Digital Voice Echo Canceller with a TMS32020

APPLICATION REPORT: SPRA129

Author David Messerschmitt, David Hedberg, Christopher Cole, Amine Haoui, and Peter Winship
Technekron Communications Systems

Digital Signal Processing Solutions
1989

TEXAS INSTRUMENTS

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 application 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.

TRADEMARKS

TI is a trademark of Texas Instruments Incorporated.

Other brands and names are the property of their respective owners.

CONTACT INFORMATION

US TMS320 HOTLINE	(281) 274-2320
US TMS320 FAX	(281) 274-2324
US TMS320 BBS	(281) 274-2323
US TMS320 email	dsph@ti.com

Digital Voice Echo Canceller with a TMS32020

Abstract

This report covers both the theory and implementation of a single chip TMS32020 digital voice echo canceller. The single-chip system can perform a 128-tap or 16-ms echo cancellation for telephone network applications. The echo canceller is implemented in accordance with the CCITT recommendation (G.165). A simulation has been performed to test the echo canceller, and the result exceeds the CCITT requirements.

Product Support on the World Wide Web

Our World Wide Web site at www.ti.com contains the most up to date product information, revisions, and additions. Users registering with TI&ME can build custom information pages and receive new product updates automatically via email.

INTRODUCTION

Echo cancellers using adaptive filtering techniques are now finding widespread practical applications to solve a variety of communications systems problems.[1] These applications are made possible by the recent advances in microelectronics, particularly in the area of Digital Signal Processors (DSPs). Cancelling echoes for long-distance telephone voice communications, full-duplex voiceband data modems, and high-performance "handsfree" audio-conferencing systems (including speakerphones) are a few examples of these applications.

The continuing deployment of all-digital toll switches, satellite-based voice and data networks, and new intercontinental long-haul circuits have been accompanied by more widespread use of all-digital voice echo cancellers in carrier systems.[2] In addition, new low-cost integrated single-channel echo cancellers are expected to see increasing application in smaller systems for audio teleconferencing and low-cost voice/data communications using private satellite earth stations.

Advancements in single-chip programmable digital signal processor technology now make it attractive to implement modular per-channel echo canceller architectures with all the functions required for a single echo canceller integrated within a single device. A programmable DSP implementation offers the advantages of a short development and test schedule and the flexibility to meet custom product requirements by extending software-based functional building blocks rather than designing new hardware.

This application report describes the implementation of an integrated 128-tap (16-ms span) digital voice echo canceller on the Texas Instruments TMS32020 programmable signal processor. The implementation features a direct interface for standard PCM codecs (e.g., Texas Instruments TCM2913) and meets the requirements of the CCITT (International Telegraph and Telephone Consultive Committee) Recommendation G.165 for echo cancellers.[3] This report presents the requirements for echo cancellation in voice transmission and discusses the generic echo cancellation algorithms. The implementation considerations for a 128-tap echo canceller on the TMS32020 are then described in detail, as well as the software logic and flow for each program module.

A hardware demonstration model of a 128-tap voice echo canceller using the TMS32020 has been constructed and tested. Figure 1 shows a photograph of the echo canceller demonstration system. The main features of this model are described within the report. The appendixes contain complete source code and a schematic for the demonstration system echo canceller module.

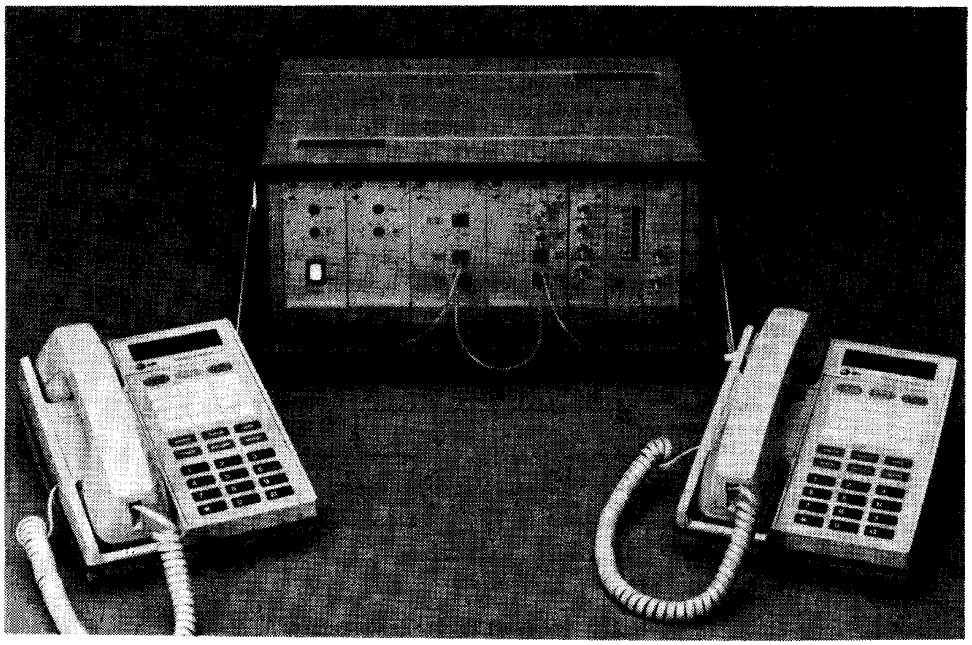


Figure 1. Echo Canceller Demonstration System

ECHO CANCELLATION IN VOICE TRANSMISSION

Echoes in the Telephone Network

The source of echoes can be understood by considering a simplified connection between two subscribers, S1 and S2, as shown in Figure 2. This connection is typical in that it contains two-wire segments on the ends, a four-wire connection in the center, and a hybrid at each end to convert from two-wire transmission to four-wire transmission. Each two-wire segment consists of the subscriber loop and possibly some portion of the local network. Over this segment, both directions of transmission are carried by the same wire pair, i.e., signals from speakers S1 and S2 are superimposed on this segment. On the four-wire section, the two directions of transmission are segregated. The speech from speaker S1 follows the upper transmission path, as indicated by the arrow, while speech originating from S2 follows the lower path. The segregation of the two signals is necessary where it is desired to insert carrier terminals, amplifiers, or digital switches.

The hybrid is a device that converts two-wire to four-wire transmission. The role of the hybrid on the right-hand side is to direct the signal energy arriving from S1 to the two-wire segment of S2 without allowing it to return to S1 via the lower four-wire transmission path. Because of impedance mismatches (unfortunately occurring in practice), some of this energy will be returned to speaker S1, who then hears a delayed version of his speech. This is the source of "talker echo."

The subjective effect of the talker echo depends on the delay around the loop. For short delays, the talker echo represents an insignificant impairment if the attenuation is reasonable (6 dB or more). This is because the talker echo is indistinguishable from the normal sidetone in the telephone. For satellite connections, the delay in each four-wire path is about 270 ms as a consequence of the high altitude of synchronous satellites. This means that the round-trip echo delay is approximately 540 ms, which makes it very disturbing to the talker, and can in fact make it quite difficult to carry on a conversation. When such is the case, it is essential to find ways of controlling or removing that echo. Since the subjective annoyance of echo increases with delay as well as echo level due to hybrid return energy, the measures for control depend on the circuit length.

For terrestrial circuits under 2,000 miles, the via net loss (VNL) plan,[4] which regulates loss as a function of transmission distance, is used to limit the maximum echo-to-signal ratio. On circuits over this length (e.g., intercontinental circuits), echo suppressors or cancellers are used. An echo suppressor is a voice-operated switch that attempts to open the path from listener to talker whenever the listener is silent. However, echo suppressors perform poorly since echo is not blocked during periods of doubletalk. They impart a choppiness to speech and background noise as the transmission path is opened and closed. Due to recent decreasing trends in DSP costs, digital echo cancellers are now viable as replacements for most of the circuits using echo suppressors.

For satellite circuits with full hop delays of 540 ms, echo suppressors are subjectively inadequate, and cancellers must be employed.

Digital Echo Cancellers in Voice Carrier Systems

The principle of the echo canceller for one direction of transmission is shown in Figure 3. The portion of the four-wire connection near the two-wire interface is shown in this figure, with one direction of voice transmission between ports A and C, and the other direction between ports D and B. All signals shown are sampled data signals that would occur naturally at a digital transmission terminal or digital switch. The far-end talker signal is denoted y(i), the undesired echo r(i), and the near-end talker x(i). The near-end talker is superimposed with the undesired echo on port D. The received signal from far-end talker y(i) is available as a reference signal for the echo canceller and is used by the canceller to generate a replica of the echo called $\hat{r}(i)$. This replica is subtracted from the near-end talker plus echo to yield the transmitted near-end signal u(i) where $u(i) = x(i) + r(i) - \hat{r}(i)$. Ideally, the residual echo error $e(i) = r(i) - \hat{r}(i)$ is very small after echo cancellation.

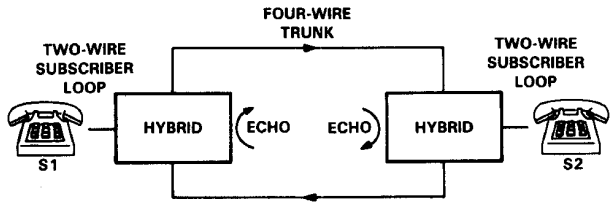


Figure 2. A Simplified Telephone Connection

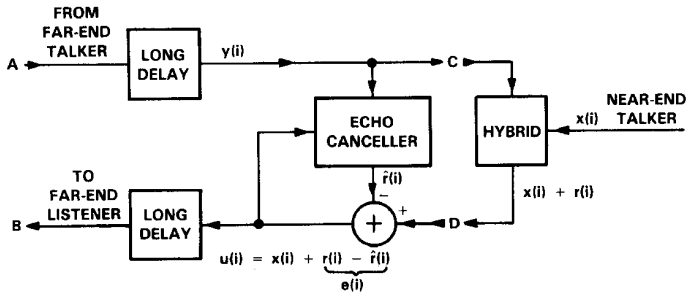


Figure 3. Echo Canceller Configuration

The echo canceller generates the echo replica by applying the reference signal to a transversal filter (tapped-delay line), as shown in Figure 4. If the transfer function of the transversal filter is identical to that of the echo path, the echo replica will be identical to the echo, thus achieving total cancellation. Since the transfer function of the echo path from port C to port D is not normally known in advance, the canceller adapts the coefficients of the transversal filter. To reduce error, the adaptation algorithm infers from the cancellation error e(i) (when no near-end signal is present) the appropriate correction to the transversal filter coefficients.

The number of taps in the transversal filter of Figure 4 is determined by the duration of the impulse response of the echo path from port C to port D. The time span over which this impulse response is significant (i.e., nonzero) is typically 2 to 4 ms. This corresponds to 16 to 32 tap positions with 8-kHz sampling. However, because of the portion of the four-wire circuit between the location of the echo canceller and the hybrid, this response does not begin at zero, but is delayed. The number of taps N, must be large enough to accommodate that delay. With N = 128, delays of up to 16 ms (or about 1,200 miles of "tail" circuit) can be accommodated.

In practice, it is necessary to cancel the echoes in both directions of a trunk. For this purpose, two adaptive cancellers are used, as shown in Figure 5, where one cancels the echo from each end of the connection. The near-end talker for one of the cancellers is the far-end talker for the other. In each case, the near-end talker is the "closest" talker, and the far-end talker is the talker generating the echo being cancelled. It is desirable to position these two "halves" of the canceller in a split configuration, as shown in Figure 5, where the bulk of the delay in the four-wire portion of the connection is in the middle. The reason is that the number of coefficients required in the echo-cancellation filter is directly related to the delay of the tail circuit between the location of the echo canceller and the hybrid that generates the echo. In the split configuration, the largest delay is not

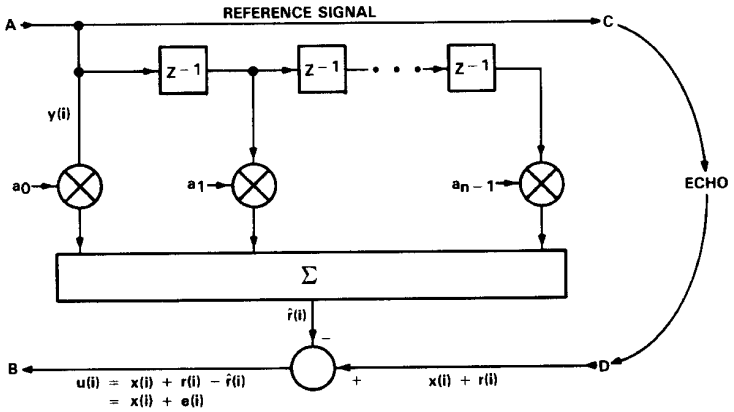


Figure 4. Echo Estimation Using a Transversal Filter

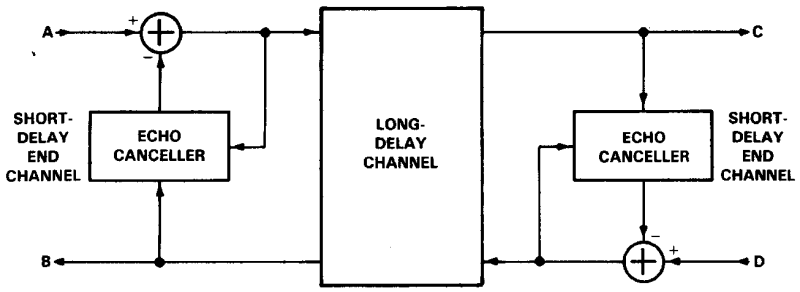


Figure 5. Split-Type Echo Canceller for Two Directions of Transmission

in the echo path of either half of the canceller. Therefore, the number of coefficients is minimized.

The digital voice echo canceller can be applied in a variety of transmission equipment configurations. Some of these are illustrated in Figures 6 through 8.

Figure 6 shows a single-channel echo canceller with a four-wire analog interface. The TMS32020 implementation described in this application report provides for the serial PCM codec interface required for this common configuration.

In digital carrier transmission systems, digital voice channels are usually carried in groups of 24 using the T1 group format.[5] As indicated in Figures 7 and 8, a T1-compatible digital voice echo canceller can be implemented with 24 single-channel echo cancellers connected directly to the serial 1.544-Mbps T1 PCM data streams for the transmit and receive groups.

Figures 9 through 11 show the appropriate architectures for applying digital voice echo cancellers to analog switching and analog transmission channel groups within the telephone network.

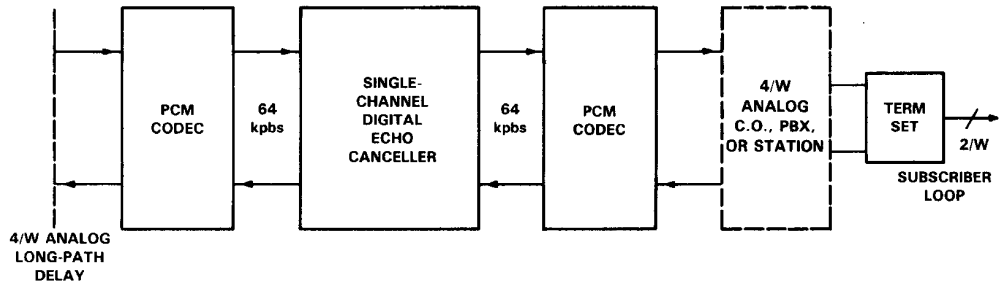


Figure 6. Single-Channel Four-Wire VF Echo Canceller

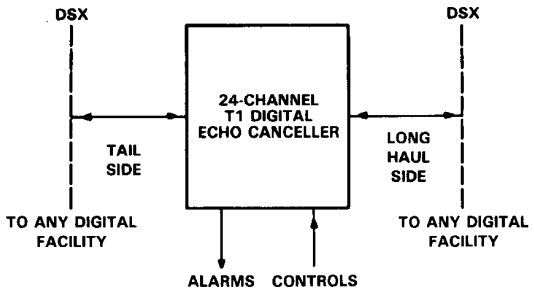


Figure 7. Standalone Digital T1 Echo Canceller

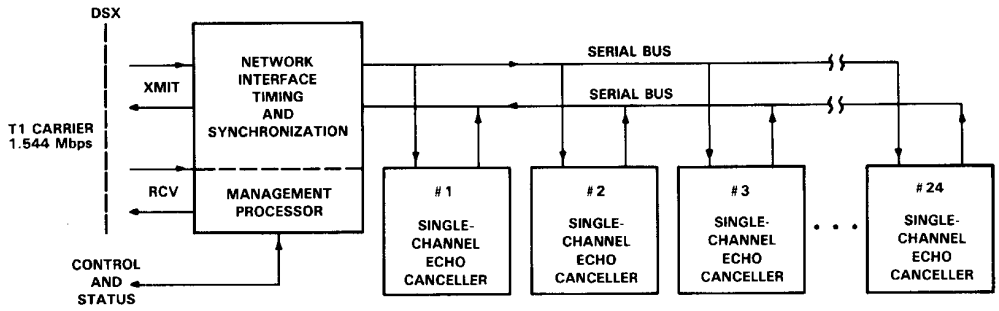


Figure 8. Per-Channel Architecture for a T1 Digital Echo Canceller

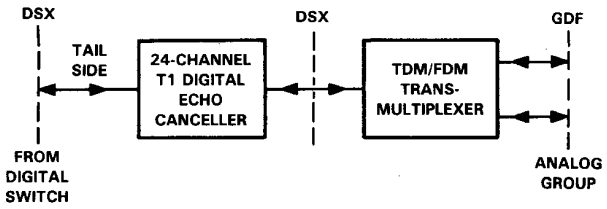


Figure 9. Digital Switch to Analog Facility

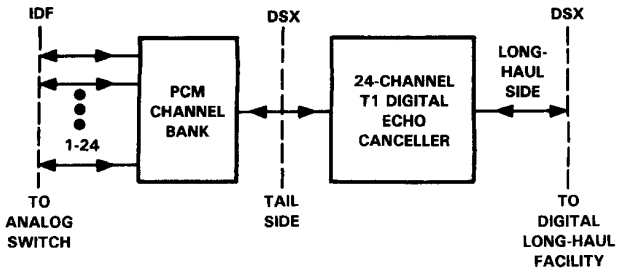


Figure 10. Analog Facility to Digital Facility

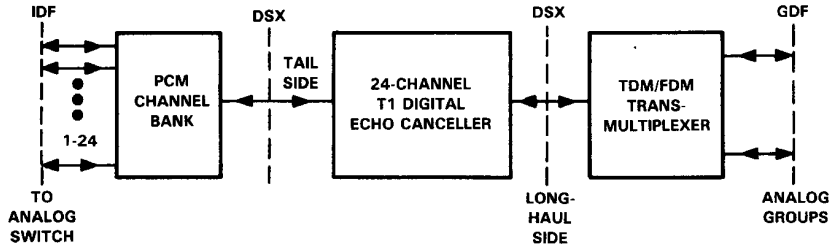


Figure 11. Analog Facility to Analog Facility

ECHO CANCELLATION ALGORITHMS

Generic algorithm requirements for each major signal processing function are discussed in this section. The signal processing flow for a single-channel digital voice echo canceller is shown in the block diagram of Figure 12.

Adaptive Transversal Filter

The reflected echo signal r(i) at time i (see Figure 3) can be written as the convolution of the far-end reference signal y(i) and the discrete representation h_k of the impulse response of the echo path between port C and D.

$$r(i) = \sum_{k=0}^{N-1} h_k \, y(i-k) \quad . \tag{1}$$

Linearity and a finite duration N of the echo-path response have been assumed. An echo canceller with N taps adapts the N coefficients a_k of its transversal filter to produce a replica of the echo r(i) defined as follows:

$$\hat{r}(i) = \sum_{k=0}^{N-1} a_k \, y(i-k) \tag{2}$$

Clearly, if $a_k = h_k$ for $k = 0,\ldots,N-1$, then $\hat{r}(i) = r(i)$ for all time i and the echo is cancelled exactly.

Since, in general, the echo-path impulse response h_k is unknown and may vary slowly with time, a closed-loop coefficient adaptation algorithm is required to minimize the average or mean-squared error (MSE) between the echo and its replica. From Figure 3, it can be seen that the near-end error signal u(i) is comprised of the echo-path error $r(i) - \hat{r}(i)$ and the near-end speech signal x(i), which is uncorrelated with the far-end signal y(i). This gives the equation

$$E(u^2(i)) = E(x^2(i)) + E(e^2(i)) \tag{3}$$

where E denotes the expectation operator. The echo term $E(e^2(i))$ will be minimized when the left-hand side of (3) is minimized. If there is no near-end speech ($x(i) = 0$), the minimum is achieved by adjusting the coefficients a_k along the direction of the negative gradient of $E(e^2(i))$ at each step with the update equation

$$a_k(i+1) = a_k(i) - \beta \frac{\partial E(e^2(i))}{\partial a_k(i)} \tag{4}$$

where β is the stepsize. Substituting (1) and (2) into (3) gives from (4) the update equation

$$a_k(i+1) = a_k(i) + 2\beta E\left[e(i)\, y(i-k)\right] \tag{5}$$

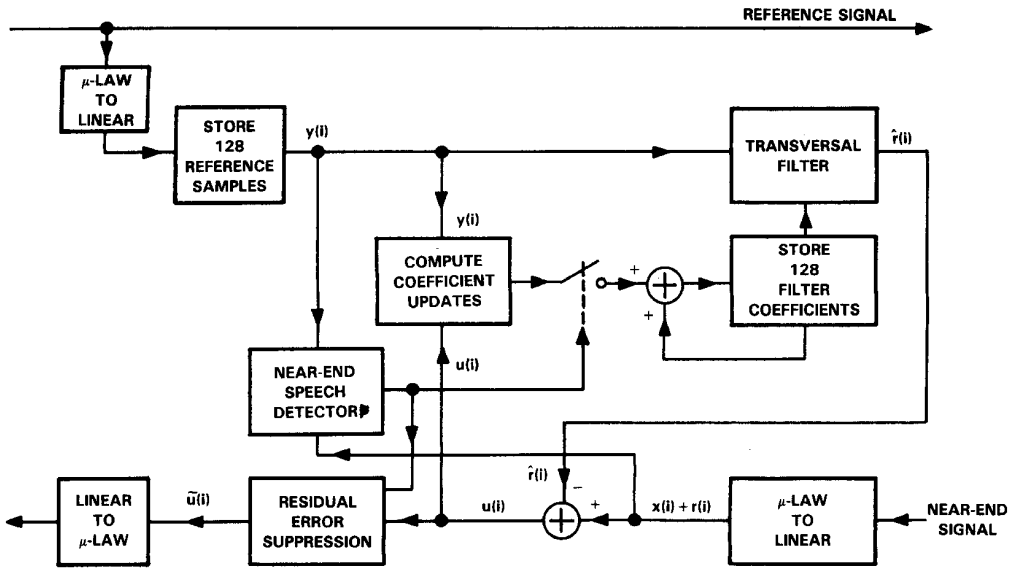


Figure 12. Signal Processing for a Digital Voice Echo Canceller

In practice, the expectation operator in the gradient term $2\beta E\,[e(i)\,y(i-k)]$ cannot be computed without a priori knowledge of the reference signal probability distribution. Common practice is to use an unbiased estimate of the gradient, which is based on time-averaged correlation error. Thus, replacing the expectation operator of (5) with a short-time average, gives

$$a_k(i+1) = a_k(i) + 2\beta \frac{1}{M} \sum_{m=0}^{M-1} e(i-m)\, y(i-m-k) \quad (6)$$

The special case of (6) for M = 1 is frequently called the least-mean-squared (LMS) algorithm or the stochastic gradient algorithm. Alternatively, the coefficients may be updated less frequently with a thinning ratio of up to M, as given in

$$a_k(i+M+1) = a_k(i) + 2\beta \sum_{m=0}^{M-1} e(i+M-m)\, y(i+M-m-k) \quad (7)$$

Computer simulations of this "block update" method show that it performs better than the standard LMS algorithm (i.e., M = 1 case) with noise or speech signals.[6] Many cancellers today avoid multiplication for the correlation function in (7), and instead use the signs of e(i) and y(i − k) to compute the coefficient updates. However, this "sign algorithm" approximation results in approximately a 50-percent decrease in convergence rate and an increase in degradation of residual echo due to interfering near-end speech.

The convergence properties of the algorithm are largely determined by the stepsize parameter β and the power of the far-end signal y(i). In general, making β larger speeds the convergence, while a smaller β reduces the asymptotic cancellation error.

It has been shown that the convergence time constant is inversely proportional to the power of y(i), and that the algorithm will converge very slowly for low-power signals.[7] To remedy that situation, the loop gain is usually normalized by an estimate of that power, i.e.,

$$2\beta = 2\beta(i) = \frac{\beta_1}{P_y(i)} \quad (8)$$

where β_1 is a compromise value of the stepsize constant and $P_y(i)$ is an estimate of the average power of y(i) at time i.

$$P_y(i) = (L_y(i))^2 \quad (9)$$

where $L_y(i)$ is given by

$$L_y(i+1) = (1-\rho)\, L_y(i) + \rho |y(i)| \quad (10)$$

The estimate $\rho_y(i)$ is used since the calculation of the exact average power is computation-expensive.

Near-End Speech Detector

When both near-end and far-end speakers are talking, the condition is termed "doubletalk." Since the error signal

u(i) of Figure 2 contains a component of the near-end talker x(i) in addition to the residual echo-cancellation error, it is necessary to freeze the canceller adaptation during doubletalk in order to avoid divergence. Doubletalk status can be detected by a near- end speech detector operating on the near-end and far-end signals y(i) and s(i), respectively.

A commonly used algorithm by A. A. Geigel[8] consists of declaring near-end speech whenever

$$|s(i)| = |x(i) + r(i)| \geq = \frac{1}{2} \max\{|\, y(i)|, |y(i-1)|, \ldots, |y(i-N)|\} \quad (11)$$

where N is the number of samples in the echo canceller transversal filter memory. It is necessary to compare s(i) with the recent past of the far-end signal rather than just y(i) because of the unknown delay in the echo path. The factor of one-half is based on the hypothesis that the echo-path loss through a hybrid is at least 6 dB. The algorithm in effect performs an instantaneous power comparison over a time window spanning the echo-path delay range.

A more robust version of this algorithm uses short-term power estimates, $\tilde{y}(i)$ and $\tilde{s}(i)$, for the power estimates of the recent past of the far-end receive signal y(i) and the near-end hybrid signal s(i), respectively.These estimates are computed recursively by the equations

$$\tilde{s}(i+1) = (1-\alpha)\, \tilde{s}(i) + \alpha |s(i)| \quad (12)$$

$$\tilde{y}(i+1) = (1-\alpha)\, \tilde{y}(i) + \alpha |y(i)| \quad (13)$$

where the filter gain $\alpha = 2^{-5}$. For this version of the algorithm, near-end speech is declared whenever

$$\tilde{s}(i) \geq \frac{1}{2} \max\,(\tilde{y}(i), \tilde{y}(i-1), \ldots, \tilde{y}(i-N)) \quad (14)$$

Since the near-end speech detector algorithm detects short-term power peaks, it is desirable to continue declaring near-end speech for some hangover time after initial detection.

Residual Echo Suppressor

Nonlinearities in the echo path of the telephone circuit and uncorrelated near-end speech limit the amount of achievable suppression in the circuit from 30 to 35 dB. Thus, there is no merit in achieving more than a certain degree of cancellation.

The use of a residual echo suppressor algorithm has been found to be subjectively desirable.[7] During doubletalk, the residual suppressor must be disabled. A common suppression control algorithm is to detect when the return signal power falls below a threshold based on the receive reference signal power. If the return signal consists only of residual echo and the canceller has properly converged, then the residual echo level will be below the threshold and the transmitted return signal will be set to zero.

The return signal power is estimated by the equation

$$L_u(i+1) = (1-\rho)\, L_u(i) + \rho |u(i)| \quad (15)$$

The reference power estimate $L_y(i)$ is given by (10). Suppression is enabled on the transmitted signal u(i) (i.e.,u(i) = 0) whenever $L_u(i)/L_y(i) < 2^{-4}$. This corresponds to a suppression threshold of 24 dB.

IMPLEMENTATION OF A 128-TAP ECHO CANCELLER WITH THE TMS32020

The TMS32020 is ideally suited for the implementation of a single 128-tap digital voice echo canceller channel since it has the capability and features to implement all of the required functions with full precision. This section discusses an implementation approach that meets or exceeds the performance of currently available products and the requirements of the CCITT G.165 recommendations.[3]

Echo Canceller Performance Requirements

Echo cancellers have the following fundamental requirements:

1. Rapid convergence when speech is incident in a new connection
2. Low-returned echo level during singletalking (i.e., echo-return loss enhancement)
3. Slow divergence when there is no signal
4. Rapid return of the echo level to residual if the echo path is interrupted
5. Little divergence during doubletalking

The CCITT recommendation G.165 specifies echo canceller performance requirements with band-limited white-noise (300 – 3400 Hz) test signals at the near-end and far-end input signal ports. The test specifications of G.165 are summarized in Table 1.

Digital voice echo canceller products are typically designed to accommodate circuits with tail delays of 16 ms or more and circuits with echo-return loss levels greater than 3 dB to 6 dB. Typical digital voice echo canceller product specifications are summarized in Table 2.

Table 1. CCITT G.165 Performance Test Specifications

CCITT TEST	DESCRIPTION	PERFORMANCE REQUIREMENT
1. Final echo return loss (ERL) after convergence; singletalk mode	Input noise level: −10 dbm0 to −30 dbm0 Circuit ERL: 10 dB Steady-state residual echo level after convergence with no near-end signal	−40 dbm0
2. Convergence rate; singletalk mode	Input noise level: −10 dbn 0 Combined echo loss after 500 ms from initialization with cleared register and with near-end signal set to zero at initialization time	≥27 dB
3. Leak rate	Degradation of residual echo after 2 minutes from time all signals are removed from fully converged canceller	≤10 dB
4. Infinite return loss convergence	Input noise level: −10 dbm0 to −30 dbm0 Circuit ERL: 10 dB Returned echo level 500 ms after echo path is interrupted	−40 dbm0

Table 2. Typical Echo Canceller Product Specifications

PARAMETER	SPECIFICATION
1. Maximum tail circuit length	16, 32, or 48 ms
2. Absolute delay	0.375 ms maximum
3. Minimum echo return loss	6 dB
4. Convergence	24 dB enhancement in 250 ms
5. Residual echo level (−30 to −10 dbm0 receive level)	−40 dbm0 (suppressor disabled) −65 dbm0 (suppressor enabled)
6. Speech detector threshold	6 dB below receive level
7. Speech detector hangover time	75 ms

Implementation Approach

In the implementation of the generic echo-cancelling algorithms discussed above, the coefficient update process dominates the computational requirement and efficiency of DSP realizations. The DSP efficiency and speed, in turn, determines the maximum number of echo canceller taps that can be achieved with the processor.

The block update approach of (7) with M = 16 was chosen for the TMS32020 implementation because it takes advantage of the efficient multiply and accumulate capabilities of the processor. Using the block update approach, a full-performance 128-tap canceller can be realized with a small margin. During each sample period (125 μs), 8 out of 128 coefficients are updated using correlation of the 16 past error and signal values.

Computer simulation studies were undertaken to verify the performance of the block update algorithm (M = 16) in comparison with the stochastic gradient algorithm (M = 1), taking into account the finite-precision and word-length limitations of the TMS32020. Figures 13 and 14 show the simulation results for three values of the compromize stepsize constant $\beta 1$, defined in (8). The curves represent the average of 600 samples for single convergence runs from a zero initial condition with white-noise input. The block update algorithm performs better than the stochastic gradient algorithm for all three values. For values of $\beta 1$ larger than 2^{-8}, the algorithm can become unstable. Therefore, for both practical and performance reasons, the value $\beta_1 = 2^{-10}$ was chosen for implementation.

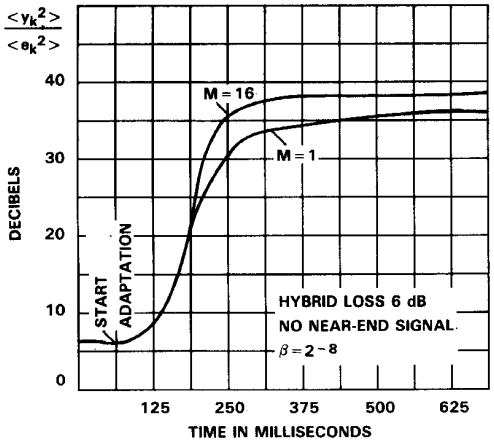


Figure 13
Convergence Performance of the Block Update Algorithm and Stochastic Gradient Algorithm

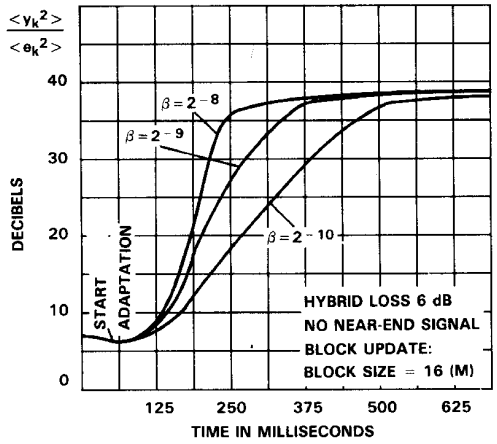


Figure 14
Convergence Performance of the Block Update Algorithm

In the TMS32020 implementation, it is convenient and desirable to normalize both the stepsize and the error variables u(i) by the square root of the power estimate $P_y(i)$, i.e., $L_y(i)$ of (9).

Normalizing u(i) and the stepsize separately enables the product term of (7) to be computed with single precision on the TMS32020 without significant loss of precision or overflow due to varying signal level.

Table 3 gives a description of the program variables together with their names and ranges, and summarizes the number formats chosen for the echo canceller implementation. One of the most important aspects of the implementation approach is the handling of the binary representation of the signal samples, algorithm variables, coefficients, and constant parameters for various stages of the processing. The notation (Q.F) is used to define the representation of either 16-bit numbers or 32-bit accumulator numbers, where F specifies the number of bits which are to the right of the implicit binary point. The assignments of Table 3 ensure that the algorithm can be executed on the TMS32020 with single-precision arithmetic and with no significant loss of precision.

Memory Requirements

The echo canceller algorithm requires the storage of both reference samples and variable coefficients in on-chip data RAM so that the required FIR and block update convolution can be performed efficiently using the RPTK and MACD instructions. Therefore, the coefficients a_k are stored in block B0, which is configured as program memory. The 16 normalized error samples for coefficient updating are also stored in B0. The 128 reference signal samples y(i) are stored in data RAM along with an additional 16 reference samples y(1 – 129), ..., y(i – 143), which are used in the update of coefficients a_{112}, ..., a_{127}. The echo canceller data memory locations are summarized in Table 4.

Software Logic and Flow

A flowchart of the TMS32020 program for a 128 – tap digital voice echo canceller is shown in Figure 15.

In Table 5, the instruction cycle and memory requirements are listed for the various blocks of the program implementation. The blocks are listed in the order of execution.

Table 3. Algorithm Number Representation on the TMS32020

VARIABLE	DESCRIPTION	BINARY REPRESENTATION	RANGE
$a_0, a_1,\ldots,a_{127}$	Filter coefficients	(Q.15)	$[-1, 1-2^{-15}]$
$y(i), y(i-1),\ldots,y(i-143)$	Reference samples	(Q.0)	$[-2^{15}, 2^{15}-1]$
$s(i)$	Near-end signal	(Q.0)	$[-2^{15}, 2^{15}-1]$
$r(i)$	Echo estimate	(Q.0)	$[-2^{15}, 2^{15}-1]$
$L_y(i)$	Average absolute value of $y(k)$	(Q.0)	$[0, 2^{15}-1]$
$L_y(i)^{-1}$		(Q.15)	$[-1, 1-2^{-15}]$
$u(i)$	Near-end signal minus echo estimate $s(k) - r(k)$	(Q.0)	$[-2^{15}, 2^{15}-1]$
$un(i),\ldots,un(i-15)$	Normalized outputs $un(i) = u(i) \times L_y(i)^{-1}$	(Q.15)	$[-1, 1-2^{-15}]$
$\tilde{s}(i)$	Short-time average of $2 \times \lvert s(i)\rvert$	(Q.0)	$[0, 2^{15}-1]$
$\tilde{y}(i)$	Short-time average of $\lvert y(i)\rvert$	(Q.0)	$[0, 2^{15}-1]$

Table 4. Echo Canceller Data Memory Locations

VARIABLE	SYMBOL	LOCATION	REMARK
$a_0,\ldots,a_{127}$	A0,...,A127	Block B0 767,766,...,640	A0 is in higher address
$y(k),\ldots,y(k-143)$	Y0,...,Y143	Block B1 768,769,...,911	Y128,...,Y143 required for block update
$un(k,\ldots,un(k-15)$	UN0,...,UN15	Block B0 512,...,527	

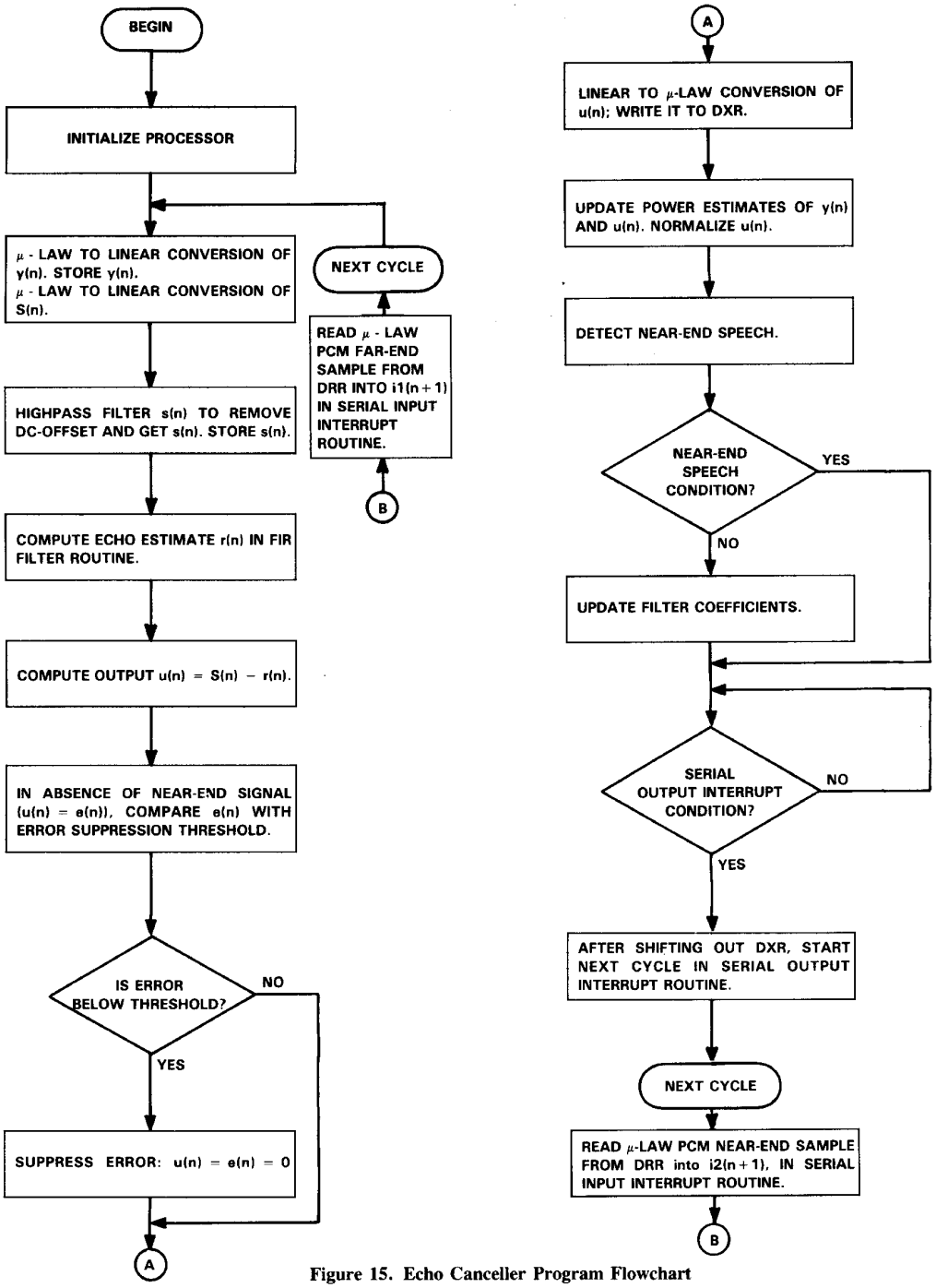


Figure 15. Echo Canceller Program Flowchart

Table 5. Program Module Requirements

STEP	MODULE FUNCTION	CODE LISTING PAGE	DESCRIPTION	CPU CYCLES	PROGRAM MEMORY LOCATIONS	DATA* MEMORY LOCATIONS
1.	Cycle Start Routine	7	μ-law to linear conversions; take absolute value of inputs and high-pan filter s(i).	32	28	11
2.	Echo Estimation Routine	9	FIR convolution of reference samples and filter coefficients to get echo replica r(i).	156	14	258
3.	Compute Output	9	u(i) = s(i) − r(i) Store u)i).	6	6	2
4.	Residual Output Suppression Routine	10	If output power below threshold, set u(i) = 0.	12	15	4
5.	Linear to μ-law Compression Routine	11	Convert u(i) to μ-law.	26	35	4
6.	Power Estimation Routine	13	Estimate short-term power of u(i) and y(i).	28	14	6
7.	Output Normalization	14	Comput $u_n(i) = \frac{u(i)}{y(i)}$ and clip it.	28	25	19
8.	Near-end Speech Detection	16	Perform maximum test for near-end speech.	54	74	16
9.	Coefficient Increment Update Routine	20	If no near-end speech, compute increments for coefficient group.	183	63	26
10.	Coefficient Update Routine	23	Add increments to coefficient group.	43	43	2
11.	Cycle End Routine	25	Wait for interrupt.	1	3	0
12.	Receive Interrupt Service Routine	25	Save status and read input sample.	2 × 14	14	3
13.	Transmit Interrupt Service Routine	25	Branch to start.	2	2	0
14.	Interrupt Branches	3		12	6	0
15.	Processor Initialization**	4	Clear memory, initialize status and set parameters.	86 **	86	0
16.	μ-law to Linear Conversion Table*	26		0	256	0
Total				614	676	351

*Locations are entered only for the routine that uses them first.

**Not in main cycle; CPU cycles not counted in total.

The program loop is executed once per I/O data sample period of 125 μs. The program loop is interrupt-driven from the output data sample mark of a T1 frame. Depending on the near-end speech detector/hangover status, the coefficient update computation module may be skipped. An input data sample interrupt mark occurs during the program loop at a time dependent on the channel location within the T1 frame. In response to the interrupt, the main program execution is interrupted and saved until the new input samples have been read into memory. At the end of each program loop, the processor waits for the next output sample interrupt.

In the following subsections, the implementation of each major block is described in detail. Each variable used in an equation is referred to by its name in the program enclosed in parentheses.

Cycle Start Routine

The voice echo canceller program has been implemented with either μ-law or A-law conversion routines as a program option.

The μ-law (or A-law) to linear input conversion routine is implemented by table lookup in order to minimize the number of instructions. The 256 14-bit two's-complement number corresponding to the 256 possible 8-bit μ-law numbers are stored in program memory. The 8 bits of the μ-law number specify the relative address of the corresponding linear number in the table, which is added to the first address in the table to form the absolute program memory address for the linear number. The TBLR instruction is then used to move the number from program to data memory.

In the cycle start routine, the μ-law input reference sample is read from memory location DRR2 and converted to its linear representation y(i) (Y0). Its absolute value is also stored in location ABSY0. The near-end input sample is then read and converted to a linear representation sdc(i) (S0DC). The sample s(i) is next put through a highpass filter to remove any residual dc offset. The highpass filter is a first-order filter with a 3 – dB frequency at 160 Hz. Its output s(i) (S0) is given by

$$s(i+1) = (1-\gamma)\, s(i) + \frac{1}{2}\,(1-\gamma)\,(sdc(i) - sdc(i-1)) \quad \text{where } \gamma = 2^{-3}. \tag{16}$$

Note that the filter implementation requires double-precision arithmetic, with S0 denoting the MSBs of s(i) and S0LSBS its LSBs.

Echo Estimation

The echo estimate $\hat{f}(i)$ (EEST) is formed by convolving the tap weight coefficients $a_0, \ldots, a_{127}$ (A0, ..., A127) with the 128 most recent reference samples y(i), ..., y(i – 127) (Y0, ..., Y127).

$$\hat{f}(i) = \sum_{k=0}^{127} a_k\, y(i-k) \tag{17}$$

This operation is most efficiently implemented on the TMS32020 using the RPTK and MACD instruction. The samples y(i),..., y(i – 127) are stored in block B1 of data memory while $a_0,\ldots, a_{127}$ are stored in block B0 configured as program memory. Since the MACD instruction also performs a data move,

$$y(i-k+1) \rightarrow y(i-k) \quad \text{for } k = 1,\ldots,128 \tag{18}$$

no data shifting is required for the computation of the next echo estimate.

The block update routine used for the coefficient adaptation requires the storage of y(i – 128), ..., y(i – 143) (Y128, ..., Y143) in addition to the most recent 128 samples used in the convolution. Since these samples are not used in the convolution, they are updated using the RPTK and DMOV instructions.

$$y(i-k+1) \rightarrow y(i-k) \quad \text{for } k = 129,\ldots,143 \tag{19}$$

The tap weight coefficients a_0, ..., a_{127} are initially set to zero, and are adjusted by the algorithm to converge to the impulse response of the echo path h_0, ..., h_{127}.

$$a_k(i) \rightarrow h_k \quad \text{for } k = 0,\ldots,127 \tag{20}$$

The $|h_k| < 1, \forall$ k,because the power gain of the echo path is smaller than unity. The binary representation for the a_k's was chosen to be of the form (Q.15) with 15 bits after the binary points. This format represents a number between -1 and $(1 - 2^{-15})$. The reference samples and the echo estimate are represented as 16-bit two's-complement integers (no binary point). The 32-bit result of the convolution is therefore of the form (Q.15), and the 16 bits of the echo estimate are the MSB of accumulator low (ACCL) and the 15 LSBs of accumulator high (ACCH). One left shift of the accumulator is required before ACCH is stored in EEST.

Residual Error Suppression

The residual cancellation error is set to zero (or suppressed) whenever the ratio of a long-time average of the absolute value of the output (ABSOUT) to a long-time average of the absolute value of the reference signal (ABSY) is smaller than a fixed threshold. The two long-time averages are updated subsequently in the program as described below. The suppression is, of course, disabled when a near-end speech signal is present (HCNTR > 0). The suppression threshold is set at 1/16 or −24 dB.

Linear to μ-Law (A-Law) Conversion

The linear to μ-law (A-law) conversion routine is an efficient adaptation to the TMS32020 of the conversion routine written for the TMS32010 and described in the application report, "Companding Routines for the TMS32010."[9]

Signal and Output Power Estimation

An estimate of the long-time average of $|u(i)|$ is required by the residual error suppression routine. This estimate $L_u(i)$ (ABSOUT) is obtained by lowpass filtering $|u(i)|$ (ABSU0) using the following infinite impulse response (IIR) filter:

$$L_u(i+1) = (1-\alpha)\, L_u(i) + \alpha|u(i)| \tag{21}$$

where $\alpha = 2^{-7}$. In terms of the program variables, the IIR filter is given by

$$\text{ABSOUT} = 2^{-16}\left(2^{16} \times \text{ABSOUT} - 2^{9} \times \text{ABSOUT} + 2^{9} \times \text{ABSU0}\right) \tag{22}$$

Similarly, the estimate $L_y(i)$ (ABSY) of the long-term average of y(i) (ABSY0) is the output of an IIR filter with the same α, but differs from the above filter by the addition of a cutoff term that prevents the estimate from taking values smaller than a desired level.

$$\text{ABSY} = 2^{-16}\,(2^{16} \times \text{ABSY} - 2^{9} \times \text{ABSY} + 2^{9} \times \text{ABSY0} + 2^{9} \times \text{CUTOFF}) \tag{23}$$

This insures that ABSY $\geq$ CUTOFF even if ABSY0 is zero for a long time.

Since $L_y(i)$ is used to normalize the algorithm stepsize, this feature is important in order to prevent excessively large stepsizes when the far-end talker is silent.

The stepsize is normalized according to

$$2\beta(i) = \frac{\beta_1}{L_y^2(i)} \tag{24}$$

In order to avoid double-precision arithmetic, this normalization is carried out in two stages (as described in the subsection on coefficient adaptation). Each of the stages requires a division by $L_y(i)$. It is more efficient to compute $L_y(i)^{-1}$ (IABSY) and replace the divisions by two multiplications.

Since ABSY is a positive integer, taking its inverse consists simply of repeating the SUBC instruction. IABSY is a positive fractional number of the form (Q.15), taking values between 0 and $1-2^{-15}$.

Output Normalization

The normalized output $u_n(i)$ (UN0) is defined as $\mu(i)/L_y(i)$ and replaces the actual error in the coefficient update routine for finite-precision considerations, described in the subsection on coefficient adaptation. In the absence of near-end speech, $u_n(i)$ is equal to a normalized cancellation error and is used in the coefficient update. In the presence of near-end speech, no coefficient update is carried out, and the normalized outputs are not used.

The block update approach requires the storage of the 16 most recent normalized outputs $u_n(i), \ldots, u_n(i-15)$ (UN0,..., UN15). In a given program cycle, only $u_n(i)$ is computed and stored, while $u_n(i-1),\ldots, u_n(i-15)$ computed in previous program cycles are only updated using the DMOV instruction.

$$u_n(i-k+1) \rightarrow u_n(i-k) \quad \text{for } k = 1,\ldots,14 \tag{25}$$

In the absence of near-end speech, the normalized output should be a number smaller than one, which is represented as a (Q.15) fraction. To insure that the representation is adequate even in the presence of a near-end signal, the normalized output is clipped at +1 or -1, i.e.,

$$\begin{aligned} &\text{if } u_n(i) > 1.0, \text{ then } u_n(i) = 1.0 \\ &\text{if } u_n(i) < -1.0, \text{ then } u_n(i) = -1.0 \end{aligned} \tag{26}$$

Near-End Speech Detection

Near-end speech is declared if

$$\bar{s}(i) \geq \max\left(\tilde{y}(i),\tilde{y}(i-1),\ldots,\tilde{y}(i-127-h(i))\right) \tag{27}$$

where $\bar{s}(i)$ (ABSS0F) is the output of a lowpass filter with input $2 \times |s(i)|$ (ABSS0). The variable $\tilde{y}(i)$ is a lowpass filtered version of $|y(i)|$, and h(i) (H) a modulo-16 counter. The lowpass filters are IIR filters with short-time constants,

$$s(i+1) = (1-\alpha)\, s(i) + \alpha \times 2 \times |s(i)| \tag{28}$$

$$y(i+1) = (1-\alpha)\, y(i) + \alpha \times |y(i)| \tag{29}$$

where $\alpha = 2^{-5}$.

The counter h(i) is incremented by one for every input sample. The routines maintain nine partial maxima m0, m1,..., m8 (M_0, M_1, ..., M_8), defined at time $i = 16m + h(i)$ by

$$\begin{aligned} m_0(i) &= \max\left((\tilde{y}(i),\ldots,\tilde{y}(i-h(i)+1)\right) \\ m_1(i) &= \max\left((\tilde{y}(i-h),\ldots,\tilde{y}(i-h(i)-15)\right) \\ &\vdots \\ m_8(i) &= \max\left(\tilde{y}(i-h-112),\ldots,\tilde{y}(i-h(i)-127)\right) \end{aligned} \tag{30}$$

Figure 16 illustrates how the partial maxima are maintained.

The condition for near-end speech declaration is then equivalent to

$$\bar{s}(i) \geq \max(m_0,\ldots,m_8) \tag{31}$$

The partial maxima are updated according to the following recursions:

$$\begin{aligned} &\text{if } h = 0, \text{ then } m_0(i) = \tilde{y}(i+1) \\ &\quad \text{and } m_j(i) = m_{j-1}(i) \\ &\quad \text{where } j = 1,\ldots,8 \end{aligned} \tag{32}$$

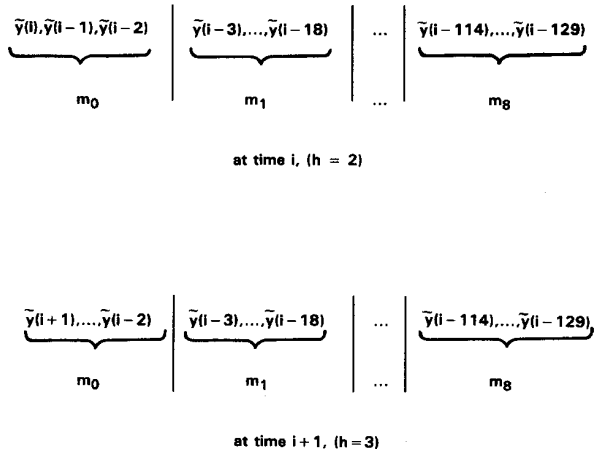


Figure 16. Partial Maxima for Near-End Speech Detection

and

if $0 < h \leq 15$, then $m_0(i+1) = \max(m_0(i), \tilde{y}(i+1))$
and $m_j(i+1) = m_j(i)$
where $j = 1,\ldots,8$

If near-end speech is declared, a hangover counter (HCNTR) is set equal to a hangover time (HANGT), which was chosen to be 600 samples or 75 ms. If no near-end speech is declared, then the hangover counter is decremented by one, unless it is zero. If the hangover counter is larger than zero, then the coefficient update routine is skipped. Moreover, if the reference signal power estimate $L_y(i)$ is smaller or equal to the cutoff value of − 48 dB, then adaptation is also disabled to avoid divergence during long silences of the far-end talker.

Coefficient Adaptation

The 128 coefficients of the transversal filter are divided into 16 groups of 8 coefficients each, as shown in Table 6.

Table 6. The Coefficient Groups

GROUP	COEFFICIENTS
0	$a_0, a_{16}, a_{32}, \ldots, a_{112}$
1	$a_1, a_{17}, a_{33}, \ldots, a_{113}$
.	.
.	.
.	.
15	$a_{15}, a_{31}, a_{47}, \ldots, a_{127}$

The coefficients in only one of the groups are updated in a given program cycle, while the other coefficients are not modified. A modulo-16 counter h(i) (H) points to the index of the group to be updated, and is incremented by one during every program cycle.

The update equation is repeated here for ease of reference.

$$a_k(i+1) = a_k(i) + \frac{\beta_1}{(L_y(i))^2} \sum_{m=0}^{15} e(i-m)\, y(i-k-m) \tag{34}$$

for k = h, h + 16, ..., h + 112, where h is the value of the counter and goes from 0 to 15. The error terms e(i − m) (m = 0, ..., 15) are the most recent cancellation errors. In this case, the errors are equal to the 15 most recent canceller outputs u(i), ..., u(i − 15) since the adaptation is carried out only in the absence of a near-end signal.

For finite-precision considerations, the actual implementation of the update equation by the routine is carried out in the following two main steps:

1. Compute eight partial updates:

$$\gamma_k(i) = \sum_{m=k}^{k+15} \frac{u(i-m)}{L_y(i)}\, y(i-k-m) \tag{35}$$

where k = h, h + 16, ..., h + 112.

The normalized outputs $u_n(i), \ldots, u_n(i-15)$ have already been computed and stored.

2. Update the coefficients:

$$a_k(i+1) = a_k(i) + \left(2^4 \times \left(L_y(i)^{-1} \times 2^G\right) \times \gamma_k(i)\right) 2^{-16} \quad (36)$$

where G (GAIN) is a program parameter that determines the stepsize of the algorithm and has the value 0, 1, 3, ..., 15.

The partial updates $\gamma_k(i)$ are computed using the MAC instruction in repeat mode. The result is rounded and stored in temporary locations INC0, ..., INC0 + 7 in block B1.

For the second step of the update, $L_y(i)^{-1}$ (IABSY) is first loaded in the T register with a left shift of G (GAIN). It is then multiplied by each of the $\gamma_k(i)$'s. SPM is set to 2 to implement the 2^4 multiplication by shifting the P register four positions to the right before adding it to the accumulator (APAC).

Interrupt Service Routines

At the end of the cycle, the program becomes idle until a receive interrupt occurs followed by a transmit interrupt that sends it back to the beginning of the cycle. The transmit interrupt routine simply enables interrupts and branches back to the start. The receive interrupt must store the status register ST0 and the accumulator, then read the received sample from DRR, zero its eight most significant bits, and store it in DRR1. It restores the accumulator and status register ST0 before returning to the main program.

External Processor Hardware Requirements

Very little external hardware is required to implement a complete single-channel 128-tap echo canceller with the TMS32020. In addition to the processor, only two external 1K x 8 PROMs and some system-dependent interface logic are required. A typical interface circuit for the demonstration system is shown in Appendix A.

The TMS32020 serial I/O ports allow direct interfacing of the echo canceller to a digital T1 carrier data stream.

Three I/O functions must be performed during each T1 frame (125 μs). The far-end and the near-end signals must be read in, and the processed near-end signal must be written out. To perform these functions, a timing circuit must extract the T1 clock and the T1 frame marks for each direction of transmission. The timing circuit uses the frame mark to generate a channel mark that selects the desired channel out of the 24 present in the T1 frame. The channel mark goes to a high level during the clock cycle, immediately preceding the eight serial bits of the desired sample.

The T1 clock, channel mark, and serial data signals are directly input into the TMS32020 serial clock (CLKR), serial input control (FSR), and serial input port (DR), respectively. Because data is read in from two directions of transmission, a triple two-to-one multiplexer (e.g., SN74LS157) is required to select one of the two sets of T1 signals to be input into the TMS32020. During each T1 frame, the multiplexer alternates once between each direction of transmission, under the control of the timing circuit.

Since data is written out in only one direction, the TMS32020 serial output port (DX) is directly tied to the outgoing T1 data line. The serial output clock (CLKX) and the serial output control (FSX) signals are the same as the near-end direction-of-transmission CLKR and FSR signals. If the far-end T1 channel-frame location overlaps the near-end T1 channel location in time, it is necessary to delay each far-end sample external to the TMS32020 to permit it to be read following the sample from the near-end direction. This requires an eight-bit serial shift register and some additional timing circuits.

Description of a Single-Channel Demonstration System

The demonstration system has been constructed in order to verify the TMS32020 implementation. Two photographs and a block diagram of the demonstration system are provided.

Figure 17 is a photograph of the front panel of the demonstration system, and Figure 18 is a closeup photograph of the single-channel echo canceller module.

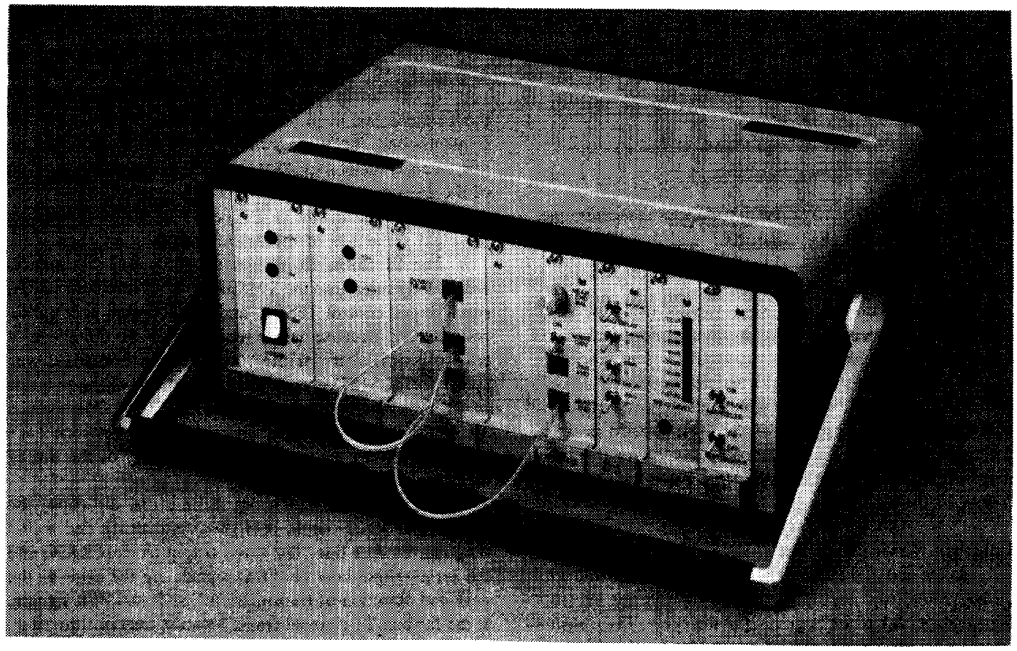


Figure 17. Front Panel of the Echo Canceller Demonstration System

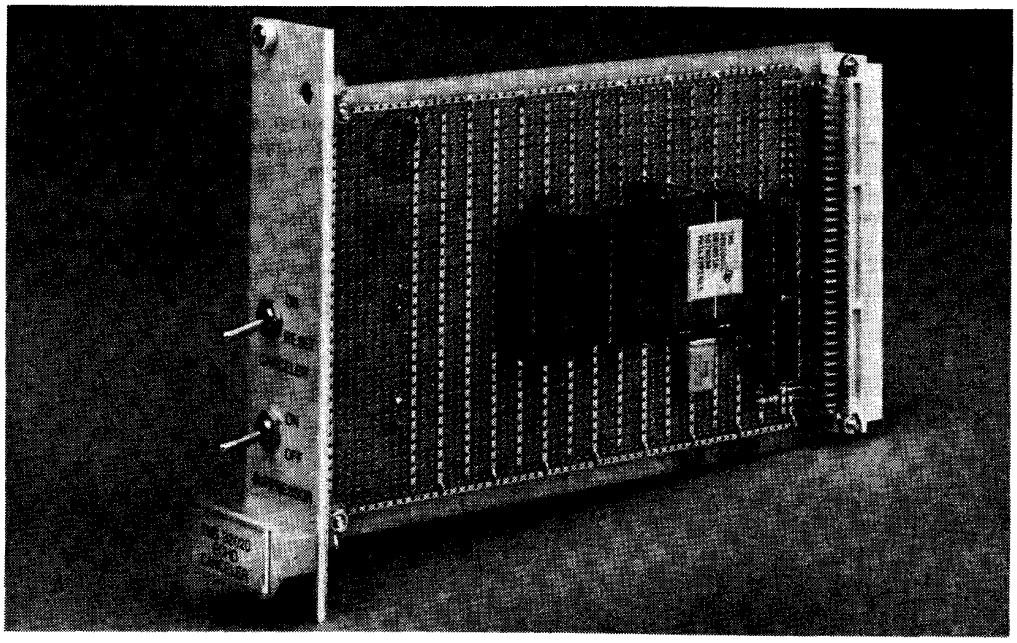


Figure 18. Single-Channel Echo Canceller Module

As shown in the block diagram of Figure 19, the demonstration system models two end offices, a delay due to a satellite link, a delay due to a terrestrial link, a typical end-loop line response, and the echo canceller. A phone is connected via a two-wire interface to each end of the path. The two-wire interfaces are converted to four-wire in electronic hybrids. The hybrids also provide the required battery voltage to power the phones. The near-end two-wire line has a series-passive line simulator. The associated hybrid has an adjustable termination to allow a variable amount of hybrid mismatch, and therefore a variable amount of near-end echo response.

At each end, the four-wire analog signal is converted to and from PCM μ-law digital representation by a codec. The PCM signaling is done in a T1 format, with appropriate timing provided by a central timing generator. Variable delay is provided in the near-end and far-end path by digital memories. The TMS32020 echo canceller is situated in the middle of the path, with signal processing done on the near-end to far-end direction of transmission. The other direction is used as the reference signal. All the TMS32020 signal I/O is performed using the T1 format. A display of the processed signal is used as an indicator of echo suppression in the absence of near-end signal. To aid the testing of the echo canceller, the far-end phone can be switched out and a noise generator switched in as a source of far-end signal.

The performance of the TMS32020 echo canceller was measured for white-noise input, as suggested in the CCITT G.165 recommendation. The measurement results are summarized in Table 7 and show that the TMS32020 echo canceller performance exceeds the CCITT requirements in all the tests described. The subjective performance on speech was also found to be very good in both singletalk and doubletalk modes, with no audible distortion of the signal.

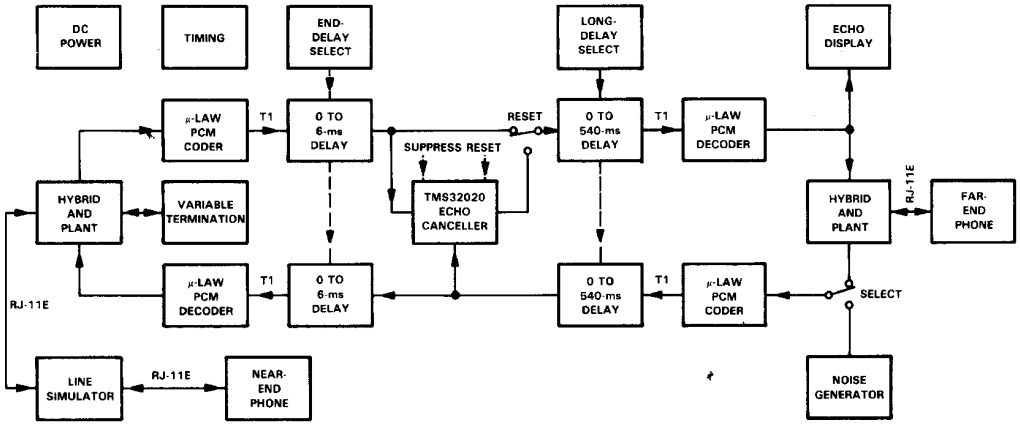


Figure 19. Block Diagram of Echo Canceller Demonstration System

Table 7. TMS32020 Echo Canceller Performance

TEST DESCRIPTION	CCITT G.165 PERFORMANCE REQUIREMENT	TMS32020 ECHO-CANCELLER PERFORMANCE
1. Final echo return loss after convergence; singletalk mode	−40 dbm0	< −48 dbm0
2. Convergence rate; singletalk mode	≥ 27 dB	> 38 dB
3. Leak rate	≤ 10 dB	≈ 0 dB
4. Infinite return loss convergence	−40 dbm0	< −48 dbm0

CONCLUSION

The development of novel variations of the generic least-mean-squared (LMS) echo cancelling algorithm and the near-end speech and residual suppression control algorithms has resulted in the implementation of a complete 128-tap single-channel echo canceller on a single TMS32020 programmable Digital Signal Processor. The echo canceller performance exceeds all requirements of the CCITT G.165 recommendations and the performance of similar currently available products. The only external hardware required are two program PROMs and a serial data multiplexer. A direct T1-rate serial interface is available to minimize component count in four-wire VF and T1 carrier configurations.

The single-channel TMS32020 echo canceller program provides a high-performance building block for low-cost systems, which can be tailored to a wide variety of system applications. Programmability offers the flexibility to implement custom requirements, such as cascaded sections for longer tail delay range, short-range multichannel versions, or other special-purpose functions.

The echo canceller application illustrates the power and versatility of the TMS32020 single-chip programmable signal processor. Applications of this technology can be expected to benefit many other complex signal processing tasks in communications products, including voiceband data modems, voice codecs, digital subscriber transceivers, and TDM/FDM transmultiplexers.

ACKNOWLEDGEMENTS

Texas Instruments and Teknekron Communications Systems wish to acknowledge the fine work done by the Teknekron project team comprised of Dr. David Messerschmitt, Dr. David Hedberg, Mr. Christopher Cole, Dr. Amine Haoui, and Mr. Peter Winship. Special appreciation is extended to Peter Winship for his excellent work in the design and construction of the prototype demonstration system and to James Hesson for his helpful suggestions on the interleaved, coefficient update technique.

Further information about the echo canceller applications may be obtained by contacting Texas Instruments or Teknekron Communications Systems, 2121 Allston Way, Berkeley, CA 94704, (415) 548-4100.

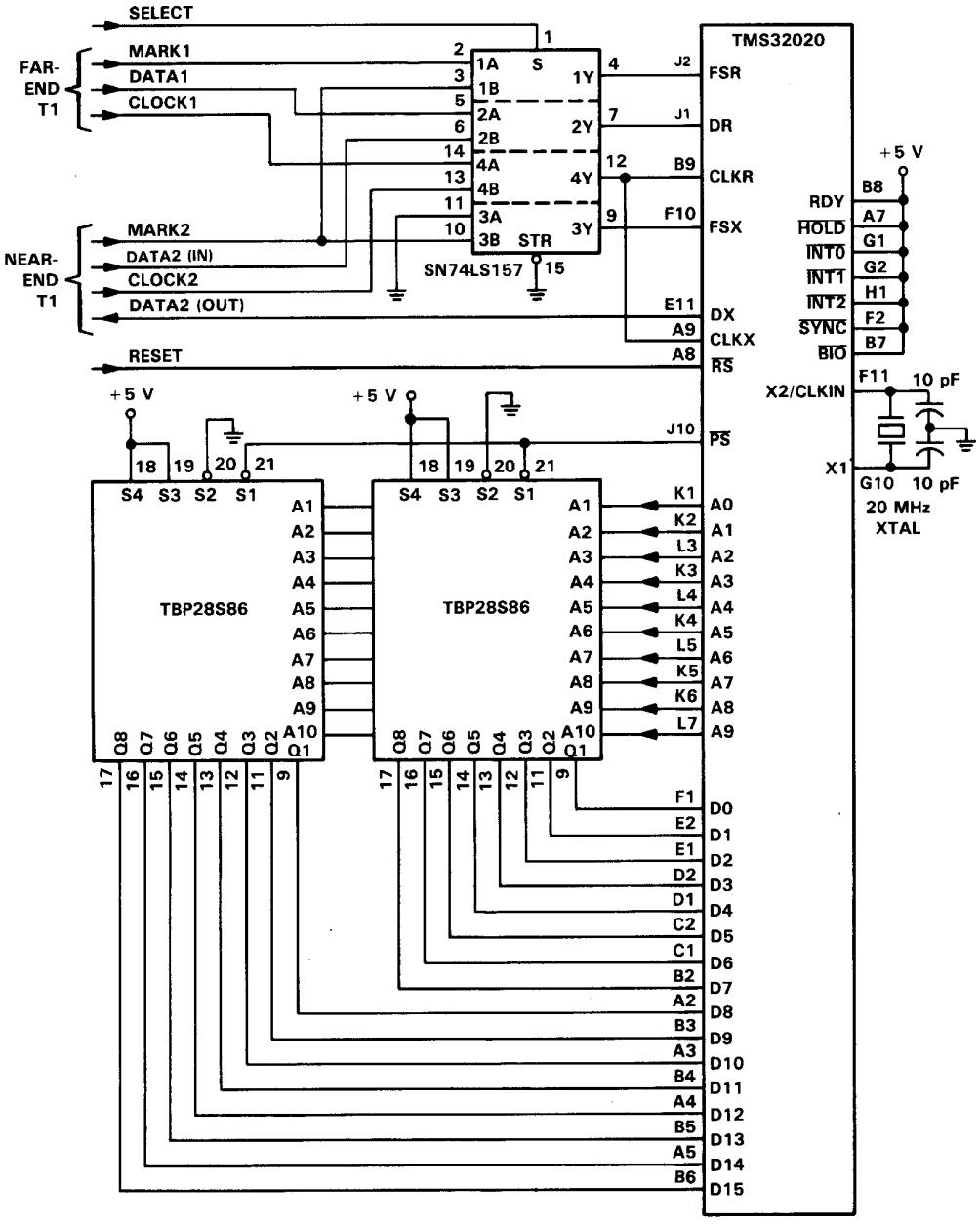
Note that Texas Instruments does not warrant or guarantee the applicability of this application report to any particular design or customer use.

REFERENCES

1. M.L. Honig and D.G. Messerschmitt, *Adaptive Filters,* Kluwer Academic Publishers (1984).
2. D.L. Duttweiler and Y.S. Chen, "A Single-Chip VLSI Echo Canceller," *Bell System Technical Journal,* **Vol** 59, No. 2, 149 (February 1980).
3. "Recommendation G.165, Echo Cancellers," *CCITT,* Geneva (1980).
4. H.R. Huntley, "Transmission Design of Intertoll Telephone Trunks," *Bell System Technical Journal,* **Vol** 32, No. 5, 1019-36 (September 1953).
5. J.C. Bellamy, *Digital Telephony*, Wiley & Sons (1982).
6. M.J. Gingell, B.G. Hay, and L.D. Humphrey, "A Block Mode Update Echo Canceller Using Custom LSI," *GLOBECOM Conference Record,* **Vol** 3, 1394-97 (November 1983).
7. D.G. Messerschmitt, "Echo Cancellation in Speech and Data Transmission," *IEEE Journal on Selected Topics in Communications,* SAC-2, No. 2, 283-303 (March 1984).
8. D.L. Duttweiler, "A Twelve-Channel Digital Voice Echo Canceller," *IEEE Transactions on Communications,* COM-26, No. 5, 647-53 (May 1978).
9. L. Pagnucco and C. Ershine, "Companding Routines for the TMS32010/TMS32020," *Digital Signal Processing Applications with the TMS320 Family*, Texas Instruments (1986).

APPENDIX A

HARDWARE SCHEMATIC OF THE SINGLE-CHANNEL DEMONSTRATION PROCESSOR



APPENDIX B

SOURCE CODE LISTING

```
0001            ************************************************************
0002            *
0003            *
0004            *               128-TAP ECHO-CANCELLER PROGRAM
0005            *               (C) COPYRIGHT TEXAS INSTRUMENTS INC., 1985
0006            *
0007            ************************************************************
0008 0000
0009                    IDT 'EC128'
0010 0000
0011            *
0012            *       ALGORITHM CONSTANTS
0013            *
0014 0000
0015      0003  GAIN    EQU     3               * COEFF UPDDATE GAIN = 2**-10
0016            *                               *   = 2**(GAIN - 13)
0017      0009  LTAU    EQU     9               * LPF LONG TAU = 16 MSECS
0018            *                               *   = 125 USECS * 2**(16-LTAU)
0019      000B  STAU    EQU     11              * LPF SHORT TAU = 4 MSECS
0020            *                               *   = 125 USECS * 2**(16-STAU)
0021      000D  HTAU    EQU     13              * HPF TAU = 1MSEC = 1/2*PI*170HZ
0022            *                               *   = 125 USECS * 2**(16-HTAU)
0023      0800  THRES0  EQU     >800            * SUPRESS THRES = 1/16 (-24DB)
0024            *                               *   = 2**-THRES0 (-6DB * THRES0)
0025      0258  HANGT0  EQU     600             * HANG OVER TIME = 75 MSECS
0026            *                               *   = 125 USECS * HANGT0
0027      0001  NER     EQU     1               * NEAR END SPEECH THRES = -6DB
0028            *                               *   = -6DB * NER
0029      0021  CUTOF0  EQU     >21             * UPDATE CUTOFF = -48DB
0030            *                               *   = -6DB * (13 - LN(CUTOF0))
0031 0000
0032            *
0033            *       PAGE 0 DATA MEMORY ALLOCATION
0034            *
0035 0000
0036      0000  PODM    EQU     0               * PAGE 0 DATA MEM ADRS
0037 0000
0038      0000  DRR     EQU     0               * SERIAL PORT DATA RECEIVE REG
0039      0001  DXR     EQU     1               * SERIAL PORT DATA TRANSMIT REG
0040      0002  TIM     EQU     2               * TIMER REG (NOT USED)
0041      0003  PRD     EQU     3               * PERIOD REG (NOT USED)
0042      0004  IMR     EQU     4               * INTERRUPT MASK REG
0043      0005  GREG    EQU     5               * MEM ALLOCATION REG (NOT USED)
0044 0000
0045      0060  DRR1    EQU     96              * STORAGE FOR S(0) BY RINT
0046      0061  DRR2    EQU     97              * STORAGE FOR Y(0) BY RINT
0047      0062  TST0    EQU     98              * STORAGE FOR ST0 BY RINT
0048      0063  TACCH   EQU     99              * STORAGE FOR ACC HIGH BY RINT
0049      0064  TACCL   EQU     100             * STORAGE FOR ACC LOW BY RINT
0050 0000
0051      0065  BADDR   EQU     101             * BASE ADR FOR MU-LAW EXPANSION
0052      0066  BIAS2   EQU     102             * BIAS FOR MU-LAW COMPRESSION
0053      0067  NEG7    EQU     103             * -7 FOR MU-LAW COMPRESSION
0054      0068  Q       EQU     104             * MU-LAW COMPRESSION MANTISSA
0055      0069  SI      EQU     105             * MU-LAW COMPRESSION SIGN
0056 0000
0057      0068  TEMP1   EQU     104             * TEMPORARY STORAGE LOCATION 1
```



```
0058      0069  TEMP2   EQU     105             * TEMPORARY STORAGE LOCATION 2
0059 0000
0060      006B  EEST    EQU     107             * ECHO ESTIMATE
0061      006C  OUTPUT  EQU     108             * OUTPUT ESTIMATE
0062      006D  THRES   EQU     109             * RESIDUAL OUTPUT SPRS THRESHOLD
0063 0000
0064      006E  ONE     EQU     110             * HOLDS 1
0065 0000
0066      006F  ADY142  EQU     111             * Y142 DATA MEM ADRS
0067 0000
0068      0070  S0      EQU     112             * NEAR-END SAMPLE MSBS
0069      0071  S0LSBS  EQU     113             * NEAR-END SAMPLE LSBS
0070      0072  S0DC    EQU     114             * INPUT NEAR-END SAMPLE (K=0)
0071      0073  S1DC    EQU     115             * INPUT NEAR-END SAMPLE (K=1)
0072 0000
0073            *
0074            *       PAGE 4 DATA MEMORY ALLOCATION
0075            *
0076 0000
0077      0200  P4DM    EQU     512             * PAGE 4 DATA MEM ADRS
0078      FF00  P4PM    EQU     65280           * PAGE 4 PROG MEM ADRS
0079 0000
0080      0000  UN0     EQU     0               * NORMALIZED OUTPUT (K=0)
0081      000F  UN15    EQU     15              * NORMALIZED OUTPUT (K=15)
0082 0000
0083            *
0084            *       PAGE 5 DATA MEMORY ALLOCATION
0085            *
0086 0000
0087      0280  P5DM    EQU     640             * PAGE 5 DATA MEM ADRS
0088      FF80  P5PM    EQU     65408           * PAGE 5 PROG MEM ADRS
0089 0000
0090      0000  A127    EQU     0               * FIR FILTER COEFFICIENT (K=127)
0091      007F  A0      EQU     127             * FIR FILTER COEFFICIENT (K=0)
0092 0000
0093            *
0094            *       PAGE 6 DATA MEMORY ALLOCATION
0095            *
0096 0000
0097      0300  P6DM    EQU     768             * PAGE 6 DATA MEM ADRS
0098 0000
0099      0000  Y0      EQU     0               * REFERENCE SAMPLE (K=0)
0100      007F  Y127    EQU     127             * REFERENCE SAMPLE (K=127)
0101 0000
0102            *
0103            *       PAGE 7 DATA MEMORY ALLOCATION
0104            *
0105 0000
0106      0380  P7DM    EQU     896             * PAGE 7 DATA MEM ADRS
0107 0000
0108      0000  Y128    EQU     0               * REFERENCE SAMPLE (K=128)
0109      000F  Y143    EQU     15              * REFERENCE SAMPLE (K=143)
0110 0000
0111      0010  TEMP3   EQU     16              * TEMPORARY STORAGE LOCATION 3
0112 0000
0113      0011  CUN0    EQU     17              * COPY OF UN0 FROM PAGE 4
0114 0000
```

```
0115      0012  AONE   EQU    18              * HOLDS 1
0116      0013  SONE   EQU    19              * HOLDS SATURATION 1
0117 0000
0118      0014  ADA0   EQU    20              * A0 DATA MEM ADRS
0119      0015  ADY1   EQU    21              * Y1 DATA MEM ADRS
0120      0016  ADINC0 EQU    22              * INC0 DATA MEM ADRS
0121      0017  ADM7   EQU    23              * M7 DATA MEM ADRS
0122      0018  ADUN14 EQU    24              * UN14 DATA MEM ADRS
0123 0000
0124      0060  H      EQU    96              * MODULO 16 COUNTER
0125 0000
0126      0061  HANGT  EQU    97              * HANG OVER COUNTER RESET VALUE
0127      0062  HCNTR  EQU    98              * HANG OVER COUNTER
0128 0000
0129      0063  ABSS0  EQU    99              * |S0|
0130      0064  ABSS0F EQU    100             * SHORT TAU LPF 2*|S0|
0131 0000
0132      0065  ABSE0  EQU    101             * |OUTPUT|
0133      0066  ABSOUT EQU    102             * LONG TAU LPF |OUTPUT| MSBS
0134      0067  AELSBS EQU    103             * LONG TAU LPF |OUTPUT| LSBS
0135 0000
0136      0068  ABSY0  EQU    104             * |Y0|
0137      0069  ABSY   EQU    105             * LONG TAU LPF |Y0| MSBS
0138      006A  AYLSBS EQU    106             * LONG TAU LPF |Y0| LSBS
0139      006B  IABSY  EQU    107             * 1/ABSY
0140      006C  CUTOFF EQU    108             * ABSY CUTOFF LVL FOR NO UPDATE
0141      006D  ABSY0F EQU    109             * SHORT TAU LPF |Y0|
0142 0000
0143      006E  M0     EQU    110             * LOCAL MAXIMA (K=0)
0144      0076  M8     EQU    118             * LOCAL MAXIMA (K=8)
0145 0000
0146      0078  INC0   EQU    120             * UPDATE INCREMENT (K=0)
0147      007F  INC7   EQU    127             * UPDATE INCREMENT (K=7)
0148 0000
0149 0000
0150      FF80  A127PM EQU    P5PM+A127       * A127 PROG MEM ADRS
0151      FF00  UN0PM  EQU    P4PM+UN0        * UN0 PROG MEM ADRS
0152 0000
0153 0000
0154            ***********************************************************
0155            *
0156            *              INTERRUPT BRANCHES
0157            *
0158            ***********************************************************
0159 0000
0160 0000             AORG   0
0161 0000
0162 0000 FF80        B      INIT             * ON HARDWARE RESET GO TO INIT
     0001 0028
0163 0002
0164 001A             AORG   26
0165 001A
0166 001A FF80        B      RXRT             * ON RINT GO TO RXRT
     001B 01C1
0167 001C
0168 001C FF80        B      TXRT             * ON TINT GO TO TXRT
     001D 01CF
```



```
0170            ***********************************************************
0171            *
0172            *              PROCESSOR INITIALIZATION ROUTINE
0173            *
0174            ***********************************************************
0175 001E
0176 0028             AORG   40
0177 0028
0178 0028 C800  INIT  LDPK   0                * INTIALIZE ST0 AND ST1
0179 0029
0180 0029 D001        LALK   >2E00            * 0010 1110 0000 0000  IN BINARY
     002A 2E00
0181 002B 6068        SACL   TEMP1            * DATA FOR ST0
0182 002C
0183 002C 5068        LST    TEMP1            * 0 -> OP    PG POINTER SET TO 0
0184            *                             * 1 -> INTM  INTERRUPTS DISABLED
0185            *                             * 1 -> OVM   OVERFLOW SATURATION
0186            *                             * 0 -> OV    OVERLOW REG CLEARED
0187            *                             * 1 -> ARP   AR POINTER SET TO 1
0188 002D
0189 002D D001        LALK   >27F8            * 0010 0111 1111 1000  IN BINARY
     002E 27F8
0190 002F 6068        SACL   TEMP1            * DATA FOR ST1
0191 0030
0192 0030 5168        LST1   TEMP1            * 0 -> PM    NO P REG SHIFTING
0193            *                             * 0 -> TXM   FSX IS AN INPUT
0194            *                             * 1 -> FO    DRR, DXR TO 8 BITS
0195            *                             * 1 -> XF    XF PIN SET TO HIGH
0196            *                             * 1 -> SXM   SIGN EXTENSION ON
0197            *                             * 0 -> TC    TC FLAG BIT RESET
0198            *                             * 0 -> CNF   B0 IS DATA MEM
0199            *                             * 1 -> ARB   AND  1 -> ARP
0200 0031
0201            *
0202            *     INITIALIZE PAGE 0
0203            *
0204 0031
0205 0031 C160        LARK   AR1,96           * LOWEST PAGE 0 LOCATION -> AR1
0206 0032
0207 0032 CA00        ZAC                     * 0 -> ACC
0208 0033
0209 0033 CB1F        RPTK   31
0210 0034
0211 0034 60A0        SACL   *+               * ZERO PAGE 0
0212 0035
0213 0035 D001        LALK   >0030            * ENABLE  XINT,RINT
     0036 0030
0214 0037 6004        SACL   IMR              * DISBALE TINT,INT0,INT1,INT2
0215 0038
0216 0038 D001        LALK   >FFFF            * MU-LAW FFFF = LINEAR 0
     0039 FFFF
0217 003A 6001        SACL   DXR
0218 003B
0219 003B 6060        SACL   DRR1
0220 003C
0221 003C 6061        SACL   DRR2
0222 003D
```

```
0223 003D D001          LALK    XT8L
     003E 0300
0224 003F 6065          SACL    BADDR
0225 0040
0226 0040 D001          LALK    132
     0041 0084
0227 0042 6066          SACL    BIAS2
0228 0043
0229 0043 D001          LALK    -7
     0044 FFF9
0230 0045 6067          SACL    NEG7
0231 0046
0232 0046 D001          LALK    THRES0
     0047 0800
0233 0048 606D          SACL    THRES
0234 0049
0235 0049 D001          LALK    1
     004A 0001
0236 004B 606E          SACL    ONE
0237 004C
0238 004C D001          LALK    P7DM+Y143-1
     004D 038E
0239 004E 606F          SACL    ADY142
0240 004F
0241 004F
0242                *
0243                *   INITIALIZE PAGE 4 AND 5
0244                *
0245 004F
0246 004F D100          LRLK    AR1,512         * LOWEST PAGE 4 ADDRESS -> AR1
     0050 0200
0247 0051
0248 0051 CA00          ZAC                     * 0 -> ACC
0249 0052
0250 0052 CBFF          RPTK    255
0251 0053
0252 0053 60A0          SACL    *+              * ZERO PAGE 4 AND 5
0253 0054
0254                *
0255                *   INITIALIZE PAGE 6 AND 7
0256                *
0257 0054
0258 0054 CBFF          RPTK    255
0259 0055
0260 0055 60A0          SACL    *+              * ZERO PAGE 6 AND 7
0261 0056
0262 0056 C807          LDPK    7
0263 0057
0264 0057 D001          LALK    1
     0058 0001
0265 0059 6012          SACL    AONE
0266 005A
0267 005A D001          LALK    >4FFF
     005B 4FFF
0268 005C 6013          SACL    SONE
0269 005D
0270 005D D001          LALK    P5DM+A0
```



```
     005E 02FF
0271 005F 6014          SACL    ADA0
0272 0060
0273 0060 D001          LALK    P6DM+Y0+1
     0061 0301
0274 0062 6015          SACL    ADY1
0275 0063
0276 0063 D001          LALK    P7DM+INC0
     0064 03F8
0277 0065 6016          SACL    ADINC0
0278 0066
0279 0066 D001          LALK    P7DM+M8-1
     0067 03F5
0280 0068 6017          SACL    ADM7
0281 0069
0282 0069 D001          LALK    P4DM+UN15-1
     006A 020E
0283 006B 6018          SACL    ADUN14
0284 006C
0285 006C D001          LALK    HANGT0
     006D 0258
0286 006E 6061          SACL    HANGT
0287 006F
0288 006F D001          LALK    >400            * >400 = 1/8 OF MAX ABSY
     0070 0400
0289 0071 6069          SACL    ABSY
0290 0072
0291 0072 D001          LALK    >20
     0073 0020
0292 0074 606B          SACL    IABSY
0293 0075
0294 0075 D001          LALK    CUTOF0
     0076 0021
0295 0077 606C          SACL    CUTOFF
0296 0078
0297 0078 D001          LALK    >400
     0079 0400
0298 007A 606D          SACL    ABSY0F
0299 007B
0300 007B 606E          SACL    M0
0301 007C
0302 007C CE00          EINT
0303 007D
0304 007D FF80          B LOOP
     007E 01BE
```

```
0306                ****************************************************
0307                *
0308                *               CYCLE START ROUTINE
0309                *
0310                ****************************************************
0311 007F
0312 007F C800  START  LDPK    0
0313 0080
0314                *
0315                *       CONVERT MU-LAW INPUT REFERENCE SAMPLE TO LINEAR  (Y0)
0316                *
0317 0080
0318 0080 4161         ZALS    DRR2            * MU-LAW Y(0) -> ACC
0319 0081
0320 0081 0065         ADD     BADDR           * ADD MU-LAW TABLE BASE ADDRESS
0321 0082
0322 0082 C806         LDPK    6
0323 0083
0324 0083 5800         TBLR    Y0              * LINEAR Y(0) -> Y0
0325 0084
0326                *
0327                *       COMPUTE ABSOLUTE VALUE OF Y0
0328                *
0329 0084
0330 0084 2000         LAC     Y0              * Y0 -> ACC
0331 0085
0332 0085 CE1B         ABS
0333 0086
0334 0086 C807         LDPK    7
0335 0087
0336 0087 6068         SACL    ABSY0           * |Y0| -> ABSY0 ON PAGE 7
0337 0088
0338 0088 C800         LDPK    0
0339 0089
0340                *
0341                *       CONVERT MU-LAW NEAR END SAMPLE TO LINEAR (S0DC)
0342                *
0343 0089
0344 0089 4160         ZALS    DRR1            * MU-LAW S(0)DC -> ACC
0345 008A
0346 008A 0065         ADD     BADDR           * ADD MU-LAW TABLE BASE ADDRESS
0347 008B
0348 008B 5872         TBLR    S0DC            * LINEAR S(0)DC -> S0DC
0349 008C
0350                *
0351                *       COMPUTE HIGH PASS FILTERED NEAR END SAMPLE (S0)
0352                *
0353 008C
0354 008C 4171         ZALS    S0LSBS          * S0LSBS -> LOW ACC
0355 008D
0356 008D 4870         ADDH    S0              * S0 (MSBS) -> HIGH ACC
0357 008E
0358 008E 1D70         SUB     S0,HTAU         * ACC - S0 * 2**HTAU -> ACC
0359 008F
0360 008F 4872         ADDH    S0DC            * ACC + S0DC * 2**16 -> ACC
0361 0090
0362 0090 1C72         SUB     S0DC,HTAU-1     * ACC - S0DC * 2**HTAU-1 -> ACC
```



```
0363 0091
0364 0091 4473         SUBH    S1DC            * ACC - S1DC * 2**16 -> ACC
0365 0092
0366 0092 DC73         ADD     S1DC,HTAU-1     * ACC + S1DC * 2**HTAU-1 -> ACC
0367 0093
0368 0093 6071         SACL    S0LSBS          * LOW ACC -> S0LSBS
0369 0094
0370 0094 6870         SACH    S0              * HIGH ACC -> S0 (MSBS)
0371 0095
0372 0095 5672         DMOV    S0DC            * S0DC -> S1DC
0373 0096
0374                *
0375                *       COMPUTE ABSOLUTE VALUE OF S0
0376                *
0377 0096
0378 0096 CE1B         ABS
0379 0097
0380 0097 C807         LDPK    7
0381 0098
0382 0098 6863         SACH    ABSS0           * |S0| -> ABSS0 ON PAGE 7
```

```
0384                ************************************************************
0385                *
0386                *               ECHO ESTIMATION ROUTINE
0387                *
0388                ************************************************************
0389 0099
0390 0099 C800  EESTR  LDPK    0
0391 009A
0392                *
0393                *       MOVE Y128,Y129,...,Y142 TO NEXT HIGHER MEMORY LOCATION
0394                *
0395 009A
0396 009A 5589         LARP    AR1             * 1 -> AR POINTER
0397 009B
0398 009B 316F         LAR     AR1,ADY142      * ADY142 -> AR1
0399 009C
0400 009C CB0E         RPTK    14              * K = 142,141,...,128
0401 009D
0402 009D 5690         DMOV    *-              * Y(K) -> Y(K+1)
0403 009E
0404                *
0405                *       CONVOLVE REFERENCE SAMPLES WITH FIR COEFFICIENTS
0406                *
0407 009E
0408 009E 2E6E  FIR    LAC     ONE,14          * ROUND-OFF OFFSET -> ACC
0409 009F
0410 009F A000         MPYK    0               * 0 -> P
0411 00A0
0412 00A0 CE05         CNFP                    * AR1 STILL POINTS AT Y127
0413 00A1
0414 00A1 CB7F         RPTK    127             * K = 127,126,...,0
0415 00A2
0416 00A2 5C90         MACD    A127PM,*-       * Y(K) * A(1-K) + ACC -> ACC
     00A3 FF80
0417 00A4
0418 00A4 CE04         CNFD
0419 00A5
0420 00A5 CE15         APAC                    * P + ACC -> ACC
0421 00A6
0422 00A6 696B         SACH    EEST,1          * 2 * HIGH ACC -> EEST
0423 00A7
0424                *
0425                *       COMPUTE THE OUTPUT
0426                *
0427 00A7
0428 00A7 2070         LAC     S0              * S0 -> ACC
0429 00AB
0430 00AB 106B         SUB     EEST            * ACC - EEST -> ACC
0431 00A9
0432 00A9 606C         SACL    OUTPUT          * ACC -> OUTPUT
0433 00AA
0434 00AA C807         LDPK    7
0435 00AB
0436 00AB CE1B         ABS
0437 00AC
0438 00AC 6065         SACL    ABSE0           * ACC -> ABSE0 ON PAGE 7
```



```
0440                ************************************************************
0441                *
0442                *               RESIDUAL OUTPUT SUPPRESSION ROUTINE
0443                *
0444                ************************************************************
0445 00AD
0446 00AD
0447 00AD 3C6B  SPRS   LT      IABSY           * IABSY -> T REG
0448 00AE
0449 00AE 3866         MPY     ABSOUT          * ABSOUT * IABSY -> P REG
0450 00AF
0451 00AF 2062         LAC     HCNTR           * NEAR END SPEECH FLAG -> ACC
0452 00B0
0453 00B0 C800         LDPK    0
0454 00B1
0455 00B1 F180         BGZ     WOUT            * IF N.E. SPEECH NO SPRS
     00B2 00BB
0456 00B3
0457 00B3 CE14         PAC                     * P REG -> ACC
0458 00B4
0459 00B4 106D         SUB     THRES           * ACC - THRES -> ACC
0460 00B5
0461 00B5 F180         BGZ     WOUT            * IF THRES EXCEEDED SKIP SPRS
     00B6 00BB
0462 00B7
0463 00B7 FAB0         BIOZ    WOUT            * IF BIO PIN LOW SKIP SPRS
     00B8 00BB
0464 00B9
0465 00B9 CA00         ZAC                     * 0 -> ACC
0466 00BA
0467 00BA 606C         SACL    OUTPUT          * ACC -> OUTPUT
0468 00BB
0469 00BB 3C6C  WOUT   LT      OUTPUT          * OUTPUT -> T REG   (FOR UNO)
```

```
0471                    ************************************************************
0472                    *
0473                    *               LINEAR TO MU-LAW COMPRESSION ROUTINE
0474                    *
0475                    ************************************************************
0476 00BC
0477 00BC 406C  CMPRS   ZALH    OUTPUT          * OUTPUT -> ACC
0478 00BD
0479 00BD CE18          SFL
0480 00BE
0481 00BE CE18          SFL                     * LEFT JUSTIFY ACC
0482 00BF
0483 00BF F380          BLZ     NEGCMP          * IF ACC < 0 THEN GO TO NEGCMP
     00C0 00D0
0484 00C1
0485 00C1 4866  POSCMP  ADDH    BIAS2
0486 00C2
0487 00C2 3167          LAR     AR1,NEG7
0488 00C3
0489 00C3 CB06          RPTK    6               * FIND MSB
0490 00C4
0491 00C4 CEA2          NORM
0492 00C5
0493 00C5 DE04          ANDK    >F000,14        * ZERO 2 MSBS AND ALL LSBS
     00C6 F000
0494 00C7
0495 00C7 6868          SACH    Q
0496 00CB
0497 00C8 7169          SAR     AR1,SI
0498 00C9
0499 00C9 4069          ZALH    SI
0500 00CA
0501 00CA CE1B          ABS
0502 00CB
0503 00CB 0268          ADD     Q,2
0504 00CC
0505 00CC D406          XORK    >FF00,4         * INVERT ALL BITS
     00CD FF00
0506 00CE
0507 00CE FF80          B       TXOUT
     00CF 00DE
0508 00D0
0509 00D0
0510 00D0 CE1B  NEGCMP  ABS                     * LEFT JUSTIFIED OUTPUT IN ACC
0511 00D1
0512 00D1 4866          ADDH    BIAS2
0513 00D2
0514 00D2 3167          LAR     AR1,NEG7
0515 00D3
0516 00D3 CB06          RPTK    6               * FIND MSB
0517 00D4
0518 00D4 CEA2          NORM
0519 00D5
0520 00D5 DE04          ANDK    >F000,14        * ZERO 2 MSBS AND ALL LSBS
     00D6 F000
0521 00D7
0522 00D7 6868          SACH    Q
```



```
0523 00D8
0524 00D8 7169          SAR     AR1,SI
0525 00D9
0526 00D9 4069          ZALH    SI
0527 00DA
0528 00DA CE1B          ABS
0529 00DB
0530 00DB 0268          ADD     Q,2
0531 00DC
0532 00DC D406          XORK    >7F00,4         * INVERT ALL BITS IN Q
     00DD 7F00
0533 00DE
0534 00DE 6C01  TXOUT   SACH    DXR,4           * 2**4 * HIGH ACC -> DXR
```

```
0536                ****************************************************************
0537                *
0538                *                POWER ESTIMATION ROUTINE
0539                *
0540                ****************************************************************
0541 00DF
0542 00DF C807  NORM   LDPK    7               * T REG STILL CONTAINS OUTPUT
0543 00E0
0544                *
0545                *      UPDATE LONG TAU OUTPUT POWER ESTIMATE (ABSOUT)
0546                *
0547 00E0
0548 00E0 4066         ZALH    ABSOUT          * ABSOUT -> HIGH ACC
0549 00E1
0550 00E1 4967         ADDS    AELSBS          * AELSBS -> LOW ACC
0551 00E2
0552 00E2 1966         SUB     ABSOUT,LTAU     * ACC - ABSOUT * 2**LTAU -> ACC
0553 00E3
0554 00E3 0965         ADD     ABSE0,LTAU      * ACC + ABSE0 * 2**LTAU -> ACC
0555 00E4
0556 00E4 6866         SACH    ABSOUT          * HIGH ACC -> ABSOUT
0557 00E5
0558 00E5 6067         SACL    AELSBS          * LOW ACC -> AELSBS
0559 00E6
0560                *
0561                *      UPDATE LONG TAU REFERENCE POWER ESTIMATE (ABSY)
0562                *
0563 00E6
0564 00E6 4069         ZALH    ABSY            * ABSY -> HIGH ACC
0565 00E7
0566 00E7 496A         ADDS    AYLSBS          * AYLSBS -> LOW ACC
0567 00E8
0568 00E8 1969         SUB     ABSY,LTAU       * ACC - ABSY * 2**LTAU -> ACC
0569 00E9
0570 00E9 0968         ADD     ABSY0,LTAU      * ACC + ABSY0 * 2**LTAU -> ACC
0571 00EA
0572 00EA 096C         ADD     CUTOFF,LTAU     * ACC + CUTOFF * 2**LTAU -> ACC
0573 00EB
0574 00EB 6869         SACH    ABSY            * HIGH ACC -> ABSY
0575 00EC
0576 00EC 606A         SACL    AYLSBS          * LOW ACC -> AYLSBS
0577 00ED
0578                *
0579                *      COMPUTE 1/ABSY (DIVIDE 1 BY ABSY)
0580                *
0581 00ED
0582 00ED 4012         ZALH    AONE
0583 00EE
0584 00EE CB0E         RPTK    14
0585 00EF
0586 00EF 4769         SUBC    ABSY
0587 00F0
0588 00F0 606B         SACL    IABSY
```



```
0590                ****************************************************************
0591                *
0592                *              OUTPUT NORMALIZATION ROUTINE
0593                *
0594                ****************************************************************
0595 00F1
0596                *
0597                *      MOVE UN0,UN1,....,UN14 TO NEXT HIGHER MEMORY LOCATION
0598                *
0599 00F1
0600 00F1 3118         LAR     AR1,ADUN14      * ADUN14 -> AR1
0601 00F2
0602 00F2 CB0D         RPTK    13              * K=14,13,....1
0603 00F3
0604 00F3 5690         DMOV    *-              * UN(K) -> UN(K+1)
0605 00F4
0606 00F4 5680         DMOV    *               * UN(0) -> UN(1)
0607 00F5
0608                *
0609                *      COMPUTE NORMALIZED OUTPUT (UN0)
0610                *
0611 00F5
0612 00F5 386B         MPY     IABSY           * IABSY * T REG(OUTPUT) -> P REG
0613 00F6
0614 00F6 CE14         PAC                     * P REG  (UN0) -> ACC
0615 00F7
0616                *
0617                *      SATURATE NORMALIZED OUTPUT (UN0) AT +/- 1.0
0618                *
0619 00F7
0620 00F7 F480         BGEZ    POSUN0          * IF UN0 > 0 THEN GO TO POSUN0
     00F8 0100
0621 00F9
0622 00F9 0013  NEGUN0 ADD     SONE            * ACC + SONE -> ACC
0623 00FA
0624 00FA F480         BGEZ    SMLUN0          * IF -1.0 < UN0 < 0 THEN NO SATR
     00FB 0106
0625 00FC
0626 00FC CA00         ZAC                     * 0 -> ACC
0627 00FD
0628 00FD 1013         SUB     SONE            * ACC - SONE -> ACC
0629 00FE
0630 00FE FF80         B       SAVUN0
     00FF 0107
0631 0100
0632 0100 1013  POSUN0 SUB     SONE            * ACC - SONE -> ACC
0633 0101
0634 0101 F280         BLEZ    SMLUN0          * IF 0 < UN0 < 1.0 THEN NO SATR
     0102 0106
0635 0103
0636 0103 2013         LAC     SONE            * SONE -> ACC
0637 0104
0638 0104 FF80         B       SAVUN0
     0105 0107
0639 0106
0640 0106 CE14  SMLUN0 PAC                     * P REG  (UN0) -> ACC
0641 0107
```

```
0642 0107 6011  SAVUN0 SACL    CUN0            * ACC -> CUN0
0643 0108
0644 0108 6080         SACL    *               * ACC -> UN(0)
```



```
0646                   ************************************************************
0647                   *
0648                   *               NEAR-END SPEECH DETECTION ROUTINE
0649                   *
0650                   ************************************************************
0651 0109
0652 0109 C807  NESP   LDPK    7
0653 010A
0654                   *
0655                   *       UPDATE SHORT TAU REFERENCE POWER ESTIMATE (ABSY0F)
0656                   *
0657 010A
0658 010A 4060         ZALH    ABSY0F          * ABSY0F * 2**16 -> ACC
0659 010B
0660 010B 1B6D         SUB     ABSY0F,STAU     * ACC - ABSY0F * 2**STAU -> ACC
0661 010C
0662 010C 0B68         ADD     ABSY0,STAU      * ACC + ABSY0 * 2**STAU -> ACC
0663 010D
0664 010D 686D         SACH    ABSY0F          * HIGH ACC -> ABSY0F
0665 010E
0666                   *
0667                   *       UPDATE SHORT TAU NEAR END POWER ESTIMATE (ABSS0F)
0668                   *
0669 010E
0670 010E 4064         ZALH    ABSS0F          * ABSS0F * 2**16 -> ACC
0671 010F
0672 010F 1B64         SUB     ABSS0F,STAU     * ACC - ABSS0F * 2**STAU -> ACC
0673 0110
0674 0110 0C63         ADD     ABSS0,STAU+NER  * ACC + ABSS0*2**STAU+NER -> ACC
0675 0111
0676 0111 6864         SACH    ABSS0F          * HIGH ACC -> ABSS0F
0677 0112
0678                   *
0679                   *       UPDATE MODULO 16 COUNTER (H)
0680                   *
0681 0112
0682 0112 2060         LAC     H               * H -> ACC
0683 0113
0684 0113 0012         ADD     AONE            * ACC + 1 -> ACC
0685 0114
0686 0114 D004         ANDK    >000F           * IF ACC = 16 THEN 0 -> ACC
     0115 000F
0687 0116
0688 0116 6060         SACL    H               * ACC -> H
0689 0117
0690 0117 F180         BGZ     NESP1           * IF H > 0 THEN GO TO NESP1
     0118 011F
0691 0119
0692                   *
0693                   *       MOVE M0,M1,...,M7 TO NEXT HIGHER MEMORY LOCATION
0694                   *
0695 0119
0696 0119 3117         LAR     AR1,ADM7        * ADM7 -> AR1
0697 011A
0698 011A CB07         RPTK    7               * K=7,6,...,0
0699 011B
0700 011B 5690         DMOV    *-              * M(K) -> M(K+1)
```

```
0701 011C
0702 011C 566D          DMOV    ABSYOF          * ABSYOF -> M0
0703 011D
0704 011D FF80          B       NESP3           * ON MEMORY MOVES SKIP DETECTION
     011E 0149
0705 011F
0706                *
0707                *   UPDATE MOST RECENT LOCAL MAXIMA (M0)
0708                *
0709 011F
0710 011F 206D  NESP1   LAC     ABSYOF          * ABSYOF -> ACC
0711 0120
0712 0120 106E          SUB     M0              * ACC - M0 -> ACC
0713 0121
0714 0121 F280          BLEZ    NESP2           * IF M0 > ABSYOF THEN NO UPDATE
     0122 0124
0715 0123
0716 0123 566D          DMOV    ABSYOF          * ABSYOF -> M0
0717 0124
0718                *
0719                *   COMPARE REFERENCE POWER TO NEAR-END POWER
0720                *
0721 0124
0722 0124 2064  NESP2   LAC     ABSSOF          * ABSSOF -> ACC
0723 0125
0724 0125 106E          SUB     M0              * ACC - M0 -> ACC
0725 0126
0726 0126 F280          BLEZ    NESP3           * NO N.E. SPEECH IF M0 > ABSSOF
     0127 0149
0727 0128
0728 0128
0729 0128 2064          LAC     ABSSOF          * ABSSOF -> ACC
0730 0129
0731 0129 106F          SUB     M0+1            * ACC - M1 -> ACC
0732 012A
0733 012A F280          BLEZ    NESP3           * NO N.E. SPEECH IF M1 > ABSSOF
     012B 0149
0734 012C
0735 012C
0736 012C 2064          LAC     ABSSOF          * ABSSOF -> ACC
0737 012D
0738 012D 1070          SUB     M0+2            * ACC - M2 -> ACC
0739 012E
0740 012E F280          BLEZ    NESP3           * NO N.E. SPEECH IF M2 > ABSSOF
     012F 0149
0741 0130
0742 0130
0743 0130 2064          LAC     ABSSOF          * ABSSOF -> ACC
0744 0131
0745 0131 1071          SUB     M0+3            * ACC - M3 -> ACC
0746 0132
0747 0132 F280          BLEZ    NESP3           * NO N.E. SPEECH IF M3 > ABSSOF
     0133 0149
0748 0134
0749 0134
0750 0134 2064          LAC     ABSSOF          * ABSSOF -> ACC
0751 0135
```



```
0752 0135 1072          SUB     M0+4            * ACC - M4 -> ACC
0753 0136
0754 0136 F280          BLEZ    NESP3           * NO N.E. SPEECH IF M4 > ABSSOF
     0137 0149
0755 0138
0756 0138
0757 0138 2064          LAC     ABSSOF          * ABSSOF -> ACC
0758 0139
0759 0139 1073          SUB     M0+5            * ACC - M5 -> ACC
0760 013A
0761 013A F280          BLEZ    NESP3           * NO N.E. SPEECH IF M5 > ABSSOF
     013B 0149
0762 013C
0763 013C
0764 013C 2064          LAC     ABSSOF          * ABSSOF -> ACC
0765 013D
0766 013D 1074          SUB     M0+6            * ACC - M6 -> ACC
0767 013E
0768 013E F280          BLEZ    NESP3           * NO N.E. SPEECH IF M6 > ABSSOF
     013F 0149
0769 0140
0770 0140
0771 0140 2064          LAC     ABSSOF          * ABSSOF -> ACC
0772 0141
0773 0141 1075          SUB     M0+7            * ACC - M7 -> ACC
0774 0142
0775 0142 F280          BLEZ    NESP3           * NO N.E. SPEECH IF M7 > ABSSOF
     0143 0149
0776 0144
0777 0144
0778 0144 2064          LAC     ABSSOF          * ABSSOF -> ACC
0779 0145
0780 0145 1076          SUB     M0+8            * ACC - M8 -> ACC
0781 0146
0782 0146 F280          BLEZ    NESP3           * NO N.E. SPEECH IF M8 > ABSSOF
     0147 0149
0783 0148
0784                *
0785                *   NEAR-END SPEECH DETECTED  SET HANGOVER COUNTER (HCNTR)
0786                *
0787 0148
0788 0148 5661          DMOV    HANGT           * HANGT -> HCNTR
0789 0149
0790                *
0791                *   CHECK AND UPDATE HANGOVER COUNTER
0792                *
0793 0149
0794 0149 2062  NESP3   LAC     HCNTR           * HCNTR -> ACC
0795 014A
0796 014A F680          BZ      NESP4           * IF HCNTR = 0 THEN GO TO NESP4
     014B 0150
0797 014C
0798 014C 1012          SUB     AONE            * ACC - 1 -> ACC
0799 014D
0800 014D 6062          SACL    HCNTR           * ACC -> HCNTR
0801 014E
0802 014E FF80          B       LOOP            * GO TO CYCLE END
```

```
     014F 01BE
0803 0150
0804                *
0805                *       CHECK IF LTAU REFERENCE POWER ESTIMATE IS BELOW CUTOFF
0806                *
0807 0150
0808 0150 2069      NESP4   LAC     ABSY            * ABSY -> ACC
0809 0151
0810 0151 106C              SUB     CUTOFF          * ACC - CUTOFF -> ACC
0811 0152
0812 0152 F280              BLEZ    LOOP            * IF ABSY < CUTOFF THEN LOOP
     0153 01BE
```



```
0814                ***************************************************
0815                *
0816                *               COEFFICIENT INCREMENT UPDATE ROUTINE
0817                *
0818                ***************************************************
0819 0154
0820 0154 2015      UPINC   LAC     ADY1            * ADY1 -> ACC  (Y0 IS NOW IN Y1)
0821 0155
0822 0155 0060              ADD     H               * ACC + H -> ACC
0823 0156
0824 0156 6010              SACL    TEMP3
0825 0157
0826 0157 3110              LAR     AR1,TEMP3       * ADY1 + H -> AR1
0827 0158
0828 0158 CE05              CNFP
0829 0159
0830 0159 3C11              LT      CUN0            * UN0 -> T REG
0831 015A
0832 015A
0833 015A 2F12              LAC     AONE,15         * ROUND-OFF OFFSET -> ACC
0834 015B
0835 015B 38A0              MPY     *+              * UN(0) * Y(0+H) -> P REG
0836 015C
0837 015C CB0E              RPTK    14              * K = 1,2,...,15
0838 015D
0839 015D 5DA0              MAC     UN0PM+1,*+      * UN(K) * Y(K+H) + ACC -> ACC
     015E FF01
0840 015F
0841 015F 3D11              LTA     CUN0            * P REG + ACC -> ACC    UN0 -> T
0842 0160
0843 0160 6878              SACH    INC0            * HIGH ACC -> INC(0)
0844 0161
0845 0161
0846 0161 2F12              LAC     AONE,15         * ROUND-OFF OFFSET -> ACC
0847 0162
0848 0162 38A0              MPY     *+              * UN(0) * Y(16+H) -> P REG
0849 0163
0850 0163 CB0E              RPTK    14              * K = 1,2,...,15
0851 0164
0852 0164 5DA0              MAC     UN0PM+1,*+      * UN(K) * Y(K+16+H) + ACC -> ACC
     0165 FF01
0853 0166
0854 0166 3D11              LTA     CUN0            * P REG + ACC -> ACC    UN0 -> T
0855 0167
0856 0167 6879              SACH    INC0+1          * HIGH ACC -> INC(1)
0857 0168
0858 0168
0859 0168 2F12              LAC     AONE,15         * ROUND-OFF OFFSET -> ACC
0860 0169
0861 0169 38A0              MPY     *+              * UN(0) * Y(32+H) -> P REG
0862 016A
0863 016A CB0E              RPTK    14              * K = 1,2,...,15
0864 016B
0865 016B 5DA0              MAC     UN0PM+1,*+      * UN(K) * Y(K+32+H) + ACC -> ACC
     016C FF01
0866 016D
0867 016D 3D11              LTA     CUN0            * P REG + ACC -> ACC    UN0 -> T
```

```
0868 016E
0869 016E 687A          SACH    INC0+2          * HIGH ACC -> INC(2)
0870 016F
0871 016F
0872 016F 2F12          LAC     AONE,15
0873 0170
0874 0170 38A0          MPY     *+
0875 0171
0876 0171 CB0E          RPTK    14
0877 0172
0878 0172 5DA0          MAC     UN0PM+1,*+
     0173 FF01
0879 0174
0880 0174 3D11          LTA     CUN0
0881 0175
0882 0175 687B          SACH    INC0+3
0883 0176
0884 0176
0885 0176 2F12          LAC     AONE,15
0886 0177
0887 0177 38A0          MPY     *+
0888 0178
0889 0178 CB0E          RPTK    14
0890 0179
0891 0179 5DA0          MAC     UN0PM+1,*+
     017A FF01
0892 017B
0893 017B 3D11          LTA     CUN0
0894 017C
0895 017C 687C          SACH    INC0+4
0896 017D
0897 017D
0898 017D 2F12          LAC     AONE,15
0899 017E
0900 017E 38A0          MPY     *+
0901 017F
0902 017F CB0E          RPTK    14
0903 0180
0904 0180 5DA0          MAC     UN0PM+1,*+
     0181 FF01
0905 0182
0906 0182 3D11          LTA     CUN0
0907 0183
0908 0183 687D          SACH    INC0+5
0909 0184
0910 0184
0911 0184 2F12          LAC     AONE,15
0912 0185
0913 0185 38A0          MPY     *+
0914 0186
0915 0186 CB0E          RPTK    14
0916 0187
0917 0187 5DA0          MAC     UN0PM+1,*+
     0188 FF01
0918 0189
0919 0189 3D11          LTA     CUN0
0920 018A
```



```
0921 018A 687E          SACH    INC0+6
0922 018B
0923 018B
0924 018B 2F12          LAC     AONE,15
0925 018C
0926 018C 38A0          MPY     *+
0927 018D
0928 018D CB0E          RPTK    14
0929 018E
0930 018E 5DA0          MAC     UN0PM+1,*+
     018F FF01
0931 0190
0932 0190 3D11          LTA     CUN0
0933 0191
0934 0191 687F          SACH    INC0+7
0935 0192
0936 0192
0937 0192 CE04          CNFD
```

```
0939                ********************************************************
0940                *
0941                *       COEFFICIENT UPDATE ROUTINE
0942                *
0943                ********************************************************
0944 0193
0945 0193 C010          LARK    AR0,16          * 16 -> AR0   (AR2 INCREMENT)
0946 0194
0947 0194 3116          LAR     AR1,ADINC0      * ADINC0 -> AR1
0948 0195
0949 0195 2014          LAC     ADA0            * ADA0 -> ACC
0950 0196
0951 0196 1060          SUB     H               * ACC - H -> ACC
0952 0197
0953 0197 6010          SACL    TEMP3
0954 0198
0955 0198 3210          LAR     AR2,TEMP3       * ADA0 - H -> AR2
0956 0199
0957 0199 CE0A          SPM     2               * SET 4 BIT LEFT SHIFT OF P REG
0958 019A
0959 019A 236B          LAC     IABSY,GAIN      * IABSY * 2**GAIN -> ACC
0960 019B
0961 019B 6010          SACL    TEMP3           * ACC -> TEMP3
0962 019C
0963 019C 3C10          LT      TEMP3           * TEMP3 -> T REG
0964 019D
0965 019D
0966 019D 38AA          MPY     *+,AR2          * INC(0) * T REG -> P REG
0967 019E
0968 019E 4080          ZALH    *               * A(H) * 2**16 -> ACC
0969 019F
0970 019F CE15          APAC                    * P REG + ACC -> ACC
0971 01A0
0972 01A0 68D9          SACH    *0-,0,AR1       * HIGH ACC -> A(H)
0973 01A1
0974 01A1
0975 01A1 38AA          MPY     *+,AR2          * INC(1) * T REG -> P REG
0976 01A2
0977 01A2 4080          ZALH    *               * A(16+H) * 2**16 -> ACC
0978 01A3
0979 01A3 CE15          APAC                    * P REG + ACC -> ACC
0980 01A4
0981 01A4 68D9          SACH    *0-,0,AR1       * HIGH ACC -> A(16+H)
0982 01A5
0983 01A5
0984 01A5 38AA          MPY     *+,AR2          * INC(2) * T REG -> P REG
0985 01A6
0986 01A6 4080          ZALH    *               * A(32+H) * 2**16 -> ACC
0987 01A7
0988 01A7 CE15          APAC                    * P REG + ACC -> ACC
0989 01A8
0990 01A8 68D9          SACH    *0-,0,AR1       * HIGH ACC -> A(32+H)
0991 01A9
0992 01A9
0993 01A9 38AA          MPY     *+,AR2
0994 01AA
0995 01AA 4080          ZALH    *
```



```
0996 01AB
0997 01AB CE15          APAC
0998 01AC
0999 01AC 68D9          SACH    *0-,0,AR1
1000 01AD
1001 01AD
1002 01AD 38AA          MPY     *+,AR2
1003 01AE
1004 01AE 4080          ZALH    *
1005 01AF
1006 01AF CE15          APAC
1007 01B0
1008 01B0 68D9          SACH    *0-,0,AR1
1009 01B1
1010 01B1
1011 01B1 38AA          MPY     *+,AR2
1012 01B2
1013 01B2 4080          ZALH    *
1014 01B3
1015 01B3 CE15          APAC
1016 01B4
1017 01B4 68D9          SACH    *0-,0,AR1
1018 01B5
1019 01B5
1020 01B5 38AA          MPY     *+,AR2
1021 01B6
1022 01B6 4080          ZALH    *
1023 01B7
1024 01B7 CE15          APAC
1025 01B8
1026 01B8 68D9          SACH    *0-,0,AR1
1027 01B9
1028 01B9
1029 01B9 38AA          MPY     *+,AR2
1030 01BA
1031 01BA 4080          ZALH    *
1032 01BB
1033 01BB CE15          APAC
1034 01BC
1035 01BC 68D9          SACH    *0-,0,AR1
1036 01BD
1037 01BD
1038 01BD CE08          SPM     0               * SET NO SHIFT OF P REG
```

```
1040                    ********************************************************
1041                    *
1042                    *               CYCLE END ROUTINE
1043                    *
1044                    ********************************************************
1045 01BE
1046 01BE CE1F  LOOP    IDLE                    * WAIT IN LOOP UNITL RINT/XINT
1047 01BF
1048 01BF CE1F          IDLE                    * EXTRA IDLE FOR TWO RINT
1049 01C0
1050 01C0 5500          NOP
1051 01C1
1052                    ********************************************************
1053                    *
1054                    *               RINT SERVICE ROUTINE
1055                    *
1056                    ********************************************************
1057 01C1
1058 01C1 7862  RXRT    SST     TST0            * SAVE ST0
1059 01C2
1060 01C2 C800          LDPK    0               * 0 -> PAGE POINTER
1061 01C3
1062 01C3 6863          SACH    TACCH           * SAVE HIGH ACC
1063 01C4
1064 01C4 6064          SACL    TACCL           * SAVE LOW ACC
1065 01C5
1066 01C5 5660          DMOV    DRR1            * DRR1 -> DRR2
1067 01C6
1068 01C6 4100          ZALS    DRR             * DRR -> ACC
1069 01C7
1070 01C7 D004          ANDK    >00FF           * MASK-OFF MSB BYTE
     01C8 00FF
1071 01C9
1072 01C9 6060          SACL    DRR1            * ACC -> DRR1
1073 01CA
1074 01CA 4164          ZALS    TACCL           * RESTORE LOW ACC
1075 01CB
1076 01CB 4863          ADDH    TACCH           * RESTORE HIGH ACC
1077 01CC
1078 01CC 5062          LST     TST0            * RESTORE ST0
1079 01CD
1080 01CD CE00          EINT                    * INTERRUPTS ENABLED
1081 01CE
1082 01CE CE26          RET                     * RETURN TO PROGRAM
1083 01CF
1084 01CF
1085                    ********************************************************
1086                    *
1087                    *               XINT SERVICE ROUTINE
1088                    *
1089                    ********************************************************
1090 01CF
1091 01CF CE00  TXRT    EINT                    * INTERRUPTS ENABLED
1092 01D0
1093 01D0 FF80          B       START           * BRANCH TO PROGRAM START
     01D1 007F
```



```
1095                    ********************************************************
1096                    *
1097                    *           MU-LAW EXPANSION LOOKUP TABLE
1098                    *
1099                    ********************************************************
1100 01D2
1101 0300               AORG    >300
1102 0300
1103      0300  XTBL    EQU     $
1104 0300
1105 0300 E0A1          DATA    >E0A1           * NEGATIVE VALUES FIRST
1106 0301 E1A1          DATA    >E1A1           *  (FF, FE, ETC.)
1107 0302 E2A1          DATA    >E2A1
1108 0303 E3A1          DATA    >E3A1
1109 0304 E4A1          DATA    >E4A1
1110 0305 E5A1          DATA    >E5A1
1111 0306 E6A1          DATA    >E6A1
1112 0307 E7A1          DATA    >E7A1
1113 0308 E8A1          DATA    >E8A1
1114 0309 E9A1          DATA    >E9A1
1115 030A EAA1          DATA    >EAA1
1116 030B EBA1          DATA    >EBA1
1117 030C ECA1          DATA    >ECA1
1118 030D EDA1          DATA    >EDA1
1119 030E EEA1          DATA    >EEA1
1120 030F EFA1          DATA    >EFA1
1121 0310 F061          DATA    >F061
1122 0311 F0E1          DATA    >F0E1
1123 0312 F161          DATA    >F161
1124 0313 F1E1          DATA    >F1E1
1125 0314 F261          DATA    >F261
1126 0315 F2E1          DATA    >F2E1
1127 0316 F361          DATA    >F361
1128 0317 F3E1          DATA    >F3E1
1129 0318 F461          DATA    >F461
1130 0319 F4E1          DATA    >F4E1
1131 031A F561          DATA    >F561
1132 031B F5E1          DATA    >F5E1
1133 031C F661          DATA    >F661
1134 031D F6E1          DATA    >F6E1
1135 031E F761          DATA    >F761
1136 031F F7E1          DATA    >F7E1
1137 0320 F841          DATA    >F841
1138 0321 F8B1          DATA    >F8B1
1139 0322 F8C1          DATA    >F8C1
1140 0323 F901          DATA    >F901
1141 0324 F941          DATA    >F941
1142 0325 F981          DATA    >F981
1143 0326 F9C1          DATA    >F9C1
1144 0327 FA01          DATA    >FA01
1145 0328 FA41          DATA    >FA41
1146 0329 FA81          DATA    >FA81
1147 032A FAC1          DATA    >FAC1
1148 032B FB01          DATA    >FB01
1149 032C FB41          DATA    >FB41
1150 032D FB81          DATA    >FB81
1151 032E FBC1          DATA    >FBC1
```

```
1152 032F FC01            DATA  >FC01
1153 0330 FC31            DATA  >FC31
1154 0331 FC51            DATA  >FC51
1155 0332 FC71            DATA  >FC71
1156 0333 FC91            DATA  >FC91
1157 0334 FCB1            DATA  >FCB1
1158 0335 FCD1            DATA  >FCD1
1159 0336 FCF1            DATA  >FCF1
1160 0337 FD11            DATA  >FD11
1161 0338 FD31            DATA  >FD31
1162 0339 FD51            DATA  >FD51
1163 033A FD71            DATA  >FD71
1164 033B FD91            DATA  >FD91
1165 033C FDB1            DATA  >FDB1
1166 033D FDD1            DATA  >FDD1
1167 033E FDF1            DATA  >FDF1
1168 033F FE11            DATA  >FE11
1169 0340 FE29            DATA  >FE29
1170 0341 FE39            DATA  >FE39
1171 0342 FE49            DATA  >FE49
1172 0343 FE59            DATA  >FE59
1173 0344 FE69            DATA  >FE69
1174 0345 FE79            DATA  >FE79
1175 0346 FE89            DATA  >FE89
1176 0347 FE99            DATA  >FE99
1177 0348 FEA9            DATA  >FEA9
1178 0349 FEB9            DATA  >FEB9
1179 034A FEC9            DATA  >FEC9
1180 034B FED9            DATA  >FED9
1181 034C FEE9            DATA  >FEE9
1182 034D FEF9            DATA  >FEF9
1183 034E FF09            DATA  >FF09
1184 034F FF19            DATA  >FF19
1185 0350 FF25            DATA  >FF25
1186 0351 FF2D            DATA  >FF2D
1187 0352 FF35            DATA  >FF35
1188 0353 FF3D            DATA  >FF3D
1189 0354 FF45            DATA  >FF45
1190 0355 FF4D            DATA  >FF4D
1191 0356 FF55            DATA  >FF55
1192 0357 FF5D            DATA  >FF5D
1193 0358 FF65            DATA  >FF65
1194 0359 FF6D            DATA  >FF6D
1195 035A FF75            DATA  >FF75
1196 035B FF7D            DATA  >FF7D
1197 035C FF85            DATA  >FF85
1198 035D FF8D            DATA  >FF8D
1199 035E FF95            DATA  >FF95
1200 035F FF9D            DATA  >FF9D
1201 0360 FFA3            DATA  >FFA3
1202 0361 FFA7            DATA  >FFA7
1203 0362 FFAB            DATA  >FFAB
1204 0363 FFAF            DATA  >FFAF
1205 0364 FFB3            DATA  >FFB3
1206 0365 FFB7            DATA  >FFB7
1207 0366 FFBB            DATA  >FFBB
1208 0367 FFBF            DATA  >FFBF
```



```
1209 0368 FFC3            DATA  >FFC3
1210 0369 FFC7            DATA  >FFC7
1211 036A FFCB            DATA  >FFCB
1212 036B FFCF            DATA  >FFCF
1213 036C FFD3            DATA  >FFD3
1214 036D FFD7            DATA  >FFD7
1215 036E FFDB            DATA  >FFDB
1216 036F FFDF            DATA  >FFDF
1217 0370 FFE2            DATA  >FFE2
1218 0371 FFE4            DATA  >FFE4
1219 0372 FFE6            DATA  >FFE6
1220 0373 FFE8            DATA  >FFE8
1221 0374 FFEA            DATA  >FFEA
1222 0375 FFEC            DATA  >FFEC
1223 0376 FFEE            DATA  >FFEE
1224 0377 FFF0            DATA  >FFF0
1225 0378 FFF2            DATA  >FFF2
1226 0379 FFF4            DATA  >FFF4
1227 037A FFF6            DATA  >FFF6
1228 037B FFF8            DATA  >FFF8
1229 037C FFFA            DATA  >FFFA
1230 037D FFFC            DATA  >FFFC
1231 037E FFFE            DATA  >FFFE
1232 037F 0000            DATA  >0
1233 0380
1234 0380 1F5F            DATA  >1F5F         * POSITIVE VALUES NEXT
1235 0381 1E5F            DATA  >1E5F         *  (POLARITY BIT = 1)
1236 0382 1D5F            DATA  >1D5F
1237 0383 1C5F            DATA  >1C5F
1238 0384 1B5F            DATA  >1B5F
1239 0385 1A5F            DATA  >1A5F
1240 0386 195F            DATA  >195F
1241 0387 185F            DATA  >185F
1242 0388 175F            DATA  >175F
1243 0389 165F            DATA  >165F
1244 038A 155F            DATA  >155F
1245 038B 145F            DATA  >145F
1246 038C 135F            DATA  >135F
1247 038D 125F            DATA  >125F
1248 038E 115F            DATA  >115F
1249 038F 105F            DATA  >105F
1250 0390 0F9F            DATA  >F9F
1251 0391 0F1F            DATA  >F1F
1252 0392 0E9F            DATA  >E9F
1253 0393 0E1F            DATA  >E1F
1254 0394 0D9F            DATA  >D9F
1255 0395 0D1F            DATA  >D1F
1256 0396 0C9F            DATA  >C9F
1257 0397 0C1F            DATA  >C1F
1258 0398 0B9F            DATA  >B9F
1259 0399 0B1F            DATA  >B1F
1260 039A 0A9F            DATA  >A9F
1261 039B 0A1F            DATA  >A1F
1262 039C 099F            DATA  >99F
1263 039D 091F            DATA  >91F
1264 039E 089F            DATA  >89F
1265 039F 081F            DATA  >81F
```

```
1266 03A0 07BF          DATA  >7BF
1267 03A1 077F          DATA  >77F
1268 03A2 073F          DATA  >73F
1269 03A3 06FF          DATA  >6FF
1270 03A4 06BF          DATA  >6BF
1271 03A5 067F          DATA  >67F
1272 03A6 063F          DATA  >63F
1273 03A7 05FF          DATA  >5FF
1274 03A8 05BF          DATA  >5BF
1275 03A9 057F          DATA  >57F
1276 03AA 053F          DATA  >53F
1277 03AB 04FF          DATA  >4FF
1278 03AC 04BF          DATA  >4BF
1279 03AD 047F          DATA  >47F
1280 03AE 043F          DATA  >43F
1281 03AF 03FF          DATA  >3FF
1282 03B0 03CF          DATA  >3CF
1283 03B1 03AF          DATA  >3AF
1284 03B2 038F          DATA  >38F
1285 03B3 036F          DATA  >36F
1286 03B4 034F          DATA  >34F
1287 03B5 032F          DATA  >32F
1288 03B6 030F          DATA  >30F
1289 03B7 02EF          DATA  >2EF
1290 03B8 02CF          DATA  >2CF
1291 03B9 02AF          DATA  >2AF
1292 03BA 028F          DATA  >28F
1293 03BB 026F          DATA  >26F
1294 03BC 024F          DATA  >24F
1295 03BD 022F          DATA  >22F
1296 03BE 020F          DATA  >20F
1297 03BF 01EF          DATA  >1EF
1298 03C0 01D7          DATA  >1D7
1299 03C1 01C7          DATA  >1C7
1300 03C2 01B7          DATA  >1B7
1301 03C3 01A7          DATA  >1A7
1302 03C4 0197          DATA  >197
1303 03C5 0187          DATA  >187
1304 03C6 0177          DATA  >177
1305 03C7 0167          DATA  >167
1306 03C8 0157          DATA  >157
1307 03C9 0147          DATA  >147
1308 03CA 0137          DATA  >137
1309 03CB 0127          DATA  >127
1310 03CC 0117          DATA  >117
1311 03CD 0107          DATA  >107
1312 03CE 00F7          DATA  >F7
1313 03CF 00E7          DATA  >E7
1314 03D0 00DB          DATA  >DB
1315 03D1 00D3          DATA  >D3
1316 03D2 00CB          DATA  >CB
1317 03D3 00C3          DATA  >C3
1318 03D4 00BB          DATA  >BB
1319 03D5 00B3          DATA  >B3
1320 03D6 00AB          DATA  >AB
1321 03D7 00A3          DATA  >A3
1322 03D8 009B          DATA  >9B
```



```
1323 03D9 0093          DATA  >93
1324 03DA 008B          DATA  >8B
1325 03DB 0083          DATA  >83
1326 03DC 007B          DATA  >7B
1327 03DD 0073          DATA  >73
1328 03DE 006B          DATA  >6B
1329 03DF 0063          DATA  >63
1330 03E0 005D          DATA  >5D
1331 03E1 0059          DATA  >59
1332 03E2 0055          DATA  >55
1333 03E3 0051          DATA  >51
1334 03E4 004D          DATA  >4D
1335 03E5 0049          DATA  >49
1336 03E6 0045          DATA  >45
1337 03E7 0041          DATA  >41
1338 03E8 003D          DATA  >3D
1339 03E9 0039          DATA  >39
1340 03EA 0035          DATA  >35
1341 03EB 0031          DATA  >31
1342 03EC 002D          DATA  >2D
1343 03ED 0029          DATA  >29
1344 03EE 0025          DATA  >25
1345 03EF 0021          DATA  >21
1346 03F0 001E          DATA  >1E
1347 03F1 001C          DATA  >1C
1348 03F2 001A          DATA  >1A
1349 03F3 0018          DATA  >18
1350 03F4 0016          DATA  >16
1351 03F5 0014          DATA  >14
1352 03F6 0012          DATA  >12
1353 03F7 0010          DATA  >10
1354 03F8 000E          DATA  >E
1355 03F9 000C          DATA  >C
1356 03FA 000A          DATA  >A
1357 03FB 0008          DATA  >8
1358 03FC 0006          DATA  >6
1359 03FD 0004          DATA  >4
1360 03FE 0002          DATA  >2
1361 03FF 0000          DATA  >0
1362 0400
1363                    END
NO ERRORS, NO WARNINGS
```