



ATC-10B/Fixed Point C Source Code Report

Prepared by CIE Engineering, Inc. (02/28/2003)
FA100-00099

Abstract: This report describes the early testing and evaluation results for the ATC-10B/Fixed Point C Source Code provided by Digital Voice Systems, Inc. (DVSI) to the Federal Aviation Administration (FAA). A general description of the distribution is provided. Results of source code compilation for the ADSP-2188M and Intel x86 platforms are provided (including execution performance). Bit stream compatibility tests were conducted between the C-source code (fixed point implementation) and the VC-20 hardware platform (floating point implementation). Conclusions are provided at the end of this report.

ATC-10B/C Source Code Overview

Digital Voice Systems, Inc. (DVSI) (Burlington, MA) provided C Source Code to the Federal Aviation Administration for the fixed-point implementation of the ATC-10B voice compression algorithm.

Distribution Contents

The ATC-10B/C Source Code distribution contains:

- Documentation: A directory includes an algorithm description and a software overview description.
- ATC-10B Source Code: The voice encoder/decoder (vocoder) source code.
- Sample Application: A sample PC application is provided that works with file input/output and illustrates how to interface with the vocoder source code.
- Test Vectors: Sample linear and compressed voice files are provided.

The vocoder source code is supplied in 76 different C source files (*.c) and 135 C header files (*.h). The input/output source code is supplied in 10 different C source files (*.c) and 21 C header files (*.h). The input/output source code works with PC-based files (see "Test Vector Files" below).



CIE ENGINEERING INC.



The distribution isolates the vocoder code from the sample input/output code. This structure facilitates integration by allowing modifications to be made only to the input/output code without affecting the vocoder code.

SPECIAL CAPABILITIES

The distribution supports soft-decision detection using up to 4-bits to represent each compressed channel bit. With 4-bit soft-decision, a 0x0 (hexadecimal) represents a confident '0' compressed channel bit and a 0xF (hexadecimal) represents a confident '1' compressed channel bit.

The distribution sample application supports bit error rate modeling. Noise can be added to the channel bits to 'simulate' RF channel degradation. Available models include 'ideal', 'gauss', 'rician', and 'markov'. At least six bit error rate levels are available for each model.

TEST VECTOR FILES

The distribution input/output test vectors are supplied as PC-based files. The distribution includes three types of vectors.

- Raw PCM Linear Voice (*.DAT): These binary files contain raw 16-bit linear data streams with no headers or special formatting. These files serve as input vectors for the ATC-10B vocoder. A number of multiple speaker files are provided.
- ATC-10B Compressed Voice (*.BIT): These binary files contain compressed voice with each byte containing one bit stored in soft decision format (4-bits per channel bit).
- Synthesized PCM Linear Voice (*.SYN): These binary files contain 16-bit linear data streams that have been processed by the ATC-10B decoder.

PROCESSING STRUCTURE

The algorithm produces 96 compressed voice bits for every 20 milliseconds of linear voice data sampled at 8000 samples/second (16-bits/sample). The 20-millisecond data window is called a frame. The encoder processes a frame in two pieces to reduce overall delay, i.e., two 10-millisecond subframes. Forward error correction (FEC) is added to create the 96-bit encoded frame. The decoder is similarly structured using subframe processing after the FEC decoder.

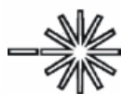


CIE ENGINEERING INC.



SOURCE CODE APPROACH

The ATC-10B/C Source Code approach uses C routines to implement 16-bit fixed point processing. In addition to standard DSP functions (e.g. FFT), it also uses C routines to mimic DSP computational units such as the DSP accumulator, multiplier/accumulator, and shifter computational units. Some proprietary functions are implemented using these computational unit emulation functions. This approach may support traceability between C and optimized assembly implementations of the ATC-10B algorithm. At the very least, it facilitates easy translation of C routines to DSP-specific assembly routines.



CIE ENGINEERING INC.



ADSP-2188M Processor Implementation

CIE Engineering compiled and linked the ATC-10B vocoder code to execute on an Analog Devices ADSP-2188M digital signal processor (DSP). The input/output sample code was not used. CIE also conducted simulations to measure the execution performance of the algorithm on the DSP.

Note: The NEXCOM Prototype RIU, developed by CIE for the FAA, uses the ADSP-2188M DSP processor.

Code Compilation (ADSP-2188M)

ADSP-2188M DESCRIPTION

The ADSP-2188M is a 16-bit fixed point DSP with arithmetic logic unit, multiplier/accumulator unit (40-bit), and barrel shifter unit. It executes instructions at a rate up to 75 million instructions/second (MIPS). The DSP maintains separate program memory (24-bit wide) and data memory (16-bit) spaces. The ADSP-2188M includes 48K words of on-board program memory and 56K words of on-board data memory (organized into 8K pages).

VISUAL DSP DESCRIPTION

Visual DSP 2.0 (VDSP) is an integrated development environment (IDE) tool used to build and manage C/assembly based programs for the ADSP-2188M. VDSP also includes a simulator that can be used to measure/determine code execution time.

Initially, CIE Engineering compiled (but did not link) the ATC-10B Fixed Point C Source for the ADSP-2188M. The total code size for the vocoder was over 60 kwords of program memory. This is 12 kwords more than what is available on-board. In addition, this total does not include the required input/output code or program core code. The core code includes interrupt handlers, overlay handlers (used for paged memory management) and serial port drivers (for maintenance/debug port).

Due to the large program size, CIE Engineering separated the code into two pieces to run a dual DSP processor configuration. The encoder was placed in one DSP and the decoder was placed in the other DSP. This approach was minimally successful in that the ATC-10B vocoder code was successfully linked. However, the remaining memory was not sufficient to support the vocoder input/output or core functions.

The ADSP-2188M Compiler reported a few warnings and errors. All warnings and errors were extremely minor (e.g., missing new line at end of file) except for errors resulting from the use of the 'long long' type variables. The ADSP-2188M Compiler does not support this (or any other) 64-bit data type. Fortunately, the use of this type is limited to one file within the ATC-10B source code. The 64-bit type was used to implement 40-bit DSP math. These functions can be translated into ADSP-2188M assembly to take advantage of the processor's 40-bit multiplier/accumulator (MAC).

Note: For the purposes of obtaining initial memory and execution performance profiles, the problematic sections of code were commented out.



CIE ENGINEERING INC.

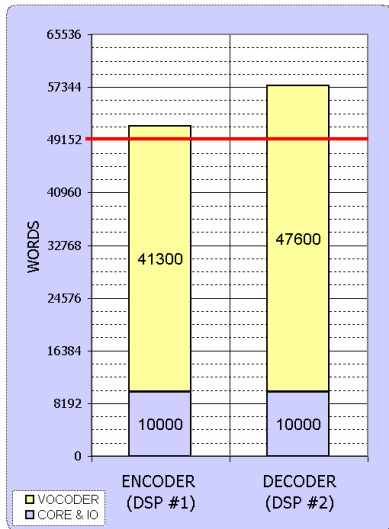


Figure 1. Program Memory Profile (ADSP-2188M). Even a dual processor configuration requires more than the available 48K of on-board memory.

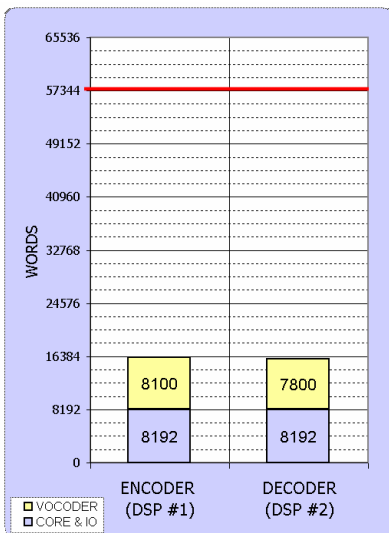


Figure 2. Data Memory Profile (ADSP-2188M). The data memory requirements are minimal. The 56K of on-board data memory is not fully utilized.

Memory Profile (ADSP-2188M)

This section provides detailed ADSP-2188M program and data memory usage requirements for the ATC-10B vocoder. The profile is based on a dual processor configuration. The encoder program is executed on one DSP and the decoder program is executed on another. While the data memory requirements for both programs are minimal, the program memory requirements are relatively large (especially for the decoder).

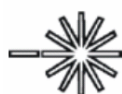
PROGRAM MEMORY

Figure 1 presents the program memory utilization requirements for the encoder and decoder programs. The encoder requires 41.3 kwords. The decoder requires 47.6 kwords.

An estimated 10.0 kwords is required to support voice input/output and core program functions. Voice input/output functions include DSP SPORT bus driver and queue routines. Core program functions include an interrupt handler, queues, RS-232 serial port driver, overlay manager (for memory page management), and a minimal command line processor.

DATA MEMORY

Figure 2 presents the data memory utilization requirements for the encoder and decoder programs. The data memory requirements are minimal when compared to the program memory requirements. The encoder requires 8.1 kwords (or 14%) of data memory. The decoder requires 7.8 kwords (or 14%) words of data memory. The voice input/output and core program functions require an estimated 8.2 kwords of data memory.



CIE ENGINEERING INC.

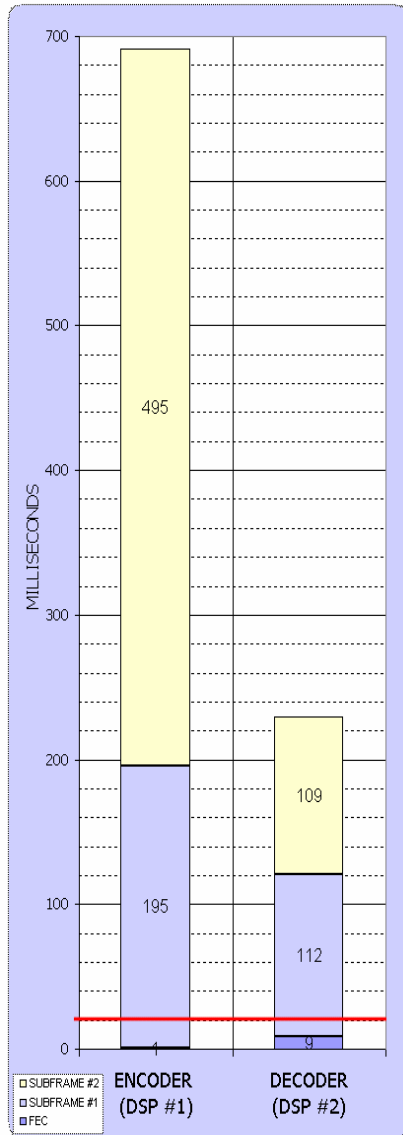


Figure 3. Execution Profile (ADSP-2188M). Both encoder and decoder execution profiles exceed the 20 ms limit by orders of magnitude. The encoder requires a 35:1 improvement and the decoder requires a 12:1 improvement.

Execution Profile (ADSP-2188M)

CIE Engineering simulated the execution time of the encoder and decoder software using Visual DSP.

RESULTS

Figure 3 presents the execution time profiles for the encoder and decoder. With a 13.5 ns instruction period, the encoder required 692 ms to compress 20 ms of linear audio. The decoder required 230 ms to generate 20 ms of linear audio. Clearly, the C source code cannot run in real time on this DSP.

Note: These results do not include execution time for input/output or core program functions.

POTENTIAL OPTIMIZATION

The source code was examined to determine if key DSP-related routines could be optimized to allow the algorithm to run within the time available, i.e. 20 ms. There are identifiable DSP routines, such as FFT and DFT routines, that require large execution times when implemented as C routines.

For example, in the first encoder subframe (195 ms execution time), a single DFT routine executes multiple times and requires an aggregate execution time of 135 ms. If the routine were replaced with an ADSP-2188M assembly routine, the estimated aggregate execution time would be less than 1 ms. Notwithstanding, the remaining functions in this subframe still require 60 ms to execute and the processing time is distributed across many functions. Many, if not all, of these functions would need to be converted to assembly routines.

The optimization issue for the second subframe (495 ms execution time) is much worse than the first subframe.

The code cannot be reasonably optimized to run in the time allotted without eliminating much, if not all, of the C source code.



CIE ENGINEERING INC.



Pentium Processor Implementation

CIE Engineering compiled the sample application (with the vocoder source code) to execute on a standard Pentium computer. Microsoft Visual Studio 6.0 was used to compile and link the code.

Code Compilation

CIE successfully compiled and linked the sample application which includes the ATC-10B vocoder source code.

Execution Profile

A small C timing utility was written to measure the execution time of the encode/decode functions using the ATC-10B sample application. The utility also counted the data samples in the linear voice input/output files and displayed the corresponding 1X playback time.

Figure 4 shows execution performance for seven different computers. The input test vector (linear audio) contained 73.984 seconds of voice sampled at 8 kHz (refer to the bold line in the figure).

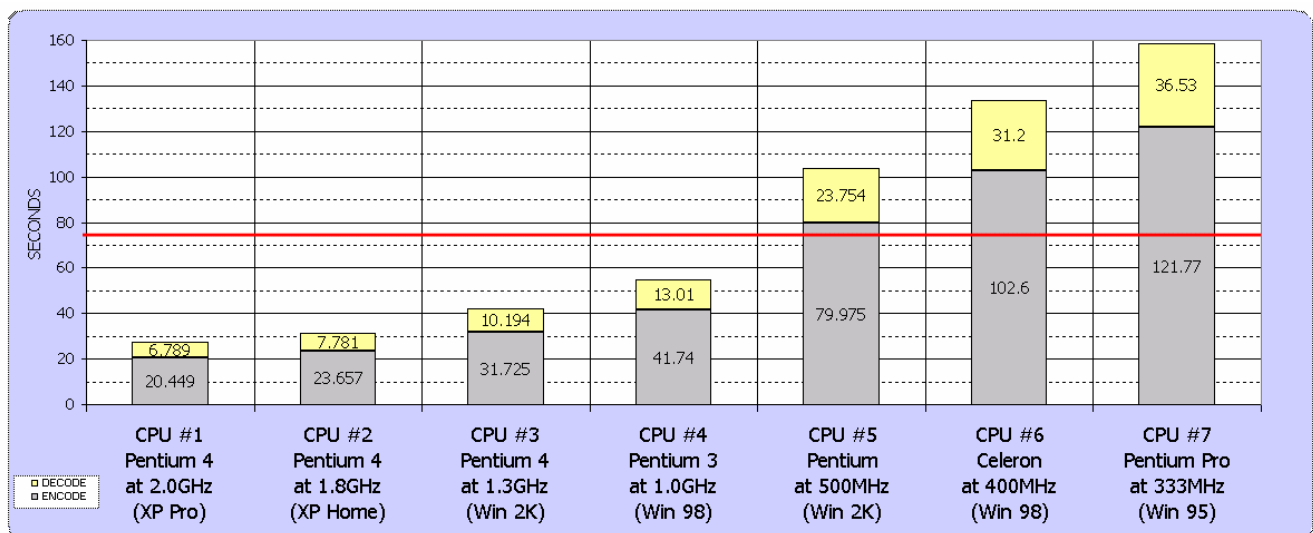
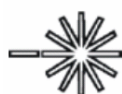


Figure 4. Pentium Machine Execution Profile. The encoder requires roughly 3 times more time to execute than the decoder.



CIE ENGINEERING INC.



The sample application enables the encoder and decoder to be executed separately. The sample application does include file input/output operations. These operations were judged to add an insignificant amount of time to the total execution time. The bit error rate models were not used.

The timing tests were run a few times on each machine. The timing results were fairly consistent, varying by less than 2%. Figure 5 shows that execution time is proportional to processor speed. The error in predicted performance using the linearity was less than 4%. Given these results, a Pentium running at a minimum of 750-800 MHz is required to run the algorithm in real-time.

A few different voice sample files were also tested. Execution times varied only slightly for those files tested. For example, a silence voice file (all zeros for 74 seconds) was applied to CPU #3. The execution time for encoder and decoder was 40.728 seconds versus the 41.919 seconds for active voice content (a difference of less than 3%).

Note: Although no other primary applications were executing on any of the machines, no special efforts were made to stop the execution of operating system services or background applications automatically started on power up.

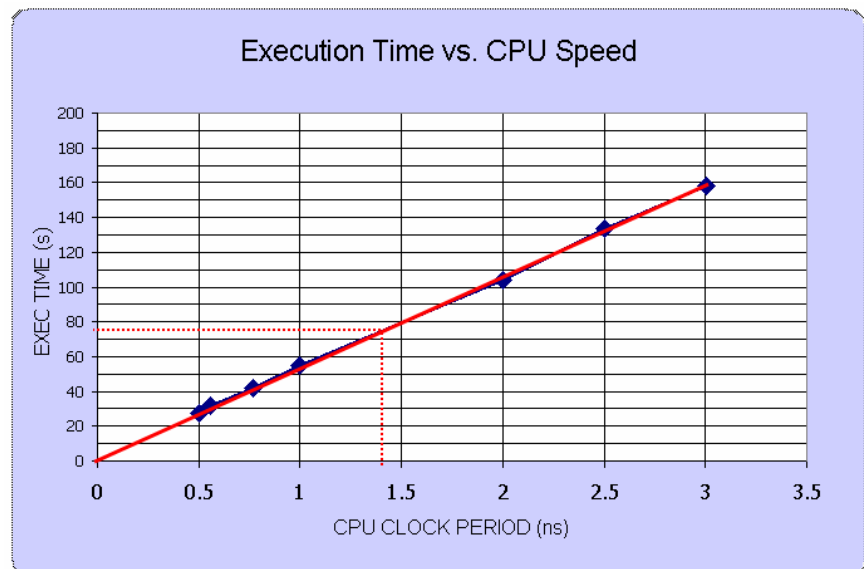


Figure 5. Execution Time vs CPU Speed. The execution time is proportional to CPU speed. A 750-800 MHz machine is required to run the algorithm in real time.



CIE ENGINEERING INC.



VC-20 Compatibility

CIE conducted bit-stream compatibility testing between the 16-bit fixed point ATC-10B/C Source Code and the floating point VC-20 hardware.

The CIE-designed Vocoder Development Support Tool (VDST) platform was used to provide VC-20 functionality. The VDST includes an integral VC-20 board and supports compressed voice capture/playback capabilities over a host serial port.

Test Setup

Figure 6 presents the test setup used to verify VC-20 compatibility. The test objective was to transfer voice in both directions (VC-20→C-SOURCE and C-SOURCE→VC-20).

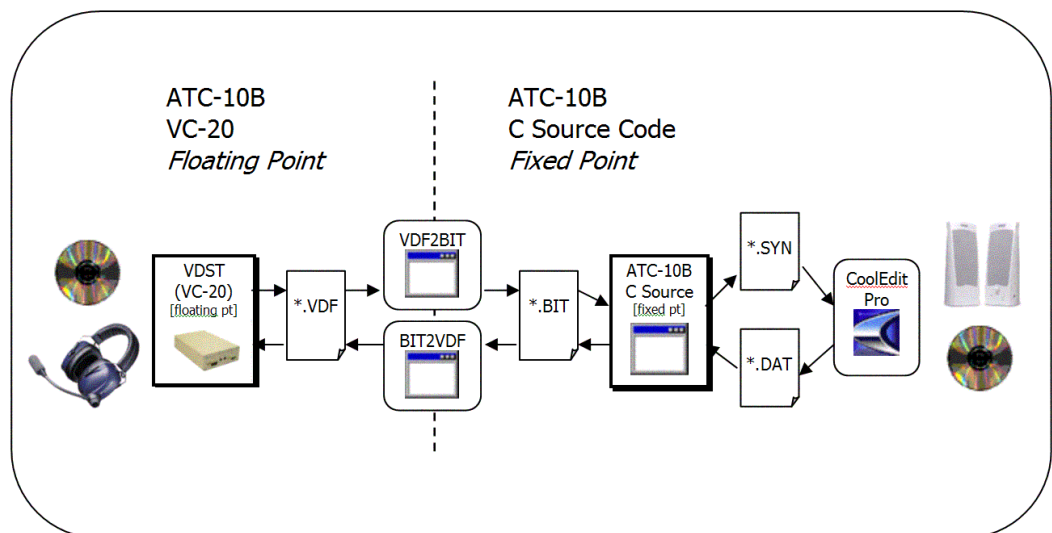


Figure 6. VC20/C Source Code Compatibility Test. The CIE VDST and a personal computer were used to test compatibility between the VC-20 and the Fixed Point C Source Code. Translation software was written to convert VDST compressed voice files (*.VDF) to DVS1 compressed voice files (*.BIT).

TEST #1: VC-20® C-SOURCE

For the first test, an air traffic voice sample track was played (using a CD player) through the VDST. The VDST was configured to send the compressed voice stream (generated by the internal VC20) out to the VDST host serial port. HyperTerminal (a terminal emulation software package) was used to capture the ASCII bit stream to a file. The VDST user manual describes the compressed voice ASCII format.



CIE ENGINEERING INC.



A file format translation utility, VDF2BIT, was written to convert the VDST compressed voice file format (*.VDF) into the DVSI compressed voice file format (*.BIT). Hard decision bit encoding was used. Note: Although the VC-20 software can internally support soft decision, the VC-20 hardware interface specification limits the compressed voice to 1 soft-bit per channel-bit, i.e. hard decision.

The ATC-10B/C Source Code sample application (which provides a file I/O interface for the software vocoder) was used to decode the compressed voice file. Cool Edit Pro 2.0 (a multi-track sound recording and editing software program) was used to read the raw linear voice data files and to play the decoded voice on the computer speakers.

TEST #2: C-SOURCE® VC-20

For the second test, the air traffic voice sample track was converted to *.DAT format using Cool Edit. This file was encoded by the ATC-10B/C Source Code sample application.

A file format translation utility, BIT2VDF, was written to convert the DVSI compressed voice file format (*.BIT) to the VDST compressed voice file format (*.VDF).

HyperTerminal was used to play the file out the computer serial port to the VDST host serial port. The VDST was configured to route the compressed voice through the internal VC20 and to play the decoded audio out the analog line interface. The audio was monitored using a computer headset.

RESULTS: NIBBLE BIT ORDER INCOMPATIBILITY

The initial tests were completely unsuccessful. After some research and trial bit stream rearranging, CIE discovered the bit streams needed to flip the nibble bit order for the voice paths to work properly.

The VDST is compliant with the DO-224A standard which states:

Section 3.3.5.2.3: Vocoder Frame Bit Ordering. The 96 bits of the vocoder frame shall be ordered as follows: For the DVSI VC-20 ATC-10B implementation, the vocoder encoder frame shall consist of the concatenation of 4bit words output from the vocoder board each transmitted msb first. The VC-20 ATC-10B vocoder decoder frame shall consist of the concatenation of twenty-four 4-bit words input to the vocoder board each received msb first. For the VC-20 ATC-10B implementation, the 4-bit control bytes which precede each vocoder frame shall not be considered part of the 96-bit vocoder frame.

The C Source Code bit order appears to be consistent with the Department of Defense (DOD) Consortium Text Fixture (CTF) compatible units submitted by DVSI for vocoder algorithm evaluation.



CIE ENGINEERING INC.



Figure 7 presents the differences between VC-20 bit order and the C Source Code bit order.

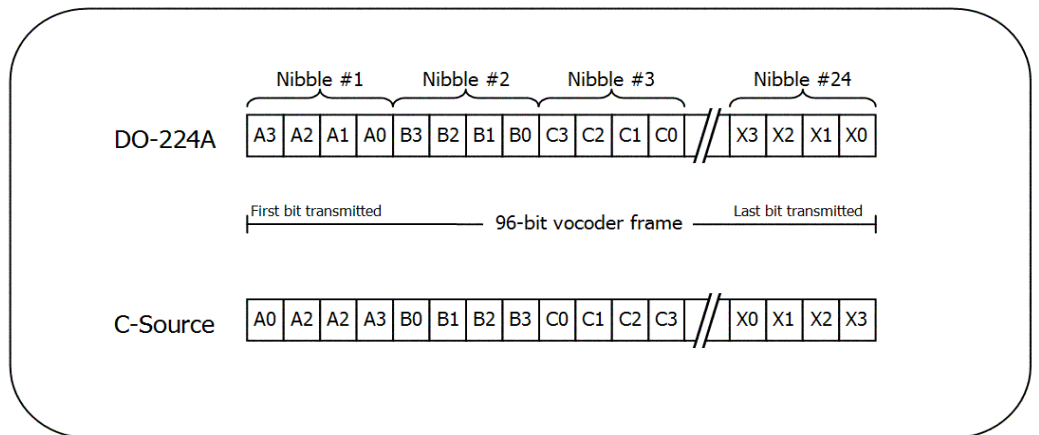


Figure 7. Bit Order Incompatibility. The DO-224A standard specifies MSB first transmission of nibbles. While there is not a nibble interface for the C-Source code, the figure depicts the relative bit order generated by the code. The nibble bit order is flipped.

The file format translation utilities, VDF2BIT and BIT2VDF, were modified to compensate for the bit order problem. Once this was accomplished, both voice paths worked properly. Given that DO-224A is the standard, the source code interface will need to be modified to comply with DO-224A.



Conclusions

CIE presents the following conclusions based on the testing and evaluation conducted thus far:

- **ATC-10B/C Source Code Structure:** The C source is reasonably documented and well structured.
- **ATC-10B/C Source Code Portability:** The C source code is not directly compatible with the ADSP-2188M Compiler. The code uses 64-bit variable types that are not supported by the processor (though the use of this variable type is limited to one source file). The C source code is directly compatible with the Visual C++ Compiler (for Intel x86 platform).
- **ATC-10B/C Source Code Execution Time:** The C source cannot run in real-time on any reasonably powered embedded processor platform. A Pentium processor running at an estimated speed of 750-800MHz (or better) can run the C-source vocoder in real-time.
- **ATC-10B/C Source Code Optimization:** While the C source is structured for translation to DSP assembly, many if not all of the files would need to be translated (hand-optimized) to run on an embedded processor. This would be an extremely time consuming task.
- **ATC-10B/C Source Code Bit Stream Compatibility:** The C source bit stream order is not directly compatible with the VC-20. After the bit order is rearranged, the C code is interoperable with the VC-20.

CIE plans to continue testing and evaluation of the ATC-10B vocoder source code.



CIE ENGINEERING INC.