# *Theory and Implementation of the Digital Cellular Standard Voice Coder: VSELP on the TMS320C5x*

## *Application Report*

***Jason Victor Macres***
***DSP Software Engineering, Incorporated***

SPRA136
October 1994



Printed on Recycled Paper

## IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software 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.

Certain applications using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

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

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI 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 semiconductor products or services might be or are used.

## Introduction

TIA subcommittee TR45.3 has adopted vector sum excited linear prediction (VSELP) as the voice coding standard for U.S. digital cellular communications. Motorola was responsible for the design and development of the VSELP algorithm. Additionally, Motorola has kept implementation details of the VSELP algorithm proprietary. This paper explains an interoperable VSELP alternative algorithm and the implementation of this algorithm on a TMS320C5x digital signal processor. The interoperable algorithm is developed using reference [1] as a guideline.

The VSELP algorithm is a type of code excited linear predictive coding (CELP) algorithm that has been adopted as the standard for digital cellular communications. The VSELP vocoder encodes speech at a bit rate of 7950 bits/second. An additional 5050 bits/second are utilized for error protection and synchronization, bringing the total bit rate to 13,000 bits/second. This paper describes only the voice coding portion of the vocoder. A brief overview of the VSELP algorithm is presented for background.

## Overview of VSELP

Structurally, the VSELP algorithm closely resembles the CELP algorithm. The difference lies in the form and structure of the code books. Whereas CELP uses a stochastically overlapped code book (each entry shares all but two samples with its neighboring entries), VSELP utilizes two sets of basis vectors to generate the space of candidate vectors. Thus, the stochastic code book search of CELP corresponds to two code book searches in VSELP. There are seven basis vectors for each search. Each basis vector contains 40 elements. The selection of the basis vectors is fundamental to deriving fast code book search procedures. The basis vectors chosen provide for *fast* orthogonalization of the entire space. By orthogonalizing each of the seven vectors with a vector V, the entire 128 ($2^7$) space, defined by the seven basis vectors, is also orthogonalized.

An open-loop LPC analysis is performed on a frame of speech to derive a set of LPC filter coefficients. These coefficients are bandwidth expanded for use in perceptual error weighting filters, H(z) and W(z), where $H(z) = 1/A(z)$ and $W(z) = A(z)/A(z/\gamma)$. The input frame of speech is filtered through the filter W(z) to obtain a perceptually weighted frame of speech. The analysis by synthesis proceeds with three code books (unlike CELP, which proceeds with two). First, the adaptive code book is searched and the resulting best entry and gain are found. This entry multiplied by its gain factor is orthogonalized with the first set of seven basis vectors. Thus, the second code book search can be performed independently of the first code book search. The new set of basis vectors is used form the code book for the second search. The best entry and gain are found for this code book and orthogonalized with the second set of basis vectors. Finally, the third code book search is performed. The gains of each of the three code book searches are jointly quantized and transmitted with the three code book indices to the receiver.

The basic blocks in the VSELP coder are:

- Tenth-order LPC analysis (spectrum predictor)
- Long term (pitch) predictor
- Adaptive (pitch) code book search
- First basis vector code book search
- Second basis vector code book search
- Vector quantization of the code book gains

The primary VSELP parameters are outlined in Table 1.

**Table 1. Primary VSELP Parameters**

| Symbol | Parameter | Value |
|---|---|---|
| SR | Sampling rate | 8 kHz |
| $N_F$ | Samples per frame | 160 |
| $N_{SF}$ | Samples per subframe | 40 |
| $N_P$ | LPC filter order | 10 |
| $M_1$ | No. basis vectors (1) | 7 |
| $M_2$ | No. basis vectors (2) | 7 |
| BWEXP | Bandwidth expansion | 0.8 |
| LTFORD | Long term filter order | 1 |

The VSELP algorithm has been developed from references [1] and [2]. These references contain information pertaining to the high-level description of the algorithm and provide no actual implemented software (high-level or assembly).

### Bit Allocations

Table 2 shows the bit allocation for the VSELP frame. The frame energy (R0) and reflection coefficients (LPC1–LPC10) are sent once per frame, while the pitch lag (LAG1–LAG4), code book indices (CODE1_1–CODE1_4, CODE2_1–CODE2_4), and gain indices (GSP0_1–GSP0_4) are sent four times per frame.

The total number of bits per 20-millisecond speech frame is 159, yielding a voice coder bit rate of 7950.

**Table 2. VSELP Frame Bit Allocation**

| Parameter | Bits | Description |
|---|---|---|
| R0 | 5 | Frame energy |
| LPC1 | 6 | 1st reflection coefficient |
| LPC2 | 5 | 2nd reflection coefficient |
| LPC3 | 5 | 3rd reflection coefficient |
| LPC4 | 4 | 4th reflection coefficient |
| LPC5 | 4 | 5th reflection coefficient |
| LPC6 | 3 | 6th reflection coefficient |
| LPC7 | 3 | 7th reflection coefficient |
| LPC8 | 3 | 8th reflection coefficient |
| LPC9 | 3 | 9th reflection coefficient |
| LPC10 | 2 | 10th reflection coefficient |
| LAG1 | 7 | Lag, SF 1 |
| LAG2 | 7 | Lag, SF 2 |
| LAG3 | 7 | Lag, SF 3 |
| LAG4 | 7 | Lag, SF 4 |
| CODE1_1 | 7 | 1st CB index, SF 1 |
| CODE1_2 | 7 | 1st CB index, SF 2 |
| CODE1_3 | 7 | 1st CB index, SF 3 |
| CODE1_4 | 7 | 1st CB index, SF 4 |
| CODE2_1 | 7 | 2nd CB index, SF 1 |
| CODE2_2 | 7 | 2nd CB index, SF 2 |
| CODE2_3 | 7 | 2nd CB index, SF 3 |
| CODE2_4 | 7 | 2nd CB index, SF 4 |
| GSP0_1 | 8 | Gain index, SF 1 |
| GSP0_2 | 8 | Gain index, SF 2 |
| GSP0_3 | 8 | Gain index, SF 3 |
| GSP0_4 | 8 | Gain index, SF 4 |

### Perceptual Weighting

Perceptual weighting of the input speech signal (or the error signal) improves the performance of the coder. The high-energy formant regions of the speech spectrum mask noise better than lower energy portions of the spectrum. The error signal generated by each synthesizer pass is weighted appropriately to capitalize on this perceptual effect. The filter amplifies the error signal spectrum in nonformant regions of the speech spectrum and attenuates the error signal spectrum in formant regions. Thus, an error signal whose spectral energy is concentrated in formant regions of the speech is considered better than one whose spectral energy is not located under formants.

### Open-Loop LPC Analysis

Each incoming speech frame is processed through an open-loop LPC analysis to generate the filter coefficients used in the remaining portions of the algorithm. The input speech is first windowed using a Hamming window, then an autocorrelaion is performed and the result is normalized based on the energy of the first coefficient of the autocorrelation.

The autocorrelation coefficients are then windowed for bandwidth expansion and spectral smoothing using a rectangular (in frequency) window. The smoothed autocorrelations are the input to a Leroux-Guegan routine, which transforms the autocorrelation parameters into reflection coefficients. The Leroux-Guegan algorithm was chosen because it is ideal for fixed-point implementation and is very efficient.

A stability check is performed in the Leroux-Guegan algorithm by monitoring the rms value. If the rms falls below 0, the Leroux-Guegan is terminated, and the previous reflection coefficients are used. This instability can occur from ill-conditioned autocorrelation coefficients.

### Interpolation

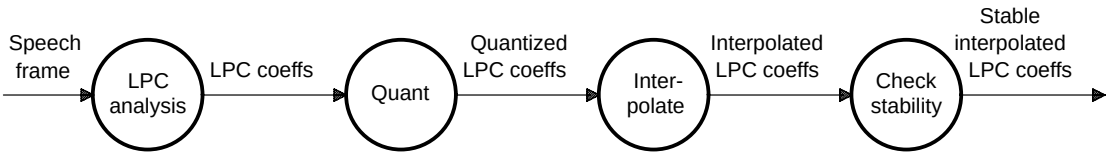
Because the reflection coefficients generated by the LPC analysis represent the spectrum of the speech for one frame centered over the fourth subframe, the coefficients for the remaining subframes are interpolated from the current and the previous frame's coefficients. The direct form-filter coefficients are linearly interpolated. The following table shows the interpolation scheme:

| | |
|---|---|
| $a_i = (0.75)a_{i(previous)}\ (0.25)a_{i(current)}$ | subframe 1 formula |
| $a_i = (0.50)a_{i(previous)}\ (0.50)a_{i(current}$ | subframe 2 formula |
| $a_i = (0.25)a_{i(previous)}\ (0.75)a_{i(current}$ | subframe 3 formula |
| $a_i = a_{i(current)}$ | subframe 4 formula |

Interpolating the direct form coefficients can result in an unstable filter; therefore, the resulting coefficients must be checked for stability. For the first, second, and third subframes, the filter coefficients are converted to reflection coefficients. If any of the resulting reflection coefficients' magnitudes are greater than 1, then the interpolation process has produced an unstable filter. To remedy this instability, the filter coefficients for the subframe are replaced by the uninterpolated filter coefficients. For the first subframe, the previous frame's uninterpolated filter coefficients are used. For the third subframe, the current frame's uninterpolated filter coefficients are used. The second subframe uses the uninterpolated filter coefficients from the frame (previous or current) that has the higher energy. For the case when the energies are equal, subframe 2 uses the uninterpolated filter coefficients from the previous frame.

The following data flow illustrates the procedure for quantization and interpolation of the LPC filter coefficients.

**Figure 1. LPC Filter Coefficient Quantization and Interpolation**




## Long-Term Predictor

The long-term filtering operation (adaptive code book search) for VSELP is similar to the general CELP long-term filtering operation. The long-term filter is given by:

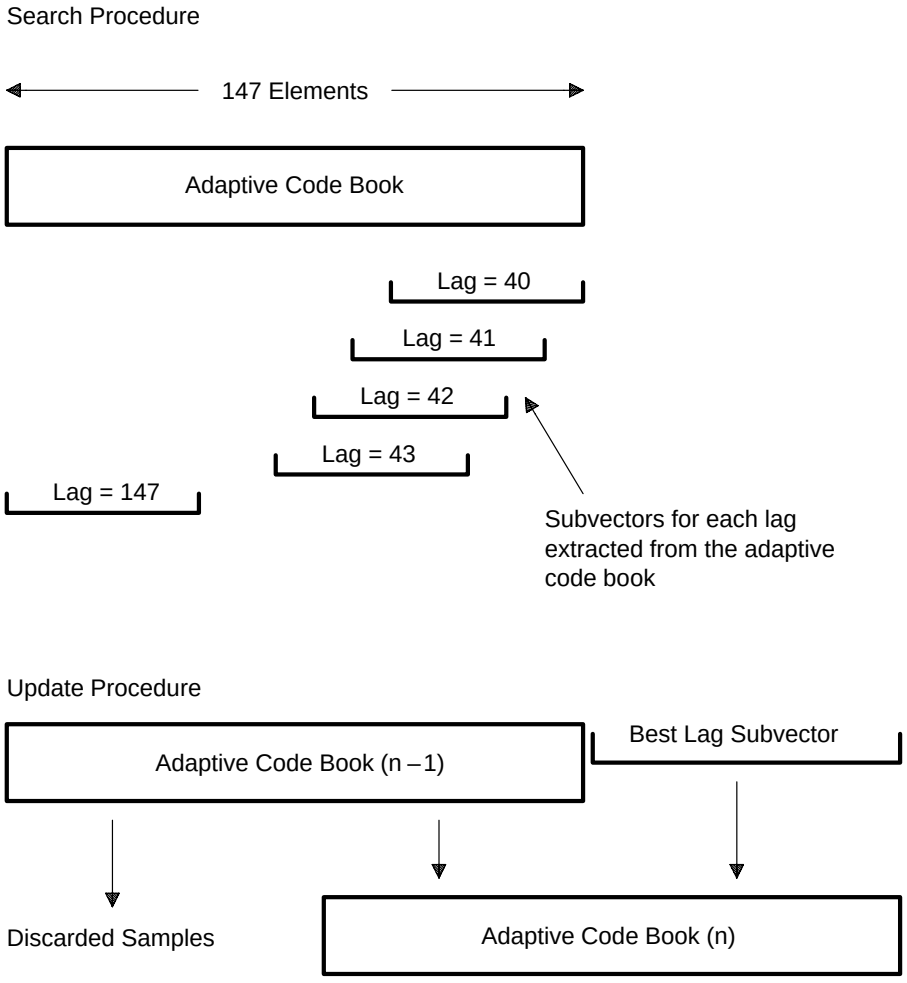
$$B(z) = \frac{1}{1-\beta z^{-L}} \tag{1}$$

To accommodate lags less than the subframe size (L < NSF), the equation is modified such that the filter's output is only a function of the filter state at the start of a subframe.

$$B(z) = \frac{1}{1-\beta_z{}^{-\mathrm{flr}(\frac{n+L}{L})L}} \tag{2}$$

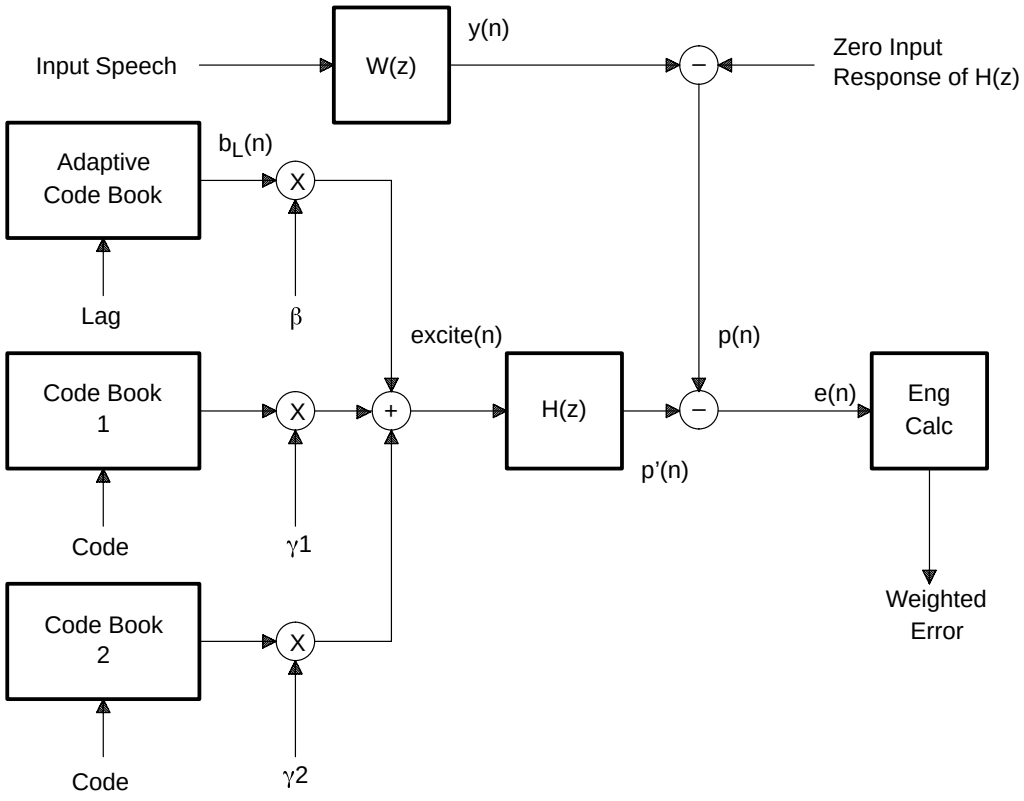
The flr(x) function truncates the fractional portion of x, returning only the integer portion of x. For $L \leq$ NSF, the equations are identical. For L < NSF, the flr function will evaluate to 2 when n = L, as depicted in Figure 2.

**Figure 2. Adaptive Code Book Search**




In Figure 2, the portion of the adaptive code book utilized (call this subvector $b_L$) is of length NSF and starts at the index defined by the current lag value in the search procedure. For $L \geq NSF$, this procedure is straightforward because the length of $b_L$ fits (see Figure 2) inside the adaptive code book. The VSELP algorithm supports lags from 20 to 147; therefore, a special situation exists when the lag (L) is less than NSF. In this case, the $b_L$ vector is placed such that a portion of it hangs over the adaptive code book. These elements of the adaptive code book (long-term filter state) do not exist yet. The flr function of equation [2] remedies this by doubling the lag (code book index value). This results in copying the first NSF – L elements of the $b_L$ vector to the ending NSF – L elements.

**Figure 3. Code Book Search Signal Flow**




For each lag ($20 \leq L \leq 146$), a vector called $b_L(n)$ of length NSF is extracted from the adaptive code book. This vector is filtered through the bandwidth-expanded LPC filter H(z). The resulting vector, $b'_L(n)$, is compared to the input vector p(n). The p(n) vector is the perceptually weighted input speech vector minus the zero-input response of H(z). The zero-input response is subtracted from the input speech to remove any of the ringing of the H(z) filter caused by the previous subframe. The $b_L$ vector that produces the minimum mean square error (MSE) (or maximum match score) compared to p(n) is chosen as the best vector from the adaptive code book. The lag L that produced this $b_L$ vector is transmitted to the receiver. The match score is defined as:

$$MS = \frac{(C_L)^2}{G_L} \tag{3}$$

where:

$$G_L = \sum_{n=0}^{NSF-1} (b'_L(n))^2 \tag{4}$$

$$C_L = \sum_{n=0}^{NSF-1} b'_L(n)p(n) \tag{5}$$

In digital cellular VSELP, $\beta$ is restricted to positive numbers; therefore, only lags with a positive $C_L$ are considered in the search procedure. If no lag with a positive $C_L$ can be found, the adaptive code book is disabled. The lag is coded using seven bits, yielding 128 possible lag values. Since only 127 of these values are valid ($20 \leq L \leq 146$), one lag value is reserved to disable the adaptive code book search in the decoder. It should be noted that the gain coefficient is not coded at this time. After all three code vectors are determined, a joint optimization is performed on the three gain terms, $\beta$, $\gamma 1$, and $\gamma 2$.

Our implementation precomputes all of the correlations and energies and stores them. The temporary storing of these parameters is not strictly necessary; however, it allows us to find a scale factor so the search can be performed utilizing maximum dynamic range. Preserving dynamic range is very important for a proper pitch search.

### Code Search Algorithm

Each of the two code books is constructed from a set of M basis vectors. These vectors are combined linearly to form a code book of size $2^M$. The code book vectors are described by:

$$u_i(n) = \sum_{m=1}^{M} \theta_{im} v_m(n) \tag{6}$$

where $v_m$ is the mth basis vector and $u_i$ is the ith code-book vector. The value of $\theta$ is either +1 or –1 and is formulated as follows. Each of the code book vectors, $u_i$, is indexed by i. If the indices are viewed in binary form, M bits are required to represent the index space. If the LSB of the index is defined as bit 1 and the MSB is defined as bit M, then $\theta_{im}$ can be defined as:

If (bit m of index i = 1)

then $\theta_{im} = +1$

If (bit m of index i = 0)

then $\theta_{im} = -1$

The following provides an example for the trivial case when M = 2. This defines a code book size of $2^2$, or 4. In this case, only two basis vectors are required, namely $v_1$ and $v_2$. Each of the four code book vectors is developed below.

$$u_i = \theta_{i1} \times v_1 + \theta_{i2} \times v_2$$

$$u_0 = u_{00} = \theta_{01} \times v_1 + \theta_{02} \times v_2 = v_1 + v_2$$
$$u_1 = u_{01} = \theta_{11} \times v_1 + \theta_{12} \times v_2 = -v_1 + v_2$$
$$u_2 = u_{10} = \theta_{21} \times v_1 + \theta_{22} \times v_2 = v_1 - v_2$$
$$u_3 = u_{11} = \theta_{31} \times v_1 + \theta_{32} \times v_2 = -v_1 - v_2$$

It should be noted that $u_0 = -u_3$ and $u_1 = -u_2$. These are called complementary code book vectors, and this property is exploited in the code book search to reduce computational requirements.

The VSELP code book structure was defined above for a static single code book. The formula below expands the notation to describe a VSELP structure with multiple static code books. From equation (6):

$$u_{k,i}(n) = \sum_{m=1}^{M} \theta_{im} v_{k,m}(n) \tag{7}$$

For digital cellular VSELP, k = 1 or 2; that is, two static code books are used. The three code books are searched sequentially. First, the adaptive code book is searched for the optimal vector assuming $\gamma 1 = 0$ and $\gamma 2 = 0$. The technique used in searching the adaptive code book is described above. For the stochastic code book searches, it is necessary to generate the zero-state response of each code vector to H(z). This is accomplished by filtering each of the M (M=7) basis vectors for each code book through H(z) with the history of H(z) set to 0 prior to filtering each vector. The resulting code vectors are defined by equation (8):

$$f_{k,I}(n) = \sum_{m=1}^{M} \theta_{im} q_{k,m}(n) \tag{8}$$

where $q_{k,m}(n)$ is the zero-state response of H(z) to the basis vector $v_{k,m}(n)$.

The result of the first search is the optimal lag value and the optimal $b_L(n)$ vector. The $b_L(n)$ vector times its gain, $\beta$, represents the adaptive code book's contribution to the excitation signal. Next, the first stochastic code book is searched, given $b_L(n)$. This results in an optimal code vector and corresponding index (I) for the first code book, $f_{1,I}$. Finally, the second code book is searched given $b_L(n)$ and $f_{1,I}(n)$. This results in an optimal code vector and corresponding index (H) for the second code book, $f_{2,H}(n)$.

All of the searches in this implementation take full advantage of the 'C5x MAC instructions and are optimized for speed.

### Orthogonalization of the Code Vectors

The error signal generated after each of the code vectors from each code book is selected is:

$$e(n) = p(n) - \beta b_L(n) - \gamma 1 f_{k,I}(n) - \gamma 2 f_{k,I}(n) \tag{9}$$

and

$$\text{Total weighted error} = \sum_{n=0}^{NSF-1} e^2(n) \tag{10}$$

Given $\gamma 2 = 0$ and $b_L(n)$ for the first code book search, optimal values for $\beta$, $\gamma 1$, and $f_{1,I}(n)$ must be found. This however, would be too computationally expensive for real-time performance. If the $b'_L$ vector and the each of the code vectors $f_{1,I}$ are orthogonal, then $\gamma 1$ and the code vector can be jointly optimized independent of $\beta$. By orthogonalizing each of the basis vectors to the $b'_L(n)$ vector, the entire space of code vectors is orthogonalized. The Grahm-Schmidt algorithm is used to perform this orthogonalization as follows:

$$\Gamma = \sum_{n=0}^{NSF-1} (b'_L(n))^2 \tag{11}$$

and

$$\Psi_m = \sum_{n=0}^{NSF-1} b'_L(n) q_{1,m}(n) \qquad \text{for } 1 \le m \le M \tag{12}$$

The orthogonalized, filtered basis vectors for the first code book are defined by:

$$q'_{1,m}(n) = q_{1,m}(n) - \frac{\Psi_m}{\Gamma} b'_L(n) \tag{13}$$

The orthogonalized, filtered code vectors for the first code book are defined by:

$$f'_{1,i}(n) = \sum_{m=1}^{M} \theta_{im} q'_{1,m}(n) \quad \text{for } 0 \le i \le 2^M - 1 \tag{14}$$

The new expression for the total weighted error for the first code book search is

$$E'_{1,i} = \sum_{n=0}^{NSF} (p(n) - \gamma_1 f'_{1,i}(n))^2 \tag{15}$$

This expression is independent of b and $b_L$ and also assumes no contribution from the second code book. The value for the gain is computed for each code vector but is not encoded yet. As stated previously, the value for the gains of each of the vectors contributing to the excitation vector are jointly optimized after all searches are complete.

The second stochastic code book search is identical to the first except that the basis vectors for the second code book are orthogonalized to both the $b_L(n)$ vector and to the optimum code vector from code book 1, $f'_{1,I}(n)$. This orthogonalization can be performed sequentially. The filter basis vectors, $q_{2,m}(n)$, are first orthogonalized to $b_L(n)$. The resulting vectors are then orthogonalized to $f'_{1,I}(n)$.

The orthogonalized, filtered code vectors for the second code book are defined by:

$$f'_{2,i}(n) = \sum_{m=1}^{M} \theta_{im} q'_{2,m}(n) \quad \text{for } 0 \le i \le 2^M - 1 \tag{16}$$

The new expression for the total weighted error for the second code book search is

$$E'_{2,i} = \sum_{n=0}^{NSF} (p(n) - \gamma_2 f'_{2,i}(n))^2 \tag{17}$$

For the implementation of the fixed-point VSELP, a modified Grahm-Schmidt algorithm was used. The difference between this Grahm-Schmidt and the one just presented is that this one is scaled by an energy constant. This scale washes out in the code book search, yet avoids an expensive division and preserves dynamic range.

### Gray Code Search

In this section, a fast search procedure for finding the best code vector from the stochastic code book is developed. As with the adaptive code book search, the vector that minimizes the MSE (that is, that maximizes the match score) is sought. Note that the subscript denoting the first or second code book has been dropped for clarity. The code search procedures are identical for each code book. The match score is defined as:

$$MS = \frac{(C_i)^2}{G_i} \tag{18}$$

The search procedure calculates the match score for each vector in the code book. The best code vector (indexed by i) will have the highest match score of all code vectors in the code book. The computational requirements for one subframe search of one code book is $2 \times NSF$ multiply-accumulates (MACS). This results in a code book search computational requirement of:

$$2 \times NSF \times 2^M(\frac{MACS}{code\ book}) \times 2(\frac{codebooks}{subframe}) \times 4(\frac{subframes}{frame}) \times 50(\frac{frames}{s}) \tag{19}$$

$$= 4.1 \text{ x } 10^6(\frac{MACS}{s})$$

To reduce this complexity, the structure of the VSELP code books is exploited. Defining the correlation between the p(n) vector and the filtered code vector, $f'_i(n)$:

$$C_i = \sum_{n=0}^{NSF} f'_i p(n) \tag{20}$$

Expanding f'i(n) using equation (8) yields:

$$= \sum_{n=0}^{NSF-1} \sum_{m=1}^{M} \theta_{im} q'_m(n) p(n) \tag{21}$$

Rearranging the summations yields:

$$= \sum_{m=1}^{M} \theta_{im} \sum_{n=0}^{NSF-1} q'_m(n) p(n) \tag{22}$$

Defining

$$R_m = 2 \sum_{n=0}^{NSF-1} q'm(n) p(n) \tag{23}$$

then substituting this back into 22 yields:

$$C_i = \frac{1}{2} \sum_{m=1}^{M} \theta_{im} R_m \tag{24}$$

Defining the gain of the filtered code vector, $f'_i(n)$:

$$G_i = \sum_{n=0}^{NSF-1} (f'_i(n))^2 \tag{25}$$

Expanding $f'_i(n)$ using equation (8) yields:

$$= \sum_{n=0}^{NSF-1} \left( \sum_{m=1}^{M} \theta_{im} q'_{m(n)} \right) \left( \sum_{j=1}^{M} \theta_{ij} q'_{j(n)} \right) \tag{26}$$

Rearranging the summations yields:

$$= \sum_{m=1}^{M} \sum_{j=1}^{M} \theta_{im} \theta_{ij} \sum_{n=0}^{NSF-1} q'_j(n) q'_m(n) \tag{27}$$

Defining

$$D_{mj} = 4 \sum_{n=0}^{NSF-1} q'_m(n) q'_j(n) \tag{28}$$

and substituting back into equation (27) yields:

$$G_i = \sum_{m=1}^{M} \sum_{j=1}^{M} \theta_{im} \theta_{ij} \frac{D_{mj}}{4} \tag{29}$$

Because:

$$\theta_{ij} \theta_{im} = \theta_{im} \theta_{ijj}$$

and:

$$\theta_{ij} \theta_{im} = 1 \quad \text{for } j = m$$

the equation can be expanded to:

$$G_i = \frac{1}{2} \sum_{j=2}^{M} \sum_{m=1}^{j-1} \theta_{im} \theta_{ij} D_{mj} + \frac{1}{4} \sum_{j=1}^{M} D_{jj} \tag{30}$$

Given two code words indexed by i and u such that u differs from i by only one bit (that is, bit position v), then:

$$\theta_{uv} = -\theta_{iv} \tag{31}$$

$$\theta_{um} = \theta_{im} \quad \text{for } m \mathrel{!}= v \tag{32}$$

The correlations $C_i$ and $C_u$ are related by:

$$C_u = C_i + \theta_{uv} R_v \tag{33}$$

The gains $G_i$ and $G_u$ are related by:

$$G_u = G_i + \sum_{j=1}^{v-1} \theta_{uj} \theta_{uv} D_{jv} + \sum_{j=v+1}^{M} \theta_{uj} \theta_{uv} D_{vj} \tag{34}$$

If the code book is searched in a sequence such that the code vector index changes by only one bit from the previous code vector index, then the previous set of equations leads to a very efficient method to search the code book. By sequencing the indices using a Gray code, only one bit will change as the indices are generated. In addition, only half of each code book needs to be searched because the other half is the complementary set of code vectors (differing only by sign). The sign of $C_i$ is checked to determine which of the complementary code vectors yields a positive gain $\gamma$. The resulting computational requirements are now reduced to:

$$CR = 2 \times 4 \times 50 \times \{[2 \times M1 \times NSF + M1 + 28] + [\frac{2^M}{2} \times (M1 + 2)]\}$$

$$= 0.468 \times 10^6 \text{ MACS} \tag{35}$$

### Gain Quantization

The gain values for each of the three code book contributions to the excitation vector are jointly optimized using a vector quantization table. The development of the quantization procedure can be found in [1]. The parameters required for the joint vector quantization of the gain values are:

$$R_{cc}(j, k) = \sum_{n=0}^{N-1} c'_k(n)c'_j(n) \quad k = 0, 2; \; j = k, 2 \tag{36}$$

where $c'_k(n)$ denotes the kth (k = [0...2]) excitation contribution vector filtered through the H(z) synthesis filter. Therefore, the upper triangular matrix $R_{cc}$ is the crosscorrelation matrix of the three filtered code book excitation contributions.

$$R_{pc}(k) = \sum_{n=0}^{N-1} p(n)c'_k(n) \quad k = 0, 2 \tag{37}$$

where p(n) is the perceptually weighted speech minus the ringing in the synthesis filter from the previous frame. The three-element vector $R_{pc}$ is the crosscorrelation vector of the three filtered code book excitation contributions with the p(n) vector.

$$R_x(k) = \sum_{n=0}^{N-1} c^2_k(n) \quad k = 0, 2 \tag{38}$$

where $c_k(n)$ denotes the kth (k = [0...2]) excitation contribution vector (not filtered). Thus, the vector $R_x(k)$ denotes the energy in each of the three code book excitation contributions.

Equation (39) defines the parameter RS, the energy in the LPC filter's residual signal.

$$RS = NSF \times R'_q(0) \times \prod_{i=1}^{NP}(1 - r_i^2) \tag{39}$$

where $R'_q(0)$ is the average power in the current subframe of speech and the product series is the normalized error power in the synthesis filter. $R'_q(0)$ is interpolated from $R_q(0)$ at the subframe rate using the strategy in Equations 40 – 42.

$$R'_q(0) = R_q(0)_{\text{previous frame}} \quad \text{for subframe 1} \tag{40}$$

$$R'_q(0) = R_q(0)_{\text{current frame}} \quad \text{for subframes 3, 4} \tag{41}$$

$$R'_q(0) = \sqrt{R_q(0)_{\text{previous frame}} R_q(0)_{\text{current frame}}} \quad \text{for subframe 2} \tag{42}$$

The error equation used in searching the quantization tables is:

$$\begin{aligned} E = &-a\sqrt{GS\ P0} - b\sqrt{GS\ P1} - c\sqrt{GS\ (1-P0-P1)} \\ &+ d\ GS\sqrt{P0\ P1} + e\ GS\ \sqrt{P0(1-P0-P1)} + f\ GS\sqrt{P1(1-P0-P1)} \\ &+ g\ GS\ P0 + h\ GS\ P1 + i\ GS\ (1-P0-P1) \end{aligned} \tag{43}$$

where $P_0$ is the fraction of the coder excitation energy due to the adaptive code book contribution, $P_1$ is the fraction of the coder excitation energy due to the first stochastic code book, and GS is an energy *tweak* parameter (GS = R/RS). Note: (1–P0–P1) is the fraction of the coder excitation energy due to the second stochastic code book. The definitions of *a* through *i* follow:

$$a = 2R_{pc}(0)\ \sqrt{\frac{RS}{R_x(0)}} \tag{44}$$

$$b = 2R_{pc}(1)\ \sqrt{\frac{RS}{R_x(1)}} \tag{45}$$

$$c = 2R_{pc}(2)\ \sqrt{\frac{RS}{R_x(2)}} \tag{46}$$

$$d = \frac{2R_{cc}(0,1)RS}{\sqrt{R_x(0)R_x(1)}} \tag{47}$$

$$e = \frac{2R_{cc}(0,2)RS}{\sqrt{Rx(0)R_x(2)}} \tag{48}$$

$$f = \frac{2R_{cc}(1,2)RS}{\sqrt{R_x(1)R_x(2)}} \tag{49}$$

$$g = \frac{R_{cc}(0,0)RS}{R_x(0)} \tag{50}$$

$$h = \frac{R_{cc}(1,1)RS}{R_x(1)} \tag{51}$$

$$i = \frac{R_{cc}(2,2)RS}{R_x(2)} \tag{52}$$

The values P0, P1, and GS are vector quantized in a three-column table of length 256. For each subframe, the index of the elements that minimize the error equation (43) is selected. The resulting code book gains are defined by the following equations, where the subscript vq indicates the index of the best table entry.

$$\beta_q = \sqrt{\frac{RS\ GS_{vq}\ P0_{vq}}{R_x(0)}} \tag{53}$$

$$^{th}\gamma_{1q} = \sqrt{\frac{RS\ GS_{vq}\ P1_{vq}}{R_x(1)}} \tag{54}$$

$$\gamma_{2q} = \sqrt{\frac{RS\ GS_{vq}\ (1-P0_{vq}-P1_{vq})}{R_x(1)}} \tag{55}$$

For the fixed-point implementation, the energies are calculated and converted up front to floating-point format. The parameters are then calculated in floating point because of the wide dynamic range. These parameters are then scaled back to the 16-bit integer domain according to the largest of the parameters (hence, the ratios between parameters are maintained.)

## Speech Decoder

The speech decoder resembles the encoder with the following exceptions:

- The coefficients for the LPC synthesis filter are not the bandwidth-expanded ones. They are taken from the RC coefficients in the RX bitstream.
- There is no closed-loop search procedure.
- There is an adaptive postfilter in the signal flow.

The coefficients for the filter A(z) are interpolated at the subframe rate from the reflection coefficients received at the frame rate. For each frame, the quantized reflection coefficients specified by the bitstream are converted to direct form-filter coefficients. They are then interpolated using the same scheme as defined in the interpolation section. The three code book indices are used to look up the correct vector in each of the code books. Each selected vector is multiplied by its corresponding gain value as calculated using equations (53), (54), and (55). The three scaled code book contributions are then summed to form the excitation signal and applied as input to the LPC synthesis filter A(z). In addition, this excitation signal is fed back into the adaptive code book. The output of the LPC synthesis filter is called the nonpostfiltered speech vector. To mask the effects of quantization in the coder, the speech is filtered through a spectral postfilter.

### Adaptive Postfilter

The adaptive postfilter shapes the noise spectrum to match the speech spectrum, thus hiding the effects of quantization in the VSELP coder beneath the formants of the speech signal [12]. Given the speech synthesis filter, $1/A(z)$, the postfilter is defined as:

$$H(z) = \frac{A(\frac{z}{bwf1})}{A(\frac{z}{bwf2})} \tag{56}$$

where $0 \leq bwf1 \leq bwf2 < 1$. With bwf1 and bwf2 defined as bandwidth expansion factors (like the bandwidth factors used in the perceptual-weighting filter), this filter boosts the formants in the speech signal. Several methods exist for the implementation of the postfilter. Two methods are outlined below.

### TIA Postfilter

A problem with the postfilter described above is the accentuation of the speech signal's spectral tilt. This results in the attenuation of the higher frequencies of the speech spectrum. The method described in [1] requires the use of a Levinson-Durbin recursion after the bandwidth expansion of the speech correlation coefficients. The denominator coefficients are converted to autocorrelation coefficients and then bandwidth expanded by $w(i) = 0.923077^{(i \times i)}$. Finally, these autocorrelation coefficients are converted back to filter coefficients via a Levinson-Durbin recursion. This proves to be computationally expensive and provides no quality improvement compared to the method described below. In addition to the spectral shaping filter, a brightness filter is used to boost the high frequencies. The speech, after passing through the filter H(z), is scaled to remove any gain introduced by the filter.

$$Scale = \sqrt{\frac{\sum_{n=0}^{NSF-1} (s_{in}(n))^2}{\sum_{n=0}^{NSF-1} (s_{out}(n))^2}} \tag{57}$$

The scale value is then passed through a first order low-pass filter to remove discontinuities:

$$Scale'(n) = = 0.9875 \times Scale'(n-1) + 0.125 \times Scale \tag{58}$$

### Modified Postfilter

Rather than adjusting for the spectral tilt in the postfilter via adjusted numerator coefficients, this method utilizes an adaptive brightness filter. The first reflection coefficient of the numerator filter is used as the coefficient for the brightness filter. This method is described in [14]. This results in the same spectral effect as the specified method, yet it is computationally less expensive. This is the method we used for our implementation.

## Features of VSELP

The code book described above allows a fast code book search to be conducted. Memory requirements are also reduced since only the basis vectors are stored (not the entire code book). The selected code book index is robust to channel errors because an error in the index changes only the sign of one of the basis vectors. Most importantly, the gains associated with each of the vectors contributing to the excitation vector are jointly optimized and quantized.

## TMS320C5x Real-Time Implementation

The DSPSE implementation of VSELP on the TMS320C5x is written entirely in assembly code so that it can fit on one 'C5x running at 20 MIPS. The two main functions, analysis and synthesis, are completely modular and C callable. The memory and MIPS requirements are listed below.

### Processing Requirements

The table below lists the processor utilization requirements for the TMS320C5x VSELP vocoder software.

**Table 3. VSELP Vocoder Processor Requirements**

| Application | MIPS Maximum | Utilization at 20 MIPS† | MIPS Average | Utilization at 20 MIPS† |
|---|---|---|---|---|
| Analysis | 16.10 | 81% | 15.30 | 77% |
| Synthesis | 3.60 | 18% | 3.32 | 17% |

† Values reflect execution from zero-wait-state external SRAM and use of TMS320C5x internal RAM.

### Memory Requirements

The table below lists the memory requirements for the TMS320C5x VSELP vocoder software. All memory specifications are in units of 16-bit words.

**Table 4. VSELP Vocoder Memory Requirements**

| Function | ROM | On-Chip RAM | External RAM | Total RAM |
|---|---|---|---|---|
| Analyzer | 8.2K | 1.5K | 0.23K | 1.73K |
| Synthesizer | 3.32K | 1.1K | 0.23K | 1.33K |
| Full Duplex VSELP | 9.0K | 1.55K | 0.42K | 1.97K |

The three on-chip memory blocks are b0, b1, and b2 and are used as follows:

Block b0 is a special block in that it is the only segment of RAM that can be switched into program memory. This feature is useful for filtering operations such as the MACD instruction. Because this memory is dynamically switched as program or data memory, no static variables reside in this block. However, this block is used as temporary memory in the code book searches.

Block b1 is used in two ways. The first 350 locations are used as temporary scratch-pad memory. The remaining locations are used for time-critical buffers such as the intermediate weighted excitation vectors and the stack.

Block b2 is used to overlay local temporary variables. This strategy not only saves memory but also allows all local variables to be placed in fast dual-access RAM for maximum DSP performance.

### Speech Coder Quality

Quality measures were used to compare the speech output of the fixed point VSELP (TMS320C5x) with a C model of the TIA reference synthesizer. The input bitstream for each of five speakers (three male and two female) produced five reference files, both postfiltered and nonpostfiltered. This same bitstream was used as input to the 'C5x implementations of the VSELP coder. The resulting speech files were compared to the reference files using the SNR measure described below.

### SNR Measurements

To track the progress of algorithmic modification, the *segmental* SNR measure was used. The segmental SNR is the average of the each subframe's SNR over some segment of speech.

$$\text{SegSNR} = \frac{1}{L}\sum_{i=0}^{i=L-1} 10 * \log_{10}\left(\frac{\sum_{n=0}^{NSF-1} s_i(n)^2}{\sum_{n=0}^{NSF-1} (s_i(n)-s_p(n))^2}\right) \quad (59)$$

where L is the length of the speech segment in subframes, $s_i$ is the input speech, and $s_p$ is the synthetic speech. This measure is used in testing vocoder implementations against the reference vocoder. For the five reference files, the output of the synthesizer was compared to the output of the reference vocoder's synthesizer. All the SNR values for the fixed-point implementation were distributed between 25 and 30 dB.

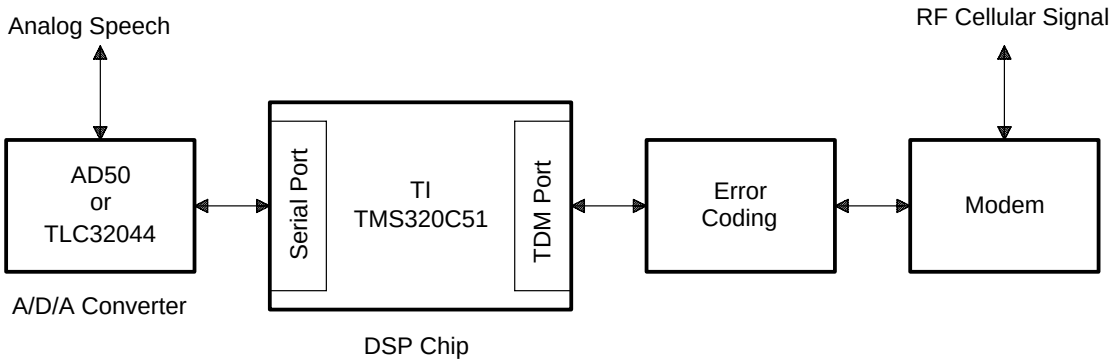
### DTMF Performance

The VSELP algorithm must pass the dual-tone multifrequency (DTMF) signals to allow for remote signaling and dialing. Several DTMF files were recorded and processed through the algorithm. The Fourier spectra were analyzed for proper frequency content. In addition, the resulting files were used to signal the central office and correctly initiate a telephone connection.

## A Typical Digital Cellular Vocoder Configuration

Figure 4 illustrates a possible digital cellular system configuration. Analog speech sampled by the A/D converter is processed by the TMS320C51 digital signal processor to produce a VSELP coded bitstream. This bitstream is passed through the error-coding block to protect the data against channel errors. Finally, the error-coded VSELP bitstream is modulated and transmitted to the cellular base station. Since the digital cellular telephone is full duplex, incoming RF data is simultaneously processed in the reverse order to produce speech. The incoming signal is demodulated and error corrected before the VSELP synthesis processing and D/A conversion.

**Figure 4. Possible Digital Cellular System Configuration**




## Code Availability

The associated software is available for licensing from DSP Software Engineering Incorporated, 165 Middlesex Turnpike, Suite 206, Bedford, MA 01730

## References


1. "Vector Sum Excited Linear Prediction (VSELP) 7950 Bit Per Second Voice Coding Algorithm", *Technical Description*, Motorola, Inc., November 1989.

2. *Cellular System: Dual-Mode Mobile Station – Base Station Compatibility Standard*, IS-54 Project Number 2215, Electronic Industries Association, December 1989.

3. Schroeder, M.R., and Atal, B.S., "Code-Excited Linear Prediction (CELP): High Quality Speech at Very Low Bit Rates", *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, March 1985, pp. 937–940.

4. Davidson, G. and Gersho, A., "Complexity Reduction Methods for Vector Excitation Coding", *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, April 1986, pp. 3055–3058.

5. Gerson, I.A., and Jasiuk, M., "Vector Sum Excited Linear Prediction (VSELP)", *IEEE Workshop on Speech Coding for Telecommunications*, September 1989, pp. 66–68.

6. Gerson, I.A., "Method and Means of Determining Coefficients for Linear Predictive Coding", U.S. Patent #4,544,919, October 1985.

7. Cumani, A., "On a Covariance-Lattice Algorithm for Linear Prediction", *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, May 1982, pp. 651–654.

8. Tohkura, Y., Itakura, F., and Hashimoto, S., "Spectral Smoothing Technique in PARCOR Speech Analysis-Synthesis", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Volume ASSP–26, December 1978, pp. 587–596.

9. Atal, B.S., Schroeder, M.R., "Predictive Coding of Speech Signals and Subjective Error Criteria", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Volume ASSP–27, June 1979, pp. 247–254.

10. Kroon, P., Deptrettere, Ed.F., and Sluyter, R.J., "Regular-Pulse Excitation: A Novel Approach to Effective and Efficient Multipulse Coding of Speech", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Volume ASSP–34, October 1986, pp. 1054–1063.

11. Linde, Y., Buzo, A., and Gray, R.M., "An Algorithm for Vector Quantizer Design", *IEEE Transactions*, Communications Volume COM–28, January 1980, pp. 84–95.

12. Chen, Juin-Hwey, and Gersho, Allen, "Real-Time Vector APC Speech Coding at 4800 bps With Adaptive Postfiltering", *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, April 1987, pp. 51.3.1–51.3.4.

13. Kemp, D., Sueda, R., and Tremain, T., "An Evaluation of 4800 bps Voice Coders", *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Glasgow, 1989, pp. 200–203.

14. Fenichel, R., Proposed "Federal Standard 1016 (Second Draft)", National Communications System, Office of Technology and Standards, Washington, DC 20305-2010, November 1989.

15. Campbell, J., Welch, V., and Tremain, T., "An Expandable Error Protected 4800 bps CELP Coder", *Proceedings of ICASSP*, Glasgow, 1989, pp. 735-738.

16. Kroon, P., and Atal, B., "On Improving the Performance of Pitch Predictors in Speech Coding Systems", *Abstracts of the IEEE Workshop on Speech Coding for Telecommunications*, 1989, pp. 49–50.

17. Kroon, P., and Atal, B., "Strategies for Improving the Performance of CELP Coders at Low Bit Rates", *Proceedings of ICASSP*, 1988, pp. 151–154.

18. Campbell, J., Tremain, T., and Welch, V., "The DoD 4.8 kbps Standard (Proposed Federal Standard 1016) in Advances in Speech Coding", edited by B. Atal, V. Cuperman, and A Gersho, submitted to Kluwer Academic Publishers, 1990.

19. Macres, J., "The First Real-Time Implementation of U.S. Federal Standard 4800 bps CELP version 3.1", submitted to the *Proceeding on Speech Technology '90*, 1990.

20. Parsons, T.W., *Voice and Speech Processing*, McGraw-Hill, New York , New York, 1987.

21. Kemp, D., Sueda, R., and Tremain, T., "Processing of Military and Government Speech Technology '89", *Media Dimensions*, 1989, pp. 86–90.

## IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software 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.

Certain applications using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

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

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI 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 semiconductor products or services might be or are used.