Application Report
SPRA734 - February 2001

A Case Study in DSP Systems Integration: The TI Third-Party Vocoder Demonstration Kit

A. Campbell
X. Fu
Digital Signal Processing Solutions

ABSTRACT

The Texas Instruments Third-Party Vocoder Technology Demonstration Kit (TDK) illustrates recent advances in DSP software systems integration. The tool allows users to graphically configure multiple channels of several speech-coding algorithms running on a TMS320C6000™ Digital Signal Processor (DSP). A key differentiating feature, however, is that the application framework embraced an algorithm standard which greatly reduced the overall system assembly time.

This paper presents a case study in DSP systems integration, and aims to solve issues when using algorithms from multiple sources. At its heart lies the TMS320™ DSP Algorithm Standard. A component developed in accordance with this standard adheres to a set of rules designed to ensure that it will inter-operate with algorithms from different vendors in virtually any application.

The end-result is a speech coder framework running on an inexpensive C6211/C6711 DSP Starter Kit (DSK). At present more than 30 Standard-compliant algorithms from four TI third parties have been successfully integrated. A common set of Application Programming Interfaces (API) further enables one vendor's implementation to be re-linked in place of another. No source code modifications are required.

The intended audience is primarily systems integrators keen to using commercial off-the-shelf (COTS) algorithms. Technical highlights in the reduced TDK development cycle are presented with reference to the enabling algorithm-standard features. Consumers evaluating different third-party solutions also benefit from this work, since real-time algorithm performance metrics are clearly displayed in the Graphical User Interface (GUI).

Contents

TMS320C6000 and TMS320 are trademarks of Texas Instruments.

Trademarks are the property of their respective owners.

List of Figures

1 Introduction

The primary goal of the Texas Instruments Vocoder TDK is to demonstrate the high performance of various speech coding algorithms on the TMS320C6000 DSP platform. The application is capable of processing up to 10 channels concurrently, with each assigned up to three algorithms. For example, five channels of the International Telecommunications Union (ITU) standard G.723.1 algorithm may execute alongside three channels of G.726 and two of G.729. Each may have an encode, decode pairing and/or a Line Echo Cancellor (LEC). Running on an inexpensive C6211/C6711 DSP Starter Kit (DSK) board, high channel densities can be achieved with mainly C-code based algorithm implementations due to the strength of the compiler technology.

However, the Vocoder TDK has recently acquired a second, equally important role. It has been further developed into a tool to highlight recent advances in DSP Systems integration. The new TMS320 DSP Algorithm Standard [1] acts as the catalyst to construct applications consisting of algorithm components from potentially different vendors. Rules for symbol naming conventions, and an interface to uniformly address memory resource requirements, allow multiple vendor algorithms to co-exist in the same application (see Figure 1).

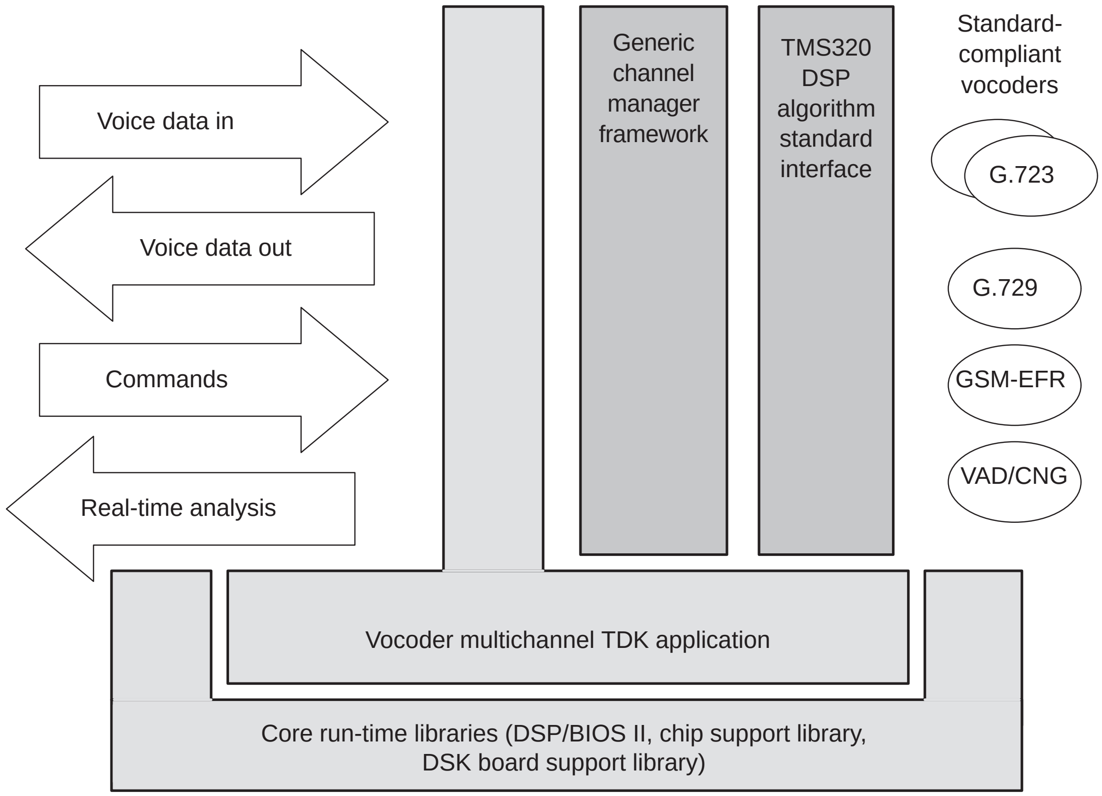

Figure 1. Distinct System Layers Showing Separation of Concerns – Core Run-time Functions, Compliant Algorithms, and an Application Calling Upon a Framework

A common set of agreed Application Programming Interfaces (APIs) enables customers to swap out and compare one third party's vocoders for another. Zero source code modifications are necessary; merely a re-link to target the different standard-compliant libraries. Four third parties and over 30 algorithms have now been successfully integrated into the TDK, a task made relatively straightforward by the existence of an algorithm standard.

This paper describes the implementation of the Vocoder TDK, and focuses on technical aids available to integrate third-party algorithms. Mention is also made of the generic Channel Manager framework responsible for the creation, deletion, execution, and configuration of all algorithm instances.

2 Review of the TMS320 DSP Algorithm Standard

The TI TMS320 DSP Algorithm Standard was officially launched in October 1999. As part of the eXpressDSP initiative, its primary goal is to enable DSP system integrators to better construct applications consisting of algorithm components from potentially different vendors [2].

A fundamental requirement is a convention for how algorithms request memory. All standard-compliant algorithms must implement an abstract interface named IALG to define their memory resource requirements. It is intended to give system integrators more freedom in the placement of data buffers. Some may be critical and should be placed on-chip, while other less-used buffers might be deferred to external memory. This uniform memory management scheme enables multiple algorithms to co-exist without contention in a single application [3].

Rules are also defined to formalize commonsense guidelines, such as the requirement for vendors to produce re-entrant code (i.e., the same algorithm may be used concurrently by two or more threads). Furthermore, the inclusion of naming conventions avoids clashes between software from different vendors. The standard ensures uniqueness of external identifiers by requiring module and vendor name prefixes.

An extract of a third-party TDK symbol table (shown in Figure 2) clearly demonstrates usage of the naming convention.

```
80094990    _G723_SIGSOFT_tabgain170
80094ae8    _G723_SIGSOFT_tabgain85
000000a0    _G726D_FRAME_LEN
ffffffff    _G726D_IG726D
000000a0    _G726E_FRAME_LEN
ffffffff    _G726E_IG726E
00000020    _G728DEC_FRAME_LEN
80096508    _G728DEC_IG728DEC
800c5c78    _G728DEC_SIGSOFT_Config
800c56c0    _G728DEC_SIGSOFT_Control
800c57a8    _G728DEC_SIGSOFT_Decod
60096508    _G728DEC_SIGSOFT_IALG
80096508    _G728DEC_SIGSOFT_IG728DEC
800c5ca0    _G728DEC_SIGSOFT_Process
800c5ae0    _G728DEC_SIGSOFT_Reset
800ee400    _G728DEC_SIGSOFT_alloc
800c6220    _G728DEC_SIGSOFT_block32
800c603c    _G728DEC_SIGSOFT_block33
800c62e4    _G728DEC_SIGSOFT_block34
800c63f4    _G728DEC_SIGSOFT_block35
800c6588    _G728DEC_SIGSOFT_block50dec
800c673c    _G728DEC_SIGSOFT_block71_72
800c6d28    _G728DEC_SIGSOFT_block73_74_75
800c6e74    _G728DEC_SIGSOFT_block76_77
800c6f10    _G728DEC_SIGSOFT_block81
800c721c    _G728DEC_SIGSOFT_block82
800c7eac    _G728DEC_SIGSOFT_block83
800c8220    _G728DEC_SIGSOFT_block84
800c82d4   _G728DEC_SIGSOFT_bloock85
800ee8e0    _G728DEC_SIGSOFT_free
800ed900    _G728DEC_SIGSOFT_initObj
800c8434    _G728DEC_SIGSOFT_loop85
000000a0    _G728ENC_FRAME_LEN
800964dc    _G728ENC_IG728ENC
800c00e8    _G728ENC_SIGSOFT_Coder
```

Figure 2. Symbol-Map File Extract of Vocoder TDK Built With Several Signals and Software Algorithms

The tag '_SIGSOFT_' refers to the third-party Signals and Software based in the United Kingdom. All vendor IDs are stored in a TI database. No other third party can use this tag, thus ensuring uniqueness. The extract implies the existence of at least three algorithms: G.723, G.726, and G.728, all of which are speech coders specified by the ITU. In this instance, only libraries for G.723 and G.728 have been linked in to the final executable. The remaining symbols for the G.726 algorithm pertain to the top-level application handling all vendor G.726 implementations.

It can be seen that the third party has chosen to separate encode (G728ENC) and decode (G728DEC) modules for the G.728 algorithm. This is primarily a business decision. The customer may require 1 side of the coder in isolation.

Finally, the signature of the IALG memory interface is exposed by the symbol G728DEC_SIGSOFT_IALG. In accordance with the rules, it has been further extended to define the decode interface in G728DEC_SIGSOFT_IG728DEC.

3 Third-Party Multichannel Vocoder TDK Design

The Vocoder TDK consists of 2 distinctive parts, as shown in Figure 3:

- The target-side DSP application running on a C6211 or C6711 DSK
- A host-side application with a Graphical User Interface (GUI)

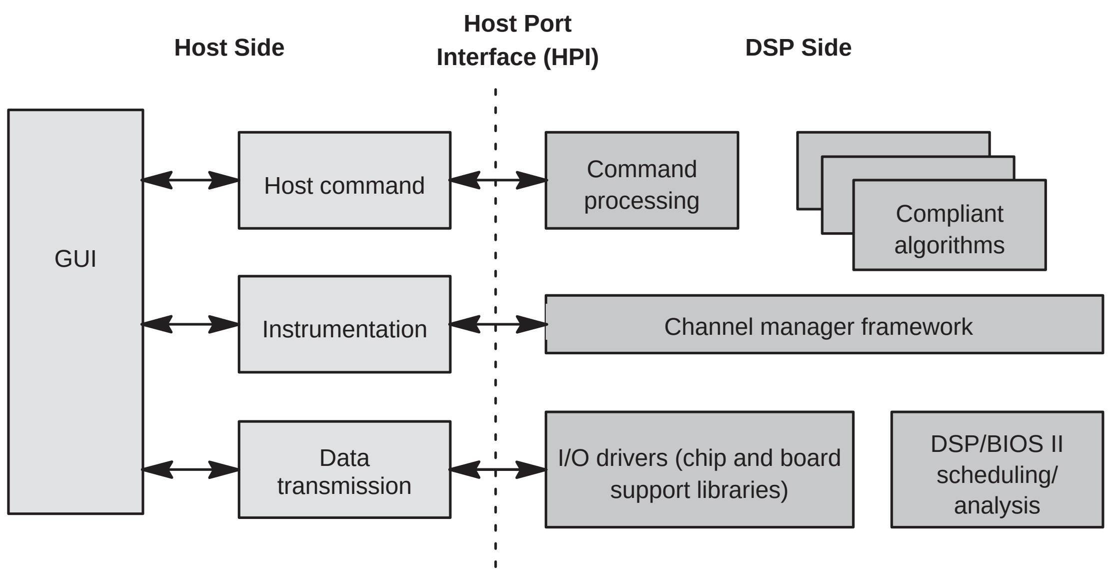

Figure 3. System Diagram of the Vocoder TDK

The host application sends commands to the DSP for run-time channel and I/O configuration [4]. It is also responsible for real-time data transmission between the host and DSP, when a user chooses to process a static voice data file in preference to the latest buffer from the DSP's input serial ports. The host is able to determine which vocoders are linked into the target executable via a handshaking mechanism over the Host Port Interface (HPI). This allows for dynamic configuration. The user can drag-and-drop multiple channels of a vocoder, or execute different algorithms.

The DSP application consists of up to 10 concurrently active processing channels [5]. Each active channel has at least one input I/O stream, one output stream, and up to three different algorithms. Typically, a vocoder will accept a single data buffer, process it, and return a modified output buffer. However, Line Echo Cancellation is also supported. This algorithm feeds on two input streams, termed near- and far-end, then performs calculations to reduce the near-end echo on the single-output send path. For each channel, the underlying framework gets input data from the assigned input port, uses assigned algorithms to process the data, and sends the outputs to the assigned output port.

The DSP application is built upon the DSP/BIOS II kernel, Chip Support Library (CSL), Board Support Library (BSL), and a Channel Manager framework (CM). Scheduling and real-time analysis is performed by DSP/BIOS II, allowing events to be prioritized and trace data to be received. The CSL and BSL provide all chip-level and board-level peripheral accesses, including DMA, serial port, audio codec, and HPI. The Channel Manager will be discussed under the Integration Successes topic.

4 Algorithm Components

The beauty of the TDK initiative is that a third party need only contribute one or more speech coders to participate in the program. The Channel Manager ignores a channel with no algorithms assigned to it. Third parties can build up their vocoder software portfolio on the TMS320C6000 DSP using the Vocoder TDK to test for basic re-entrancy, correct IALG implementation, and interoperability. Customers benefit enormously from having a simple tool to benchmark multiple vendor telecommunications algorithms.

The following algorithm components are currently supported in the Multichannel Vocoder TDK. All modules must be compliant with the TMS320 DSP Algorithm Standard.

- ITU-T recommendation G.723.1, dual-rate speech coding algorithm operating at 5.3/6.3kbps
- ITU-T recommendations G.729/A/AB, a Conjugate Structure Algebraic Code-Excited Linear Prediction (CS-ACELP) algorithm. Annexes A and AB process a reduced complexity version, and silence descriptors respectively.
- ITU-T recommendation G.726, an Adaptive Differential Pulse Code Modulation (ADPCM) speech coding algorithm operating at 16/24/32/40 kbps
- ITU-T recommendation G.728, Low-Delay CELP at 16kbps
- ITU-T recommendation G.711, Pulse-Code modulation operating at 64/56/48kbps
- IS-127 Enhanced Variable Rate Speech Coder (EVRC), an international standard for “Service Option 3 for Wideband Spread Spectrum Digital Systems”
- Full-Rate (FR) and Enhanced FR GSM, international standards for wireless communications defined by the European Telecommunications Standards Institute (ETSI)
- Line Echo Cancellation, compliant with ITU-T recommendation G.165
- Voice Activity Detection (VAD), and Comfort Noise Generation (CNG)

5 System Integration Success Stories

With a total of 24 algorithms supported, how is it possible to create a generic framework capable of instantiating and executing the various components? If individual algorithm interface methodologies need to be considered, then the integration effort may stretch over many, many man-months. Debugging and system robustness concerns arise in such cases, since the system integrator cannot re-use code written to run one vocoder, on the next [6]. Worse still, the combination of algorithms may fail. One algorithm may assume the use of certain resources that are then "stolen" by another vendor's algorithm in real time.

Fortunately, the TMS320 DSP Algorithm Standard addresses these concerns. The integration successes are described below.

5.1 Algorithm Initialization

The standard's IALG memory interface defines various types and constants with the key element being a global structure of type IALG_Fxns. It contains a set of pointers commonly denoted as the v-table. Some of the functions are optional, while algAlloc(), algInit(), and algFree() must always be implemented.

The algAlloc() function returns a table of memory records that describe the size, alignment, type and memory space of all buffers required by an algorithm. Based on this information, the application allocates the requested memory before calling the algInit() function. algInit() performs all the initialization necessary to complete the run-time creation of an algorithm's instance. After a successful return from algInit(), the object is ready to process data.

Significantly, all standard-compliant algorithms can be instantiated in this manner. Hence, a uniform method of initializing all the vocoders is feasible.

The Vocoder TDK defines a function ALGO_New() to perform precisely the sequence specified. Each algorithm is started with a call to ALGO_New(), irrespective of the vocoder type, or third party software vendor.

An additional benefit is the reduction in program code. The use of a single, common API in place of 24 separate initialization functions potentially saves a great deal of program memory.

5.2 The Channel Manager Framework

The job of the framework writer can be different to that of the system integrator. Complex DSP applications can often be broken down into two layers. The first is application-specific managing system initialization, feeding of data to the various tasks, and controlling peripherals. The second layer, however, can in fact be independent of specific applications, algorithms, and essentially the DSP hardware. It acts as the framework: the glue that holds together the application, algorithms, and the underlying infrastructure or operating system.

The Vocoder TDK Channel Manager framework provides APIs to:

- Open and close channels
- Configure channels
- Execute channels

- Assign algorithms to channels
- Register algorithms with the library
- Get and set algorithm parameters in a channel

Evidence that a framework is reusable can be found in the new Imaging TDK [7]. It uses the same channel manager as the Vocoder TDK. As a result, the overall system assembly time for new TDKs is greatly reduced.

Again, the core of the framework lies in the algorithm standard. A random mixture of API formats across the various vocoders would have prevented a uniform approach to executing algorithms. Indeed, the Channel Manager is capable of executing any standard-compliant algorithm with a processing method of the form:

```
Void (*process)(IALG_Handle handle, Void *inData, Void *outData);
```

All of the vocoders have such an API format, with the exception of Voice Activity Detect and Comfort Noise Generation. It was a relatively straightforward task, however, to ‘wrap’ VAD and CNG with a layer matching the framework format. Neither the framework writer nor the algorithm vendor needed to be involved in this process.

5.3 Intechangeable Algorithms

One of the goals of the DSP Algorithm Standard is defined as:

- Integration of algorithms does not require recompilation of the client application; reconfiguration and re-linking may be required however.

This is a very powerful feature. An application can directly swap out an algorithm for a different vendor implementation, perhaps with improved MIPS and/or memory footprint.

For example, a system integrator could obtain a baseline C-code version of the ETSI GSM-EFR algorithm, adapt it to the algorithm standard, and execute it within the application framework. However, the lack of optimization for the specific DSP platform may prevent channel density goals being reached.

The ability to slot in a different vendor’s implementation of EFR opens up the make or buy debate. The system integrator can spend many man-months trying to hand-optimize the algorithm or buy compatible code from a third party. Availability of highly optimized third party software, and time-to-market pressures make the latter option very appealing.

Swapping one vendor implementation for another is a trivial task in the TDK. An example is given in Figure 4 for the G.723 algorithm available from TI and several third parties on the C6000 platform.

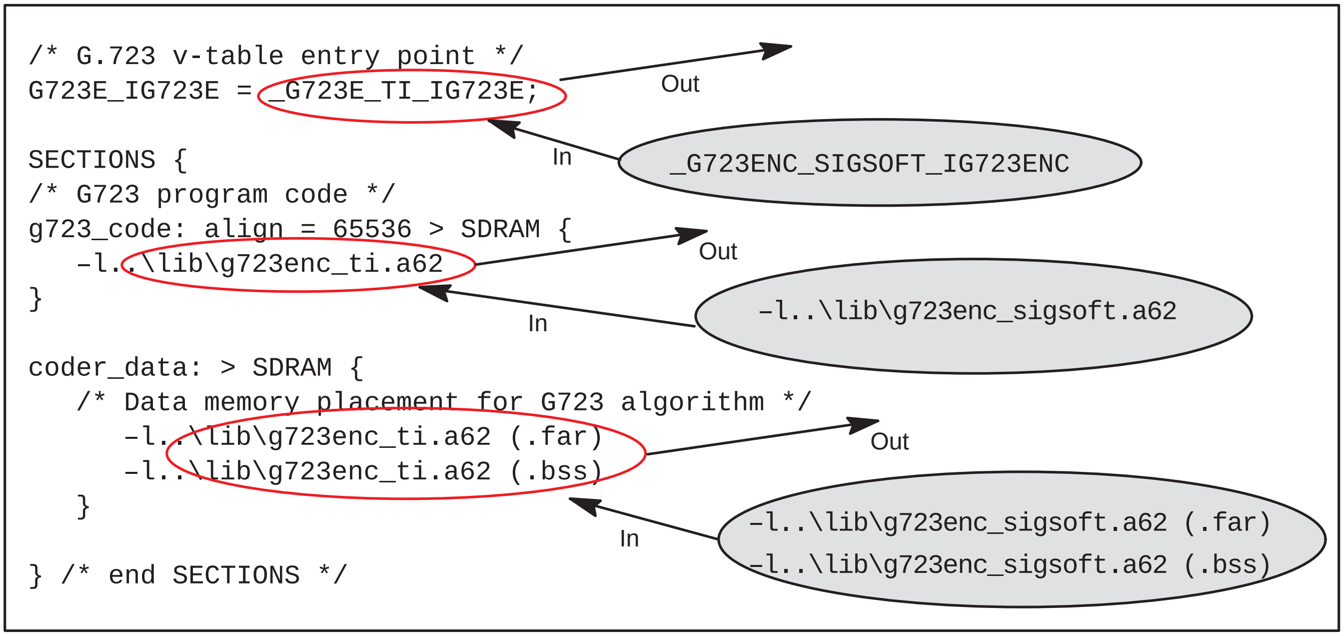

Figure 4. Linker Command File Swapping of One G.723 Implementation for Another

Customers now have a tool for benchmarking different speech coders. No source code changes are required to reconfigure the set of algorithms.

The availability of a suggested set of vocoder abstract interfaces within the Standard, enabled third parties to write to a common standard. When a customer swaps out one G.723 implementation for another, the bits change underneath but the calling convention does not.

Relatively new algorithms in other domains such as fingerprint recognition may also make use of this inter-changeability feature. One third party can define a new abstract interface which others may follow. Alternatively, if a vendor chooses to extend its features, only minor application source code changes are required to incorporate modifications.

6 Results

A fully featured multichannel, multi-algorithm speech coding application has been developed on the C6211/C6711 DSP Starter Kit. Code reuse was the major theme; leverage of the TMS320 DSP Algorithm Standard, DSP/BIOS II kernel, CSL, BSL, and Channel Manager framework vastly reduced system integration time. The Graphical User Interface is shown in Figure 5 for an example vocoder configuration.

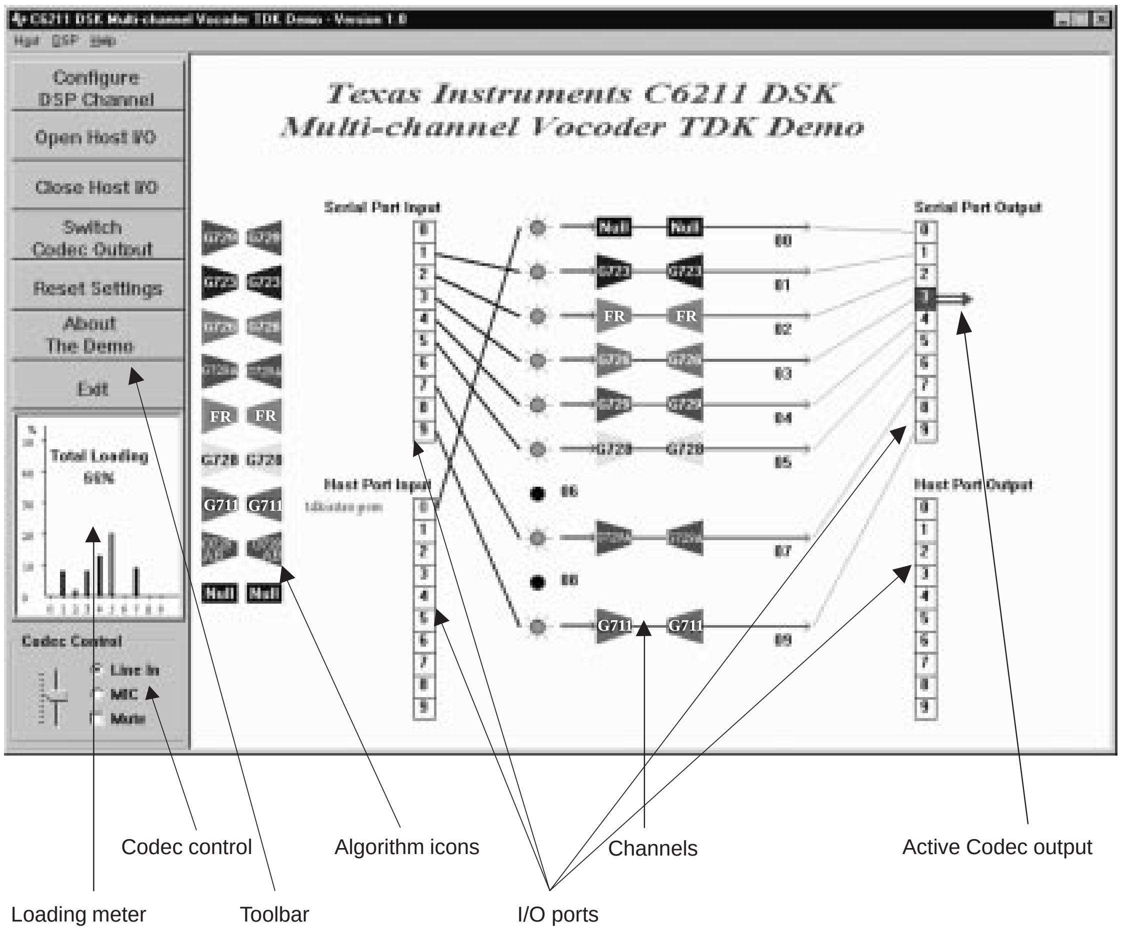

Figure 5. Main Graphical User Interface to Third-Party Vocoder TDK

7 Conclusions

The TI Third-Party Vocoder TDK is a powerful tool for demonstrating a wide range of speech coding algorithms on the TMS320C6000 DSP. Clear benefits arise for both producers and consumers participating in the TDK initiative. Algorithm vendors producing vocoder software have a vehicle to demonstrate their solutions to the mass market as well as a facility to check the basic re-entrancy, interoperability, and voice quality of their components. Telecommunications consumers benefit from greater choice and the ability to benchmark algorithms.

Central to the Vocoder TDK case study was the impact of the TMS320 DSP Algorithm Standard. As the system integrator, commonality in algorithm initialization and API formats was fully exploited. A generic Channel Manager framework was written to handle creation, deletion, configuration, and execution of Standard-Compliant components. Proof of a reduction in overall system assembly time can be found in the new Imaging TDK, which uses the same framework to run compliant imaging algorithms.

8 References

1. *TMS320 DSP Algorithm Standard Rules and Guidelines* (SPRU352), September 2000.
2. "TI Algorithm Standard Slashes DSP Product Development Time", A. Campbell, J. Dubin, *Global Semiconductor*, 2000.
3. *Achieving Zero Overhead With the TMS320 DSP Algorithm Standard IALG Interface* (SPRA716), November, 2000.
4. *TMS320C6000 Multichannel Vocoder TDK Host Side Design* (SPRA558), February 2000.
5. *TMS320C6000 Multichannel Vocoder (TDK) Target Side Design* (SPRA560), February 2000.
6. *The TMS320 Algorithm Standard* (SPRA581), January 2000.
7. *C6000 Imaging TDK Application Framework*, to be published.

IMPORTANT NOTICE

Texas Instruments and its subsidiaries (TI) reserve the right to make changes to their products or to discontinue any product or service without notice, and advise customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgment, including those pertaining to warranty, patent infringement, and limitation of liability.

TI warrants performance of its products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Customers are responsible for their applications using TI components.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards must be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance or customer product design. TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such products or services might be or are used. TI's publication of information regarding any third party's products or services does not constitute TI's approval, license, warranty or endorsement thereof.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations and notices. Representation or reproduction of this information with alteration voids all warranties provided for an associated TI product or service, is an unfair and deceptive business practice, and TI is not responsible nor liable for any such use.

Resale of TI's products or services with *statements different from or beyond the parameters* stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service, is an unfair and deceptive business practice, and TI is not responsible nor liable for any such use.

Also see: Standard Terms and Conditions of Sale for Semiconductor Products. www.ti.com/sc/docs/stdterms.htm

Mailing Address:

Texas Instruments
Post Office Box 655303
Dallas, Texas 75265