**

In order to provide reliable test data on which to build a database, the software has been installed on Motorola Astro Spectra digital radios using both W7 and W9 control heads in the Project54 offices. It is running on a dedicated embedded PC with the radio connected to an antenna on the roof of the building. The program monitors radio traffic on one fixed channel. As mentioned earlier, this fixed channel is referred to as the "base channel" and is configurable through the "RadioTA\Parameters\BaseChannel" registry setting.

**Data Analysis Tool:**

The generated log files, which are in tab delimited format, can simply be imported into a database structure created with Microsoft Access. A user interface within the database has been created for displaying reports and statistics about radio usage, so once the data is imported to a table, pre-made data analysis plots can be viewed easily. This interface can be seen in Figure 5.

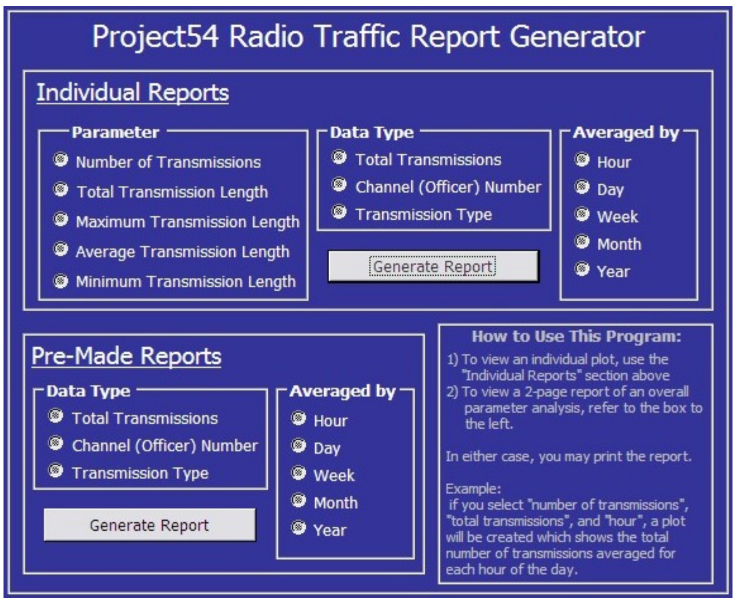


**Figure 5 - Data Processing Interface**

By selecting a parameter, data type, and time period for averaging, the user can display a graph of that particular data. For instance, if the user was interested in finding the average daily number of transmissions for data and voice traffic, he/she would select "Number of Transmissions" for the Parameter, "Transmission Type" for the Data Type and "Day" for the Averaged by box.

While this interface seems simple, there are more than 30 queries and almost 100 charts which drive it. As long as the data is imported into a table named "Sample radio data", the queries will update themselves and the charts they support automatically, so no additional database work is necessary. All of the data is currently displayed in bar graphs based on a user-defined time interval, but as the software is implemented, charts such as the number of transmissions from each car graphed over time can be included (depending on their relevance to the radio usage studies).

The data used with the sample database was from 9/26/2004 to 10/02/2004 (Week #40 of 2004) and was recorded on the test logger setup in the Project54 offices. When making the queries, it was determined that the best way to make the charts display valuable data was to average the data over a certain time period. For instance, if the user selects the chart for "Average Number of Transmissions Per Hour", the total number of transmissions per hour for every hour are averaged together. This means that the number of transmissions for every 0900-1000 hour of every day are being averaged together and displayed on the graph. This yields a result which is a better indicator of radio traffic than if the total hourly traffic were displayed. That is, if every 0900-1000 hour of every day was added together and displayed on the graph – this would create runaway totals which would solely depend on the number of hours of data being used and would not provide any useful information.

The type of transmission graphs only display two bars, one for data and the other for voice. The charts based on the channel/officer/license plate number usually can display dozens of bars, so, within the queries, the number of bars has been limited to a more reasonable 20-25.

**Preliminary Results:**

From the week of sample data that was collected, some preliminary data analysis has been done with the database structure created using Microsoft Access. Three sample plots are given to illustrate the range of information which can be garnered from the log files of radio data.

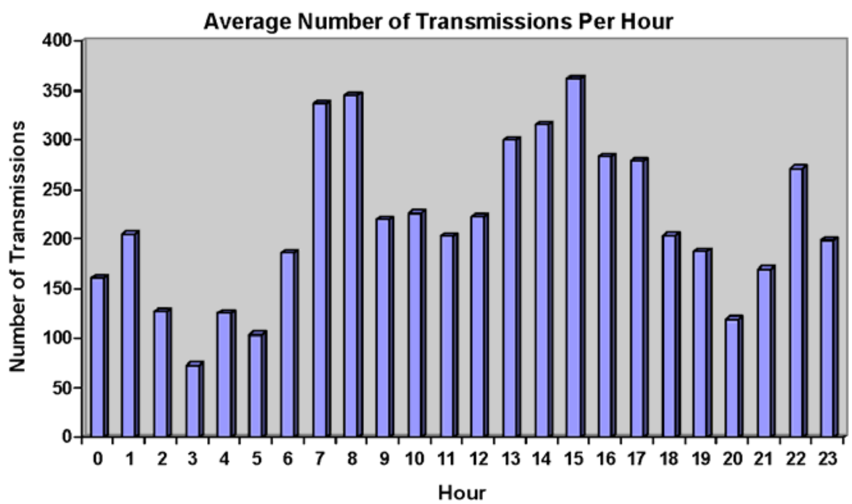


**Figure 6 - Average Number of Transmissions per Hour**

Figure 6 illustrates the average number of radio transmissions made per hour on the "A1TRP_A_" channel. For example, there was an average of just over 200 transmissions on this channel during the 0100 hour. It is apparent that the peak amounts of radio traffic occur between 0700 and 0800 hours and again between 1300 and 1700 hours. These times roughly correspond to the morning and afternoon commutes to and from work when traffic is heaviest.

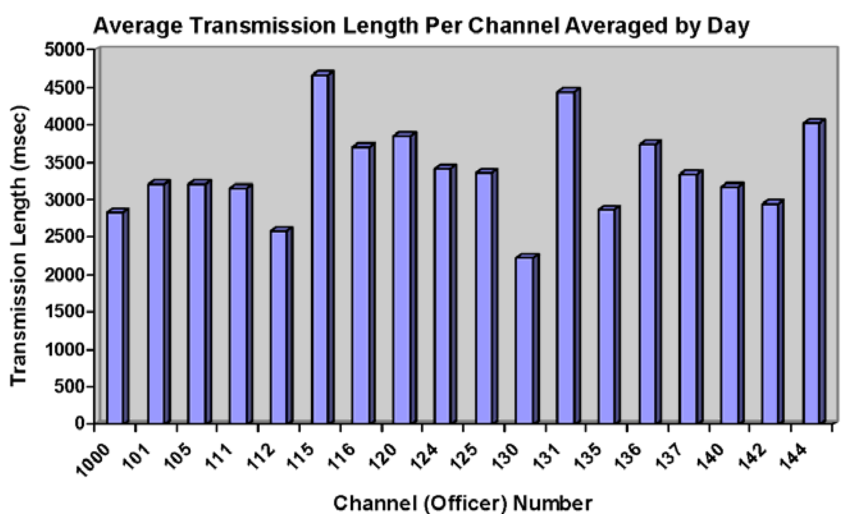


**Figure 7 - Average Transmission Length per Channel Averaged by Day**

The next graph, Figure 7, shows the average length of several officers' (as identified by their car's plate number) transmissions as a daily average on the "A1TRP_A_" channel. This was calculated by first finding the average daily transmission length for a given officer and then averaging all of the daily averages together to get the magnitude shown in Figure 7. For example, the officer in state police car #120 talked on the radio for an average duration of 4 seconds per transmission.

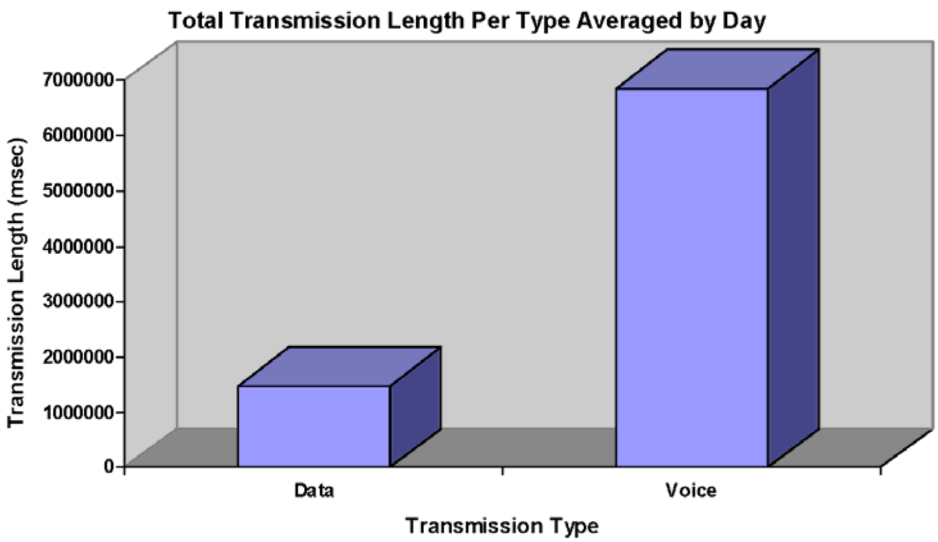


**Figure 8 - Total Transmission Length per Type Averaged by Day**

Data analysis can even be done according to the type of transmission: data or voice. Figure 8 illustrates the average daily transmission length for each type. It shows that voice traffic makes up for a considerably higher portion (almost 2 hours/day on average) of the overall radio bandwidth than data traffic (around 0.5 hours/day on average).

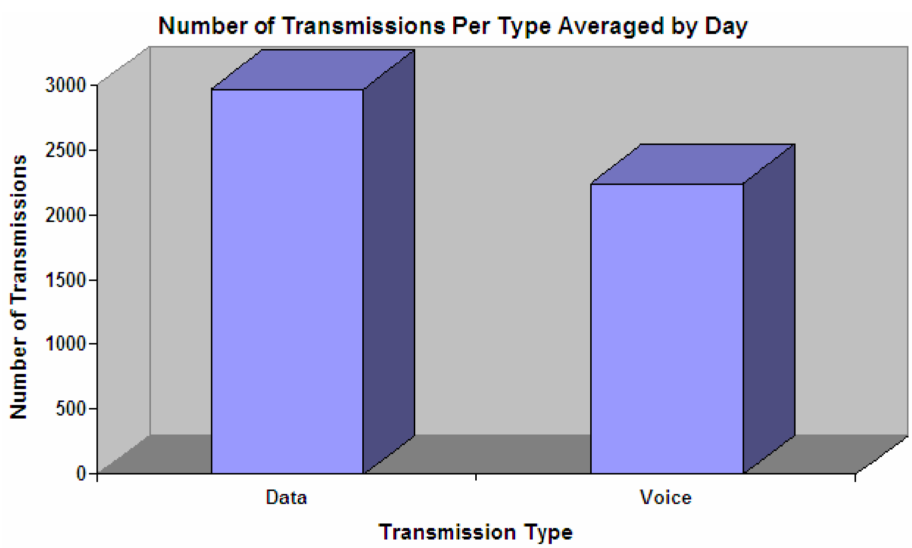


**Figure 9 - Number of Transmissions per Type Averaged by Day**

Figure 9 shows the average daily number of transmissions for both data and voice traffic and spans the same time period as Figure 8. Although there are over 500 more data transmissions than voice transmissions, the total amount of bandwidth consumed by data traffic is still an hour and a half less than that consumed by the voice broadcasts. It is clear from these two plots that using data packets rather than voice packets to transmit information results in a considerable reduction in radio bandwidth usage. This frees up the radio network to other uses such as AVL data.

**Future Research & Applications:**

This piece of software can be used in several situations in the future – at its most basic level, the RadioTrafficAnalyzer application can monitor the bandwidth usage of any digital radio communication. More specifically however, the program can be applied to analyze the impact of sending AVL data [2] or, as outlined earlier, to help determine the relative reduction in radio bandwidth as a result of the Project54 system being implemented.

Part of determining the success of the Project54 system at reducing radio bandwidth is maintaining an accurate record of data packets being transmitted. If this program proves to be insufficient for doing so, more work will need to be done to find out if there are any data packets not being logged. For example, in type 3 traffic the “STATUS BUSY OFF” message is not an accurate indicator of the transmission end. The problem with this type of traffic is that the channel is reset to the base channel before the “STATUS BUSY OFF” message is received (the opposite of type 2 traffic). In fact, “STATUS BUSY OFF” is received anywhere from 100 milliseconds to 4.5 seconds after the channel is reset (most often the delay is discernable just by looking at the control head display). There is more than enough time for several data packets to be transmitted while the light remains on without any way of detecting them. This means that the radio may, in fact, still be busy, just not with voice traffic but with data traffic. Because there is currently no way of sensing if data is being transmitted or not within that 100 millisecond to 4.5 second gap, it is just assumed that data is not being sent and the transmission is ended on the first sign of no radio activity, as described earlier.

Another shortcoming of this method is that the origins of data transmissions cannot be determined. This information may be obtainable from state police headquarters but if it is not and is deemed necessary to traffic monitoring, a method of analyzing the packet headers will need to be devised.

This software is an analysis tool to help determine the feasibility of other radio-related projects within Project54. The data it is capable of recording and the post-monitoring analysis will be valuable information to any radio-related (AVL, License queries, etc…) application and to determine the overall effectiveness of the Project54 system at reducing radio bandwidth.

**References:**


[1] A.L. Kun, W.T. Miller, III and W.H. Lenharth, "Computers in police cruisers," IEEE Pervasive Computing, Volume 3, Issue 4, Pages: 34 - 41, October-December 2004.

[2] S. Y. Kim, K. Wilson-Remmer, A. L. Kun, W. T. Miller, III, “Remote Fleet Management for Police Cruisers,” submitted to IEEE Intelligent Vehicles, Las Vegas, 2005.

[3] Motorola. Digital Spectra Models W4, W5, W7 and W9. User's Guide. User's Guide 68P81074C80-D. 1996. Motorola.

[4] W.T. Miller, III, "Project54 Client/Server Application Messaging," ECE.P54.2003.3, 1-3. 3-24-2003, University of New Hampshire, Consolidated Advanced Technologies for Law Enforcement Laboratories (CATLAB) - Project54.

[5] B. Batinic, "The Integration of a Police Mobile Radio into the Project54 System," Master of Science in Electrical Engineering University of New Hampshire, 2002.

# Appendix A:
## Message-Handling Routine

```
// handle alert message from app manager (& process feedback)
void AppHandleMessage(LPWSTR source, LPWSTR messageID, LPWSTR message_text)
{
   // process channel change messages from radio app.
   if(wcsncmp(message_text, L"STATUS CHANNEL", 14) == 0)
   {
      // if a channel change happens while the transmission timer is running,
      // the transmission has to be voice traffic
      isdata = false;

      // channel is being changed to A1TRPA
      if(wcsncmp(&message_text[15], L"A1   TRP A ", 11) == 0)
      {
         // check status of ignoreA1TRPA
         if(ignoreA1TRPA)
         {/* A1TRPA is being shunned and ignored */}

         else if (logstarted)
         {
            // stop timer & log the transmission to the file
            SetTransEnd(isdata);

            // show that the computer has received a command indicating the
            // end of a radio broadcast (channel change or BUSY OFF)
            logstarted = false;

            // if next message is STATUS CHANNEL A1 TRPA, ignore it (it will
            // be leftover from a prior radio broadcast)
            ignoreA1TRPA = true;
         }
      }

      else
      {
         // extract channel name from radio message
         wcscpy(channelname, &message_text[15]);

         if(logstarted)
         {
            // if next message is STATUS CHANNEL A1 TRPA, do not ignore it as
            // it indicates the end of another broadcast
            ignoreA1TRPA = false;
         }

         else
         {
            SetTransStart();  // set start of transmission

            // indicate that the computer is waiting for a command indicating
            // the end of a radio broadcast (channel change or BUSY OFF)
            logstarted = true;

            // if next message is STATUS CHANNEL A1 TRPA, do not ignore it as
            // it indicates the end of another broadcast
            ignoreA1TRPA = false;
         }
```

```
      }
   }

   // process start of radio transmission only if one is not already started
   else if((!logstarted) && (wcslen(message_text) == 11) &&
           (wcsncmp(message_text, L"STATUS BUSY", 11) == 0))
   {
      SetTransStart();  // set start of transmission

      // indicate that the computer is waiting for a command indicating
      // the end of a radio broadcast (channel change or BUSY OFF)
      logstarted = true;

      // if next message is STATUS CHANNEL A1 TRPA, ignore it (it will be
      // leftover from a prior radio broadcast)
      ignoreA1TRPA = true;

      // if true when a BUSY OFF is received, transmission must be data
      isdata = true;
   }

   // process end of radio transmission only if it is already started –
   // eliminates the possibility of BUSY OFF messages being processed if
   // they're sent before a BUSY message is received (as when system starts).
   else if((logstarted) &&
           (wcsncmp(message_text, L"STATUS BUSY OFF", 15) == 0))
   {
      if(isdata)
      {
         // set the channel name for a data transmission
         wcscpy(channelname, L"A1TRPA");
      }

      SetTransEnd(isdata); // stop timer & log the transmission to the file

      // indicate that the computer has received a command indicating
      // the end of a radio broadcast (channel change or BUSY OFF)
      logstarted = false;

      // if next message is STATUS CHANNEL A1 TRPA, ignore it (it will be
      // leftover from a prior radio broadcast)
      ignoreA1TRPA = true;
   }

   // turn feedback on when system is fully started
   else if(wcsncmp(message_text, L"BROADCAST STARTUP", 17) == 0)
   {
      Message(self, radio, self, L"FEEDBACK ON");     // turn feedback ON
   }

   // shutdown when system is shutting down & turn feedback off
   else if(wcscmp(message_text, L"BROADCAST SHUTDOWN") == 0)
   {
      Message(self, radio, self, L"FEEDBACK OFF");    // turn feedback OFF
      Message(self, L"", self, L"SHUTDOWN");
   }
}
```

# Appendix B
## Sample Log File matched up with actual messages

```
09/24/04 11:01:45 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY
09/24/04 11:01:46 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY OFF

      Data  A1TRPA      09/24/04    11:01:45.123      891

09/24/04 11:02:01 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY
09/24/04 11:02:01 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY OFF

      Data  A1TRPA      09/24/04    11:02:01.65 591

09/24/04 11:03:41 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY
09/24/04 11:03:42 APPMAN_MSG:radio;radiota;radiota;STATUS CHANNEL 218
09/24/04 11:03:43 APPMAN_MSG:radio;radiota;radiota;STATUS CHANNEL A1   TRP A
09/24/04 11:03:48 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY OFF

      Voice 218         09/24/04    11:03:41.720      1802

09/24/04 11:03:48 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY
09/24/04 11:03:49 APPMAN_MSG:radio;radiota;radiota;STATUS CHANNEL 1000
09/24/04 11:03:50 APPMAN_MSG:radio;radiota;radiota;STATUS CHANNEL A1   TRP A
09/24/04 11:03:50 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY OFF

      Voice 1000        09/24/04    11:03:48.830      1682

09/24/04 11:03:51 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY
09/24/04 11:03:52 APPMAN_MSG:radio;radiota;radiota;STATUS CHANNEL 218
09/24/04 11:03:56 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY OFF
09/24/04 11:03:56 APPMAN_MSG:radio;radiota;radiota;STATUS CHANNEL A1   TRP A

      Voice 218         09/24/04    11:03:51.835      4457

09/24/04 11:03:56 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY
09/24/04 11:03:56 APPMAN_MSG:radio;radiota;radiota;STATUS CHANNEL 1000
09/24/04 11:03:57 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY OFF
09/24/04 11:03:57 APPMAN_MSG:radio;radiota;radiota;STATUS CHANNEL A1   TRP A

      Voice 1000        09/24/04    11:03:56.572      751

09/24/04 11:04:16 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY
09/24/04 11:04:16 APPMAN_MSG:radio;radiota;radiota;STATUS CHANNEL 115
09/24/04 11:04:18 APPMAN_MSG:radio;radiota;radiota;STATUS CHANNEL A1   TRP A
09/24/04 11:04:20 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY OFF

      Voice 115         09/24/04    11:04:16.650      1873

09/24/04 11:04:20 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY
09/24/04 11:04:21 APPMAN_MSG:radio;radiota;radiota;STATUS CHANNEL 1000
09/24/04 11:04:21 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY OFF
09/24/04 11:04:22 APPMAN_MSG:radio;radiota;radiota;STATUS CHANNEL A1   TRP A

      Voice 1000        09/24/04    11:04:20.957      841

09/24/04 11:04:23 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY
09/24/04 11:04:23 APPMAN_MSG:radio;radiota;radiota;STATUS BUSY OFF

      Data  A1TRPA      09/24/04    11:04:23.210      301
```