WinTone v2.0

Multi-Tone Decoder for Windows 95/NT

USERS GUIDE

Table of Contents

5. Tone Squelch

6. Minimizing CPU Usage

7. Troubleshooting

8. Technical Support

1 Introduction

System Requirements

Wintone is a very processor intensive application by it's very Nature. It uses a process called Fast Fourier transform to achieve tone Detection. This process must take place several times per second (In some cases even hundreds of times per second) in order to detect all of the tones that are required.
WinTone has been tested on machines as slow as 486/66Mhz with 16MB of RAM, using the optimized libraries (See Chapter 7).

The minimum hardware and software are required for WinTone to operate are as follows :

Machine	**Operating System**	**Peripherals**
486/66Mhz	Windows 95 or Windows NT	Windows compatible Sound Card

These specifications indicate the slowest machine that we have tested with, but execution on slower machines may be possible with the correct Tuning (See chapter 7 - Minimizing CPU Usage).

Summary of Installation Process

The installation of WinTone is fairly simplistic, but will require you to tune the settings for optimum performance. The program is distributed with an install program that will let you un-install the program at a later date if you wish to do so.

2 Installation

Running the Install

Run the SETUP.EXE from the installation floppy, or from the contents of the uncompressed ZIP file. You will be prompted for a directory for which to install WinTone and all of its support files. The install will then install all of the files onto your hard drive.

Optimizing for your CPU

WinTone uses Dynamic Link Libraries for most of its detection code. This allows the use of any one of five specially compiled libraries for your particular machine type. This will give WinTone the horsepower that it needs to do its processing, and hopefully leave some for you.

The Libraries are listed below :

WINTONE4.DLL	Intel 486 Optimized Module
WINTONE5.DLL	Intel Pentium Optimized Module
WINTONE6.DLL	Intel Pentium Pro Optimized Module
WINTONEM.DLL	Intel Pentium MMX Optimized Module
WINTONEX.DLL	Generic All-Processor Module

You will need to choose the best fit for your machine type, and copy that DLL to WINTONE.DLL. For example, if you have a Pentium MMX processor, you would type the following at a command prompt in the directory to which you installed WinTone :

```
C:\WINTONE> copy wintonem.dll wintone.dll
```

This will insure that WinTone is using the fastest possible methods for decoding tones on your system.

Note : By default, the WINTONEX.DLL has been copied to the WINTONE.DLL.

Installing optional Tone Modules

If you received optional tone modules for WinTone, you simply need to copy them into the same directory that you installed the program, and they will be auto-detected.

Setting up your environment

Setting up the environment is sometimes one of the hardest things to do in the installation. It is divided into several parts, which are discussed below.

Setting up your Scanner

In most cases, you will have a HAM Radio or Scanner designated as the source for your incoming signal. In this case, you will need to plug the Speaker or other audio out to the into the Microphone or

Line-In jack on your sound card (See 2-1 "Microphone or line-in" for more information) by using a compatible patch cable (Not Included).

The volume on the scanner should be set to about half the maximum. This normally produces the best results.

Discriminator Taps

Although it is not required, you may choose to hook your scanners Discriminator output into the scanner. This will result in less distortion, and therefore higher accuracy for detecting tones.

Configuring the Windows Mixer

Most windows compatible sound cards give you access to the built in audio mixer that is provided with windows. You can access this by double-clicking the speaker icon in the system tray. This will present you with many options for playing and recording sounds through your sound card. Make sure that the Recording volume under RECORDING is set to 50%-75%. Also make sure that either the Line-in or microphone inputs are selected, and their levels are about 50%-75%.

Microphone or Line-in?

In most cases, it does not matter. If you choose line-in, you should be delivering a non-amplified signal, and therefore may have to increase the recording levels in the Windows Mixer to get best results. Worst case would be that the signal would be too quiet.

Microphone on the other hand is amplified, but is being done by both the Source (Scanner) and the sound card, which may cause more noise and distortion than line-in. You may have to try them both, and decide which one works best for you.

3 Using the Program

Theory of Operation

WinTone decodes tones by sampling sound from the sound card at a given sample rate, and feeding that sound into a mathematical calculation which returns data on tone strength for given frequencies.

Frequency definitions
WinTone needs a list of frequencies for which to look for tones. These "Tonesets" are provided in DLL's which are located in the directory with all the other WinTone Files. If these DLL's are not there, or are corrupted - WinTone will not know which tones to look for.

Sessions
Decoding tones in itself would be pretty useless unless we had a way of organizing them. That's why WinTone uses "Sessions" of detected tones. A session is defined as a unit of time onto which tones are detected. The Session length will vary, depending on how long tones are detected. A session starts when a tone is detected. A session terminates when no tones have been detected in *x* amount of milliseconds. *x* is determined by the Session Threshold. So, if you have your session threshold set to 5000ms, and you have the following tones come in :

12345 <10 second delay> 67890

Then "12345" will be in a different session than "67890". This helps organize the tones by time, and give you a good idea of when tones came were detected in relation to one another.

NOTE
All post processing of tones (Triggers, Macros, Etc) will occur once a session closes.

Global Options

There are several options in the program which need to be accessed from many different tabs, these are the Global options. They are all in the form of buttons on the bottom of the screen, right above the status bar. These options are described below.

Enable/Disable detection
This option controls whether or not the detection engine is running. For most of the programs features, this should be enabled. This includes decoding live audio, monitoring and recording of WAV files, and Tone squelch features. However, it should be disabled for all decoding of WAV files. (This is done automatically) It should be noted that this feature is what makes WinTone processor intensive. So when this is enabled, you machines response to other applications may slow down.

Play WAV file
This option allows you to play a WAV file through WinTone, and decode tones while it does so. This option is identical to live decoding, except the real-time detection engine is shut down. WAV files must be either 8000Hz, 11025Hz, 22050Hz and be 8 Bit Mono. 16 Bit Stereo files are not supported in this release.

Decoding Results Screen

The decoding results tab is the main screen of the program. It contains all of the tones which are decoded, and is the source for triggers and destination of Macros. All the action occurs here.

The Decoding Grid
This is where the tones are displayed in real-time, as they are decoded. The grid consists of 3 columns : A date/time stamp of when the session was started, The length of the current session, and a list of tones that are detected.

Log and grid options
There are several options available to you on the top of this tab in the form of buttons. They are :

View Log	This runs an external viewer to examine the log file.
Clear Log	This erases the disk log and starts with a clean slate.
Print Log	This prints the current log to your default printer.
Clear Grid	This erases the contents of the detection grid.

Module Configuration

As stated before, WinTone gets its list of tones to detect from DLL's located in the WinTone directory.

The Modular design
WinTone was designed to be able to decode virtually any set of tones, regardless of their frequency. And since there are so many different Tonesets (DTMF, CCIR, ZVEI, EIA, EEA, etc.) we decided to make WinTone modular. If you need to be able to decode a different standard, simply copy that DLL into the WinTone directory and you instantly have access to tone decoding.. That simple.

Active Detection Module
As far as WinTone is concerned, there are 2 types of Tonesets. One is an active toneset, like DTMF or CCIR. These tones are normally short and numerous, and need to be decoded on the fly. On the Module configuration screen, you have a pull-down box of all Active detection modules that you have available to you. By selecting any one of these Tonesets from the pull down box, you make it active, and WinTone will attempt to decode tones based on the frequencies that are defined in that module. The tones for the selected module are listed in the grid below the pull down box, as well as that tone or tones identifier in the case it is detected. Only one active toneset may be active at any one time.

Tone Squelch Selection
The other type of ToneSet is a Tone Squelch set. These tones are normally constant in the background of any transmission, and are

usually sub-audible. An example of this type is CTCSS. They are used mostly for multi-user frequencies when you only want to hear one particular conversation, but do not want to hear anyone else. The tones are used to "Break Squelch". These tones are selected in the same way as active tones, by selecting the available modules from the pull down box. Only one of these tonesets may be active at one time, but they may be selected along with an active ToneSet.

Triggers

Detecting tones is great, but what if you want to be able to do something when those tones arrive?

What are Triggers?

Triggers are a set of "Rules" which are defined, that when activated, execute a given task. For example, I can define a trigger for the tones "123", which runs the program "NOTEPAD.EXE". Then when detection is enabled, and a session is detected which contains the tones "123", WinTone will launch that program for you. (Are you thinking remote control?)

Trigger actions

The heart of a trigger is what kind of actions it can perform when it receives the correct tones. Each different action has related parameters which it needs to operate. These are placed in the DATA field of each Trigger definition. Below is a list of all the Action Types that WinTone is capable of :

Execute External Program

Data : The full path and filename of the program to execute

This allows you to launch any program in the system. Simply include the path and filename, and you are ready to go.

Execute External Program w/Decoded Data

Data : The full path and filename of the program to execute

This works the same as "Execute External Program" with the exception that it passes the string of decoded tones (For the current session) as a parameter to your executable. It should be noted that it passes ALL the parameters, even the ones that caused the trigger. In the case that you do not want to include the tones that cause the trigger, you can get creative with the "Delays are Spaces" setting in the options window. With this, it will pass the decoded data with spaces, which will cause them to be passed a different command line parameters. Here is an example :

1) I set up a trigger for "123" to run an External Program w/decoded data, and have the data field point to a Batch File.
2) The batch file is set up to receive 2 command line parameters, %1 and %2
3) A set of tones is received : "123" <1 sec delay> "4567890"
4) The batch file is executed with 2 command line parameters, one being "123" and the other being "4567890".

Close External Program

Data : The title of the window to be closed

This command will close any window that has the title which matches the string in the data field. For example, when you run Notepad.exe by itself,

it has a title of "Untitled - Notepad". The title must match exactly, and a lot of programs change their title bars depending on what they might be doing.

Start Recording
Data : Filename of the WAV you wish to record to [, Time in Minutes]
If you do not specify a name, it will take the default name which is in the record filename field on the monitoring screen. The optional parameter is a time, in minutes, for which you want to record. After this time is reached, it will stop recording.

Stop Recording
Data : None
This stops the recording of a WAV file. If no WAV file is recording, no actions are taken.

Start Monitoring
Data : Path and file Prefix of the WAV file to start monitoring [,TIme in minutes]
If you do not specify a file prefix, it will default to the prefix defined in the Monitoring Filename box of the monitoring tab. The optional parameter is a time, in minutes, for which you want to monitor. After this time is reached, it will stop Monitoring.

Stop Monitoring
Data : None
This stops a monitoring session. If no monitoring session is active, no actions are taken.

Stop Listening
Data : None
This disables the decoding engine. Realize that if you stop the decoding engine, you will not be able to enable it again via a trigger.

Exit Program
Data : None
This simply terminates WinTone.

Macros

Macros are in essence a "Search and Replace" of sets of detected tones.

How macros work
When you define a Macro, you supply a string of numbers, and a replacement string for those numbers. For example, some emergency services such as fire engines and ambulances use a 10 tone signal before every voice communication. In this case you might here "1234567890...<Conversation>", where 12345 is the calling station (The base) and 67890 is the Fire Engine. If you define a macro for 12345 as "Dispatch" and 67890 as "Fire Engine", when you see the previous call in the detection grid (1234567890 <conversation>), it will look like "Dispatch - Fire Engine".

Defining Macros
To define a macro, simply switch to the macro tab, enter in a sequence of tones for which you wish to replace, then enter the string you wish to have it replaced

with. Then either hit the "ADD" button to add this as a new entry in the macro list or "MODIFY" to have the current entry replaced with the one you have defined.

Loading and Saving Macros

You can load and save your Macros to and from a file, to exchange with other Wintone Users. Simply hit the appropriate load or save button, and enter the file that you wish to load or save to.

Monitoring and Recording

One of the handiest features of WinTone is the ability to record the current data to a WAV file, for playback at a later time.

What is the difference?

What is the difference between recording and monitoring? Recording will simply start when you tell it to, record all the sound to a WAV file, and stop when you tell it to. It will even record silence if any exists. Monitoring is a "Smart" recorder. It will only record data to a WAV file when Sound is present, and will NOT record the silence.

Using monitoring

To use the monitoring feature, simply change to the monitoring tab, and type in a prefix for the WAV file that you wish to record. Not the this is a PREFIX ONLY, Wintone will add a sequential number to the end of each monitoring WAV file, when the monitoring is stopped.

Squelch Setting

The Squelch setting determines what is considered silence and what is considered speech. By adjusting this value, you will see the Monitoring status box either turn red for "Active" of Green for "Silent". Adjust it accordingly, so that when squelch is broken on your radio, that the box turns red.

Delay for break

The delay for break is the time which WinTone will keep the squelch open, after it has gone silent. This gives you a break between transmissions.

Using Recording

To use recording, simply specify a file name, and click the box to enable recording. The Recording is stopped when the box is unchecked.

Options

The following is a list of the options which appear under the options tab :

General Options

Format 7 and 11 character sessions as Phone Numbers

This option will add formatting characters to any 7 or 11 character session, to make it conform to (xxx) xxx-xxxx.

Ignore sessions that contain only 1 detected tone

In most cases, a session that contains only 1 tone is a false hit. in this case, you may check this box, and any 1 character session will be automatically deleted.

Warn if listening is to be disabled

When you play a WAV file, the Real time decoding engine is disabled.

Checking this box will force WinTone to pop up a dialog box when the real time engine is disabled.

Treat "E" as repeat tone

In most single tone decoding sets, you cannot have 2 of the same tone one right after another. In this case, they use an "E" as the signal for a repeat tone. This option will translate the "E" to the actual repeated tone - so in the string "1234E" it will give you "12344"

Erase log each time the application starts

This will delete the Disk log when WinTone is started. All previous recording sessions will be erased from this log, as there is no backup made.

Pauses within sessions translate to spaces

This option will monitor the detection sequence for gaps in time. When a gap exists that is longer than that specified by the tolerance indicator, it will add a space to the decoded string in the detection grid (When this option is enabled).

Eliminate duplicate tone sets between pauses

In some paging applications, the pages are repeated twice for ECC. So you may receive a string like "12345 12345 67890 67890". This option will eliminate the duplicate, giving you only the 1 set. SO in the above example you would receive "12345 67890". Note that you MUST have the "Pauses with session translate to spaces" option enabled to use this feature.

Enable Hints on all controls

If you hate to read the help file, enable this option and all controls will have pop up hints on them.

Devices

The devices boxes show you which recording and playback devices you are currently using. If you experience a problem with the current device, you may need to change to a different one, if available.

File/Path Options

There are 2 options. The log filename, which specifies the name of the decoding log, and the log viewer. The log viewer is launched when you hit the "View Log" button from the Decoding results tab.

Decoding options

From here, you can enable or disable any of the session processing features of WinTone. You can also Adjust the length of the session (In milliseconds) and the Sample rate.

4 Tuning

How tone decoding actually works

Wintone decodes DTMF tones by a method known as a Fast Fourier Transform, or FFT. With this method, Wintone has a sort of spectrum analyzer of how prevalent any given frequency is. The program then takes this information, and among other things, looks for the most prevalent tone over a given period of time. This is then the tone that is detected.

Now if you actually look at an oscilloscope of a DTMF tone, you will see that depending on the source, the waveform is rarely perfect, due to transmission conditions, and quality of the device that is generating it. These are the things that Wintone must overcome if detection is to be performed with any type of accuracy. This is where the 3 tuning options come into play.

The options and how they work

The 3 Tuning options which Wintone makes available are Threshold, Tolerance, and Bottom Threshold. These options control different aspects of how WinTone detects tones. These options are described in detail below.

Threshold for detection

To understand threshold, you need to know what WinTone does to detect the tone. Think of WinTone having a table of Frequencies. These frequencies correspond to frequencies you want to detect. Along with these frequencies, WinTone has a number which corresponds to its relative amplitude. For example :

Freq	1000	1050	1100	1150	1200	1250	1300
Value	**100**	**98**	**9000**	**92**	**993**	**1400**	**830**

In the above example, you see the frequencies and all thier associated values. Now Wintone uses an equation to determine if a tone is *a certian order of magnitude above the average.* This is how it sees if a given tone is being played. Threshold IS that magnitude. For example, if you have a threshold of 6, the value of the highest tone MUST be more than 3 times (Theshold /2) greater than the average of all the other tones. Why Divide the threshold by 2? It gives greater granularity. So lets look at the calulations wintone would do on the 1000Hz tone with 6 Threshold:

1) Average : (98+9000+92+993+1400+830)/6 = 2068
2) Calculate Value of 1000Hz multiplied by (Thresh/2) : 100*(6/2) = 300
3) Is this value greater than 2068? No, not detected.

Lets look at the same thing for 1100Hz, Same threshold :

1) Average : (100+98+92+993+1400+830)/6 = 586
2) Calculate Value of 1100Hz multiplied by (Thresh/2) : 9000*(6/2) = 27,000
3) Is this value greater than 568? YES!, Tone detected.

Now realize, that WinTone only does this calculation on the Two highest valued tones in the whole table, not on every one.

Also realize that Wintone refreshes that table may times per second (Depending on sample rate).

So what does this all mean to you? The lower the number, the more sensitive the program will be to tones. But it also increases your chances for false hits. As a general rule, 4-6 is a good range for a tone to be detected. If you raise the number, the tone will have to be clearer and more pronounced, lower the number and the tone could be choppy and still be detected (But so will random noise).

Tolerance for repeated tone

In the table described above, we saw that an 1100Hz tone was detected when the tones value was at 9000 - Plenty for a standard detection. But the value for a constant tone over time will oscillate, possibly between 9000 and 4000 (For instance in this case), several times in a second. (The higher the frequency, the faster the oscillation occurs)
If we detected the tone at a value of 9000, then it was not detected with a value of 4000, then detected again at 9000 - We would have detected the same tone twice. This would not be good. And the quieter the tone, and closer it came to "Squeezing by" the threshold, the more predominant this behavior can be.

That's where the Tolerance setting comes in. As we stated above, WinTone samples (Or refreshes the grid) several times in a second. The tolerance value can be defined as the number of samples that a tone must be undetected before it can be detected again. So lets look at another example, this time focusing on the 1100Hz tone over a period of time :

Sample	1	2	3	4	5	6
Value	9000	7000	9000	7000	5000	9000

Lets just say for the sake of argument that the value must be above 8000 to be considered detected. With a threshold of 1, we would have 3 detected tones (Sample 1, 3 and 6). Because the tone only has to have been "Inactive" for 1 sample. This fits the bill. But the same example with a tolerance of 3 would only have 1 detected tone (Sample1), because the most the tone goes "Inactive" is 2 samples (#4 and #5). So if you get 1 tone, but multiple detections, raise this tolerance. If you get 2 legitimate tones consecutively, but WinTone misses them, try lowering this.

Static bottom threshold

This feature gives a no-nonsense requirement for any tone to pass before even being considered for detection. Lets consider the following set of tones, in the "Table" format described above :

Freq	1000	1050	1100	1150	1200	1250	1300
Value	**10**	**-203**	**4**	**-10**	**500**	**34**	**83**

Now in this example, the 1200Hz tone is several orders of magnitude above the average (Which is like -15), but a tone value of 500 within itself could be nothing more than static which happens to be the strongest frequency available. Most tones distinguishable to the ear would have a strength of at least 3000-4000 (Very soft), average tones will be in the 11000 range, and strong might be in the 20000-30000 range. Wintone looks at the Bottom threshold, Multiplies this by 1000, and says that all tones which can be detected must be higher than this value in order to be detected. So a value of 1 is 1000, 3 is 3000, 5 is 5000, and so on. A good average is 3, but may need to be adjusted in a noisy environment. This should move inversely with Threshold, so if you move threshold down, move bottom tolerance up.

The Detection Spectrum

In the previous section, we spoke of a "Table" of values for frequencies, which WinTone keeps for determining which tone if any are detected. The Frequency Spectrum graphs all of the currently registered tones on a graph, with their values. Due to space constraints, we could not label the bars, so you will just have to know that the smallest frequencies are on the left, the largest to the right. This is here mainly as a tuning tool, so the actual frequency should not be that important as the value which it possesses.

Pifalls with the detection Spectrum

Great tool, really neat looking, but it has some drawbacks. WinTone will graph every frequency, at every 5th sample. When you use a detection set with 16 frequencies, that means that it may be drawing 16 boxes on the screen 20 times/second. On a slow machine this takes up ALL the CPU, and causes Wintone, if not the whole machine, to lock up. BE CAREFUL! As a rule of thumb, I suggest that anyone running a 486 NOT use the graph, unless they have a fast 486 (100Mhz or better) or unless they know how to "Tame" the graph.

Taming the Graph

The Graph has a few options associated with it. The first is the "Use Small graph" feature. This cuts the size of the graph down, and uses less CPU to redraw. The other option is the "Graph PL Tones" option. DO NOT USE THIS UNLESS YOU ARE ON A FAST MACHINE! This will graph all of the tone squelch tones, which for CTCSS alone is 38 bars. My P133 grunts when this happens.

Recovering from a lockup

If you happen to somehow change a setting, and it locks up your system, try restarting WinTone with a /NOLISTEN parameter. This will start WinTone without the real-time engine running, and therefore will not graph anything, let alone decode anything. You can then reset your options to a less CPU intensive setting, and re-enable the Real time engine from the Main Screen.

Minimizing CPU Load

For this information, see chapter 7.

5 Tone Squelch

Selecting the Tone Squelch module

To enable tone squelch, you only need to select a Tone squelch tone set from the Module Config Tab. Notice that there are 2 windows, one is for active tone selection like DTMF, and the other is for tone squelch. If you do not have any options under tone squelch sets, contact Steak Sandwich Software, and make arrangements to fix the problem.

Quirks of low frequency detection

Since WinTone uses a method called FFT to determine frequency, it has a problem inherent to all FFT based algorithms which is the problem of lower frequency detection. PL tones normally exist in the 69-250Mhz range. Winones ability to correctly identify a tone in the 69-80Hz range is minimal at best. All frequencies above this (80 Hz on up) should be accurate. If you encounter a frequency below this, it may fluctuate between a few different "Close" frequencies. But WinTone has built in a workaround for this known as Fuzzy detection, which should make Squelch act reasonably well.

Determining PL Tones on a frequency

WinTone has 2 ways of utilizing tone squelch. The first is the ability to tell you what "Tone" is currently being used. You can simply flip to the Tone squelch tab, and look at the "Live tone squelch Data" window. This has the current tone, and its corresponding strength and Identifier. If you want to know what tone is in use on any given frequency - This is all there is to it!

Using tone squelch for monitoring

The second use for Tone squelch is the ability to monitor and decode only when the given frequency is in use. To do this, simply flip to the Tone Squelch tab, select which tone(s) you wish to be the "Active" frequency, and select which options you wish to be dependant on tone squelch (Either Decoding or Monitoring).

Tuning Tone Squelch

Tuning tone Squelch is fairly simple, and works much like the options for tuning the Active tone Detection (Chapter 4).

Threshold for Tone Squelch

This Option controls at what level the most prevalent tone is considered active. This is basically the "Bottom Threshold" value (See chapter 4) for Tone squelch. Unlike active tone detection, this is the ONLY value which determines whether a tone is detected or not. By using the Tone value indicator in the Live Tone Squelch Data window, you can see what the proper value should be. A good starting point is 1500.

Sample Rate

This determines how long the buffer is that it uses fir PL tone detection. Since low tones are harder to detect, WinTone "Records" a given amount of sound, then lets you know which PL tone was detected most

for that amount of time. Shorter numbers mean WinTone will evaluate the sound more often, but less accurately. Larger numbers mean a very accurate sample, but it checks to see if that has changed less often. Wintone ONLY updates the PL frequency when a sample ends. Why would this effect you - Why not just sample every 10 seconds if it is more accurate? If you use any of the options to ONLY record or decode when a given tone is active, and WinTone only updates this information every 10 seconds, you may get conversations that you don't want. For example :

```
Now lets say that we have a 103.5Hz PL and 167.9 Hz
PL each taking 5 seconds of a 10 second blurb :

      103.5Hz                       167.9Hz
:-----------------------*-----------------------:
0 Seconds            5 Seconds               10 Seconds

:-------------------------------:----------------
```

Wintone checks every 6 seconds

Now if WinTone is set to sample every 6 seconds for a PL tone, it will start out detecting 103.5 at 0 seconds, then 167.9 at 6 seconds. Wintone would have recorded 1 second of the 167.9Hz transmission before it realized that the PL tone had changed.

Why not check every 100ms then? Because a lot of things actually make noise at low frequencies, that may be mistaken as PL Tones. By taking a sample of a larger period of time, you are able to determine which tone was constantly detected, eliminating any mistakes. I have found that any setting that gives you between 500ms and 2000ms is ample, but you can tune it to your environment.

Fuzzy Threshold

This option counteracts the unpredictable and sometimes inaccurate detection of low frequencies with an FFT (What WinTone uses). When a misdetection occurs with PL Tones, it normally detects a frequency close to the one that is actually present (Like detecting 71.9 when 74.4 is actually the correct tone). By setting the fuzzy threshold, you tell WinTone to accept a certain number of tone above and below the selected one to be acceptable to break Tone Squelch. Let me give you an example :

You have the following PL frequencies :

67.0
71.9
74.4
77.0
79.9

Now say you tell WinTone to only decode when the 74.4 tone is detected. Well, since tones below 85Hz are so unpredictable, it may detect 74.4 and 71.9 and 77.0. If you set the fuzzy detection to 1, you tell WinTone to allow 1 frequency above, and 1 below to also break squelch - Meaning 71.9 and 77.0 would still break squelch, but 79.9 or 67 would not. Set this option to 2, and you get the 2 frequencies above and 2 below to also break squelch.

This should work fine in most situations, because when you have a frequency that is shared. your shared users will normally NOT have a PL frequency that is too close to another. For example, you may have a frequency like 154.6 MHz that has users which use 103.5, 127.3, 162.2 for PL, but wont find them close together like 136.5, 141.3 and 146.2.

Delay for Break

This option is similar to that used in the Monitoring function. After a given PL tone has been detected, then NOT detected, it will still register as detected for x amount of milliseconds. Good for putting gaps in monitoring files. But if you do not want gaps, or have a busy frequency, you want this option to be set very low (Like 1).

6 Minimizing CPU Usage

WinTone, by its nature, is a CPU hog. Doing that many floating point operations/second make it unavoidable. In some cases, if your machine cannot keep up, WinTone will warn you that it is too much to handle, and will shut down the listening engine. There are some things that you can do to "Lessen" the load on your poor slow machine :

More Frequencies - More headaches
The more frequencies WinTone has to monitor, the slower it will be. If you experience lockups, try disabling Tone squelch (That's about 3 times as many tones as a standard detection set like DTMF) - or try selecting a tone set with less tones. For example, if you don't need to detect ABCD in DTMF, select Standard DTMF instead of Extended DTMF in your module config - This lets WinTone look for 12 tones instead of 16.

Also, if you add too many Tone Squelch tones to squelch on, it may Run into problems. Choose ONLY the PL tones that you need, then Use the FUZZY feature.

Feature packed - Broken back
Some of the features of Wintone are great, but can take more CPU leading to lockups on slow machines. Try the disabling PL Tone Squelch for monitoring, Trigger Processing, and Macro Processing if they are not being used. They do add overhead.

Use the right DLL!
WinTone comes with several DLL's, each optimized for a different processor. Make sure you have renamed the correct one so that WinTone can Use it! (See the Installation Chapter for more info!)

Swap file - Bad News
If you have a slow machine, or even a fast one with no memory, Windows will constantly be using your Swap file, which KILLS your system performance. Make sure in this case that you make sure that WinTone is the only thing running. Also, recording to a WAV file requires a fair amount of RAM (Not too much, just enough to buffer a few samples) - But if that gets kicked into the SWAP file, say "Goodnight Gracie".

Sample rate is GREAT!
Probably the single biggest thing you can do to improve the speed is lower your sample rate. 8000Hz is your best bet. Your recording quality is decreased, and your shortest detectable tone is about 80ms, But you will be running in style.

7 Troubleshooting

Below are some of the most common scenarios
which I have questions about :

First off, make sure you try detection with the test WAV files that were included with WinTone. This will insure that you know what tones are being sent, and that the quality is acceptable.

It misses tones, irregularly - Every so often
Example : "5551212" comes out to "5521"
It is most likely your threshold. Lower it! If you lower it, and start getting misdetections (Tones where no tones exist), raise the threshold back up, and decrease the bottom threshold. In a single tone environment, raise your tolerance. Yes, your tolerance. It is an oddity that is hard to explain. A value of 45-50 is not uncommon. Besides, single tone sets never have duplicate tones, they use "E" instead.

It misses duplicate tones,
Example : "5551212" comes out as "51212"
Lower your Tolerance.

It gives me duplicate tones.
Example : "5555551122211222"
Raise your tolerance and/or lower your threshold.

With this information in your arsenal, you should be able
to tune the program to detect even in the noisy environments!

8 Technical Support

Technical Support and Bug Reports

In the event that you wish to contact Steak Sandwich Software for support or bug reports, we have provided a list of contact methods for which you may do so.

Support Email	***support@steaksandwich.com***
Bug Reports	***bugs@steaksandwich.com***
General Comments	***admin@steaksandwich.com***

As a convenience, we have also set up an Email list server, which will automatically notify you in the case of new developments and software updates. To subscribe to this list, simply send an Email to

listserver@steaksandwich.com

With the word ADD anywhere in the subject text. You may also remove yourself from the list by emailing the same address with the word REMOVE in the subject line.

We try our best to help you get up and running with any of our software products. If you have a problem, let us know and we will do our best to get the problem solved.