Chapter 4. Fine Tuning

To fine-tune the AudioLink communications scheme, numerous parameters had to be set. Although a dependency exists among the many variables, an exhaustive search for the optimal settings over the entire parameter space would be prohibitively complicated and time-consuming. Therefore, when conducting tests to fix the parameters, only a single parameter was varied for each group of tests. Although this method is unlikely to yield the absolute best settings for all parameters collectively, it provides a good indication of which are most relevant to performance and approximately how they should be chosen. Given the limited time allotted for adjusting the settings, our final choices seem well justified.

Descriptions and results from some of the parameter tests are provided in the sections that follow. Unless otherwise noted, all table values in this chapter are given in units of data bit errors upon reception (out of 47 data bits total). Therefore a value of zero represents error-free reception and a value of “F” represents a complete communication failure (i.e., the control function itself was degraded beyond recognition and no data was received). The values for the blank cells in the table are obvious from the data trends indicated. In other words, reducing the signal to noise ratio further cannot cause more errors than a complete communication failure. Likewise, once communication is completely successful, improving the SNR cannot improve on the 100 percent reception rate.

In each group of tests, six sources of interference were added to the composite audio (filtered audio plus code) at various relative power levels. Four music and two voice samples were recorded from FM radio and saved to WAV files to be used as standard sources of interference. During each test one of these WAV files was mixed with the composite audio at a chosen relative level, and the detection process was initiated on the noisy sum of signals. It is important to note that the levels specified in the tables should not be interpreted as absolute signal to noise ratios, since the audio-plus-code signal power is used as a reference, and not the code signal alone. The code was already mixed with the filtered audio at a desired analog to digital ratio (ADR), as described in Section 3.2.2, to yield the composite signal.

The interference level is set based on the local power level of the composite audio at the point of the code insertion only. Since the code insertions are short in time duration (approximately 300 ms), the nature of the interference can vary drastically based on the temporal location of the code insertion point. This is true even within a given interference source. However, consistent and meaningful results can still be obtained by holding the insertion location constant for the test series.

Section 4.1. Sampling Rate and FFT Size

Choosing the sampling rate and the FFT size at the AudioLink receiver involved several tradeoffs. Although we would like to use high frequencies for the sinusoids in order to take advantage of reduced human sensitivity, the television and other equipment in the signal path are lowpass and limited in their bandwidth. The television standard supports an audio bandwidth up to 15 kHz [14], but in practice the rolloff occurs much sooner, especially in older and poor-quality equipment. Of course the sampling rate of the AudioLink must be sufficient to satisfy the Nyquist criterion, with room for a transition band in the anti-aliasing filter. Since the computational requirements for the DSP are strongly dependent on the sampling rate, it must be kept as low as possible. Furthermore, a crystal oscillator and a divide-by factor in a codec control the sampling rate, so only certain rates are possible (to be discussed in Section 5.1). After reviewing these various considerations, a sampling rate of 16.0 kHz was chosen.

Regardless of the sampling rate, a 1024 point FFT could not support the number of sinusoids necessary for the signaling scheme because of the separation required between them (as discussed in Section 3.1.1). Given the 16.0 kHz sampling rate, however, both 2048 and 4096 point FFTs would allow a sufficient number of sinusoids on FFT bins in the appropriate frequency region. With a 4096 point FFT the sinusoids could be placed six bins apart, but the spacing must be reduced to three bins for the 2048 point case. The test results from comparing the two FFT sizes are shown in Table 4.1 and Table 4.2 below. In both cases a rectangular window was applied to the data. As the tables show, similar performance is achieved using both FFT sizes. **The reader is reminded that all table values in this chapter are given in units of data bit errors upon reception (out of 47 data bits total)**. Therefore a value of zero represents error-free reception and a value of "F" represents a complete communication failure (i.e., the control function itself was degraded beyond recognition and no data was received). The values for the blank cells in the table are obvious from the data trends indicated.

Table 4.1 FFT Size Test - 4096 Points

Interference	Audio+Code to Interference Ratio in dB					
	-10	-5	0	5	10	15
Music 1	F	7	0	0	0	0
Music 2	12	4	2	0		
Music 3	F	F	16	4	0	
Music 4	F	3	1	0		
Voice 1	7	3	1	0		
Voice 2	0					

Table 4.2 FFT Size Test - 2048 Points

Interference	Audio+Code to Interference Ratio in dB					
	-10	-5	0	5	10	15
Music 1	F	2	0			
Music 2	F	9	5	0		
Music 3	F	F	F	9	0	
Music 4	F	10	0			
Voice 1	5	8	2	0		
Voice 2	0					

After the discussion in the previous chapter, the reader is probably wondering, how it can be possible to use the rectangular window; after all, there is a lot of sidelobe leakage and interference, especially given the tight packing of the sinusoids in the 2048 point case. However, when these tests were conducted, an analog channel was not yet inserted into the transmission path. In fact, the signal remained completely digital and in the computer for these tests. Therefore, all sinusoids existed at their nominal locations, exactly on the FFT bins, and the samples of the FFT for the multiple sinusoids occurred precisely at the peaks and zero-crossings of the sinc functions (see Figure 3.2 in Section 3.1.1 on page 53). The rectangular window affords the only possibility for resolving the densely packed sinusoids for the 2048 point case. To obtain a fair comparison between FFT sizes only, the rectangular window was applied in the 4096 point case as well. As soon as an analog channel (and in particular a frequency shifting one) was inserted into the transmission path, however, 2048 point FFTs could no longer be used successfully. Choosing an FFT size of 4096 then became simple, since it was the only possibility that still worked.

Section 4.2. Signal Block Overlap Considerations

The AudioLink samples the received audio signal at 16.0 kHz, and FFTs are performed on 4096 point (256 ms) sample blocks. As mentioned in Section 3.1.1, after a block has been processed some number of the oldest samples are discarded and the same number of fresh samples are grabbed to form a refreshed block. This block is then processed, and the cycle is repeated. The number of samples that are identical from one block to its successor is called overlap, usually expressed as a percentage of total block size. For example, in a 4096 point signal block , if 1024 of the old samples are discarded and 1024 fresh samples are inserted, the old and the new blocks contain an identical subblock of 3072 samples. In this case the overlap would be described as 3072 / 4096 * 100 = 75 percent. Several factors had to be considered when the size of the block overlap for the AudioLink was chosen.

If the overlap is chosen large, then consecutive blocks are very similar and contain most of the same signal and information. In the current application this implies that consecutive blocks would produce almost (if not) identical estimates for the received data, and little new information is obtained. Furthermore, in order to operate in real time, as is a requirement here, all processing must be performed at the block refresh rate. Therefore, for large overlap very little time is available for the DSP to complete its many calculations. Consider the extreme case of block update rate, where consecutive blocks differ by a single sample, and minimal new information is introduced into new blocks. This corresponds to a block overlap of 4095 / 4096 * 100 = 99.976 percent, and the DSP must complete all processing within 1 / 16.0 kHz = 62.5 μs. The DSP in the AudioLink requires more than 100 times this long just to compute the 4096 point FFT, never mind the many other calculations necessary for the interactive TV application.

The opposite extreme is zero overlap between blocks. (We will only consider cases where the entire signal is used in some way. Of course many samples could be jumped over between blocks, and the skipped samples could be discarded altogether. This would be inefficient for our current application, however, and would be very detrimental to our reception capability.) In the zero overlap scenario successive blocks are entirely independent, and contain no identical signal sub-blocks. Since each block contains 4096 samples, all processing must be performed within 4096 / 16.0 kHz = 256 ms. The AudioLink's DSP can easily handle such a computational load.

There are another 4095 overlap possibilities to consider between the zero and 100 percent overlap extremes. Since the code insertions are typically short in time duration, it was decided to set the overlap close to the maximum possible given the computational burden of the DSP. This provides maximum information reception potential, while ensuring real time operation. Therefore the overlap was set to 95 percent, and 205 new samples are used to refresh each signal block. This provides the DSP 12.8125 ms to perform the FFT and other required computations. Since the audio transmission channel will often be corrupted by extraneous room noises, and human voices in particular, the 12.8125 ms refresh rate is appropriate for another reason. Human speech can be considered statistically stationary in a wide sense for frames of ten to twenty milliseconds [23]. Therefore, in the presence of human voice interference, a refreshed block will often have independent interference statistics compared to its predecessor, and thus it can truly provide new and independent signal information.

Section 4.3. FFT Window Choice

As described in Section 3.1.1, the choice of data window can significantly alter the spectral estimates produced by the FFT, and thus the data reception capability. In order to choose a window for the AudioLink, several were investigated by their application during simulated data reception in the presence of the standard noise interferences. Table 4.3 through Table 4.8 below show the results of the tests for the rectangular, Hamming, Hanning, Blackman, J-, and Kaiser Windows (β=8.5), respectively. (The J-Window is the point-by-point square of the Hanning window. In the frequency domain it is thus the convolution of two Hanning windows.) During these tests all other parameters were held constant. As the tables show, the Hanning and Hamming windows provided the best overall performance in the presence of the interferences, and were approximately equivalent for these tests. Since the codes will be inserted at a low level relative to much stronger (masking) audio content, the Hanning window with its faster rolloff rate was chosen in order to minimize the leakage interference effects from the masking audio.

Table 4.3 FFT Window Test - Rectangular

	Audio+Code to Interference Ratio in dB					
Interference	**-10**	**-5**	**0**	**5**	**10**	**15**
Music 1	F	7	0	0	0	0
Music 2	12	4	2	0		
Music 3	F	F	16	4	0	
Music 4	F	3	1	0		
Voice 1	7	3	1	0		
Voice 2	0					

Table 4.4 FFT Window Test - Hamming

	Audio+Code to Interference Ratio in dB					
Interference	**-10**	**-5**	**0**	**5**	**10**	**15**
Music 1	F	3	0			
Music 2	12	5	3	0		
Music 3	F	F	F	3	0	
Music 4	F	5	1	0		
Voice 1	7	5	0			
Voice 2	0					

Table 4.5 FFT Window Test - Hanning

	Audio+Code to Interference Ratio in dB					
Interference	**-10**	**-5**	**0**	**5**	**10**	**15**
Music 1	F	3	0			
Music 2	12	5	4	0		
Music 3	F	F	F	4	0	
Music 4	F	4	1	0		
Voice 1	7	6	0			
Voice 2	0	0				

Table 4.6 FFT Window Test - Blackman

Interference	Audio+Code to Interference Ratio in dB					
	-10	-5	0	5	10	15
Music 1	F	4	1	0		
Music 2	13	6	2	1	0	
Music 3			F	5	0	
Music 4	F	4	1	1	0	
Voice 1	8	6	1	0		
Voice 2	0					

Table 4.7 FFT Window Test - J-Window

Interference	Audio+Code to Interference Ratio in dB					
	-10	-5	0	5	10	15
Music 1	F	1	1	0		
Music 2	12	8	3	1	0	
Music 3			F	7	1	0
Music 4	F	2	0			
Voice 1	8	6	1	0		
Voice 2	0					

Table 4.8 FFT Window Test - Kaiser, Beta=8.5

Interference	Audio+Code to Interference Ratio in dB					
	-10	**-5**	**0**	**5**	**10**	**15**
Music 1	F	4	1	0		
Music 2	13	6	2	1	0	
Music 3			F	5	0	
Music 4	F	4	1	1	0	
Voice 1	8	6	1	0		
Voice 2	0					

Section 4.4. Spectral Estimation Method

Square root operations are necessary when estimating a magnitude spectrum from FFT results. However, such square root calculations are extremely taxing for digital signal processors, and demand many clock cycles per square root. Since so many square roots are required even for the subset of FFT bins examined in the IVDS application, the DSP could not possibly complete them within a reasonable time. An alternative is to use the magnitude-squared spectrum instead of the magnitude spectrum when polling for the sinusoids. However, given the 16-bit data word length in the DSP, this alternative too was unacceptable since it would quickly consume the available dynamic range. Since neither approach would satisfy the problem constraints and requirements, we decided to attempt using the maximum of the absolute values of the real and imaginary parts of the complex FFT result as a magnitude estimate. If the FFT result at a particular bin is

$z = x + iy$ where $i = \sqrt{-1}$, then the true magnitude is given by $|z| = \sqrt{(x^2 + y^2)}$. Our substitute estimate is given by $|z|_{est} = \max(|x|, |y|)$. Although the magnitude estimate derived in this manner does not have a direct relationship with the true magnitude spectrum (although it could be considered a sort of lower bound, since $|z|_{est} = \max(|x|, |y|) \le \sqrt{(x^2 + y^2)} = |z|$), it proved to be a reasonably effective alternative in practice. The results of using the true magnitude spectrum and the substitute estimate in noise interference tests are provided below in Table 4.9 and Table 4.10 respectively. As the results show, very little performance penalty is incurred by using the substitute magnitude estimate. However, the substitute is much less taxing on the DSP, and so the processing can be performed within the desired block refresh time.

It should also be noted that when comparing magnitude levels to make bit decisions, the four dB threshold used corresponds to a scale factor of about 1.584. The bin magnitude estimates are *not* converted to dB and then compared by subtraction, since the additional computational requirements would overwhelm the DSP for no reason. Rather, the scale factor is used directly for the comparisons.

Table 4.9 Spectral Estimate Test - Magnitude in dB

Interference	Audio+Code to Interference Ratio in dB					
	-10	**-5**	**0**	**5**	**10**	**15**
Music 1	F	4	0			
Music 2	10	3	1	0		
Music 3	F	F	11	1	0	
Music 4	F	2	0			
Voice 1	F	6	2	0		
Voice 2	0	0				

Table 4.10 Spectral Estimate Test - Max(|Real|,|Imag|) in dB

Interference	Audio+Code to Interference Ratio in dB					
	-10	**-5**	**0**	**5**	**10**	**15**
Music 1	F	5	0			
Music 2	10	4	1	0		
Music 3	F	F	9	2	0	
Music 4	F	2	0			
Voice 1	F	7	2	0		
Voice 2	0	0				

Section 4.5. Weighting of Received Bits

As discussed in Section 3.1.2, the control function not only serves a synchronization purpose, it also provides an indication of the quality of the received data in each signal block. Therefore, when the data is being tabulated over successive signal blocks, the results of each block are weighted according to the value of the control function in that block. For example, we have more confidence in the data when the control function is 21/21 versus when it is 14/21. The data associated with such blocks should be weighted accordingly.

Although an infinite number of possibilities exist for weighting the received data based on the control function, we only investigated two options – linear and quadratic weighting. In the linear weighting case, the data bits in a block are assigned the control function value from that block. Thus digital ones are represented by positive control function values and digital zeros by negative control function values. For the quadratic case, the data bits in a block are assigned the square of the control function value from that block. Thus digital ones are represented by positive squared control function values, and digital zeros are represented by negative squared control function values. Note that in both cases (linear and quadratic) positive values are used for digital ones, and negative values for digital zeros. However, the weighting of the data in the two cases is different (control function versus squared control function). As with the other parameter investigations, noise tests were conducted with the standard interference sources, and the results are provided in Table 4.11 and Table 4.12 below.

Table 4.11 Data Weighting Test - Linear Control Function

Interference	Audio+Code to Interference Ratio in dB					
	-10	-5	0	5	10	15
Music 1	F	3	0			
Music 2	12	5	4	0		
Music 3	F	F	F	4	0	
Music 4	F	4	1	0		
Voice 1	7	6	0			
Voice 2	0	0				

Table 4.12 Data Weighting Test - Quadratic Control Function

Interference	Audio+Code to Interference Ratio in dB					
	-10	-5	0	5	10	15
Music 1	F	3	0			
Music 2	12	5	4	0		
Music 3	F	F	F	2	0	
Music 4	F	4	1	0		
Voice 1	7	6	1	0		
Voice 2	0					

The tests indicated that the quadratic weighting provided slightly better reception capability in some instances. However, the performance enhancement was too slight to warrant the extra processing required for the quadratic over the linear weighting.

Furthermore, with the 16 bit word length of the DSP, overflow could occur with the quadratic weighting if too many data votes were received over the course of about two seconds. For these reasons we chose to linearly weight the data bits by the control function. If the performance gain had been more significant for the quadratic weighting case, methods of avoiding the potential overflow problems would have been pursued, and the extra processing required would have been tolerated and accommodated.

Section 4.6. Sinusoid Detection Threshold Levels

As presented in Section 3.1.1 the presence or absence of a sinusoid, and thus a digital one or zero, is determined by comparing a candidate FFT signal bin with its neighbors two away on each side. If the signal bin's magnitude exceeds both its reference neighbors' by four dB or more, a digital one is indicated. If the condition is not satisfied, a digital zero is received.

After numerous hours of operation and testing in various environments, it was found that the criterion used to detect the presence of a sinusoid was sometimes satisfied even when no sinusoids were present. Further investigations revealed that ironically it was the low noise cases that most often produced such occurrences. After additional testing we found that when the noise floor is low, the difference between some very small FFT bin value and an even smaller bin value can be large in dB (or scale). For example, $1=10^0$ and $10=10^1$ would both be considered very small values in this application (essentially both are equal to zero, since the magnitude estimates can range from 0 to

32,768). However, since there is a 20-dB difference between them, the sinusoid detection criterion would easily be satisfied by these numbers.

In order to protect against such false sinusoid detections, the detection criterion was amended to include the condition that the magnitude of the candidate signal bin must also be within fifteen dB of the magnitude of the nearest control bin. This additional constraint successfully prevents false detections in very low noise environments.

Section 4.7. Hard Versus Soft Decoding

When decoding the triplication code as described in Section 3.1.4, another possibility exists besides the procedure presented earlier. In the method already discussed, hard bit decisions are made within each of the three data subbands, and the triplication code is undone after these hard decisions yield binary ones and zeros. Refer again to Table 3.2, which for convenience is also provided below as Table 4.13. The row labeled "Final Values" contains the sum obtained from the bit voting in time. These sums are compared with a threshold of zero to yield bit decisions within the subbands, and the triplication code is undone as the final step.

Table 4.13 Example of the Data Decoding Process

FFT Block	Control Function	Bit 1-1	Bit 2-1	Bit 1-2	Bit 2-2	Bit 1-3	Bit 2-3
1	4	0	0	0	0	0	0
2	3	0	0	0	0	0	0
3	14	14	-14	14	-14	-14	-14
4	18	18	-18	18	-18	-18	-18
5	21	21	-21	21	-21	21	-21
6	20	20	-20	20	-20	20	-20
7	16	-16	-16	16	-16	-16	-16
8	13	0	0	0	0	0	0
9	7	0	0	0	0	0	0
10	2	0	0	0	0	0	0
Final Values		57	-89	89	-89	-7	-89
Bit Decisions		1	0	1	0	0	0

However, the data decoding can also be performed in a “soft” manner. This alternative strategy involves combining the decimal “Final Values” from the three subbands, and then making the final hard bit decisions only as a last step after the triplication decoding is done “softly.” For example, in the table Bit-1 has received values of 57, 89, and –7 within the three subbands. Instead of converting these values to [1 1 0] and undoing the triplication code to reveal a final value of 1, the decimal values can be combined to undo the triplication. So Bit-1 would have a value of 57 + 89 + (-7) = 139. Since this final value of 139 is greater than zero, it would indicate a digital one. In other words, the triplication can be undone softly based on the decimal values, and the hard bit decisions can be postponed until the final step.

In order to evaluate the performance of the two decoding schemes, the standard interference tests were conducted. The results are provided in Table 4.14 and Table 4.15 below.

Table 4.14 Data Decoding Test - Hard Decisions

Interference	Audio+Code to Interference Ratio in dB						
	-5	0	5	10	15	20	25
Music 1	F	8	1	0			
Music 2	F	9	6	3	2	0	
Music 3		F	F	F	F	2	0
Music 4				F	2	0	
Voice 1		F	7	5	4	0	
Voice 2		F	4	1	0		

Table 4.15 Data Decoding Test - Soft Decisions

Interference	Audio+Code to Interference Ratio in dB						
	-5	0	5	10	15	20	25
Music 1	F	7	0	0			
Music 2	F	9	3	2	0		
Music 3		F	F	F	F	2	0
Music 4				F	0		
Voice 1		F	7	6	1	0	
Voice 2		F	4	1	0		

Although the test results indicate an improvement in performance of about five dB by using soft instead of hard decisions, the AudioLink decoding scheme was already fixed by the time the latter tests were completed. Therefore, current AudioLinks are using the hard decoding strategy, but future generations of AudioLinks could be re-programmed to decode with the soft decision scheme instead.

Section 4.8. Audio Signal Quantization

After inserting a digital signature into a target audio signal, the samples of the composite result are quantized to some number of bits and saved to a binary file. The number of bits required to accurately represent the samples of the sinusoids and the audio was investigated by saving the signal with varying sample resolutions, and performing the standard interference tests. Since all signals received at the AudioLink will also be quantized (to 16 bits), these tests have further significance. If the received analog signal utilizes the full available dynamic range before quantization, then all (or nearly all) of the 16 bits will be used. However, if the received analog signal consumes only a small part of the available dynamic range before quantization, then only a few of the 16 bits will be used in signal representation. Without effective automatic gain control this situation will exist whenever the received audio is of low volume level. Even if the digitized signal is subsequently scaled to full range, the effective signal quantization remains as the number of bits used during the analog-to-digital (A/D) conversion.

To determine the number of bits necessary for adequate signal representation in this application, composite (audio plus code) signals were saved for various sample resolutions (between four and sixteen bits per sample), and the standard interference tests were performed. The results of the tests are tabulated below in Table 4.16 through Table 4.20, and indicate that at least eight full bits of signal resolution are required for proper reception in typical operating environments. Since most signals utilize less than the full range during quantization, eight full bits implies ten or eleven actual bits of sample resolution. Given that sixteen bits are used in the AudioLink codec, even signals that deviate from maximum range by approximately a factor of 100 can be handled without difficulty. Automatic gain control would increase this effective signal range even further, assuming that saturation is avoided.

Table 4.16 Audio Signal Quantization Test - 4 Bits Per Sample

	Audio+Code to Interference Ratio in dB					
Interference	**-10**	**-5**	**0**	**5**	**10**	**15**
Music 1	F	F	F	F	F	F
Music 2						F
Music 3						F
Music 4						F
Voice 1						F
Voice 2						F

Table 4.17 Audio Signal Quantization Test - 6 Bits Per Sample

Interference	Audio+Code to Interference Ratio in dB					
	-10	**-5**	**0**	**5**	**10**	**15**
Music 1	F	7	4	0	0	0
Music 2	16	7	5	1	0	
Music 3	F	F	F	7	0	
Music 4	F	11	3	3	2	0
Voice 1	9	6	1	0		
Voice 2	0	0				

Table 4.18 Audio Signal Quantization Test - 8 Bits Per Sample

Interference	Audio+Code to Interference Ratio in dB					
	-10	**-5**	**0**	**5**	**10**	**15**
Music 1	F	3	0			
Music 2	12	5	3	0		
Music 3	F	F	F	2	0	
Music 4	F	3	1	0		
Voice 1	8	5	0			
Voice 2	0	0				

Table 4.19 Audio Signal Quantization Test - 12 Bits Per Sample

Interference	Audio+Code to Interference Ratio in dB					
	-10	**-5**	**0**	**5**	**10**	**15**
Music 1	F	3	0			
Music 2	12	5	4	0		
Music 3	F	F	F	4	0	
Music 4	F	3	1	0		
Voice 1	6	5	0			
Voice 2	0	0				

Table 4.20 Audio Signal Quantization Test - 16 Bits Per Sample

Interference	Audio+Code to Interference Ratio in dB					
	-10	**-5**	**0**	**5**	**10**	**15**
Music 1	F	3	0			
Music 2	12	5	4	0		
Music 3	F	F	F	4	0	
Music 4	F	4	1	0		
Voice 1	7	6	0			
Voice 2	0	0				

Section 4.9. Miscellaneous Parameters

Aside from the parameters already discussed, there are numerous others that also affect AudioLink performance and these were investigated accordingly. Some of these parameters and investigations are briefly described below.

With the final choices of the Hanning window, the 16.0 kHz sampling rate, and an FFT size of 4096 points, a sufficient number of sinusoids - spaced six bins apart - could be packed within the target frequency region of 2.4 to 6.4 kHz. But sinusoidal spacings of four or five bins were also possible under the above conditions. Tests revealed, however, that a six bin separation provided the best performance overall. Other limited tests showed that the ordering of the data bits was not critical to performance, as long as redundant bits did not occur close together in frequency. Therefore the data bits were ordered LSB to MSB across increasing frequency, with the same ordering repeated within each triplication subband. Varying the amount of redundancy was also investigated momentarily. However, the amount of redundancy could not be increased any further if all data bits were to have equal weight, since the length of the digital signature consumed the entire available frequency range for the double redundancy alone.

Since digital ones are represented by the presence of sinusoids and digital zeros by their absence, energy is inserted into the audio for the ones but not the zeros. In order to somewhat balance the energy for the two binary states, we investigated flipping the bits in the central data subband. However, no performance gain was achieved by such bit flipping, and so it was abandoned. In a similar pursuit we investigated adding a bias to the bit decision threshold after the block-by-block voting in time, effectively moving it up or down from zero, and favoring zeros or ones respectively. Here too no consistent performance gain was observed, although better performance was possible in specific circumstances. In addition, we investigated using different independent sinusoid detection thresholds for the control and data bins. Experiments revealed that the best performance was achieved when they were both set to a level of approximately four dB.

When bandstop filtering the audio in preparation for the code insertion, it is possible to filter beyond the insertion location in one or both directions. This creates a filter buffer before and/or after the code location, where the audio content at the code frequencies has been removed. During detection, as the FFT blocks encroach upon the code location and contain more and more of the code energy, frequency content of the original audio at the code frequencies competes with and obscures the sinusoids. By removing the frequency content prior to and following the code location, such interference is reduced or eliminated, and cleaner data estimates are obtained for an extended duration. However, the removal of such frequencies before and after the code location severely reduces the efficacy of the psychoacoustic masking, and the audio distortion is too noticeable. Therefore, no pre- or post-filtering is applied when the codes are inserted.

Another choice that influences the perceived quality of the modified audio is the relative phases of the inserted sinusoids. If all the sinusoids are chosen at zero relative phase, they tend to produce a periodic sound. When the phases are assigned randomly from a uniform distribution over $[-\pi,\pi)$, their sum produces a more noise-like sound which is more easily hidden and is less noticeable to observers. Aligning the sinusoids with zero relative phase also produces the worst case maximum peak value for the code signal component. Since the filtered audio and the scaled code signals are mixed and saved to a binary file, overflow is less likely if the phases are assigned randomly.